

***Proceedings of the Twenty-second
Midwest Artificial Intelligence and
Cognitive Science Conference***

***April 16 – 17, 2011
University of Cincinnati
Cincinnati, Ohio***

***Edited by
Sofia Visa, Atsushi Inoue, and Anca Ralescu***

Omnipress – Madison, WISCONSIN

Contents

Contents	ii
Preface	v
MAICS-2011 Organization	vi
MAICS-2011 Sponsors	vi
Plenary Lecture	1
Fuzzy Bio-interface: Can fuzzy sets be an interface with brain?	2
<i>Isao Hayashi and Suguru N. Kudoh</i>	
Cognitive Science Society Special Session for Best Student Paper Award	7
Fuzzy Relational Visualization for Decision Support	8
<i>Brian Zier and Atsushi Inoue</i>	
A Machine Learning Approach to Identifying Sections in Legal Briefs	16
<i>Scott Vanderbeck, Joseph Bockhorst and Chad Oldfather</i>	
Automated Speech Act Classification for Online Chat	23
<i>Cristian Moldovan, Vasile Rus and Arthur C. Graesser</i>	
A Study of Query-based Dimensionality Reduction	30
<i>Augustine S. Nsang and Anca Ralescu</i>	
Expert Systems and Fuzzy Logic	39
Page Ranking Refinement Using Fuzzy Sets and Logic	40
<i>Andrew Laughlin, Joshua Olson, Donny Simpson and Atsushi Inoue</i>	
Computational Intelligence for Project Scope	47
<i>Joseph M. McQuighan and Robert J. Hammell II</i>	
Discovering Causality in Suicide Notes Using Fuzzy Cognitive Maps	54
<i>Ethan White and Lawrence J. Mazlack</i>	
Agent Systems and Evolutionary Algorithms	61
Robotic Dancing: Exploring Agents that use a Stratified Perceive-Decide-Act Cycle of Interaction	62
<i>James Benze and Jennifer Seitzer</i>	
Using a Genetic Algorithm to Evolve a D* Search Heuristic	67
<i>Andrew Giese and Jennifer Seitzer</i>	

Mining High Quality Association Rules Using Genetic Algorithms	73
<i>Peter P. Wakabi-Waiswa and Venansius Baryamureeba</i>	
Support for Agent Based Simulation of Biomolecular Systems	79
<i>Harika Korukonda and Carla Purdy</i>	
Special Session: Artificial Intelligence in Biometrics and Identity Sciences I...	85
GEFeWS: A Hybrid Genetic-Based Feature Weighting and Selection Algorithm for Multi-Biometric Recognition	86
<i>Aniesha Alford, Khary Popplewell, Gerry Dozier, Kelvin Bryant, John Kelly, Josh Adams, Tamirat Abegaz and Joseph Shelton</i>	
Iris Quality in an Operational Context	91
<i>James S. Doyle, Jr. and Patrick J. Flynn</i>	
Fusion of Face and Iris Biometrics from a Stand-Off Video Sensor	99
<i>Ryan Connaughton, Kevin W. Bowyer and Patrick Flynn</i>	
Learning and Classification	107
The Classification of Imbalanced Spatial Data	108
<i>Alina Lazar and Bradley A. Shellito</i>	
Simplifying Probability Elicitation and Uncertainty Modeling in Bayesian Networks	114
<i>Patrick Paulson, Thomas E. Carroll, Chitra Sivaraman, Peter Neorr, Stephen D. Unwin and Shamina Hossain</i>	
Confusion Matrix-based Feature Selection	120
<i>Sofia Visa, Brian Ramsay, Anca Ralescu and Esther van der Knaap</i>	
A Preliminary Study on Clustering Student Learning Data	128
<i>Haiyun Bian</i>	
Learning Morphological Data of Tomato Fruits	133
<i>Joshua Thomas, Matthew Lambert, Bennjamen Snyder, Michael Janning, Jacob Haning, Yanlong Hu, Mohammad Ahmad and Sofia Visa</i>	
A Qualitative Analysis of Edge Closure in Information Networks	138
<i>Hareendra Munimadugu and Anca Ralescu</i>	
Identifying Interesting Postings on Social Media Sites	142
<i>Swathi Seethakkagari and Anca Ralescu</i>	
Scientific Computing and Applications	145
Evolutionary Computation on the Connex Architecture	146
<i>István Lőrentz, Mihaela Malița and Răzvan Andonie</i>	

Towards a Technique of Incorporating Domain Knowledge for Unit Conversion in Scientific Reasoning Systems	154
<i>Joseph Phillips</i>	
New Features and Many Improvements to Analyze Morphology and Color of Digitalized Plant Organs Are Available in Tomato Analyzer 3.0	160
<i>Gustavo Rodriguez, David Francis, Esther van der Knaap, Jaymie Strecker, Itai Njanji, Josh Thomas and Atticus Jack</i>	
Information Retrieval	165
Towards Agent-Oriented Knowledge Base Maintenance for Description Logic ALCN	166
<i>Stanislav Ustymenko and Daniel G. Schwartz</i>	
Extracting Micro Ontologies from Interrogative Domains for Epistemic Agents	172
<i>Tracey Hughes and Cameron Hughes</i>	
An Ontological Semantic Account of Relative Quantification in English	178
<i>Whitney R. Vandiver</i>	
Intelligent Systems	187
Spatiotemporal Knowledge Representation and Reasoning under Uncertainty for Action Recognition in Smart Homes	188
<i>Farzad Amirjavid, Kevin Bouchard, Abdenour Bouzouane, Bruno Bouchard</i>	
Characteristics of Computational Intelligence (Quantitative Approach)	195
<i>Shiva Vafadar and Ahmad Abdollahzadeh Barfouroush</i>	
Toward Robust Features for Remote Audio-Visual Classroom	202
<i>Isaac Schlittenhart, Jason Winters, Kyle Springer and Atsushi Inoue</i>	
Hybrid Direct Neural Network Controller with Linear Feedback Compensator	208
<i>Sadhana K. Chidrawar and Balasaheb M. Patre</i>	
Special Session: Artificial Intelligence in Biometrics and Identity Sciences II.	215
Comparison of Genetic-based Feature Extraction Methods for Facial Recognition	216
<i>Joseph Shelton, Gerry Dozier, Kelvin Bryant, Lasanio Smalls, Joshua Adams, Khary Popplewell, Tamirat Abegaz, Damon L. Woodard and Karl Ricanek</i>	
Genetic-based Selection and Weighting for LBP, oLBP, and Eigenface Feature Extraction ..	221
<i>Tamirat Abegaz, Gerry Dozier, Kelvin Bryant, Joshua Adams, Brandon Baker, Joseph Shelton, Karl Ricanek and Damon L. Woodard</i>	
Ethnicity Prediction Based on Iris Texture Features	225
<i>Stephen Lagree and Kevin W. Bowyer</i>	
Author Index	231

Preface

Welcome to the twenty-second Midwest Artificial Intelligence and Cognitive Science Conference (MAICS2011)! This is the third time this conference is being held in Cincinnati.

This year MAICS received over 40 submissions. Over 30 papers are included in the program and the conference proceedings. Although MAICS is considered a regional conference, researchers from seven countries submitted papers to the conference. We consider this as a global recognition of the continuously improving quality of the MAICS conferences. The quality of the conference owes a great deal to the work of the program committee whose members worked diligently. They supplied thorough reviews including extended comments and suggestions for the authors. Given the diversity of topics covered by the papers submitted, the review task was rather challenging.

In keeping with the tradition of the previous conferences, this year too, graduate students and junior researchers were encouraged to submit their work to be presented along with that of more senior researchers. We are delighted by the contribution and participation of students. About 50% of accepted papers are primarily authored by students and, as a result of competitive review scores, the Program Committee nominates four student papers for the best student paper award sponsored by Cognitive Science Society.

This year the conference features two key current topics – *Brain* and *Security*. Dr. Isao Hayashi, from Kansai University, Japan, who will deliver the plenary lecture, sponsored by Omron, Japan. He shares with us recent advancements in Brain-Computer Interface and his vision for this domain. The *Security* topic is covered in the special session on Artificial Intelligence in Biometrics and Identity, whose organizers are Dr. Bryant, Dr. Dozier, Dr. Ricanek, Dr. Savvides, and Dr. Woodard. We hope that everybody will find these events informative and inspirational.

Thanks are due to many. First to York F. Choo, who (as for MAICS2008), designed the conference web page and the front cover of the conference proceedings. We thank our students, at *Eastern Washington University*, *University of Cincinnati*, and the *College of Wooster*, helped us in many ways. We thank our sponsors for their generosity and especially for their administrative support – *CINCO Credit Union*, *the Cognitive Science Society*, *College of Wooster*, *Eastern Washington University*, *Omron*, and *University of Cincinnati*.

We hope that MAICS2011 will be interesting and that, at its conclusion, you will have enjoyed its scientific aspect, engaged in lively and interesting discussions, and forged new friendships and possible collaborations.



Atsushi Inoue, General Chair



Anca Ralescu, General Chair



Sofia Visa, Program Chair

MAICS-2011 Organization

Conference Chair

Atsushi Inoue, *Eastern Washington University*

Anca Ralescu, *University of Cincinnati*

Program Chair

Sofia Visa, *College of Wooster*

Special Design Advisor

York Fang Choo

Program Committee

Razvan Andonie, USA	Byoung J. Lee, USA
Hani Abu-Salem, USA	Logan Mayfield, USA
Valentina E. Balas, Romania	Mihaela Malita, USA
Denise Byrnes, USA	Carla Purdy, USA
Alina Campan, USA	Muhammad A. Rahman, USA
Asli Celikyilmaz, USA	Anca Ralescu, USA
Dale E. Courte, USA	Dan Ralescu, USA
Susana Irene Daz, Spain	Vasile Rus, USA
Simona Doboli, USA	Ingrid Russell, USA
Martha Evens, USA	Tomasz Rutkowski, Japan
Michael Glass, USA	Michael R. Scheessele, USA
Fernando Gomide, Brazil	Tom Sudkamp, USA
Vasant Honavar, USA	Noriko Tomuro, USA
Suranga Hettiarachchi, USA	Traian M. Truta, USA
Atsushi Inoue, USA	Dan Vance, USA
Beomjin Kim, USA	Sofia Visa, USA
Jung H. Kim, USA	Dana Vrajitoru, USA
Alina Lazar, USA	

Student volunteers (in alphabetical order)

Trisha Fultz, *College of Wooster*
Mojtaba Kohram, *University of Cincinnati*
Hareendra Munimadugu, *University of Cincinnati*
Manali Pangaonkar, *University of Cincinnati*
Ahmad Rawashdeh, *University of Cincinnati*
Swathi Seetakkagari, *University of Cincinnati*
Mengxia Wang, *University of Cincinnati*

Special thanks for administrative support

Jane Runck, *School of Computing Sciences and Informatics, University of Cincinnati*

Kelly Voght, *CINCO Federal Credit Union, Cincinnati*

Sponsoring Organization

School of Computing Sciences and Informatics, University of Cincinnati

Department of Computer Science and College of Science, Health and Engineering, Eastern Washington University

Mathematics and Computer Science Department, College of Wooster

OMRON Corporation

Cognitive Science Society

CINCO

Cover Pages. Front cover design: by York Fang Choo. Back cover photograph: Built in 1911, Baldwin Hall, UC's first engineering building underwent a massive renovation and was rededicated in April 2002 (photographer Dottie Stover).

Plenary Lecture

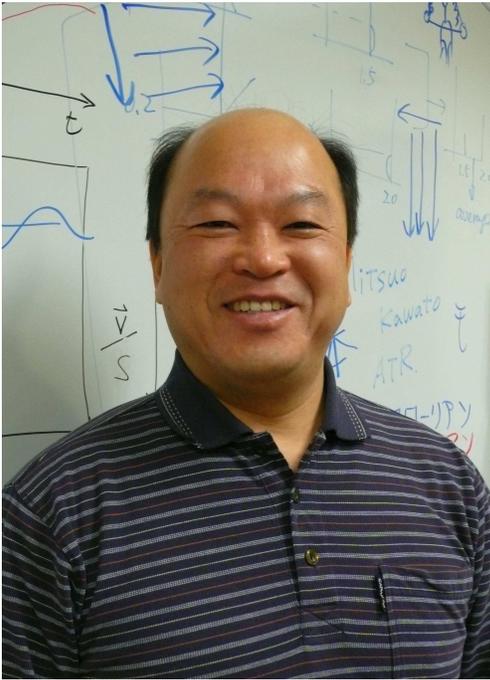
Chair: Atsushi Inoue

Fuzzy Bio-interface: Can fuzzy sets be an interface with brain?

Isao Hayashi † and Suguru N. Kudoh ‡

† Faculty of Informatics, Kansai University, Takatsuki, Osaka 569-1095, JAPAN

‡ School of Science and Technology, Kwansai Gakuin University, Sanda, Hyogo 669-1337, JAPAN



ABSTRACT

Recently, many attractive brain-computer interface and brain-machine interface have been proposed[1,2]. The outer computer and machine are controlled by brain action potentials detected through a device such as near-infrared spectroscopy (NIRS) and electroencephalograph (EEG), and some discriminant model determines a control process. However, under the condition where spontaneous action-potentials and evoked-action potentials are contained in brain signal asynchronously, we need a model that serves as an interface between brain and machine for a better stable control in order to prevent runaway reaction of machine. This interface plays a very important role to secure the stability of outer computer and machine. The interface has

two kinds of functions: (1) a decoding of the response action potentials to the control signal of outside machine and computer, and (2) an encoding of the sensor signal of the outside machine and computer to pattern of stimuli in brain and neuronal networks. Unfortunately, it is very difficult to identify such a function for the interface between machine and living brain and neuronal networks. Here we consider such an interface within the framework of fuzzy system. As a result, our study is supportive of this framework as a strong tool of the bio-interface. During the Japanese fuzzy boom in 1990's, fuzzy logic has been proven effective to translate human experience and sensitivity into control signals of machines. Tsukamoto[3] has argued a concept of fuzzy interface such that fuzzy sets is regarded as a useful tool to intermediate between language and mathematics. We believe that the framework of fuzzy system is essential for BCI and BMI, thus name this technology “fuzzy bio-interface.”

In this lecture, we introduce a fuzzy bio-interface between a culture dish of rat hippocampal neurons and the khepera robot. We propose a model to analyze logic of signals and connectivity of electrodes in a culture dish[4], and show the bio-robot hybrid we developed[5,6]. Rat hippocampal neurons are organized into complex networks in a culture dish with 64 planar microelectrodes. A multi-site recording system for extracellular action potentials is used in order to record their activities in living neuronal networks and to supply input from the outer world to the vitro living neural networks. The living neuronal networks are able to express several patterns

independently, and such patterns represent fundamental mechanisms for intelligent information processing[7].

First, we discuss how to indicate the logicity and connectivity from living neuronal network in vitro. We follow the works of Bettencourt et al.[8] such that they classify the connectivity of action potentials of three electrodes on multi-site recording system according to their entropies and have discussed the characteristic of each classification. However, they only discuss the static aspects of connectivity relations among the electrodes but not the dynamics of such connectivity concerning how the strength of electrode connection changes when a spike is fired. To address this issue, we develop a new algorithm using parametric fuzzy

connectives, that consist of both t-norms and t-conorms[9,10], in order to analyze those three electrodes (Figure 1). We have obtained the experimental result such that the parameter(s) of fuzzy connectives become infinity. Given this result, we conclude that a pulse at the 60th channel (60el) propagates to the spreading area: (51el, 59el), (43el, 50el) and (35el, 42el); and that the logic of signals among the electrodes was shifted to the logical sum from the drastic product. Consequently, the logic of signals among electrodes drastically changes from the strong AND-relation to the weak OR-relation when a crowd of the pulses was fired.

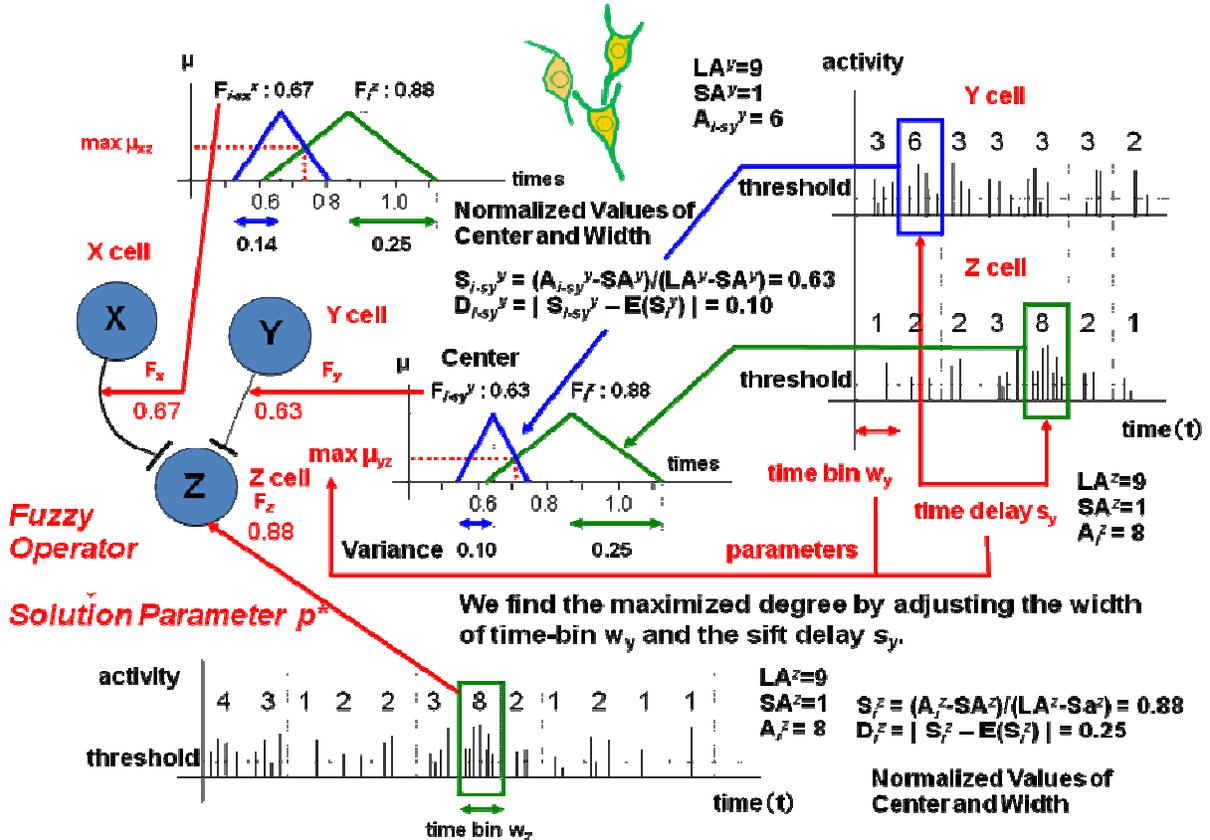


Figure 1: Algorithm to Analyze Action Potentials in Cultured Neuronal Network

Next, to control a robot, several characteristics of the living neuronal networks are represented as fuzzy IF-THEN rules. There are many works of robots that are controlled by the responses from living neuronal networks[11-15]. Unfortunately, they have not yet achieved a certain task that experimenter desired. We show a robot system that controlled by a living neuronal network through the fuzzy bio-interface in order to achieve such a task (Figure 2). This fuzzy bio-interface consists of two sets of fuzzy IF-THEN rules: (1) to translate sensor signals of robot into stimuli for the living neuronal network, and (2) to control (i.e. to determine the action of) robot based on the responses from the living neuronal network. We estimated the learning of living neuronal networks with an example of straight running with neuro-robot hybrid. Among 20 trials, the robot completed the task 16 times, and it crashed on the wall and stopped there 4 times. In this result, we may conclude that the logic of signals among living neuronal networks represented as fuzzy IF-THEN rules for the fuzzy bio-interface is rather efficient and effective comparing to the other similar works. In such works, the success rate of 80% is considered extremely high.

ACKNOWLEDGEMENT

I would like to express my gratitude to my collaborators, Megumi Kiyotoki, Kansai University, Japan and Ai Kiyohara, Minori Tokuda, Kwansai Gakuin University, Japan. This work is partially supported by the Ministry of Education, Culture, Sports, Science, and Technology of Japan under Grant-in-Aid for Scientific Research 18500181, 19200018, and 18048043 and by the Organization for Research and Development of Innovative Science and Technology (ORDIST) of Kansai University.

REFERENCES

[1] M.A.Lebedev, J.M.Carmera, J.E.O'Doherty, M.Zacksenhouse, C.S.Henriquez, J.C.Principe, and M.A.L.Nicolelis: Cortical

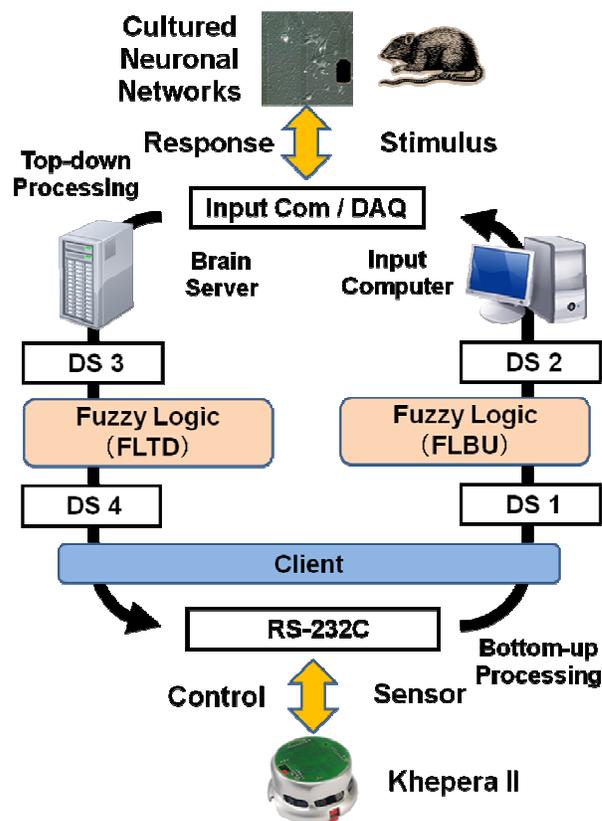


Figure 2: Living Neuronal Network and Robot

Ensemble Adaptation to Represent Velocity of an Artificial Actuator Controlled by a Brain-machine Interface, *Journal of Neuroscience*, Vol.25, No.19, pp.4681-4693, 2005.

- [2] L.R.Hochberg, M.D.Serruya, G.M.Friebs, J.A.Mukand, M.Saleh, A.H.Caplan, A.Branner, D.Chen, R.D.Penn, J.P.Donoghue: Neuronal Ensemble Control of Prosthetic Devices by a Human with Tetraplegia, *Nature*, Vol.442, pp.164-173, 2006.
- [3] Y.Tsukamoto: Fuzzy Sets as an Interface between Language Model and Mathematics Model, Proc. of the 24th Fuzzy System Symposium, Vol.251-254, 2008.
- [4] I.Hayashi, M.Kiyotoki, A.Kiyohara, M.Tokuda, and S.N.Kudoh: Acquisition of Logicity in Living Neuronal Networks and

- its Operation to Fuzzy Bio-Robot System, Proc. of 2010 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE2010) in 2010 IEEE World Congress on Computational Intelligence (WCCI2010), pp.543-549, 2010.
- [5] S.N.Kudoh, I.Hayashi, and T.Taguchi: Synaptic Potentiation Re-organized Functional Connections in a Cultured Neuronal Network Connected to a Moving Robot, *Proc. of the 5th International Meeting on Substrate-Integrated Micro Electrode Arrays (MEA2006)*, pp.51-52, Reutlingen, Germany, 2006.
- [6] S.N.Kudoh, C.Hosokawa, A.Kiyohara, T.Taguchi, and I.Hayashi: Biomodeling System - Interaction between Living Neuronal Network and Outer World, *Journal of Robotics and Mechatronics*, Vol.19, No.5, pp.592-600, 2007.
- [7] S.N.Kudoh and T.Taguchi: Operation of Spatiotemporal Patterns Stored in Living Neuronal Networks Cultured on a Microelectrode Array, *Advanced Computational Intelligence and Intelligent Informatics*, Vol.8, No2, pp.100-107, 2003.
- [8] L.M.A.Bettencourt, G.J.Stephens, M.I.Ham, and G.W.Gross: Functional Structure of Cortical Neuronal Networks Grown in Vitro, *Physical Review*, Vol.75, p.02915, 2007.
- [9] B.Schweizer and A.Sklar: Associative Functions and Statistical Triangle Inequalities, *Publicationes Mathematicae Debrecen*, Vol.8, pp.169-186, 1961.
- [10] I.Hayashi, E.Naito, and N.Wakami: Proposal for Fuzzy Connectives with a Learning Function Using the Steepest Descent Method, *Japanese Journal of Fuzzy Theory and Systems*, Vol.5, No.5, pp.705-717, 1993.
- [11] D.J.Bakkum, A.C.Shkolnik, G.Ben-Ary, P.Gamblen, T.B.DeMarse, and S.M. Potter: Removing Some 'A' from AI: Embodied Cultured Networks, in *Embodied Artificial Intelligence*, edited by F.Iida, R.Pfeifer, L.Steels, and Y.Kuniyoshi, New York, Springer, pp.130-145, 2004.
- [12] T.B.DeMarse and K.P.Dockendorf: Adaptive Flight Control with Living Neuronal Networks on Microelectrode Arrays, *Proc. of 2005 IEEE International Joint Conference on Neural Networks (IJCNN2005)*, pp.1549-1551, Montreal, Canada, 2005.
- [13] Z.C.Chao, D.J.Bakkum, and S.M. Potter: Shaping Embodied Neural Networks for Adaptive Goal-directed Behavior, *PLoS Comput Biol*, Vol.4, No.3, e1000042, 2008.
- [14] P.Marks: Rat-brained Robots take Their First Steps, *New Scientist*, Vol.199, No.2669, pp.22-23, 2008.
- [15] K.Warwick: Implications and Consequences of Robots with Biological Brains, *Journal of Ethics and Information Technology*, Vol.12, No.3, pp.223-234, 2010.

BIOGRAPHICAL SKETCH

Isao Hayashi is Professor of Informatics at Kansai University, Japan. After he received his B.Eng. degree in Industrial Engineering from Osaka Prefecture University, he worked at Sharp Corporation, Japan. After he received his M.Eng. degree from Osaka Prefecture University in 1987, he was a Senior Research Fellow of the Central Research Laboratory of Matsushita Electric Industrial (Panasonic) Co. Ltd and proposed a neuro-fuzzy system on intelligent control.

He received his D.Eng. degree based on his contributions to the neuro-fuzzy model from Osaka Prefecture University in 1991. He then joined Faculty of Management Information of Hannan University in 1993 and joined Faculty of Informatics of Kansai University in 2004. He is an editorial member of International Journal of Hybrid Intelligent Systems, Journal of Advanced Computational Intelligence and Intelligent Informatics, and has served on many conference program and organizing committees. He is the president of Kansai Chapter of Japan Society for Fuzzy Theory and Intelligent Informatics (SOFT), and the chair of the Technical Group on Brain and Perception in SOFT. He research interests include visual models, neural networks, fuzzy systems, neuro-fuzzy systems, and brain-computer interface.

Suguru N. Kudoh received his Master's degree in Biophysical Engineering in 1995 and PhD from the Osaka university in 1998. He was a research fellow of JST(Japan science and technology agency) from 1997 to 1998, and a research scientist of National Institute of Advanced Industrial Science and Technology (AIST) from 1998 to 2009. Now he is an associate professor at Kwansei Gakuin university.

The aim of his research is to elucidate relationship between dynamics of neuronal network and brain information processing. He analyses spatio-temporal pattern of electrical activity in rat hippocampal cells cultured on multi-electrode arrays or acute slice of basal ganglia. He is also developing Bio-robotics hybrid system in which a living neuronal network is connected to a robot body via control rules, corresponding to a genetically provided interfaces between a brain and a peripheral system. He believes that mind emerges from fluctuation of dynamics in hierarchized interactions between cells.

***Cognitive Science Society Special Session
for Best Student Paper Award***

Chair: Sofia Visa

Fuzzy Relational Visualization for Decision Support

Brian Zier and Atsushi Inoue *

Eastern Washington University
Cheney, WA 99004 USA

Abstract

A study on fuzzy relational visualization in system development aspects is presented. The front-end enables dynamic and scalable changes in visualization according to user's expertise and inspiration. Integrative management of various data and knowledge is handled by the back-end at any scale in cloud computing environment. Extended Logic Programming is used as the core of fuzzy relational management in the back-end, and is capable of consistent uncertainty management among probabilistic reasoning and fuzzy logic while maintaining asymptotically equivalent run-time with the ordinary Logic Programming. A multi-view relational visualization is being implemented and important graphical features are highlighted in this paper.

Keywords: Visualization, Probabilistic Reasoning, Fuzzy Logic, Logic Programming.

Introduction

Given the rapid advancement and penetration of information technologies, visualization becomes more significant in many domains. In sciences, this is utilized in many aspects such as populations, evolutions, radiations, transformations and structures (Wattenberg and Kriss 2006). In business, this is often found instrumental in various decision making, e.g. sales charts, change of markets and customer relational charts. In mathematics, modern education demands visualization as an essential element, e.g. demonstration of three dimensional functions as free surfaces. In engineering, this is a very critical, mandatory tool for system design, e.g. fluid analysis for nuclear power plants, aerodynamics for aircraft and heat radiation for CPU units to list a few.

Unfortunately, the majority of conventional visualization tends to be application-specific and its analytical model is rather static in terms of their relational representation and visualization configuration. For example, MS Excel limits its visualization within limited dimensions (1, 2 or 3), its analytical model is limited to statistical, and it only accepts tabular data. While this indeed serves in many applications, there is a fatal limit in critical decision making support, e.g. infrastructure assurance where integrated leverage of knowledge concerning policies and factual (sensory) data is essential (Inoue 2010).

On the other hand, many studies indicate effectiveness of sharing various visualization among a group in order to study extensive exploitation and in-depth understanding of data sets (Heer 2006; Heer, Viegas, and Wattenberg 2007; Viegas et al. 2007; Wattenberg and Kriss 2006). In this framework, group consensus is made as a result of sharing different interpretations through various visualizations. This suggests necessity of a general visualization platform that is capable of visualizing subjects with a dynamic and scalable change of configuration (Shneiderman 1996) via interaction with users (Shneiderman 1998; Zhang 2008).

Our general application framework for intelligent systems deploys Extended Logic Programming (ELP) and a multi-view visualization scheme, and its efficiency and effectiveness have been demonstrated throughout a showcase of various applications (Springer, Henry, and Inoue 2009). In this paper, we discuss the system development of fuzzy relational visualization for decision support within this application framework. First, the specification and progress are reported. Then the management of fuzzy relations (back-end) and their visualization scheme (front-end) are described respectively.

Specifications and Progress

The ultimate goal of this development is a fuzzy relational visualization with the following general specification:

- S1.** Various relations are dynamically visualized in various aspects.
- S2.** Various types of data are visualized, e.g. tabular, texts, images, multimedia streams, diagrams, etc.
- S3.** Relations can be uncertain, i.e. probabilistic, possibilistic, and perceptual (subjective).
- S4.** Knowledge and data are integratively managed throughout a canonical representation and process.
- S5.** This is scalable in cloud computing environment.

In this specification, the first two (S1 and S2) are considered as matters of the front-end that provide graphical interfaces and interactions with users, and the rest (S3, S4 and S5) as matters of the back-end that manage relations among all data and knowledge.

Two major works on the front-end are visualization scheme and human-computer interaction. There are two

*E-mail: inoueatsushij@gmail.com

major progresses on the visualization scheme: first, the Logic Programming (LP) visualizer for educational purpose (Henry and Inoue 2007), then the visualization scheme for reasoning under uncertainty (Springer and Inoue 2009). For this fuzzy relational visualization, we further specify this scheme in order to realize dynamic and scalable visualization as a result of developing an interactive graphic interface. Studies on more sophisticated human-computer interaction, including integration of this relational visualization scheme with various conventional visualization, e.g. geographical, spacial, statistical and functional, are planned and upcoming.

The back-end consists of ELP and extraction functionalities, and they are placed in a cloud computing environment. ELP is developed with Support Logic Programming (SLP) (Baldwin, Martin, and Pilsworth 1995) and a simple extension of fuzzy probability (Inoue 2008). Extraction functionalities consists of translators from various types of data into ELP and utilities to manage massiveness and high dimensions (Codd 1970; Nugues 2006; Moore and Inoue 2008; Yager 1982). Parallelization of ELP in a cloud computing environment is currently underway (Joxan and Maher 1994).

Management of Fuzzy Relations: Back-End

This section describes management of fuzzy relations from computational aspects: representation and query processing of ELP, as well as how various types of data are extracted and translated into this representation, i.e. extended Horn clauses.

Representation

Table 1 shows how fuzzy relations can be represented in ELP. Like ordinary LP, Horn clauses are used to represent relations in general. Two extensions are made in those Horn clauses. One is the various probability annotation such as a point (e.g. 0.62), an interval (e.g. [0.6, 0.68]) and a fuzzy (i.e. linguistic) (e.g. 'very_low' and 'high'). The interpretation of those annotated Horn clauses, i.e. probabilistic events, is $P(h) \in [0, 1]$, where P is the annotated probability and h is the Horn clause. We interpret $P(h) = 1$ when no probability is annotated. The other is use of fuzzy predicates such as 'Tall', 'BigFeet', 'ProceedAtPace' and 'Level' in this table. This is simply a matter of fuzzy predicates observed in the Horn clause h , and such predicates are specially processed in their unification.

Fuzzy probability is formally defined as a normal, convex fuzzy set defined over interval $[0, 1]$ (i.e. a fuzzy number), s.t. $\mu_p(x \in [0, 1])$. In addition, a linguistic label is associated with such a fuzzy set for our advantage, i.e. the linguistic extension of annotated probability in ELP.

Fuzzy predicate is formally expressed s.t. $p_f(x_1, \dots, x_n)$ and its semantics is determined by a corresponding fuzzy set s.t. $\mu_{p_f}(x_1, \dots, x_n)$, where $(x_1, \dots, x_n) \in U$, the universe of discourse for this fuzzy set. Truth values of such a predicate, by its nature, are partial, i.e. $\tau \in [0, 1]$ where $\tau = 0$ corresponds to false and $\tau = 1$ corresponds to true, as well as other values

Table 1: Fuzzy Relations in ELP

Probability	Event	
	Crisp	Fuzzy
None	Human(Kyle) \leftarrow Student(Kyle)	Tall(Kyle)
Point	Awake(Kyle) \leftarrow Healthy(Kyle): 0.62	BuysTallPants(Kyle) \leftarrow Tall(Kyle): 0.4
Interval	Awake(Kyle) \leftarrow Healthy(Kyle), Weekend(): [0.6, 0.68]	Tall(Kyle) \leftarrow BigFeet(Kyle), TallerThanObserver(Kyle): [0.8, 0.92]
Fuzzy	LightningStrike(Kyle): very_low	ProceedAtPace(vehicle) \leftarrow Level(terrain), HighTraction() : high

in-between correspond to partial truth. Currently we do not consider cases that fuzzy terms (i.e. fuzzy sets) appear in its arguments. They are rather unorthodox in Fuzzy Logic framework and, if necessary, can be translated into fuzzy predicates $p_{f_i}(x'_i)$, where f_i represents the i -th fuzzy term appearing in the arguments, to be properly inserted into the original Horn clause.

Query Processing

The most critical advantage of ELP is its computational efficiency, that is asymptotically equivalent with that of LP while the extensions of uncertainty management are indeed in a part of its computation. Consider the following simple extended Horn clauses, with query a and unification of some fuzzy predicates such as a and a' as well as c and c' , in order to demonstrate this efficiency:

- | | |
|---|--------------|
| h1: $a \leftarrow b \wedge c \wedge d : p1$ | h6: $b : p6$ |
| h2: $a \leftarrow e : p2$ | h7: $d : p7$ |
| h3: $c \leftarrow e \wedge f : p3$ | h8: e |
| h4: $c \leftarrow d : p4$ | h9: f |
| h5: $a' \leftarrow c' \wedge d : p5$ | |

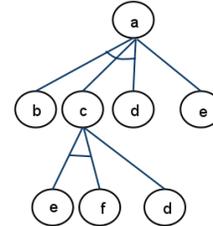


Figure 1: Snapshot of processing query a in LP

First, we consider ordinary LP in order to process query a with the assumption of all annotated probabilities $p1, \dots, p7$ to be 1 (i.e. the equivalence of no annotations). Figure 1 indicates the snapshot of this query processing in an AND-OR tree. In general, the query processing in LP is Depth-First Search starting from node a . In LP, we only consider symbolic unification so that there is no partial unification such as a and a' , as well as c and c' . Further, we only need one Horn clause to be proven true (so-called an *existential query*) – ei-

ther h1 or h2 (together with either h3 or h4 in order to prove sub-query c) in this query. Sometimes, we need to prove all possible cases (so-called a *universal query*), i.e. both h1 and h2 (together with both h3 and h4) in this query. The selection between existential and universal queries depends on applications. LP assumes *close world assumption* (i.e. negation as failure). Since recent knowledge representation technologies often deploy *open world assumption* such as Web Ontology Language (OWL), this is often considered as a shortcoming.

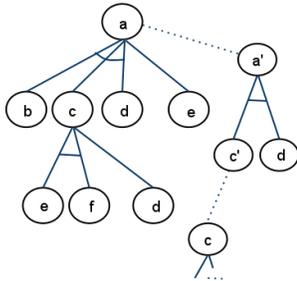


Figure 2: Snapshot of processing query a in ELP

Query processing in ELP remains in Depth-First Search starting from node a as indicated in Figure 2. In order to process query a in ELP, we need to disjunctively combine all partial truth of h1 and h2, as well as h5 (i.e. the case of fuzzy predicates) s.t. $P_{h1}(a) \cup P_{h2}(a) \cup P_{h5}(a)$. Therefore, all Horn clauses need to be proven, i.e. the equivalence of the universal query in LP.

The partial truth in ELP is represented as a probability: either one of point, interval and fuzzy, and this is computed according to Jeffreys' rule (Jeffrey 1965) s.t. $P(c) = P(c|h) \cdot P'(h) + P(c|\neg h) \cdot P'(\neg h)$. Therefore, proof by satisfying sub-queries in ELP is a matter of computing such probabilities in a chain reaction. Let a conditional Horn clause be $c \leftarrow h : p = P(c|h)$ and the result of a sub-query (or simply a fact) be $h : p' = P'(h)$. Then the partial truth of query c , i.e. $P(c)$, is computed depending on the type of annotated probability according to Jeffreys' rule as shown in Table 2. Note that $P(c|\neg h) = 0$ (i.e. false) is expected for close world assumption (i.e. negation as failure) and $P(c|\neg h) = [0, 1]$ (i.e. unknown) for open world assumption.

Table 2: Partial truth of query c , i.e. $P(c)$

Probability	negation as failure	open world assumption
Point	$p \cdot p'$	$[p \cdot p', p \cdot p' + 1 - p']$
Interval: $p = [l, u]$	$[l \cdot l', u \cdot u']$	$[l \cdot l', u \cdot u' + 1 - u']$
Fuzzy (trapezoidal) core of μ_p : $[lc, uc]$ support of μ_p : $[ls, us]$	core: $[lc \cdot lc', uc \cdot uc']$ support: $[ls \cdot ls', us \cdot us']$	core: $[lc \cdot lc', uc \cdot uc' + 1 - uc']$ support: $[ls \cdot ls', us \cdot us' + 1 - us']$

When h in the conditional Horn clause $c \leftarrow h : p$ consists

of multiple predicates, s.t. $h = h_1 \wedge \dots \wedge h_n$, we compute $p' = P'(h) = P(h_1) \cdot \dots \cdot P(h_n)$.

Partial truth between fuzzy predicates f and f' (e.g. dotted lines between a and a' , and between c and c' in Figure 2) is determined by applying Mass Assignment Theory, the conditional mass assignment $m_{f|f'}$ that yields an interval of probability¹ (Baldwin, Martin, and Pilsworth 1995). This is so-called *semantic unification* as opposed to symbolic unification. Importantly this is not symmetric unlike symbolic unification, i.e. $m_{f|f'} \neq m_{f'|f}$. In the query processing aspect of ELP, this is considered as insertion of Horn clauses $f \leftarrow f' : p$ and $f \leftarrow \neg f' : \bar{p}$, where $p = P(f|f') = m_{f|f'}$ and $\bar{p} = P(f|\neg f') = m_{f|\neg f'}$. Note that neither close world assumption nor open world assumption holds in any query process with semantic unification. This is indeed consistent with Fuzzy Logic.

Computing partial truth adds a few simple arithmetic to unification as shown in Table 2. While this may increase a coefficient of run-time, its asymptotic complexity still remains the same. Similarly to semantic unification in comparison with symbolic unification, its computation depends on the shape of fuzzy sets (i.e. #pivotal points) but not on the number of Horn clauses or that of predicates within those. Furthermore, this computation even becomes less as those fuzzy sets are more simply represented (e.g., trapezoidal-only 4 points).

Extraction

Extraction functionalities translate various types of data into a collection of extended Horn clauses. Following the concept of deductive databases, any data in tabular forms are translated into a collection of unconditional Horn clauses, i.e. facts, and any relational queries, e.g. SQL, are translated into a collection of Horn clauses (Codd 1970; Ceri, Gottlob, and Tanca 1990). Unstructured texts are to be translated into a collection of Horn clauses as a result of applying Natural Language Processing (NLP) such as tagging, syntax parsing and semantic processing in LP (Nugues 2006). Semi-structured data such as XML, E-mail and Electric Data Interchange (EDI) have a high compatibility with Horn clauses (Almendros-jimenez, Becerra-tern, and j. Enciso-baos 2008). As a consequence of this, anything that can be represented in XML is translated, e.g. diagrams.

Multimedia data such as images, audio and video are handled through their summarization, e.g. color histograms, edge and shape extractions and any other image processing. Tagged information and attributes are straightforwardly translated into Horn clauses. Texts such as captions are translated by applying NLP. Their contents can be efficiently summarized and, in a sense, compressed by applying Granular Computing and linguistic summary (Moore and Inoue 2008; Yager 1982). This can also be applied to any other data that are massively large and highly dimensional.

Overall, a rich set of extraction functionalities serves as a strong interface because Horn clauses are considered rather

¹The conditional mass assignment, i.e. semantic unification, may also yield a point probability (Baldwin, Martin, and Pilsworth 1995). However, we do not consider this in ELP.

as a pivotal language (thus, users do not have to be extensively exposed to ELP). In knowledge management for infrastructure assurance, various factual (sensory) data can be entered in tabular forms and XML. Knowledge such as policies and scheduling rules can be entered in texts. Then, minor modification and refinement are to be made as deemed necessary through some human-computer interaction for visualization in decision making.

Visualization Scheme: Front-End

The concepts of creating a visualization with various views represent the data at different levels of detail. We chose to implement a global view and a local view. The global view displays the relations in a wide range. This view allows a user to gain a broad understanding of the various components and relationships as a whole. Additionally, it is important for the user to be able to more closely understand particular subsets of the whole, particularly when the visualization is large. This necessitates the local view, which allows the user to drill down to a particular subsection of the global visualization and view the details of the relations.

We designed and implemented the prototype front-end application with several things in mind. This included the capacity to utilize the program on various platforms, leaving the door open for future expansion. For example, we wanted to ensure that this application was independent of any specific operating system. We also kept in mind that the future (or even the current) trend of technology is moving to service-based applications in cloud computing. Because the potential for this visualization system could grow to very large applications, having a powerful back-end system performing the operations and calculations could be beneficial, requiring only minimal processing power of the front-end system. Additionally, the system would be universally available and accessible regardless of where the user is. Due to these future possibilities, we designed the input specification around XML and implemented the visualization components in the Java programming language.

Input file format design

We had to develop a format which would include all necessary information about the reasoning processes to be visualized. Because the reasoning process can easily be represented in a tree structure, we chose an XML format for the input file.

Fuzzy probabilities in the local view



Figure 3: Point probability

In the aforementioned research, two methods for presenting probabilities in the local view are offered (Springer and Inoue 2009). One of these methods represents a crisp, single-point probability; for example: 0.8. The paper suggests that a rectangular box is displayed. Inside this box,

a smaller rectangular bar is displayed representing the possible range of probabilities (from 0.0 to 1.0) using a color spectrum or gradient. Therefore, the color at the left end of the bar represents a probability of 0, and the color at the other end represents 1. Any color in between is then easily seen as representing a probability somewhere between these possible values. The outer box is then filled with the color representing the point probability for that particular event. (See figure 3).

The second method involves representing an interval probability. This method is very similar to the last, in that we still have a rectangular box with a smaller bar with the spectrum representing the range of probabilities. The difference is that the outer box is filled also with a gradient over the probability interval for that event. So if the probability was $[0.1, 0.4]$, then the outer box would be filled with a gradient ranging over the colors represented within the spectrum between those values. (See figure 4).

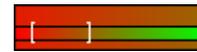


Figure 4: Interval probability

Both of these methods are very good visual representations of the probability. These visualizations make it quick to easily identify the probability of a particular event. They are also easy to compare, even between the single point probability and the interval probability. The challenge which we faced was determining an equally good method of visualizing a fuzzy probability. In this case, the probability of a particular event is represented by a fuzzy set. This means that each probability will have a membership value based on the membership function which defines the fuzzy set. After some discussion, we came up with three feasible representations of fuzzy probability for this particular project. There obviously could be many more ways to represent fuzzy probabilities; however, we needed ways that would be easy to directly compare with the other two representations.

The method which came to mind first was to represent the shape of the fuzzy set. This was quickly modified to include the gradient of color to enhance this representation. Inside the rectangular box used for the other two methods, the shape of the fuzzy set would be drawn and filled with the portion of the gradient which fit within that shape. (See figure 5).



Figure 5: Fuzzy probability using the shape of the fuzzy set

The second method which we consider is to represent the probability with color gradients layered based on particular α -cuts of the fuzzy set. After some experimentation, we discovered that the most effective method for representing in this way was to use the maximum number of α -cuts based on the height of the containing rectangular box (in pixels).

So, if the containing rectangular box was 24 pixels tall, we take 24 α -cuts of the fuzzy set and paint that row of pixels with the gradient representing the probability interval at that α -cut. This results in a color pattern which we will describe as a two dimensional gradient. (See figure 6).



Figure 6: Fuzzy probability using interval gradients for each α -cut

The third and final method we considered was to represent the fuzzy set by changing the color value of the gradient based on the fuzzy set. For example, decreasing the saturation or the brightness of the color based on the membership value of the particular point. For this representation, we developed three variations, one which decreased the saturation, another the brightness, and the other a combination of brightness and saturation. (See figure 7). After comparing these three options, we found that the method which decreased the brightness was the most clear and intuitive (see figure 8).



Figure 7: Comparing variations of color value



Figure 8: Fuzzy probability with decreasing brightness

After examining all three of these methods (shape of the fuzzy set, two dimensional gradients, and color value), we determined that certain people will view each of these with different degrees of usefulness. One person may find the first option the most intuitive. However, others may find the second or third options most intuitive. We decided that it would be most beneficial to include all three representations of fuzzy probability (see figure 9) in the local view program and allow the user to toggle between them depending on their personal preference or intuition. This will allow the user to choose an option which suits their eye and allows them to easily compare between point, interval, and fuzzy probabilities.

Local view development

For this development, we chose to use the Java programming language for several reasons. First and most importantly, Java is platform independent. Java GUI programs can also easily be converted to web applets, which could make the application even more portable by making it available through a web interface online.



Figure 9: Comparison among shape of the fuzzy set, α -cuts and decreasing brightness

Layout The local view was developed to be a box-in-box style layout. There is a top panel, a left panel, and a bottom right panel. The top panel is used for displaying information about the node, currently just the node's name. The left panel is used for displaying the calculated probability panel underneath the name of the node. The bottom right panel is then used as a container for any children of the node. All nodes with dependencies and children are given this same three-part layout. This layout is then added to the parent's bottom right panel, creating an embedded box-in-box style as specified. For the leaf nodes with no children, we simply display a single panel which contains the name of the node and a given probability panel to the right. The given probability panels for the deepest leaf nodes are drawn to touch the right border of their enclosing box, while other leaf nodes that are not as deep are indented to the left to allow quick vertical comparison between different levels. According to the input file specification, we can have several different 'branches' (i.e. separate Horn clauses) or dependencies grouped together by 'and's (i.e. conjunctively connected predicates within a Horn clause). This is represented in the local view by a slightly thicker border between the different children. Figure 10 shows a complete local view.



Figure 10: Complete local view

Global view development

The global view is conceptually straight-forward, and simpler than the local view. However, implementation turns out to be more challenging. The global view is a representation of the entirety of the reasoning process. Ideally with this visualization application, a user will be able to view the whole reasoning process in the global view with little to no detail and, in order to see more detail, look at a particular subsection of the reasoning process in the local view. This means that the global view should accommodate a large number of nodes in a small amount of space, while still pro-

viding significant information regarding the reasoning process. Our previous work determined that the global view should be, what we call, a circular tree. It "combines the relationship visibility of a *standard tree structure* with the radial organization and space efficiency of a *tree ring structure*." (Springer and Inoue 2009).

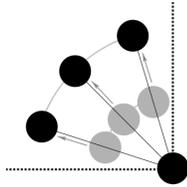


Figure 11: With a small radius, the nodes are too close together, but by increasing the radius, we create more space between the nodes while maintaining the same angles of placement

Node Positioning In order to develop this view, we needed to confront challenging issues. The first and the most critical issue is the node placement. We had to determine a method of calculating the position of each node. We considered a couple of different options, but decided that it was simplest to divide the space among the child nodes evenly. For example, the root node will begin with 360 degrees of space, which it will divide evenly among its children. Each of those nodes will then be given a placement angle as well as a certain number of degrees to allocate to their children. One issue with dividing the 'arc space' evenly among the children is that we could have one branch with many children and descendants and another with very few. However, both branches would be given an equal amount of space. This requires us to ensure that all nodes are given enough space in their angle on the circumference. If we have nodes of a particular size and which have been given a certain angle with which to work, the only thing left to manipulate in order to give enough space is the radius (see figure 11). We decided to use a consistent distance between levels of the tree. This was the simplest to implement, as well as, what we believe to be, the most clear visually. So in order to calculate node positions, we must determine the required circumference to give enough space for the most nodes in the smallest angle as follows (equation 1, where C is the set of children, d is the node diameter, a is the node's given arc space, and l is the node level).

$$\frac{1.5 \cdot |C| \cdot d}{2\pi \cdot a \cdot (l - 1)} - d \quad (1)$$

Once we know the circumference, we can determine the radius, and since we know the depth level of the nodes, we can also determine the distance between levels, or what we call the link length. With this information, we can now easily calculate the positions of each node because we know its specified angle, its depth level, and the link length (i.e. the distance between each level). After calculating the placement of each node, we must generate the links between them. Each node had a reference to its parent node, and

both nodes know their own positions, so we simply have each node draw a link from its position to its parent.

'And Arcs' With both the nodes and links in place, we must also draw connecting arcs for the branches which are grouped by 'and's. These 'and arcs' must connect the links from the first child node to the last which are part of the conjoined dependencies. To implement this, we must know the point of origin (i.e. the parent node's position), the angle of the first child in relation to the parent, and the angle of the last child in relation to the parent. This was a challenge as we have the angle for each node from the root node, not the angle with respect to the parent node. However, because we can calculate the coordinates of the child node as well as the parent node, we can calculate the angle from the parent as follows (equation 2, where A_s is the start angle, A_e is the end angle, (x_s, y_s) is the start location, (x_e, y_e) is the end location, and (x_p, y_p) is the parent location. For coding, we assume $A_s, A_e \geq 0$ and make necessary conversion, e.g. adding 360 degrees.).

$$\begin{aligned} A_s &= \tan^{-1} \frac{y_s - y_p}{x_s - x_p} \\ A_e &= \tan^{-1} \frac{y_e - y_p}{x_e - x_p} \end{aligned} \quad (2)$$

After calculating the angle from the parent to both the first and last child in the 'and' group, we can then simply draw an arc from the first to the last.

Zooming and Fuzzy Nodes The capability to zoom in or out on the global view is very critical in visualization. As far as coding was concerned, this is fortunately very simple because all of the position calculations are based on the node diameter and angles. The zoom feature simply scales the node diameter, which effectively scales everything else. It has been designed as a slider control in the bottom of the window, but could easily be changed to be any other type of interface control as well.

Lastly for the global view, we added a simple indication of fuzzy nodes. For those nodes (represented in the local view with italicized names), we drew a dashed white circle just outside the node. This allows the user to easily differentiate between fuzzy and crisp events, but does not detract from the ability to see the coloring of the node. A completed global view is shown in figure 12.

Coloring

The color calculations for the probabilities are made by a simple scale of the two primary color (red and green in our case) components by the probability. Because the probability is between 0 and 1, this scales each value between 0 and 255 in the RGB color space. To calculate the red component, we invert the calculation such that the higher the probability is, the lower the red value becomes. The combination of the scaled red and green values gives us our color.

For the interval probabilities, the calculations are the same, except that we just have to do the one for the low end of the interval and the other for the high end. To create the gradient, we generate the start color for the low probability and the end color for the high probability. This creates a gradient from the low to the high.

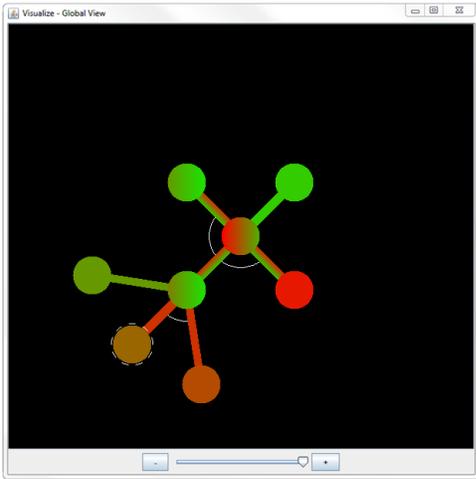


Figure 12: Completed global view

The concept of coloring for the local view had already been well-defined in Springer's previous work; however, the global view had not been specified aside from remaining consistent with the coloring in the local view. We determined that the leaf nodes should display their given probabilities and the non-leaf nodes should display their calculated probabilities. Then the links between nodes display the given probability for their appropriate 'and' branch. This is consistent with the local view in which the leaf nodes display their given probabilities and non-leaf nodes their calculated probabilities.

In order to represent point and interval probabilities on a node, we simply color the node the appropriate solid or gradient color in the scheme previously discussed. For fuzzy probabilities, we take the core, which is an α -cut where $\alpha = 1$ —representing the best-case interval. This produces an interval, which we represent with a gradient over the color spectrum discussed above. The links are colored in the same way, with the gradient going across the width of the link rather than down its length (see figure 13). Otherwise it could appear that the probability changes from one node to the next, when we are trying to represent a probability of the rule or event as a whole, not some transitional probability.

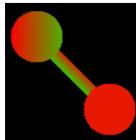


Figure 13: Node and link coloring

Transition between local and global views

A future goal of this project is to allow a user to easily make transition between the local view and global view, and also to potentially include more detail in the global view as the user zooms and manipulates the view. First ideas include:

the ability to look at a particular subsection or branch of the tree in the global view (from the global view), the ability to look at a particular subsection or branch of the tree in the local view (from the global view), the ability to go back to the complete global view (from the local view), and popup windows which include more specific information about the nodes and 'and' branches in both views (particularly the probabilities). We will briefly describe each of these ideas further.

The ability to look at a particular subsection or branch in the global view could be implemented in such a way that you click on a particular node in the global view and it makes this node into the root node in the center of the screen and repositions each of its children around it in the same circular fashion. This allows the user to more closely examine a particular branch without yet having to see all detail associated with the nodes (as in the local view). For example, assume we have a tree with a root node that has five children, and each of these branches has hundreds of descendants. The user could click on one of the five children, which would then become the root and take the center position, and its descendants would then be repositioned all around it, giving each more space and hopefully making it clearer to see the dependency links.

The ability to look at a particular subsection or branch in the local view would be very similar to the previous idea. However, once a user has found a particular (small) branch which a user wishes to examine more closely, the user can choose to view a particular node (and all dependencies/children) in the local view. This would transition them seamlessly and allow them to see all of the information the local view presents which the global view does not. Following this concept is the idea that a user should be able to easily return to the global view after examining a subsection or branch of the whole tree. Ideally the user should be returned to the same subsection from which they came in the global view to maintain a consistent frame of reference between views. However, this could also be accomplished by highlighting or outlining the nodes (in the global view) of the particular subsection the user was examining in the local view so the user can easily recognize and find the nodes in the scope of the rest of the tree.

Finally, we have had some study about various popups in each view. In the global view, a user could click a node and initiate a popup indicating the node's name together with calculated or given probability. Likewise, if the user were to click a link, a popup could indicate the given probability for that particular branch. Currently the given probability for an 'and' group is not shown in the local view, but a popup could display that probability for its branch. There are many ways in which a popup-style window could enhance both of the views. We plan to study those throughout a challenging knowledge management case.

We have a few other potential enhancements. As an alternative to the popup for displaying given probabilities of 'and' groups in the local view, we have studied effectiveness of placing additional probability panels to the left of the children which are a part of the 'and' branch. These panels would span the height of the children in the group

and probably be narrower than the regular probability panels. However, this would distinguish these probabilities from those directly associated with a node, and would allow easy comparison between several branches at the same level of the tree. This could be fairly easily implemented by utilizing the existing probability panel classes which we have already implemented. A screenshot of such panels is shown in figure 14.



Figure 14: Given probability panels for 'and' groups (conceptual screenshot)

As mentioned earlier in the paper, we have a plan to develop this application in a Java applet on a web server for cloud computing.

Concluding Summary

Our study on fuzzy relational visualization in system development aspects is presented. Our recent work has resulted in a prototype proof-of-concept that is capable of dynamic and scalable visualization. The visualization scheme and the first set of fundamental human-computer interactions have been developed. With the ever increasing complexity of various decision making, visualization is becoming more and more essential. There are many areas in which work still needs to be done; however, this prototype pushes the boundaries of current visualization techniques and limitations in the directions as introduced at the beginning of this paper.

While completing and further enhancing elements discussed in this paper, we are anticipating the following future works throughout challenging knowledge management domains, e.g. infrastructure security and health informatics:

- More sophisticated human-computer interaction and study on its effectiveness from aspects of cognitive sciences.
- Parallelization and scalability in aspects of concurrent logic programming for the back-end and that of distributed computing features in Java for the front-end.

Acknowledgment

This work is conducted for and partially supported by NSF-IUSTF International Collaboration on Infrastructure Security and Health Informatics Technology and Management Initiative at Eastern Washington University. Computational resources are provided by Computer Science Department at Eastern Washington University.

References

Almendros-jimnez, J.; Becerra-tern, A.; and j. Enciso-baas, F. 2008. Querying XML Documents in Logic Pro-

gramming. *Theory and Practice of Logic Programming* 8(3).

Baldwin, J. F.; Martin, T.; and Pilsworth, B. 1995. *FRIL: Fuzzy and Evidential Reasoning in AI*. Research Studied Press.

Ceri, S.; Gottlob, G.; and Tanca, L. 1990. *Logic Programming and Databases*. Springer-Verlag.

Codd, E. F. 1970. A Relational Model of Data for Large Shared Data Banks. *Communications of the ACM* 13(6):377–387.

Heer, J.; Viegas, F.; and Wattenberg, M. 2007. Voyagers and Voyurs: Supporting Asynchronous Colalbollative Information Visualization. In *CHI2007*.

Heer, J. 2006. Socializing Visualization. In *CHI2006 Workshop on Social Visualization*.

Henry, M. D., and Inoue, A. 2007. Visual Tracer for Logic Programming. In *ISIS2007*.

Inoue, A. 2008. Uncertainty Management by Extension of Logic Programming. In *FSS2008*.

Inoue, A. 2010. Toward a Comprehensive Knowledge Management for Infrastructure Assurance. In *iWiA2010/IFIP TM2010*, 90–96.

Jeffrey, R. 1965. *The Logic of Decision*. McGraw-Hill.

Joxan, J., and Maher, M. J. 1994. Constraint Logic Programming: a Survey. *Journal of Logic Programming* 19/20:503–581.

Moore, Z. I., and Inoue, A. 2008. Effectiveness of Value Granulation in Machine Learning for Massively Large and Complex Domain. In *IPMU2008*.

Nugues, P. 2006. *An Introduction to Language Processing with Perl and Prolog*. Springer-Verlag.

Shneiderman, B. 1996. The Eyes have It: A Task by Data Type Taxonomy for Information Visualizations. In *VL96*, 336–343.

Shneiderman, B. 1998. *Designing the User Interface (3rd eds)*. Addison-Wesley.

Springer, K., and Inoue, A. 2009. Novel Visualization Scheme for Reasoning With Uncertainty. In *NAFIPS2009*.

Springer, K.; Henry, M.; and Inoue, A. 2009. A General Application Framework for Intelligent Systems. In *MAICS2009*, 188–195.

Viegas, F.; Wattenberg, M.; Ham, F.; Kriss, J.; and McKeon, M. 2007. ManyEyes: a Site for Visualization at Internet Scale. *IEEE Trans. on Visualization and Computer Graphics* 13(6):1121–1128.

Wattenberg, M., and Kriss, J. 2006. Designing for Social Data Analysis. *IEEE Trans. Visualization and Computer Graphics* 12(4):549–557.

Yager, R. 1982. A New Approach to the Summarization of Data. *Information Sciences* 28:69–86.

Zhang, J. 2008. *Visualization for Information Retrieval*. Springer-Verlag.

A Machine Learning Approach to Identifying Sections in Legal Briefs

Scott Vanderbeck and Joseph Bockhorst

Dept. of Elec. Eng. and Computer Science
University of Wisconsin - Milwaukee
P.O. Box 784 , 2200 E. Kenwood Blvd.
Milwaukee, WI 53201-0784

Chad Oldfather

Marquette University Law School
P.O. Box 1881
Milwaukee, WI 53201-1881

Abstract

With an abundance of legal documents now available in electronic format, legal scholars and practitioners are in need of systems able to search and quantify semantic details of these documents. A key challenge facing designers of such systems, however, is that the majority of these documents are natural language streams lacking formal structure or other explicit semantic information. In this research, we describe a two-stage supervised learning approach for automatically identifying section boundaries and types in appellee briefs. Our approach uses learned classifiers in a two-stage process to categorize white-space separated blocks of text. First, we use a binary classifier to predict whether or not a text block is a section header. Next, we classify those blocks predicted to be section headers in the first stage into one of 19 section types. A cross-validation experiment shows our approach has over 90% accuracy on both tasks, and is significantly more accurate than baseline methods.

Introduction

Now that most of the briefs, opinions and other legal documents produced by court systems are routinely encoded electronically and widely available in online databases, there is interest throughout the legal community for computational tools that enable more effective use of these resources. Document retrieval from keyword or Boolean searches are key tasks that have long been a focus of natural language processing (NLP) algorithms for the legal domain. However, the simple whole document word-count representations and document similarity measures that are typically employed for retrieval limits their relevance to a relatively narrow set of tasks. Practicing attorneys and legal academics are finding that the existing suite of tools fall short of meeting their growing and complex information needs.

Consider, for example, Empirical Legal Studies (ELS), a quickly growing area of legal scholarship that aims to apply quantitative, social-science research methods to questions of law. ELS research studies are increasingly likely to have a component that involves computational processing of large collections of legal documents. One example, are studies of the role of ideological factors that assign an ideological value to legal briefs (*e.g.*, conservative or liberal (Evans *et al.* 2006)). One problem that may arise in settings like

this that employ a general similarity measure not tailored to the task at hand is that documents are more likely to group by topics, for instance the type of law, than by, say, ideology.

One general technique that has the potential to improve performance on a wide range of ELS and retrieval tasks is to vary the influence of different sections of a document. For example, studies on ideology, may reduce the influence of content in the “Statement of Facts” section while increasing the influence of the “Argument” section. However, although most briefs have similar types of sections, there are no formal standards for easily extracting them. Computational techniques are needed. Toward that end, we describe here a machine learning approach to automatically identifying sections in legal briefs.

Problem Domain

Our focus here is on briefs written for appellate court cases heard by the United States Courts of Appeals. The appeals process begins when one party to a lawsuit, called the appellant, asserts that a trial court’s action was defective in one or more ways by filing an appellant brief. The other party (the appellee) responds with an appellee brief, arguing why the trial courts action should stand. In turn, the appeals court provides its ruling in a written opinion. While there is good reason to investigate methods for identifying structure in all three kinds of documents, for simplicity we restrict our focus here to appellee briefs. We conduct our experiment using a set of 30 cases heard by the First Circuit in 2004.

In the federal courts, the Federal Rules of Appellate Procedure require that appellant briefs include certain sections, and that appellees include some corresponding sections while being free to omit others. There is, however, no standard as to section order or how breaks between sections are to be indicated. Moreover, parties often fail to adhere to the requirements of the rules, with the result being that authors exercise considerable discretion in how they structure and format the documents.

Related Work

Many genres of text are associated with particular conventional structures. Automatically determining all of these types of structures for a large discourse is a difficult and unsolved problem (Jurafsky & Martin 2000). Much of the

previous NLP work in the legal domain concerns Information Retrieval (IR) and the computation of simple features such as word frequency (Grover *et al.* 2003).

Additional work has been done in the legal domain with the focus on summarizing documents. Grover *et al.* developed a method for automatically summarizing legal documents from the British legal system. Their method was based on a statistical classifier that categorized sentences in the order that they may be seen as a candidate text excerpt in a summary (Grover *et al.* 2003).

Farzindar and Lapalme (2004) also described a method for summarizing legal documents. As part of their analysis, they performed thematic segmentation on the documents. Finding that more classic method for segmentation (Hearst 1994; Choi 2000) did not provide satisfactory results, they developed a segmentation process based on specific knowledge of their legal documents. For their study groups of adjacent paragraphs were grouped into blocks of text based on the presence of section titles, relative position within the document and linguistic markers.

The classic algorithm for topic segmentation is TextTiling where like sentences and topics are grouped together (Hearst 1997). More general methods for topic segmentation of a document are generally based on the cohesiveness of adjacent sentences. It is possible to build lexical chains that represent the lexical cohesiveness of adjacent sentences in a document based on important content terms, semantically related references, and resolved anaphors (Moens & De Busser 2001). Lexical chains and cohesiveness can then be used to infer the thematic structure of a document.

In contrast to approaches such as these that are based on inferring the relatedness of sentences in section bodies, our approach focuses identifying and categorizing section headers. These general approaches are complementary as it would be relatively straightforward to construct a combined method that considers both headers and bodies.

Overview

Our analysis begins with a pre-processing step that converts documents to sequences of text blocks, roughly at the paragraph level (see below for details). We next construct feature vector representations for all blocks. Labeled training sets and supervised learning methods are used to induce two kinds of classifiers: one for distinguishing section header blocks from non-header blocks, and one for classifying the section type of headers. Figure 1 shows a flowchart of the processing for classifying a block of text in the test set. Note that although the type of non-header blocks is not predicted directly, after classifying of all blocks in a document the predicted section for a non-header block is given by the type of the nearest preceding section header.

Models and Methods

Dataset

Appellee briefs in our dataset are available as HTML files. The HTML is not well formed or standardized and provides little insight into the structure of the briefs. The HTML elements do not contain attributes, block level elements, id's,

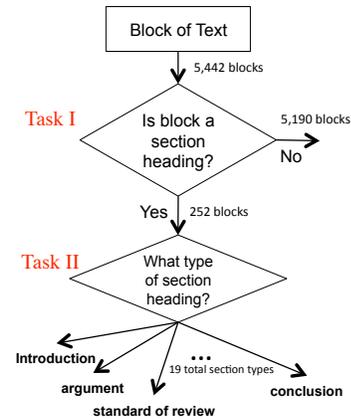


Figure 1: Flowchart of our two stage process for classifying text blocks. The first stage predicts whether or not a block of text is a section header. No further processing is done on blocks classified as non-headers. Blocks classified as headers are passed to the next stage, which predicts the section type. Numbers next to the arrows denote the total number of blocks in our annotated dataset that assort to that point.

classes, *etc.* that may indicate section breaks or section types. Further, document formatting is inconsistent and non-standardized. For example, one author may use italics for section headings, another bold, while yet another uses inline text. Formatting sometimes even varies from section to section within the same document. Thus, we ignore formatting such as italics or bold, and focus our analysis on the word and character sequence.

Preprocessing was performed on the documents to divide the documents into blocks of text. A block of text is essentially a continuous sequence of text from the original document with a line break immediately before and after. We extract blocks by converting each HTML document to an XML document that recognizes all of the line breaks and white spaces from the original HTML. Examples of document elements that correspond to blocks extracted from the XML include paragraphs, section headings, section sub-headings, footnotes, and table-of-contents entries.

The XML files were manually reviewed and annotated by the author (SV). Each block is assigned two class labels:

1. `is_header` A binary value indicating whether or not a block is a section heading.
2. `section_type` A discrete value that for section headers only indicates section type. As we only predict the type of header blocks, the value of “None” is assigned to non-headers. Table 1 shows the section types we identified in our dataset.

Feature Vector Representation

Along with the two class labels, we represent each block of text with a 25 element vector of features values. Ta-

Argument	Notice To Adverse Party	Statement of Parent Companies
Bond	Prayer	And Public Companies
Conclusion	Preliminary Statement	Statement of The Case
Corporate Disclosure Statement	Procedural History	Summary of The Argument
Introduction	Relief Sought	Table of Authorities
Issue Presented For Review	Standard of Review	Table of Contents
Jurisdictional Statement	Statement of Facts	None

Table 1: The 20 section types in our dataset. Each predicted header block is classified as one of the 19 types other than “None.”

(a)

Feature Name	Domain	Description
leadingAsterisk	binary	True if the block begins with an asterisk (*)
leadingNumeral	binary	True if the block begins with an Arabic or Roman numeral (optionally preceded by an asterisk).
endsInPeriod	binary	True if the block ends with a period (.)
endsInNumeral	binary	True if the block ends with an Arabic or Roman numeral.
stringLength	integer	Number of characters in the block.
percentCaps	continuous, in [0,1]	The % of alpha characters that are capitalized.
ellipses	binary	True if the block contains an ellipses (i.e. “...”).
contains(“argument”)	binary	Each of these features is an indicator for a specific string. The feature contains(<i>s</i>) is true if the block contains a word that begins with the string <i>s</i> and false otherwise.
contains(“authori”)	binary	
contains(“case”)	binary	
contains(“conclusion”)	binary	
contains(“contents”)	binary	
contains(“corporate”)	binary	
contains(“disclosure”)	binary	
contains(“fact”)	binary	
contains(“issue”)	binary	
contains(“jurisdiction”)	binary	
contains(“of”)	binary	
contains(“prayer”)	binary	
contains(“present”)	binary	
contains(“review”)	binary	
contains(“standard”)	binary	
contains(“statement”)	binary	
contains(“summary”)	binary	
contains(“table”)	binary	

(b)

leadingAsterisk: FALSE	contains(“of”): TRUE
endsInPeriod: FALSE	contains(“table”): TRUE
stringLength: 21	contains(“contents”): TRUE
percentCaps: 1	(all other string match features): FALSE
leadingNumeral: TRUE	
endsInNumeral: FALSE	is_header: TRUE
ellipses: FALSE	section_type: Table of Contents

Table 2: (a) Features we use to represent blocks of text. (b) An example showing feature and class values for the block of text “II. TABLE OF CONTENTS”

ble 2(a) lists the features we use, Table 2(b) shows the feature and class values for the block of text “III. TABLE OF CONTENTS”.

The features chosen were engineered through visual inspection of section headings, intuition, and trial and error. Other attributes were considered such as the length and percentage of capital letters of the previous and next blocks of text, however, these did not improve model performance. The group of features named *contains(s)* are string matching features, which are true if the block of text contains exactly one word that begins with the string *s*. We construct a string match feature from all words that occur five or more times in the 252 header blocks.

Learning

The task of identifying section headers and the type of section is divided into two steps (Figure 1). The first step classifies a block of text as either a section heading or not a section heading. For this task, supervised machine learning algorithms are used to learn a binary classifier. The second task takes each block of text classified in the first step as a heading and uses a second classifier to predict the specific type of section. Again supervised machine learning is used to learn a classifier, this time with 19 classes. For both tasks, multiple types of classifiers including naive Bayes, logistic regression, decision trees, support vector machines and neural networks were considered.

Evaluation

With the abundance of legal documents available, it is important that they be structured in ways usable by computers (Wynera 2010). We hypothesize the task of structuring our legal documents into relevant sections can be accomplished with a supervised machine learning classifier that first identifies section headers, and then assigns a section type to the header.

To test this hypothesis we have conducted an experiment on 30 appellee briefs from cases heard by the US 1st Circuit in 2004. No effort was made to restrict the cases to a particular area of the law, and indeed a variety of different types of cases is represented in this set. The legal briefs were obtained as HTML files through WestLaw (www.westlaw.com). In the 30 documents, a total of 252 section headers were identified. Note that subsection headers are not included as part of this task as there is very little commonality in authors use of subsections. Additionally, subsections are generally specific to the legal case being addressed, and not the overall document. Of the 252 total section headers, 116 unique strings were identified (not accounting for any difference in formatting or upper / lower case). Manual inspection of the 116 variations revealed that the headers cluster into the 19 different section types listed in Table 2(b). A 20th section type “None” was added to be used as the class label for blocks of text that do not represent section headers.

We conducted a leave-one-case-out cross-validation experiment. That is, in each experiment all blocks from one of our documents was held out of the training set and used

as test data to estimate our models’ ability to generalize to unseen documents.

For the first task, all blocks of text in the training set are used. For the second task, only training set blocks of text labeled as section headings are used for training. This decision was made because we only wish to use the second classifier to label the section type of true section headers. Also, this approach sidesteps the inconsistency that arises when a block of text is identified as a heading in the first stage, but as section type “None” in the second stage. We may revisit this decision in future work as a “None” prediction in stage two could potentially be used to catch false positives from the first stage. With the current dataset, however, it was found that the number of correctly identified headings being labeled as “None” vs. the correction of false positives was not worth the tradeoff. Therefore, we take the approach described above.

We evaluate models on the first task by the percentage of headings or non-headings correctly classified as well as precision and recall rates where:

$$precision = \frac{\#true\ positives}{\#true\ positives + \#false\ positives}$$

and

$$recall = \frac{\#true\ positives}{\#true\ positives + \#false\ negatives}$$

Note blocks of text that are a section header represent our positive class. Precision and recall are both of particular importance for our first task. Examining our dataset, 95.4% of blocks of text are non-headings. The extreme case of classifying all blocks of text as non-headings would then result in very high overall accuracy and 100% recall rate for non-headings, at the expense of poor precision.

We compare our machine learning approach to a regular expression baseline. The regular expression used for this baseline approach may be summarized as the concatenation of the following list of parts:

1. The beginning of the string
2. An optional asterisk
3. An optional Roman Numeral or Natural Number followed by an optional period and space
4. A list of zero or more all capitalized words
5. The end of the string

Blocks that contain a match to the regular expression are predicted to be headers. This regular expression should correctly identify many section headings as many are entirely capitalized, while excluding false positives such as table of contents entries that are generally followed by a page or section number of some form.

Our second task is then evaluated in two ways. The first is the overall percentage of predicted headings that are assigned the correct section heading type. The second metric is an adjusted metric that does not penalize the second task for errors made in the first task. If the input to the second classification task was a non-heading to begin with, this classifier would inherently fail as it is attempting to determine

the section heading type when no such type actually exists. Therefore, we account for this disparity in our results and also present the number of section heading types predicted correctly divided by actual headings correctly classified by the first task.

A baseline approach is only considered for the first task of identifying whether or not a block of text is a section heading. A baseline approach for the secondary task of assigning a label of one of our 20 classes could be developed through a complicated regular expression or a form of sequential logic, but was not considered in this project. Our most frequent section heading type, "Argument", accounts for 12% of cases. Therefore, that level of accuracy could be achieved by simply always predicting "Argument".

Last, a combined metric is presented where we merged the results from both steps of classification to determine the overall percentage of section headings that are correctly identified and assigned the correct type.

Results

Task 1 - Identifying Section Headings

A total of 5,442 blocks of text were identified in our dataset. Table 3 shows a comparison of the baseline method with our supervised machine learning based approach for the task of identifying if a block of text is a section heading or not. With the exception of naive Bayes (which performed worse), all other classifiers performed similarly.

	Baseline	Learning Based
Total Blocks of Text:	5442	5442
Correctly Classified:	5288	5409
Percentage Correct:	97.2%	99.4%

Table 3: Results classifying section headings vs. non section headings

As expected the baseline approach performed very well with 97.2% accuracy. This, represents a small gain over calling all blocks non-headings (95.4%). As we hypothesized, the learning based classifier performed much better with 99.4% accuracy. As seen in the confusion matrix in Table 4, the logistic regression classifier had a similar number of false positives and false negatives. Precision and recall statistics are presented in Table 5. As seen in the table, there is a significant difference in the recall rates of headings (92.1% vs. 61.5%) which is of great importance to the ultimate goal.

Actual/ Predicted	Learning Based		Baseline	
	Heading	Non-Heading	Heading	Non-Heading
Heading	232	20	155	97
Non-Heading	13	5177	57	5133

Table 4: Confusion matrix for Task 1

	Precision	Recall	F-Measure
Learning Based	0.947	0.921	0.934
Baseline	0.731	0.615	0.668

Table 5: Precision and recall of headings for learning based classifier vs. baseline approach

Examining incorrectly classified blocks, the most frequent was "Standard of Review" and accounted for 24% of all errors. Examination of this reveals that the "Standard of Review" is often included as a subsection of the "Argument" section of the brief by many authors, while others choose to make a standalone section. For example, the block of text "1. STANDARD OF REVIEW" was incorrectly classified as a heading in one instance. In this case the author did not use a numbering scheme for the primary section ("Argument" in this case), but numbered the sub-sections on the document confusing our model. Similar errors occurred for the section type "Statement of Facts" and accounted for 12% of all errors. With additional post processing of the classification, it may be possible to account for these types of errors further increasing model performance.

Task 2 - Predicting Section Type

Table 6 summarizes the result of the secondary classifier that assigns section types to any block of text classified as a heading by the first task. The first task identified 245 blocks of text as headings. Of these, only 18 were assigned an incorrect section heading type for an overall accuracy of 92.7%. However, 13 of these 18 were not actually classes to begin with so the secondary classifier could not have assigned a correct class label. Adjusting for this, 232 blocks of text were correctly identified as headings and of these only 5 were given an incorrect label for an adjusted 97.8% accuracy.

	Count	Correctly Labeled	Percent Correct
Total Headings Identified	245	227	92.7%
Actual Headings Identified	232	227	97.8%

Table 6: Results of secondary classifier assigning class labels

Combined Accuracy

Combining accuracy from each of the two tasks results in an overall recall rate of 90.1% as seen in Table 7. Of 252 total labels, 232 were correctly identified as labels. Of those identified, 227 were assigned their correct actual class.

Conclusion

We presented a supervised machine learning approach for structuring legal documents into relevant sections. Our approach is based on two steps. The first step identifies blocks of text that are section headings. In the second step, blocks

Actual Headings	Correctly Identified	Recall Rate	Correct Class	Overall Recall
252	232	92.1%	227	90.1%

Table 7: Combined accuracy for identifying and classifying section headings

of text classified as section headings are then input into a second step to predict section type.

We evaluated our approach with a cross-validation experiment. The first task of identifying section headers using a binary logistic regression classifier was shown to perform with 99.4% accuracy. The secondary task is then used with 92.7% accuracy to determine the type of section one is looking at. The NLP approach provides a 2.2% improvement in accuracy over the baseline regular expression based approach, and more importantly provides a significantly higher recall rate in identifying section headings vs. non section headings.

While it may be possible to create a non-learning based approach (more complex than the baseline approach presented) to perform the given subtask, it has been shown that a machine learning and NLP approach are very well suited for this problem. This paper only researched appellee briefs, but there is ample reason to believe that this approach would provide similar results for appellant briefs, the judges written opinion, and other similar documents.

The significance of our learned models having significantly higher recall rates than baseline models becomes of even greater importance when one considers that approaches would be available to correct or account for false positives (i.e. non-headings classified as headings), however, it would be far more difficult, if even possible, to correct for false negatives (i.e. actual headings classified as non-headings).

While not formally discussed in this paper, it is possible to implement secondary logic to correct for some of the classification errors we encountered. For instance, our most frequent error in the first task was the “Standard of Review. Logic could be implemented as a post processing step that says if a block of text is called a section heading and classified with the section heading type “Standard of Review, but is preceded by the section type “Argument, remove this as a section heading. In our dataset this correction would correct 5 of 7 mistakes made labeling “Standard of Review“ and improve accuracy for the first task to 99.5% and 94.6% for the second task.

In addition, allowing the secondary classifier to identify sections that it assigns the class label “None could correct some false positives incorrectly classified as section headings by the first task. In our dataset, 4 such corrections could have been made further improving accuracy. However, if implementing this change one must consider the implications of giving an actual section break heading the section type “None versus the improvement from corrections.

We considered 20 different potential class labels for each section. For specific tasks it may be found that this number can be reduced to even as few as two (i.e. relevant or non-relevant) sections. This could be done as part of the classifi-

cation or as part of a post process mapping the classifications output by the classifier to a smaller groups of classes for the ultimate task. This may potentially further improve overall performance.

In our approach, the secondary task was treated as individual classifications. It may be possible to treat the secondary classification problem as a Hidden Markov Model or Continuous Random Field. Doing so may improve performance as when an author does include a section in his/her legal briefs, they are generally in a consistent order.

Last, the majority of misclassifications in both tasks appears to be the result of sparse data and infrequently used section headings. While learning curves were not created, it is suspected that additional data could provide the classifier with information about many these sections and improve overall model performance.

With the current model, and the potential for further future improvements, section related information can reliably be identified with supervised machine learning based methods in poorly structured legal documents.

References

- Choi, F. Y. Y. 2000. Advances in Domain-Independent Linear Text Segmentation. In *Proc. Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 26–33.
- Evans, M. C.; McIntosh, W. V.; Lin, J.; and Cates, C. L. 2006. Recounting the Courts? Applying Automated Content Analysis to Enhance Empirical Legal Research. *SSRN eLibrary*.
- Farzindar, A., and Lapalme, G. 2004. Legal text summarization by exploration of the thematic structures and argumentative roles. In *In Text Summarization Branches Out Conference held in conjunction with ACL 2004*, 27–38.
- Grover, C.; Hachey, B.; Hughson, I.; and Korycinski, C. 2003. Automatic summarisation of legal documents. In *Proceedings of the 9th international conference on Artificial intelligence and law, ICAIL '03*, 243–251. New York, NY, USA: ACM.
- Hearst, M. A. 1994. Multi-paragraph segmentation of expository text. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics, ACL '94*, 9–16. Stroudsburg, PA, USA: Association for Computational Linguistics.
- Hearst, M. A. 1997. TextTiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics* 23(1):33–64.
- Jurafsky, D., and Martin, J. H. 2000. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition (Prentice Hall Series in Artificial Intelligence)*. Prentice Hall, 1 edition. neue Auflage kommt im Frhjahr 2008.
- Moens, M.-F., and De Busser, R. 2001. Generic topic segmentation of document texts. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '01*, 418–419. New York, NY, USA: ACM.

Wynera, A. 2010. Weaving the legal semantic web with natural language processing. <http://blog.law.cornell.edu/voxpath/2010/05/17/weaving-the-legal-semantic-web-with-natural-language-processing>.

Automated Speech Act Classification For Online Chat

Cristian Moldovan and Vasile Rus

Department of Computer Science
Institute for Intelligent Systems
The University of Memphis
Memphis, TN 38152, USA
cmlldovan|vrus@memphis.edu

Arthur C. Graesser

Department of Psychology
Institute for Intelligent Systems
Institute for Intelligent Systems
The University of Memphis
art.graesser@gmail.com

Abstract

In this paper, we present our investigation on using supervised machine learning methods to automatically classify online chat posts into speech act categories, which are semantic categories indicating speakers' intentions. Supervised machine learning methods presuppose the existence of annotated training data based on which machine learning algorithms can be used to learn the parameters of some model that was proposed to solve the task at hand. In our case, we used the annotated Linguistic Data Consortium chat corpus to tune our model which is based on the assumption that the first few tokens/words in each chat post are very predictive of the post's speech act category. We present results for predicting the speech act category of chat posts that were obtained using two machine learning algorithms, Naïve Bayes and Decision Trees, in conjunction with several variants of the basic model that include the first 2 to 6 words and their part-of-speech tags as features. The results support the validity of our initial assumption that the first words in an utterance can be used to predict its speech act category with very good accuracy.

Introduction

The task of speech act classification involves classifying a discourse contribution, e.g. an utterance, into a speech act category selected from a set of predefined categories that fulfill particular social discourse functions. Examples of speech act categories are Questions, Statements, or Greetings. For instance, the hearer infers from the following utterance *How did you do that?* that the speaker is asking a Question, which informs the hearer to prepare an answer. Sometimes the speaker just states something as in the following Statement, *The situation is getting worse every day.* or greets someone as in *Hello!* .

In this paper, we propose an automated method to classify online chat posts into speech act categories. The proposed automated method relies on a model that emphasizes the use of the first tokens or words in an utterance to decide their speech act category. For instance, a Question can be distinguished from a Statement based on the first words because usually a Question starts with question words such as *How* which is followed by an auxiliary verb such as *did*. Our model is based on the assumption that humans do infer speakers' intentions early on when they hear the first few words of an utterance. To automate the process, we framed our problem as a supervised machine learning problem in

which we map the previously described model into a set of features and then use machine learning algorithms to learn the parameters of the model from annotated training data. We test in this paper this hypothesis and report how well the first 2 to 6 words of an utterance can diagnose its speech act. The tuned models are then evaluated on separate test data sets. In particular, we work with online chat conversations in which participants in online chatrooms converse with each other via computer networks. Each online chatroom participant can see everyone else's dialogue turns, or chat posts, and respond.

The rest of the paper is organized as in the followings. The next section presents theoretical background on speech acts as well as an overview of various speech act taxonomies. The *Approach* section offers the details of our approach. The following section describes related work addressing the task of speech act classification in similar contexts, e.g. online chats. The *Experiments and Results* section provides a summary of the experiments and results. The *Conclusions* section ends the paper.

Language As Action - Speech Acts

Speech act theory has been developed based on the language as action assumption which states that when people say something they do something. Speech act is a term in linguistics and the philosophy of language referring to the way natural language performs actions in human-to-human language interactions, such as dialogues. Its contemporary use goes back to John L. Austin's theory of *locutionary*, *illocutionary* and *perlocutionary acts* (Austin 1962). According to Searle (Searle 1969), there are three levels of action carried by language in parallel: first, there is the locutionary act which consists of the actual utterance and its exterior meaning; then, there is the illocutionary act, which is the real intended meaning of the utterance, its semantic force; finally, there is the perlocutionary act which is the actual effect of the utterance, such as scaring, persuading, encouraging, etc.

It is interesting to notice that the locutionary act is a feature of any kind of language, not only natural ones, and that it does not depend on the existence of any actor. In contrast, an illocutionary act needs the existence of an environment outside language and an actor that possesses intentions, in other words an entity that uses language for acting in the outside environment. Finally, a perlocutionary

act needs the belief of the first agent in the existence of a second entity and the possibility of a successful communication attempt: the effect of language on the second entity, whether the intended one or not, is taking place in the environment outside language, for which language exists as a communication medium. As opposed to the locutionary act, the illocutionary and perlocutionary acts do not exist in purely descriptive languages (like chemical formulas), nor in languages built mainly for functional purposes (like programming languages). They are an indispensable feature of natural language but they are also present in languages built for communication purposes, like the languages of signs or the conventions of warning signals.

In a few words, the locutionary act is the act of saying something, the illocutionary act is an act performed *in* saying something, and the perlocutionary act is an act performed *by* saying something. For example, the phrase "*Don't go into the water*" might be interpreted at the three act levels in the following way: the locutionary level is the utterance itself, the morphologically and syntactically correct usage of a sequence of words; the illocutionary level is the act of warning about the possible dangers of going into the water; finally, the perlocutionary level is the actual persuasion, if any, performed on the hearers of the message, to not go into the water. In a similar way, the utterance "*By the way, I have a peanut butter sandwich with me; would you like to have a bite?*" can be decomposed into the three act levels. The locutionary act is the actual expressing of the utterance, the illocutionary act is the offer implied by the phrase, while the perlocutionary act, namely the intended effect on the interlocutor, might be impressing with own selflessness, creating a gesture of friendliness, or encouraging an activity, in this case eating.

The notion of speech act is closely linked to the illocutionary level of language. The idea of an illocutionary act can be best captured by emphasizing that "by saying something, we do something" (Austin 1962). Usual illocutionary acts are: greeting ("*Hello, John!*"), describing ("*It's snowing.*"), asking questions ("*Is it snowing?*"), making requests ("*Could you pass the salt?*"), giving an order ("*Drop your weapon!*"), making a warning ("*The floor is wet!*"), or making a promise ("*I'll return it on time.*"). The illocutionary force is not always obvious and could also be composed of different components. As an example, the phrase "*It's cold in this room!*" might be interpreted as having the intention of simply describing the room, or criticizing someone for not keeping the room warm, or requesting someone to close the window, or a combination of the above. A speech act could be described as the sum of the illocutionary forces carried by an utterance. It is worth mentioning that within one utterance, speech acts can be hierarchical, hence the existence of a division between direct and indirect speech acts, the latter being those by which one says more than what is literally said, in other words, the deeper level of intentional meaning. In the phrase "*Would you mind passing me the salt?*", the direct speech act is the request best described by "*Are you willing to do that for me?*" while the indirect speech act is the request "*I need you to give me the salt.*" In a similar way, in the phrase "*Bill and Wendy lost a lot of weight with*

a diet and daily exercise." the direct speech act is the actual statement of what happened "*They did this by doing that.*", while the indirect speech act could be the encouraging "*If you do the same, you could lose a lot of weight too.*"

In our work presented here, we assume there is one speech act per utterance and the set of speech acts used are all at the same level of depthness forming a flat hierarchy. These simplification assumptions are appropriate for a first attempt at automating the speech act classification process and testing our leading tokens model. Furthermore, the LDC data set imposed further constraints on our experiments as the LDC corpus does assume only one speech act per chat posts and also uses a flat set of speech act categories.

Speech Act Taxonomies

The task of speech act classification, the focus of our paper, requires the existence of a predefined set of speech act categories or speech act taxonomy. Researchers have proposed various speech act taxonomies over the years. We present next a summary of the most important ones as judged from historical and relevance perspectives.

The classic categorization of (Austin 1962) postulates five major speech act classes based on five categories of performative verbs: Expositives - verbs asserting or expounding views, classifying usages and references; Exercitives - verbs issuing a decision that something is to be so, as distinct from a judgement that it is so; Verdictives - verbs delivering a finding, official or unofficial, upon evidence or reason as to value or fact; Commissives - verbs committing the speaker to some course of action; and Behabitives - verbs involving the attitudinal reaction of the speaker to someone's conduct or fortunes (D'Andrade and Wish 1985).

The taxonomy proposed by (Searle 1969) consists of six major classes: Representatives - committing the speaker to something's being the case; Directives - attempt by speaker to get the hearer to do something; Commissives - committing the speaker to some course of action; Expressives - expressing the psychological state specified; Declarations - bringing into existence the state described in the proposition and Representative Declarations - giving an authoritative decision about some fact.

The category scheme proposed by (D'Andrade and Wish 1985) treats most utterances as conveying more than a speech act and does not attempt to establish a hierarchical order among multiple speech acts. The primary motivation for the speech act coding system was a desire to investigate correspondences between speech acts and adjectival "dimensions" descriptive of interpersonal behavior. In order for a classifying system to be useful for measuring interpersonal communication, the distinctions reflected by the coding scheme should be relevant to native speakers' perceptions and evaluations of interaction. Their classes are: Assertions (Expositives), Questions (Interrogatives), Requests and Directives (Exercitives), Reactions, Expressive Evaluations (Behabitives), Commitments (Commissives) and Declarations (Verdictives, Operatives).

While there seems to be some consensus on the existence of some speech acts, like greetings, questions, answers, etc., the efficiency of a particular taxonomy for solving a particu-

Table 1: Literature Speech Act Taxonomies

Name	Main Classes
Austin	Expositives, Exercitives, Verdictives, Commissives, Behabitives
Searle	Representatives, Directives, Commissives, Expressives, Declarations, Representative Declarations
D’Andrade and Wish	Expositives, Interrogatives, Exercitives, Reactions, Verdictives, Commissives, Behabitives
VerbMobil	Request, Suggest, Convention, Inform, Feedback

lar problem ultimately rests on the task at hand. For instance (Olney, et al. 2003) uses a taxonomy that divides questions into 16 subcategories and has only 3 classes for the rest of the utterances, which is suitable for an Intelligent Tutoring environment. The 16 subclasses of Questions are: Verification, Disjunctive, Concept Completion, Feature Specification, Quantification, Definition, Example, Comparison, Interpretation, Causal Antecedent, Causal Consequence, Goal Orientation, Instrumental/Procedural, Enablement, Expectational and Judgmental.

In the case of Verbmobil, a longterm interdisciplinary Language Technology research project with the aim to develop a system that can recognize, translate and produce natural utterances, the taxonomy used takes into consideration in which of the five dialogue phases the actual speech acts occur. The main classes of their taxonomy tree are: Request, Suggest, Convention, Inform and Feedback which all ramify into subclasses. For instance, the Convention class is composed of the following subclasses: Thank, Deliberate, Introduce, Politeness Formula and Greeting. (Alexandersson, et al. 1997)

A summary of the theoretical speech act taxonomies and the Verbmobil taxonomy mentioned above are presented in Table 1. In our work, we will use the LDC set of speech act categories, which are described later.

The Approach

As we already mentioned, we adopted a supervised machine learning method to automate the process of speech act classification. Machine learning methods imply the design of a feature set which can then be used together with various machine learning algorithms. We used two such algorithms, Naïve Bayes and Decision Trees, to learn the parameters of the basic model and induce classifiers that can categorize new utterances into speech act categories. Naïve Bayes are statistical classifiers that make the naïve assumption of feature independence. While this assumption means models that are too simplistic at times, it helps with better estimating the parameters of the model which in turn leads to good classifiers in general. Decision Trees are based on the idea of organizing the features in a hierarchical decision tree based on information gain. More informative features are always higher in the tree.

In the automated speech act classification literature, researchers have considered rich feature sets that include the actual words (possibly lemmatized or stemmed) and n-grams (sequences of consecutive words). In almost every such case, researchers apply feature selection methods because considering all the words might lead to overfitting and,

in the case of n-grams, to data sparseness problems because of the exponential increase in the number of features. Besides the computational challenges posed by such feature-rich methods, it is not clear whether there is need for so many features to solve the problem of speech act classification.

We believe that humans infer speakers’ intention after hearing only few of the leading words of an utterance. One argument in favor of this assumption is the evidence that hearers start responding immediately (within milliseconds) or sometimes before speakers finish their utterances ((Jurafsky and Martin 2009) - pp.814). This paper is a first step towards exploring the validity of such a hypothesis within the context of automated speech act classification of online chat posts.

Intuitively, the first few words of a dialog utterance are very informative of that utterances speech act. We could even show that some categories follow certain patterns. For instance, Questions usually begin with a *wh-* word while speech acts such as Answers, Accepting, or Rejecting, contain a semantic equivalent of *yes* or *no* among the first words, and Greetings use a relatively small bag of words and expressions. In the case of other classes, distinguishing the speech act after just the first few words is not trivial, but possible. It should be noted that in typed dialogue, which is a variation of spoken dialogue, some information is lost. For instance, humans use spoken indicators such as the intonation to identify the speech act of a spoken utterance.

We must also recognize that the indicators allowing humans to classify speech acts also include the expectations created by previous speech acts, which are discourse patterns learned naturally. For instance, after a first greeting another greeting, that replies to the first one, is more likely. We ignored such intonational and contextual clues so far in our work in order to explore the potential of classifying speech acts based on words alone. We do plan to incorporate contextual clues in future experiments.

A key decision when developing methods to classify speech acts is choosing the speech act taxonomy. In our work presented in this paper, we adopted the taxonomy proposed by the developers of the LDC chat corpus (Forsyth and Martell 2007). The taxonomy is presented in Table 2. We will use chat posts and their speech acts from the LDC corpus to illustrate the basic idea of our leading tokens approach. We picked examples of posts labeled as Yes/No Questions from the corpus. Selecting the first few words as features seems to be a good approach after seeing the following 12 randomly selected instances of the Yes/No Questions class: *"is 10-19-20sUser68 back yet"*, *"Any women from*

Table 2: Speech act taxonomy and frequencies in the LDC online chat corpus

Classification	Percent	Example
Statement	34.50%	10-19-40sUser11...some people have a lot of blank pages
System	17.02%	JOIN
Greet	13.40%	Hey You
Emotion	11.52%	lmao
Wh-Question	5.33%	where from@11-09-adultsUser12
Yes/No Question	5.22%	wisconsin?
Continuer	3.48%	but i didnt chance it
Accept	2.45%	ok
Reject	2.14%	I can't do newspaper.. I can't throw that far and stairs give me problems
Bye	1.57%	goodnite
Yes Answer	1.17%	yeah
No Answer	0.94%	nope 11-09-adultsUser27
Emphasis	0.48%	Ok I'm gonna put it up ONE MORE TIME 10-19-30sUser37
Other	0.43%	0
Clarify	0.34%	i mean the pepper steak lol

Nashville in here?", *"are you a male?"*, *"hey any guys with cams wanna play?"*, *"any guyz wanna chat"*, *"any single white females?"*, *"r u serious"*, *"can't sleep huh?"*, *"really?"*, *"any girls wanna chat with 24/m"*, *"22/m/wa any ladies want to chat"*, *"can i talk to him!!"*. The word "any" seems to appear often and so are the forms of the auxiliary verb "to be" and modal verbs. It would also seem very useful to use a lemmatizer or stemmer, that map morphological variations of the same word to a canonical form, adapted to the specific environment of online chat. For instance, we would like to automatically decide that "guyz" is the same as "guys" and that the words "r" and "u" may in fact be an abbreviation for "are you". Without this additional knowledge many resemblances would be lost. Also, the post "really?" has no common feature with the others, except for the question mark.

Some other speech act classes are even more suitable to this approach. As an example, we will provide 12 randomly selected instances labeled as Yes Answer in the same corpus: *"yes 10-19-20sUser30"*, *"sure 10-19-20sUser126"*, *"yes 10-19-20sUser115!!!!"*, *"yes"*, *"yep"*, *"yes...."*, *"yes i sleep"*, *"yeah..."*, *"U are Yes"*, *"Yes i would 10-19-30sUser12"*, *"yep....cool...kool..."*, *"yep..."*. The word "yes", usually on the first position in the post, is a powerful common feature, as well as the relatively short length of the posts. A common feature is also the usage of pronouns, especially "I". However, without knowing that "yes", "yep" and "yeah" are variants of the same word, any automated classification method would lose a significant amount of accuracy.

A previous attempt by (Marineau, et al. 2000) explored classification using the first three words of each utterance. We extended the range and used from the first two words up to the first six words of each post. Using more words does provide more information and thus an easier way to differentiate between classes. However, due to the nature of the corpus we used, sometimes considering too many words is a disadvantage, because many posts are only one

or two words long and labeling the missing positions with a *none* tag means encouraging a classifier to find common features between all short utterances, regardless of their different words. We must introduce artificial values such as *none* for missing positions in short posts in order to generate values for all the six features, for instance, in models where we use the first 6 words in chat posts to predict the speech acts. We only used the first 6 leading words as the average length in our LDC corpus was 4.67 words meaning models with 6 words should use up all the words in the posts, on average, to make predictions.

Related Work

Forsyth and Martell (Forsyth and Martell 2007) developed a speech act classifier on the LDC corpus, using the taxonomy of (Wu, Khan, Fisher, Shuler and Pottenger 2005). The corpus consisted of online chat sessions in English between speakers of different ages. Their prediction model relied on a set of 22 features that include: the number of chat posts ago the user last posted something, number of chat posts in the future that contain a yes/no pattern, total number of users currently logged on, the number of posts ago that a post was a JOIN (System message), total number of tokens in post, first token in post contains "hello" or variants, first token in post contains conjunctions such as "and", "but", "or", etc., number of tokens in the post containing one or more "?" and number of tokens in the post in all caps. The values for all the features were normalized. The first 9 features were based on the distance of the post to specific posts around it, while the rest of the features were based on the density of some key words in the post or in the first token of the post belonging to a specific speech act category. The machine learning algorithms they used were Backpropagation Neural Network and Naïve Bayes, with the former performing better. Neither method seemed to make a reasonable classification unless the frequency of the class was higher than 3%.

Obviously, in the classification system of Forsyth and

Table 3: 10-fold cross-validation on LDC online chat corpus

n	Naïve Bayes					Decision Trees				
	Accuracy	Kappa	Precision	Recall	F-Measure	Accuracy	Kappa	Precision	Recall	F-Measure
2	74.14	.676	.719	.741	.714	78.33	.727	.772	.783	.772
3	73.05	.662	.698	.731	.697	78.35	.727	.772	.784	.772
4	72.57	.656	.690	.726	.690	77.27	.711	.755	.773	.746
5	72.17	.651	.671	.722	.683	77.27	.711	.755	.773	.746
6	71.70	.645	.662	.717	.677	77.32	.711	.755	.773	.746

Table 4: 10-fold cross-validation on LDC online chat corpus without "System" posts

n	Naïve Bayes					Decision Trees				
	Accuracy	Kappa	Precision	Recall	F-Measure	Accuracy	Kappa	Precision	Recall	F-Measure
2	66.64	.558	.646	.666	.634	71.80	.622	.702	.718	.702
3	65.28	.543	.627	.653	.615	71.87	.623	.704	.719	.703
4	64.46	.533	.618	.645	.604	71.82	.622	.702	.718	.703
5	64.03	.527	.605	.640	.598	71.77	.621	.701	.718	.702
6	63.51	.520	.585	.635	.591	71.82	.622	.702	.718	.703

Martell, the order of posts in the chat and automatic system messages (like JOIN or PART) played a major role. As far as syntactical information is concerned, they started from the assumption that the first word of a post is very important for determining the speech act of the post, especially in the case of Wh-Question, Yes/No Question, Continuer, Yes Answer and No Answer. Also, the question mark and the exclamation mark were considered indicative.

In order to automatically classify speech acts, (Samuel, Carberry and Vijay-Shanker 1998) applied a Transformation-Based Learning machine learning algorithm on Reithinger and Klessen's training set (143 dialogues, 2701 utterances) and on a disjoint testing set (20 dialogues, 328 utterances) (Reithinger and Klesen 1997). The features investigated were punctuation marks, speaker direction (provided by the corpus), number of words in utterances, speech acts of previous and following utterances, and a feature called dialogue act cues. The latter is finding the n-grams for $n = 1, 2, 3$ that minimize the entropy of the distribution of speech acts in a training corpus. Other processing steps they used included filtering out irrelevant dialogue act cues and clustering semantically-related words. The results showed a comparison between features: manually selected cue phrases, word n-grams, and entropy-minimization cues, all combined with the additional processing steps. The best results were obtained using entropy minimization with filtering and clustering.

Experiments and Results

The LDC online chat corpus is a product of the Naval Post-graduate School (Lin 2007). It contains 10,567 posts from different online chat rooms in English. All the posts had to go through a sanitizing process in order to protect user privacy, so that the corpus could be made available to the

larger research community. All the user screen names were replaced with a mask, for example *killerBlonde51* was replaced by *101930sUser112*.

The original motivation for the development of the corpus was an attempt to automatically determine the age and gender of the poster based on their chat style, using features like average number of words per post, vocabulary breadth, use of emoticons and punctuation. Subsequently, the corpus was manually annotated with part of speech labels for each word and a speech act category per post. An automatic speech act classifier was used for this purpose and then each post was manually verified (Forsyth and Martell 2007). We take advantage in our experiments of the part of speech information available in the LDC corpus by incorporating this information in our basic model. We report results with and without part of speech information, which was included in the basic model in the form of part of speech tags for each word considered in a particular instance of the model.

The part of speech (POS) tagging used the Penn Treebank tagset with some additions specific to the problems related to a chat corpus. Abbreviations such as "*lol*" and emoticons such as "":)" are frequently encountered and since they all convey emotion they were treated as individual tokens and tagged as interjections ("*UH*"). Also, some words that would normally be considered misspelled and were practically standard online were treated as correctly spelled words and tagged according to the closest corresponding word class. For example, the word "*wont*" if treated as a misspelling would normally be tagged as "*MD^RB*", the character ^ referring to a misspelling. The same word would be tagged as "*MD*" and "*RB*" when referring to "*modal*" and "*adverb*", respectively. However, since it was highly frequent in the chat domain, "*wont*" was tagged as "*MD*". In contrast, words that were just plain misspelled and did not

Table 5: 10-fold cross-validation on LDC online chat corpus without "System" posts and without POS tags

n	Naïve Bayes					Decision Trees				
	Accuracy	Kappa	Precision	Recall	F-Measure	Accuracy	Kappa	Precision	Recall	F-Measure
2	69.40	.574	.641	.694	.641	71.79	.622	.701	.718	.703
3	66.88	.546	.632	.669	.613	71.82	.622	.703	.718	.703
4	65.74	.532	.618	.657	.598	71.85	.623	.703	.719	.703
5	64.57	.517	.594	.646	.584	71.80	.623	.702	.718	.703
6	63.89	.507	.587	.639	.576	71.78	.622	.701	.718	.702

appear frequently were tagged with the misspelled version of the tag, for example the word "interesting" was tagged as "JJ" (Forsyth and Martell 2007).

In the LDC corpus, each post was assigned a single speech act category from the 15 categories of the chat taxonomy proposed by (Wu, Khan, Fisher, Shuler and Pottenger 2005). Those categories along with examples and their frequencies in the corpus are represented in Table 2.

In order to implement the machine learning approach, we extracted for each of the 10,567 posts the first n tokens ($n = 2..6$) and their part of speech (POS) tags. Furthermore, we recorded the annotated speech act category as the correct class of the post, which is needed during training. We then use Naïve Bayes and Decision Trees (J48) from WEKA (Witten and Frank 2005) to induce classifiers based on the leading n tokens and their POS tags. We experimented with several variants of the basic model by generating an instance for each $n = 2..6$. The accuracy of the induced classifiers was measured using 10-fold cross-validation. In the cases in which the post had less than n tokens, we replaced the empty feature slots with a dummy token and a dummy POS tag. A summary of results is shown in Table 3.

The System class clearly increases the performance of the classifier due to the large number of instances and their simplicity. Practically, the System posts are "PART", "JOIN", and just a few other variants. In a second round of experiments, we wanted to investigate the validity of our approach on real posts only, i.e. we did not take into account the System posts. As already mentioned, the actual System messages are too specific to a particular chat system and they are not natural language. As before, we extracted for each post the first n tokens and the speech act category. On the remaining 7,935 posts, we applied the same Naïve Bayes and Decision Trees (J48) classifiers with 10-fold cross-validation. The results are presented in Table 4. A significant drop in accuracy can be noticed. Still, the performance of the proposed approach is very good on the natural, non-System posts. We also observe that there is no major difference among the various models when the number of leading words is varied. This may be explained in the case of chat posts by the relative short nature of these posts.

In Table 5, we provide the results obtained by the induced classifiers without System posts and without parts of speech tags for the words. These results reveal the power of our basic model alone (without part of speech information) on natural posts (System posts are not included). An interest-

ing finding is the fact that best results using Naïve Bayes are obtained when using only the first two leading words in a chat post instead of more. When using Decision Trees, results obtained with the first two leading words are as good as when using even 6 words. Thus, we can conclude that our hypothesis that the first few leading words of an utterance are very diagnostic of that utterance's speech act is true, at least for online chat posts, the focus of our experiments.

Conclusions

Our results acknowledge the fact that the first few tokens of a chat post are indicative of the speech act of the post. It is worth mentioning the chat language could be considered more challenging than natural conversation language. Indeed, online chat being an environment that apparently encourages extreme creativity and exhibits a very high tolerance to misspellings and breaking language rules. For instance, "hey, heya, heyheyhey, heys, heyy, heyyy, heyyyy, heyyyyy, heyyyyyy, heyyyyyyy, heyyyyyyyy, heyyyyyyyyy, heyyyyyyyyyy, heyyyyyyyyyyy, heyyyyyyyyyyy" are in the LDC corpus chat-specific variants of the same greeting word. To this point, we did not use a lemmatizer-like tool that could reduce the high number of variants for the same token, especially in the case of speech acts with a high emotional content, such as rejections, accepting, emotion and emphasis, and also in the case of greetings and yes/no questions/answers, which in a literary corpus would usually be represented by a relatively small number of expressions, but which in the chat environment are especially targeted by language creativity.

One future extension we plan to do is utilizing word n-grams for detecting speech acts. N-grams could better capture word order and thus better differentiate between patterns such as "do you" (a bigram), which most likely indicates a question, and "you do", which indicates a Command. Furthermore, we plan on using the dialog act cues proposed by (Samuel, Carberry and Vijay-Shanker 1998) for detecting n-grams that minimize the entropy of the distribution of speech acts in our training corpus and apply them as features for speech act classification. We also plan to test our leading words hypothesis on dialogue data that is naturally longer than chat posts. We hope to understand better from such data whether only two or three leading words are enough as opposed to six.

References

- Alexandersson, J.; Buschbeck-Wolf, B.; Fujinami, T.; Maier, E.; Reithinger, N.; Schmitz, B.; Siegel, M. 1997. *Dialogue Acts in VerbMobil-2*. volume 226, VerbMobil Report, German Research Center for Artificial Intelligence (DFKI), Saarbrücken, 1998
- Austin, J.L. 1962. *How to do Things with Words*. Oxford University Press, 1962.
- D'Andrade, R.G.; and Wish, M. 1985. *Speech Act Theory in Quantitative Research on Interpersonal Behavior*. *Discourse Processes* 8:2:229-258, 1985.
- Forsyth, E.N.; and Martell, C.H. 2007. *Lexical and Discourse Analysis of Online Chat Dialog*. International Conference on Semantic Computing (ICSC 2007), pp. 19-26.
- Jurafsky, Dan.; and Martin, J.H. 2009. *Speech and Language Processing*. Prentice Hall, 2009.
- Lin, J. 2007. *Automatic Author profiling of Online Chat Logs*. M.S.Thesis, Naval Postgraduate School, Monterey.
- Marineau, J.; Wiemer-Hastings, P.; Harter, D.; Olde, B.; Chipman, P.; Karnavat, A.; Pomeroy, V.; and Graesser, A. 2000. *Classification of speech acts in tutorial dialog*. In Proceedings of the workshop on modeling human teaching tactics and strategies at the Intelligent Tutoring Systems conference.
- Olney, A.; Louwerse, M.; Mathews, E.; Marineau, J.; Hite-Mitchell, H.; and Graesser, A. 2003. *Utterance Classification in AutoTutor*. Building Educational Applications using Natural Language Processing: Proceedings of the Human Language Technology, Philadelphia, PA.
- Reithinger, N.; and Klesen, M. 1997. *Dialogue act classification using language models*. In Proceedings of EuroSpeech-97, pages 2235-2238.
- Samuel, K.; Carberry, S.; and Vijay-Shanker, K. 1998. *Dialogue Act Tagging with Transformation-Based Learning*. In Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics, pages 1150-1156, Montreal, 1998.
- Searle, J.R. 1969. *Speech Acts*. Cambridge University Press, GB, 1969.
- Witten, I. H.; and Frank, E. 2005. *Data Mining: Practical machine learning tools and techniques, 2nd Edition*. Morgan Kaufmann, San Francisco.
- Wu, T.; Khan, F.M.; Fisher, T.A.; Shuler, L.A.; and Pottenger, W.M. 2002. *Posting Act Tagging using Transformation-Based Learning*. In Proceedings of the Workshop on Foundations of Data Mining and Discovery, IEEE International Conference on Data Mining (ICDM'02), December 2002.

A Study of Query-Based Dimensionality Reduction

Augustine S. Nsang

Computer Science Department
University of Cincinnati
Cincinnati, OH 45221-0030, USA
nsangas@mail.uc.edu

Anca Ralescu

Computer Science Department
University of Cincinnati
Cincinnati, OH 45221-0030, USA
Anca.Ralescu@uc.edu

Abstract

This paper considers two approaches to query-based dimensionality reduction. Given a data set, D , and a query, Q , the first approach performs a random projection on the dimensions of D that are not in Q to obtain the data set D_R . A new data set (D_{RQ}) is then formed comprising all the dimensions of D that are in the query Q together with the dimensions of D_R . The resulting data set ($Q(D_{RQ})$) is obtained by applying the query Q to D_{RQ} . A similar approach is taken in the second approach with the difference that the random projection method is replaced by Principal Component Analysis. Comparisons are made between these two approaches with respect to the inter-point distance preservation and computational complexity.

Introduction

Given a collection of n data points (vectors) in high dimensional space, it is often helpful to represent the data in a lower dimensional space without the data suffering great distortion (Achlioptas 2004). This operation is known as *dimensionality reduction*.

There are many known methods of dimensionality reduction, including *Random Projection (RP)*, *Singular Value Decomposition (SVD)*, *Principal Component Analysis (PCA)*, *Kernel Principal Component Analysis (KPCA)*, *Discrete Cosine Transform (DCT)*, *Latent Semantic Analysis (LSA)* and many others (Nsang & Ralescu 2009b).

In random projection, the original d -dimensional data is projected to a k -dimensional ($k \ll d$) subspace through the origin, using a random $d \times k$ matrix R whose columns have unit lengths (Bingham & Mannila 2001). If $X_{n \times d}$ is the original set of n d -dimensional observations, then

$$X_{n \times k}^{RP} = X_{n \times d} R_{d \times k}$$

is the projection of the data in a lower k -dimensional subspace. The key idea of random projection arises from the Johnson Lindenstrauss lemma (Johnson & Lindenstrauss 1984) which states that if points in a vector space are projected onto a randomly selected subspace of suitably high dimension, then the distances between the points are approximately preserved.

Given n data points as an $n \times p$ matrix X of real numbers, to find the best q -dimensional approximation for the data ($q \ll p$) using the PCA approach, the SVD of X is first obtained. In other words, PCA finds matrices U , D and V such that

$$X = UDV^T$$

where:

- U is an $n \times n$ orthogonal matrix (i.e. $U^T U = I_n$) whose columns are the left singular vectors of X ;
- V is a $p \times p$ orthogonal matrix (i.e. $V^T V = I_p$) whose columns are the right singular vectors of X ;
- D is an $n \times p$ diagonal matrix with diagonal elements $d_1 \geq d_2 \geq d_3 \geq \dots \geq d_p \geq 0$ which are the singular values of X . Note that the bottom rows of D are zero rows.
- Define V_q to be the matrix whose columns are unit vectors corresponding to the q largest right singular values of X . V_q is a $p \times q$ matrix.

The transformed data matrix is given by $X^{PCA} = X^T V_q$ (Bingham & Mannila 2001).

Dimensionality reduction has several applications in information retrieval, image data processing, nearest neighbor search, similarity search in a time series data set, clustering and signal processing (Nsang & Ralescu 2009b).

Bingham and Mannila (Bingham & Mannila 2001) suggest the use of random projections for query matching in a situation where a set of documents, instead of one particular one, were searched for. This suggests another application of dimensionality reduction, namely to reduce the complexity of the query process. Suppose, for instance, that we want to query a text document data set with say 5000 dimensions. It would be helpful if we can reduce it to 400 dimensions, say, before applying the query, provided that the dimensions represented in the query are not eliminated by the dimensionality reduction process. The complexity of the query processing is reduced while the speed is significantly increased. Besides, given the distance preserving properties of the random projection and other methods, we retain as much as possible the inter-point distances between data items in the original and reduced data sets. This means that algorithms based on

such distances (e.g. clustering, classification) will perform similarly on the original and reduced data sets.

In the first section of this paper, we discuss the original approach to query based dimensionality reduction (suggested by Bingham and Mannila) and explain why this approach fails. In the second section, we present the first alternative approach using random projections, and determine the values of g_1 and g_2 such that, if u and v are two rows of a data set D , and $f(u)$ and $f(v)$ are the corresponding rows of the data set D_{RQ} derived from D , then:

$$g_1 \|u - v\|^2 \leq \|f(u) - f(v)\|^2 \leq g_2 \|u - v\|^2$$

We also determine, in this section, the speed up in query processing due to this approach. In the third section, we outline the second alternative approach (based on PCA), and in the fourth and last section, we compare the two alternative approaches with respect to inter-point distance preservation and computational complexity.

Original Approach (Bingham & Mannila 2001)

Suppose D is a text document data set, and Q is a query. Following the idea suggested by Bingham and Mannila (Bingham & Mannila 2001), instead of applying the query Q to the data set D to obtain the query result $Q(D)$, we first apply random projection to D to obtain the reduced data set, D_R . Querying D_R with the query Q produces the set of documents $Q(D_R)$. Ideally, for this process to become successful, $Q(D_R)$ should be equal to $R(Q(D))$, where R denotes the operation of Random Projection. Fig 1 captures this relationship.

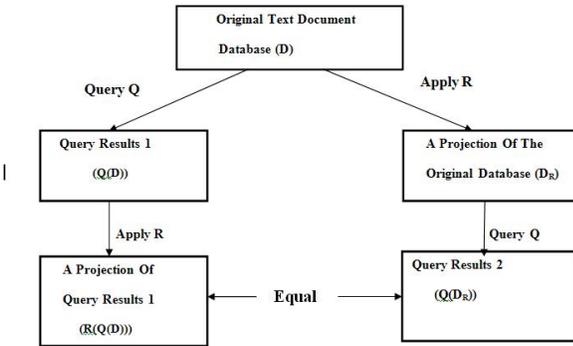


Figure 1: Original Dimensionality Reduction for Query Processing

Unfortunately, this approach fails. This is because when the random projection is applied to the original data set D , all or some of the attributes occurring in a query Q may have been eliminated, and therefore they do not occur in the reduced data set D_R . This can be illustrated by an explicit example (Nsang & Ralescu 2009a). Thus, an alternative approach to reducing the complexity of the query process while not eliminating possibly relevant records from the data set, is needed.

Query-based Dimensionality Reduction Using Random Projections

In this approach, we first perform a random projection on the dimensions of D that are NOT in Q to obtain the data set D_R . A new data set (D_{RQ}) is then formed comprising all the dimensions of D that are in the query Q together with the dimensions got by performing a random projection on the dimensions NOT in Q . The resulting data set ($Q(D_{RQ})$) is obtained by applying the query Q to D_{RQ} (see Fig 2). Thus $Q(D_{RQ})$ is the dimensionality reduced form of $Q(D)$, which is the result of applying the query Q to the text document data set D .

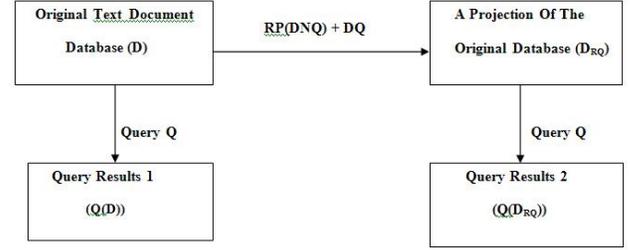


Figure 2: First Alternative Approach

More formally, we have the following. Given A_D , the set of attributes of the data set D , and A_Q , the set of attributes corresponding to query Q , $A_D \setminus A_Q$, the set of attributes of the data set D which are NOT in query Q , then

$$A_{D'} = A_Q \cup RP(A_D \setminus A_Q)$$

where D' corresponds to the data set D_{RQ} in Fig 2. For example, consider the text document data set, D , in Table 1 and the query

$$Q = \text{List all documents which have more than two occurrences of each of the terms } \mathbf{Augustine}(A_4) \text{ and } \mathbf{Ill}(A_7)$$

In this case

$$A_D = \{\text{My, Name, Is, Augustine, He, Was, Ill}\}$$

and

$$A_Q = \{\text{Augustine, Ill}\}.$$

Thus

$$A_D \setminus A_Q = \{\text{My, Name, Is, He, Was}\}.$$

Now if

$$RP(A_D \setminus A_Q) = \{A_1, A_2, A_3\},$$

then

$$A_{D'} = A_Q \cup RP(A_D \setminus A_Q) = \{\text{Augustine, Ill, } A_1, A_2, A_3\}.$$

A natural question which arises is why we need to keep all the attributes NOT in the query (in some reduced form) instead of just discarding them. The answer is that in this way, given the distance preserving properties of the random projection, we retain as much as possible the inter-point distances between data items in the original and reduced data

Table 1: Original data set

Id	My (A1)	Name (A2)	Is (A3)	Augustine (A4)	He (A5)	Was (A6)	Ill (A7)
1	10	0	100	5	1	3	5
2	25	1	150	9	7	9	11
3	35	0	200	15	13	15	17
4	0	25	0	10	19	21	23
5	10	0	95	70	25	40	85
6	10	16	25	14	13	15	17

Table 2: Euclidean distances between the records in original data set

	1	2	3	4	5	6
1	0	53.4	105.6	108.3	112.2	80
2	53.4	0	52.4	155.4	117.2	127.3
3	105.6	52.4	0	204.9	141.7	177.5
4	108.3	155.4	204.9	0	132.6	30.5
5	112.2	117.2	141.7	132.6	0	117
6	80	127.3	177.5	30.5	117	0

sets. This means that algorithms based on such distances (e.g. clustering, classification) will perform similarly on the original and reduced data sets.

Consider again the data set represented by Table 1. Suppose that the query is given by: *List all documents which have more than five occurrences of each of the terms My, Is and at least one occurrence of term Name*. In this case, discarding all the attributes not in the query would make the **first** and the **fifth** records much more similar in the reduced set than they were in the original set and the Euclidean distance between the first and fifth records would be much less in the reduced set than in the original set (see Tables 2 and 3).

Table 3: Euclidean distances: data set reduced only to the query attributes

	1	2	3	4	5	6
1	0	52.2	103.1	103.6	5	76.7
2	52.2	0	51	154	57	126.8
3	103.1	51	0	204.6	108	177.5
4	103.6	154	204.6	0	98.7	28.4
5	5	57.0	107.9	98.7	0	71.8
6	76.7	126.8	177.5	28.4	71.8	0

At the same time, it would make the **third** and the **sixth** records much more dissimilar in the reduced set than they were in the original set even though the Euclidean distance between the **third** and **sixth** records would be about the same in the reduced set as in the original set.

On the other hand, reducing the original data set by reducing the non-query attributes using *RP* (and appending the query attributes to the result) significantly preserves the Euclidean distances between the **first** and **fifth** records, and between the **third** and **sixth** records (see Tables 2 and 4). Table 4 was generated from the matrix obtained by multiplying the matrix representing the non-query attributes of the original data set by a 4×3 random matrix R defined by:

$$r_{ij} = \begin{cases} +1 & \text{with probability } \frac{7}{24}; \\ 0 & \text{with probability } \frac{9}{12}; \\ -1 & \text{with probability } \frac{7}{24}. \end{cases} \quad (1)$$

Table 4: Euclidean Distances in the Data Set with Non-query Attributes Reduced by RP

	1	2	3	4	5	6
1	0	52.4	103.6	104.5	125	77.3
2	52.4	0	51.4	154.3	134.2	126.9
3	103.6	51.4	0	205	158.2	177.5
4	104.5	154.3	204.9	0	158.3	30.4
5	125	134.2	158.2	158.3	0	137.2
6	77.3	126.9	177.5	30.4	137.2	0

We next determine g_1 and g_2 such that for all $u, v \in D$, $g_1(\|u - v\|^2) \leq \|f(u) - f(v)\|^2 \leq g_2(\|u - v\|^2)$ where $f(u)$ and $f(v)$ are the corresponding values to u and v in D_{RQ} , where D_{RQ} is obtained from D using the first alternative approach. Recall that for the regular random projection method, $g_1(x) = (1 - \varepsilon)x$ and $g_2(x) = (1 + \varepsilon)x$ for some value of ε .

Experiment

An experiment was carried out (in MATLAB) on the data set given by the matrix

$$D = \begin{bmatrix} 5 & 6 & 7 & 9 & 0 & 9 & 8 & 7 & 6 & 11 & 6 & 74 \\ 3 & 2 & 10 & 6 & 3 & 5 & 9 & 4 & 10 & 5 & 0 & 57 \\ 10 & 0 & 10 & 3 & 4 & 6 & 2 & 8 & 12 & 0 & 9 & 64 \\ 6 & 3 & 10 & 3 & 4 & 0 & 2 & 7 & 0 & 1 & 5 & 0 \\ 6 & 0 & 8 & 6 & 1 & 5 & 5 & 7 & 11 & 0 & 2 & 51 \\ 1 & 3 & 4 & 8 & 8 & 8 & 5 & 6 & 7 & 9 & 0 & 9 \\ 2 & 2 & 9 & 5 & 0 & 5 & 10 & 6 & 3 & 5 & 9 & 4 \\ 2 & 0 & 5 & 2 & 7 & 7 & 4 & 6 & 2 & 8 & 12 & 0 \\ 6 & 7 & 4 & 7 & 4 & 4 & 0 & 10 & 8 & 4 & 9 & 5 \\ 5 & 2 & 10 & 3 & 1 & 8 & 10 & 6 & 8 & 8 & 0 & 9 \end{bmatrix}$$

The columns in this data set represent values of 12 attributes, $A_1 - A_{12}$. The query Q for this experiment is: *Find all data points having an even number of occurrences of the attribute value for A_5* . $Q(D)$ is computed and the result obtained if we were to apply Q to D without first performing random projection is obtained.

To compute D_{RQ} , we generate D_{NQ} , the data set consisting only of the dimensions of D NOT in the query, and D_Q , the data set consisting only of the dimensions in the query. $Q(D)$, D_{NQ} and D_Q are given by:

$$Q(D) = \begin{bmatrix} 5 & 6 & 7 & 9 & 0 & 9 & 8 & 7 & 6 & 11 & 6 & 74 \\ 10 & 0 & 10 & 3 & 4 & 6 & 2 & 8 & 12 & 0 & 9 & 64 \\ 6 & 3 & 10 & 3 & 4 & 0 & 2 & 7 & 0 & 1 & 5 & 0 \\ 1 & 3 & 4 & 8 & 8 & 8 & 5 & 6 & 7 & 9 & 0 & 9 \\ 2 & 2 & 9 & 5 & 0 & 5 & 10 & 6 & 3 & 5 & 9 & 4 \\ 6 & 7 & 4 & 7 & 4 & 4 & 0 & 10 & 8 & 4 & 9 & 5 \end{bmatrix}$$

$$D_{NQ} = \begin{bmatrix} 5 & 6 & 7 & 9 & 9 & 8 & 7 & 6 & 11 & 6 & 74 \\ 3 & 2 & 10 & 6 & 5 & 9 & 4 & 10 & 5 & 0 & 57 \\ 10 & 0 & 10 & 3 & 6 & 2 & 8 & 12 & 0 & 9 & 64 \\ 6 & 3 & 10 & 3 & 0 & 2 & 7 & 0 & 1 & 5 & 0 \\ 6 & 0 & 8 & 6 & 5 & 5 & 7 & 11 & 0 & 2 & 51 \\ 1 & 3 & 4 & 8 & 8 & 5 & 6 & 7 & 9 & 0 & 9 \\ 2 & 2 & 9 & 5 & 5 & 10 & 6 & 3 & 5 & 9 & 4 \\ 2 & 0 & 5 & 2 & 7 & 4 & 6 & 2 & 8 & 12 & 0 \\ 6 & 7 & 4 & 7 & 4 & 0 & 10 & 8 & 4 & 9 & 5 \\ 5 & 2 & 10 & 3 & 8 & 10 & 6 & 8 & 8 & 0 & 9 \end{bmatrix}$$

$$D_Q = \begin{bmatrix} 0 \\ 3 \\ 4 \\ 4 \\ 1 \\ 8 \\ 0 \\ 7 \\ 4 \\ 1 \end{bmatrix}$$

Next we generate the random projection matrix, R , multiply it by D_{NQ} and append D_Q to the result to obtain D_{RQ} . Define the random projection matrix, $R = (r_{ij})$, as:

$$r_{ij} = \sqrt{3} \times \begin{cases} +1 & \text{with probability } \frac{1}{6}; \\ 0 & \text{with probability } \frac{2}{3}; \\ -1 & \text{with probability } \frac{1}{6}. \end{cases}$$

If we wanted D_{RQ} to have a dimensionality of 7, say, R will have to be a 11×6 matrix.

Thus $D_R = D_{NQ} * R$ and $D_{RQ} = DQ \cup D_R$ where \cup denotes the operation of adding to D_R the columns of DQ . The result, $Q(D_{RQ})$, of applying the query Q to D_{RQ} is now the collection of data records in D_{RQ} that satisfy the query Q .

Figures 3, 4 and 5 show the results obtained from a run of a MATLAB implementation of this procedure to obtain R , D_{RQ} and $Q(D_{RQ})$.

We now investigate the relation between the pairwise distances in the original and reduced data sets. For any two records $u, v \in D$, let $f(u)$ and $f(v)$ denote the corresponding records in D_{RQ} . The pairwise distances obtained from our sample run are shown in Table 5 below.

Because the projection matrix R is random, each run generates a different value of $\|f(u) - f(v)\|^2$ for any pair of rows $u, v \in D$ (and of course the same value of $\|u - v\|^2$). As we know, the actual value of $\|f(u) - f(v)\|^2$ corresponding to a specific value of $\|u - v\|^2$ lies somewhere between $\max(\|f(u) - f(v)\|^2)$ and $\min(\|f(u) - f(v)\|^2)$. Thus, we shall use the midpoint between these two extremes as an estimate of the value of $\|f(u) - f(v)\|^2$ which corresponds to the value of $\|u - v\|^2$.

Sixteen runs of the program were made, and for each value of $\|u - v\|^2$, the maximum and minimum values of

$$R = \begin{bmatrix} 1.7321 & 0 & 0 & -1.7321 & 0 & 0 \\ 0 & -1.7321 & 0 & 0 & 0 & 0 \\ 0 & 1.7321 & 0 & 0 & -1.7321 & 0 \\ -1.7321 & 0 & -1.7321 & 1.7321 & 0 & -1.7321 \\ 1.7321 & 0 & 0 & 0 & 0 & -1.7321 \\ 0 & 0 & 0 & 0 & 1.7321 & 0 \\ -1.7321 & 1.7321 & 1.7321 & 0 & 0 & -1.7321 \\ -1.7321 & 0 & 0 & -1.7321 & 0 & 1.7321 \\ -1.7321 & 0 & 1.7321 & -1.7321 & -1.7321 & 0 \\ -1.7321 & 0 & 0 & 0 & 0 & 1.7321 \\ 1.7321 & 0 & -1.7321 & 1.7321 & 0 & -1.7321 \end{bmatrix}$$

Figure 3: The R Matrix

$$D_{RQ} = \begin{bmatrix} 84.8705 & 13.8564 & -112.5833 & 105.6551 & -17.3205 & -150.6884 & 0 \\ 69.2820 & 20.7846 & -93.5307 & 77.9423 & -10.3923 & -107.3872 & 3.0000 \\ 83.1384 & 31.1769 & -102.1910 & 77.9423 & -13.8564 & -103.9230 & 4.0000 \\ -17.3205 & 24.2487 & 8.6603 & -6.9282 & -15.5885 & -8.6603 & 4.0000 \\ 62.3538 & 25.9808 & -86.6025 & 69.2820 & -5.1962 & -96.9948 & 1.0000 \\ -20.7846 & 12.1244 & -3.4641 & 0 & -13.8564 & -41.5692 & 8.0000 \\ -29.4449 & 22.5167 & 3.4641 & -1.7321 & -6.9282 & -13.8564 & 0 \\ -36.3731 & 19.0526 & 20.7846 & -17.3205 & -15.5885 & -1.7321 & 7.0000 \\ -39.8372 & 12.1244 & 3.4641 & -10.3923 & -13.8564 & -15.5885 & 4.0000 \\ -5.1962 & 24.2487 & 3.4641 & -15.5885 & -13.8564 & -31.1769 & 1.0000 \end{bmatrix}$$

Figure 4: The D_{RQ} Matrix

$\|f(u) - f(v)\|^2$ were obtained. These were further reduced to midpoints between these two extremes (as explained above). More precisely,

$$M_d = \max\{\|f(u) - f(v)\|_i^2 / \|u - v\|^2 = d, i = 1..16\}$$

$$m_d = \min\{\|f(u) - f(v)\|_i^2 / \|u - v\|^2 = d, i = 1..16\}$$

$$mid_d = \frac{M_d + m_d}{2}$$

where $\|f(u) - f(v)\|_i^2$ is the distance between $f(u)$ and $f(v)$ in the i^{th} run. The results obtained are summarized in Table 6, and Figure 6 which shows the values of M_d , m_d and mid_d for each value of $d = \|u - v\|^2$, $u, v \in D$ (in this table) and their linear regression lines.

Consider the value X on the $\|u - v\|^2$ axis (labeled on Figure 6). The corresponding values of $\|f(u) - f(v)\|_{min}^2$, $\|f(u) - f(v)\|_{estimate}^2$ and $\|f(u) - f(v)\|_{max}^2$ are $Y1$, Y and $Y2$ respectively. Clearly

$$Y1 \leq Y \leq Y2.$$

$$Q(D_{RQ}) = \begin{bmatrix} 84.8705 & 13.8564 & -112.5833 & 105.6551 & -17.3205 & -150.6884 & 0 \\ 83.1384 & 31.1769 & -102.1910 & 77.9423 & -13.8564 & -103.9230 & 4.0000 \\ -17.3205 & 24.2487 & 8.6603 & -6.9282 & -15.5885 & -8.6603 & 4.0000 \\ -20.7846 & 12.1244 & -3.4641 & 0 & -13.8564 & -41.5692 & 8.0000 \\ -29.4449 & 22.5167 & 3.4641 & -1.7321 & -6.9282 & -13.8564 & 0 \\ -39.8372 & 12.1244 & 3.4641 & -10.3923 & -13.8564 & -15.5885 & 4.0000 \end{bmatrix}$$

Figure 5: The $Q(D_{RQ})$ Matrix

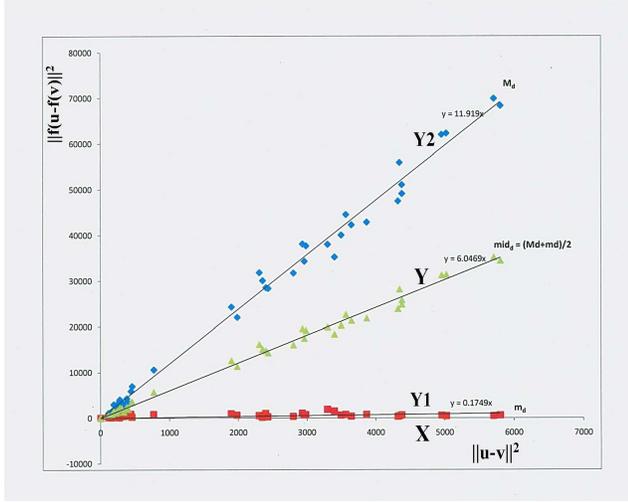


Figure 6: XY chart showing maximum, minimum and estimated values of $\|f(u) - f(v)\|^2$ for each value of $\|u - v\|^2$ (RP Approach)

But $Y1 = m_1X$ and $Y2 = m_2X$ where m_1 and m_2 are the slopes of the regression lines corresponding to $\|f(u) - f(v)\|_{min}^2$ and $\|f(u) - f(v)\|_{max}^2$ respectively. Thus,

$$m_1X \leq Y \leq m_2X$$

Generalizing, for any value of $\|u - v\|^2$, we obtain

$$m_1\|u - v\|^2 \leq \|f(u) - f(v)\|^2 \leq m_2\|u - v\|^2$$

or, letting $g_1(x) = m_1x$ and $g_2(x) = m_2x$, we obtain

$$g_1(\|u - v\|^2) \leq \|f(u) - f(v)\|^2 \leq g_2(\|u - v\|^2) \quad (2)$$

Since both g_1 and g_2 are nondecreasing functions we can prove the following result.

Proposition 1 Suppose u and v are data points in D , and $f(u)$ and $f(v)$ are their mappings in D_{RQ} obtained from D

u	v	$\ u - v\ ^2$	$\ f(u) - f(v)\ ^2$
1	2	450	3354
1	3	434	3394
1	4	5801	58117
1	5	764	5683
1	6	4376	46219
1	7	5020	56973
1	8	5705	69877
1	9	4952	60769
1	10	4342	50683
2	3	288	400
2	4	3493	34933
2	5	112	337
2	6	2428	26743
2	7	2956	34275
2	8	3561	44515
2	9	2980	37633
2	10	2348	29539
3	4	4319	38709
3	5	268	909
3	6	3396	30889
3	7	3864	38437
3	8	4377	49083
3	9	3642	42255
3	10	3296	33063
4	5	2797	29154
4	6	395	1456
4	7	185	322
4	8	216	702
4	9	211	744
..
..
8	9	227	612
8	10	373	2208
9	10	332	1626

Table 5: The values of $\|u - v\|^2$ and $\|f(u) - f(v)\|^2$ for all u, v in D (RP Approach)

using the query-based dimensionality reduction procedure.

If v is in the neighborhood of u of radius r , then $f(v)$ is in the neighborhood of $f(u)$ of radius $g_2(r)$. Conversely, if $f(v)$ belongs to the neighborhood of radius $g_1(r)$ of $f(u)$,

then v belongs to the neighborhood of radius r of u .

Proof: The proof follows trivially. Define the neighborhood of u of radius r as:

$$\eta(u, r) = \{v \in D \mid \|u - v\|^2 \leq r\} \quad (3)$$

From equation (3) it follows that if $v \in \eta(u, r)$, then $g_2(\|u - v\|^2) \leq g_2(r)$, and therefore $\|f(u) - f(v)\|^2 \leq g_2(r)$, that is, $f(v) \in \eta(f(u), g_2(r))$. Conversely, if $f(v) \in \eta(f(u), g_1(r))$ then $g_1(\|u - v\|^2) \leq g_1(r)$ and therefore $\|u - v\|^2 \leq r$, that is $v \in \eta(u, r)$.

Determining the Speed-up of the Query-based

Dimensionality Reduction

We investigate now the computational aspects of the query-based dimensionality reduction.

Complexity of the Original Approach Suppose that the query Q with q attributes is applied to the data set $D_{n \times p}$ resulting in a query result $Q(D)$ with m rows and p attributes.

Table 6: Maximum, minimum and estimated values of $\|f(u) - f(v)\|^2$ for each value of $\|u - v\|^2$ (RP Approach)

$d = \ u - v\ ^2$	M_d	m_d	$mid_d = \frac{M_d + m_d}{2}$
112	1141	151	646
153	1420	211	815.5
154	1303	121	712
164	1543	142	842.5
185	2974	301	1637.5
211	2244	201	1222.5
216	2073	321	1197
227	2406	615	1510.5
230	2917	430	1673.5
238	2377	361	1369
268	3786	123	1954.5
272	4069	694	2381.5
288	3682	406	2044
301	3298	1177	2237.5
332	2322	441	1381.5
359	3333	546	1939.5
373	4305	699	2502
395	3361	589	1975
434	5908	895	3401.5
450	6996	222	3609
764	10609	838	5723.5
1894	24288	939	12613.5
1978	22012	625	11318.5
2300	31806	528	16167
2348	30043	142	15092.5
2396	28510	1003	14756.5
2428	28336	247	14291.5
2797	31710	339	16024.5
..
..
5020	62256	441	31348.5
5705	69877	448	35162.5
5801	68296	583	34439.5

To compute the query result $Q(D)$ from the data set, D , we must compare the value of each attribute in the query with the value of the corresponding attribute in D for **each row** of D , a total of nq operations. After this, to get the query result $Q(D)$ from D we have to generate an $m \times p$ matrix, which is of complexity $O(mp)$. Thus the original query process has complexity $O(nq + mp)$.

Complexity of the Query-based Reduction We recall that D_{RQ} is computed from D by performing a random projection of the dimensions of D that are NOT in Q , and then simply copying all the columns of D corresponding to attributes in Q into the result. Again, if there are q attributes in Q , then there are $(p - q)$ attributes NOT in Q . Also, if D_{RQ} has k attributes, then D_R has $k - \text{no of dimensions in } Q = k - q$ attributes. Thus the random projection reduces an $n \times (p - q)$ matrix into an $n \times (k - q)$ matrix.

Thus, according to the result in (Fradkin & Madigan 2003), the complexity of the random projection step is given by $O((p - q)(k - q)) + O(n(p - q)(k - q))$.

Generating the rest of the matrix D_{RQ} takes $O(nq)$, according to the result in (Fradkin & Madigan 2003) again (since we are generating data having n rows and Q columns). Thus the complexity of the process of generating D_{RQ} from D is $O((n + 1)(p - q)(k - q)) + O(nq)$.

Now, after having already generated D_{RQ} , using the result in the last section, to compute the query result $Q(D_{RQ})$ from the data set D_{RQ} takes $O(nq + mk)$. Thus, the speed up is approximately

$$\frac{C_1}{C_2} = \frac{nq + mp}{nq + mk}$$

where C_1 is the complexity of the original query process and C_2 is the complexity of the process of generating $Q(D_{RQ})$ from D_{RQ} .

Query-based Dimensionality Reduction Using PCA

Another possible approach to query-based dimensionality reduction would be to use PCA in the last section instead of RP. In this case, therefore, we first apply the PCA method on the dimensions of D that are NOT in the query Q to obtain D_P . A new data set (D_{PQ}) is then formed comprising all the dimensions of D_P together with the dimensions of D that are in Q . Applying the query Q to D_{PQ} yields the result $Q(D_{PQ})$.

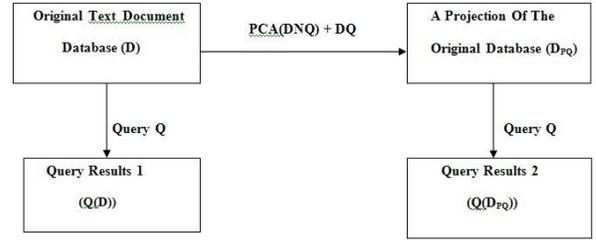


Figure 7: Query-based Dimensionality Reduction Using PCA

Implementation

To compute the projected data set, D_{PQ} , we need to first generate DNQ , the data set consisting only of the dimensions of D **not** in the query Q . We also need to compute DQ , the data set consisting only of the dimensions of D in the query Q . When applied to the data set D and query Q in the last section, the PCA reduction approach results in $DNQ = USV^T$ where U, S and V are shown in Figs 8, 9 and 10.

To compute D_{PQ} (with $k = 7$ columns) we multiply DNQ by the first $k - q$ columns of V (where q is the number of attributes in the query, 1 in this case), and append DQ to the result. $Q(D_{PQ})$ is obtained by applying the query Q to D_{PQ} . The matrices D_{PQ} and $Q(D_{PQ})$ obtained are shown in Figs 11 and 12 respectively.

Comparison of the RP and PCA Query-based Dimensionality Reductions

We now compare the performances of the first and second alternative approaches. To start with, we generate the values

$$U = \begin{bmatrix} -0.580 & -0.084 & -0.261 & 0.591 & -0.041 & 0.327 & -0.214 & -0.065 & 0.282 & -0.021 \\ -0.451 & -0.097 & -0.288 & -0.249 & -0.105 & 0.109 & 0.258 & -0.200 & -0.703 & -0.142 \\ -0.506 & -0.109 & 0.543 & -0.121 & -0.060 & -0.354 & -0.216 & -0.043 & -0.052 & 0.494 \\ -0.031 & 0.320 & 0.329 & -0.224 & -0.192 & 0.677 & -0.296 & 0.364 & -0.152 & -0.029 \\ -0.406 & -0.079 & 0.118 & -0.339 & 0.142 & -0.116 & 0.312 & 0.394 & 0.407 & -0.498 \\ -0.107 & 0.332 & -0.405 & 0.009 & 0.468 & -0.126 & 0.011 & 0.537 & -0.137 & 0.415 \\ -0.075 & 0.455 & -0.033 & -0.005 & -0.448 & 0.077 & 0.611 & -0.122 & 0.269 & 0.344 \\ -0.038 & 0.443 & 0.115 & -0.411 & -0.325 & -0.468 & -0.115 & 0.214 & -0.275 & -0.402 \\ -0.082 & 0.423 & 0.367 & 0.155 & 0.631 & 0.132 & 0.174 & -0.426 & -0.069 & -0.151 \\ -0.117 & 0.412 & -0.346 & -0.465 & -0.053 & -0.164 & -0.489 & -0.369 & 0.254 & -0.104 \end{bmatrix}$$

Figure 8: The U Matrix

$$S = \begin{bmatrix} 132.97 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 35.98 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 17.24 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 13.43 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 11.41 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 8.58 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 5.50 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3.38 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2.35 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.72 \end{bmatrix}$$

Figure 9: The S Matrix

of $u, v, \|u - v\|^2$ and $\|f(u) - f(v)\|^2$ for each pair of tuples $u, v \in D_Q$, and corresponding pair of tuples $f(u), f(v) \in D_{PQ}$. For the same example data set, the results obtained using the PCA approach are shown in Table 7.

A graph of $\|u - v\|^2$ (on the x-axis) against $\|f(u) - f(v)\|^2$ (on the y-axis) is a straight line through the origin which makes an angle of 45° with the horizontal, as shown in Figure 13 below. Thus, it is clear that the PCA approach preserves the inter-point distances much much better than our first alternative approach (with RP).

Given a p -dimensional data set with n rows, and assuming we want to have q rows in the reduced data set, the computational complexity of PCA is $(O(p^2n) + O(p^3))$ (Fradkin & Madigan 2003; Bingham & Mannila 2001), while that of RP is $O(pq) + O(npq)$ (as mentioned above). Thus the PCA approach is much more expensive computationally than the RP method.

Conclusion

In this paper, we have examined different approaches to query-based dimensionality reduction. As we observed, the original approach (suggested by Bingham and Mannila (Bingham & Mannila 2001)) which reduces the dimensionality of the entire text document data set by random pro-

$$V = \begin{bmatrix} -0.10 & 0.18 & 0.36 & -0.22 & 0.09 & 0.10 & -0.50 & -0.14 & 0.43 & 0.13 & -0.54 \\ -0.04 & 0.17 & -0.03 & 0.19 & 0.33 & 0.53 & -0.03 & -0.61 & -0.01 & 0.21 & 0.35 \\ -0.15 & 0.37 & 0.09 & -0.48 & -0.40 & 0.28 & -0.14 & 0.19 & -0.46 & 0.26 & 0.17 \\ -0.11 & 0.25 & -0.14 & 0.10 & 0.37 & 0.32 & 0.48 & 0.46 & 0.05 & 0.29 & -0.36 \\ -0.12 & 0.30 & -0.22 & 0.11 & 0.07 & -0.45 & -0.23 & 0.19 & 0.35 & 0.49 & 0.42 \\ -0.11 & 0.30 & -0.45 & -0.22 & -0.44 & 0.03 & 0.36 & -0.32 & 0.43 & -0.19 & -0.11 \\ -0.12 & 0.39 & 0.24 & -0.04 & 0.23 & 0.12 & -0.04 & 0.30 & 0.22 & -0.67 & 0.36 \\ -0.16 & 0.21 & 0.05 & -0.43 & 0.48 & -0.50 & 0.24 & -0.33 & -0.30 & -0.05 & -0.09 \\ -0.09 & 0.35 & -0.47 & 0.39 & 0.03 & -0.05 & -0.43 & -0.02 & -0.38 & -0.25 & -0.31 \\ -0.08 & 0.37 & 0.56 & 0.52 & -0.33 & -0.21 & 0.29 & -0.16 & -0.10 & 0.08 & -0.08 \\ -0.94 & -0.34 & -0.00 & 0.09 & -0.05 & 0.03 & -0.01 & 0.01 & -0.003 & -0.03 & 0.02 \end{bmatrix}$$

Figure 10: The V Matrix

$$D_{PQ} = \begin{bmatrix} -77.1476 & -3.0072 & -4.5016 & 7.9347 & -0.4665 & 2.8078 & 0 \\ -59.9135 & -3.4798 & -4.9566 & -3.3439 & -1.2025 & 0.9329 & 3.0000 \\ -67.2098 & -3.9166 & 9.3613 & -1.6209 & -0.6805 & -3.0337 & 4.0000 \\ -4.0986 & 11.5020 & 5.6657 & -3.0036 & -2.1950 & 5.8030 & 4.0000 \\ -54.0249 & -2.8511 & 2.0324 & -4.5479 & 1.6193 & -0.9928 & 1.0000 \\ -14.1784 & 11.9594 & -6.9816 & 0.1167 & 5.3415 & -1.0821 & 8.0000 \\ -9.9353 & 16.3832 & -0.5621 & -0.0651 & -5.1173 & 0.6564 & 0 \\ -5.1125 & 15.9464 & 1.9743 & 5.5184 & -3.7046 & -4.0096 & 7.0000 \\ -10.9375 & 15.2266 & 6.3227 & 2.0865 & 7.2036 & 1.1301 & 4.0000 \\ -15.5171 & 14.8357 & -5.9684 & -6.2444 & -0.6095 & -1.4054 & 1.0000 \end{bmatrix}$$

Figure 11: The D_{PQ} Matrix

jection before applying the query will not work when the original and dimensionality reduced data sets have no common attributes, making it impossible in general to query the dimensionality reduced data set using the query that was meant for the original data set.

We then looked at an approach which overcomes this problem by performing random projection only on dimensions not found in the query, Q , and simply adding all the dimensions found in the query to the result. We saw that this approach, like the regular random projection method, preserves inter-point distances to a reasonable extent.

Next, we looked at an approach which simply replaces RP in the approach just described with PCA. We realized this new approach preserves inter-point distances much much better (in fact, perfectly) than the RP approach.

However, the PCA approach is also much more expensive computationally than the RP method.

It would be worth applying the two query-based dimensionality reduction approaches discussed in this paper to image data, and comparing their performances with that of *Discrete Cosine Transform* (Bingham & Mannila 2001).

$$Q(D_{PQ}) = \begin{bmatrix} -77.1476 & -3.0072 & -4.5016 & 7.9347 & -0.4665 & 2.8078 & 0 \\ -67.2098 & -3.9166 & 9.3613 & -1.6209 & -0.6805 & -3.0337 & 4.0000 \\ -4.0986 & 11.5020 & 5.6657 & -3.0036 & -2.1950 & 5.8030 & 4.0000 \\ -14.1784 & 11.9594 & -6.9816 & 0.1167 & 5.3415 & -1.0821 & 8.0000 \\ -9.9353 & 16.3832 & -0.5621 & -0.0651 & -5.1173 & 0.6564 & 0 \\ -10.9375 & 15.2266 & 6.3227 & 2.0865 & 7.2036 & 1.1301 & 4.0000 \end{bmatrix}$$

Figure 12: The $Q(D_{PQ})$ Matrix

Table 7: The Values of $\|u - v\|^2$ and $\|f(u) - f(v)\|^2$ for all u, v in D_Q (Approach With PCA)

u	v	$\ u - v\ ^2$	$\ f(u) - f(v)\ ^2$	$\frac{\ f(u) - f(v)\ ^2}{\ u - v\ ^2}$
1	2	434	433	0.998
1	3	5801	5798	0.999
1	4	4376	4369	0.998
1	5	5020	4999	0.996
1	6	4952	4945	0.999
2	3	4319	4317	1.000
2	4	3396	3391	0.999
2	5	3864	3843	0.995
2	6	3642	3636	0.998
3	4	395	392	0.992
3	5	185	173	0.935
3	6	211	197	0.934
4	5	272	255	0.938
4	6	238	226	0.950
5	6	230	222	0.965

Acknowledgments

Augustine Nsang's work was partially supported by the Department of the Navy, Grant ONR N000140710438.

References

- Achlioptas, D. 2004. Random matrices in data analysis. In *Lecture Notes In Computer Science; Vol. 3202, Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases*, 1 – 7.
- Bingham, E., and Mannila, H. 2001. Random projections in dimensionality reduction: Applications to image

and text data. In *Conference on Knowledge Discovery in Data, Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and data mining*, 245–250.

Fradkin, D., and Madigan, D. 2003. Experiments with random projections for machine learning. In *Conference on Knowledge Discovery in Data, Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 517–522.

Johnson, W. B., and Lindenstrauss, J. 1984. Extensions of lipshitz mapping into hilbert space. *Contemporary Mathematics* 26:189–206.

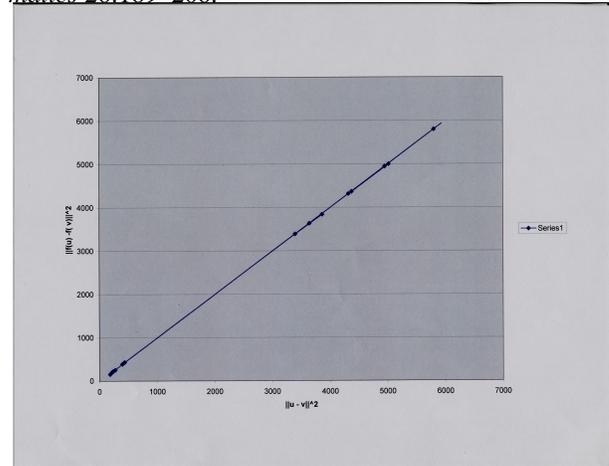


Figure 13: Graph of $\|u - v\|^2$ (on the x-axis) against $\|f(u) - f(v)\|^2$ (on the y-axis) for our alternative approach using PCA

Nsang, A., and Ralescu, A. 2009a. Query-based dimensionality reduction applied to text and web data. In *Proceedings of the Twentieth Midwest Artificial Intelligence and Cognitive Science Conference (MAICS 2009)*, 129–136.

Nsang, A., and Ralescu, A. 2009b. A review of dimensionality reduction methods and their applications. In *Proceedings of the Twentieth Midwest Artificial Intelligence and Cognitive Science Conference (MAICS 2009)*, 118–123.

This page is intentionally left blank.

Expert Systems and Fuzzy Logic

Chair: Dale Courte

Page Ranking Refinement Using Fuzzy Sets and Logic

Andrew Laughlin Joshua Olson Donny Simpson Atsushi Inoue *

Eastern Washington University
Cheney, WA 99004 USA

Abstract

This paper presents a study on personalized add-on filters applied to web search results in order to make those results more intuitive to users. Fuzzy Sets and Logic are used in order to construct such filters. Linguistic features are extracted as their universe of discourse. Three experimental filters are presented in the following specific contexts: (1) narrowing down results, (2) product specification and (3) tutorial level classification. Their performance is briefly studied mostly in qualitative manners.

Keywords: Fuzzy Sets, Fuzzy Logic, Page Ranking, Linguistic.

Introduction

Users on the Internet very likely use search engines such as *Google* and *Bing* in order to search and access to information of interest. As they enter some key words, the search engines instantaneously respond lists of web pages that are *relevant* to those key words in the order of *significance*. While the quality of search results is generally satisfactory, the users often demand finer tunings, e.g. in terms of contexts, descriptors and dependencies among key words in phrases. The following lists a few examples of such discrepancies:

Contexts. Totally different contexts intended by users, e.g. a type of coffee beans vs. software components for the key word 'java beans'.

Descriptors. Descriptors that do not symbolically match with words in target web pages, e.g. some quantifiers such as 'most' vs. actual quantities such as '99%' and qualifiers such as 'good' vs. similar ones such as 'high quality' and 'well written'.

Dependencies among words. Symbolic keyword matching in search engines is most likely performed based on regular grammars (to handle word conjugations) and often yields weak or no relevancy, e.g. 'fuzzy parser', when interpreted as a list of two key words 'fuzzy' and 'parser', vs. texts in target web pages such as 'The parser avoids fuzzy words...', 'Parsing a query e.g. founder of fuzzy logic' and '... fuzzy sets. ... C parser to compile ...'.

In theoretical aspects, web page ranking predominantly follows a fundamental ingredient in the development and

success of the *Google* search engine, and the significance is determined based on references and citations (i.e. links) made to that web page (a comprehensive survey is made in reference (Franceschet 2010)). The relevancy of a search result for given keywords is determined by this method applying to web pages containing those key words (with variations based on their conjugations)¹. Such a ranking method is effective and efficient regardless of structures and contexts of target web pages and is indeed satisfied by many users despite the above mentioned discrepancies.

On the other hand, many others hope for additional fine tunings on the search results in order to overcome those discrepancies. As having been already noticed, all of those discrepancies are caused generally by lack of various linguistic processing on target web pages—e.g. lexical and semantic processing for the matters of contexts and descriptors, and syntactic and morphological processing for those of the dependencies among words. Knowing this difference on the basis of determining significance, i.e. structural (links) versus linguistic (texts), we anticipate an effectiveness of some linguistic functionalities that compensates the structural page ranking methods.

In this paper, we consider a study on such linguistic functionalities as *personalized add-on filters* that alter web page rankings generated by conventional web engines, e.g. *Google* and *Bing*, in specific, personalized contexts. The input of such a filter is a list of (links to) web pages in an order of significance based on their structures. In text processing aspects, this is considered as stream processing of texts with a demand of real-time response (e.g. just as *Google* responds to a search query). We deploy Fuzzy Sets and Logic as the base method given its proven efficiency on stream processing (e.g. Fuzzy Controls (Mamdani and Assilian 1975; Takagi and Sugeno 1985)) and effectiveness on uncertainty management intrinsic to linguistic processing (Zadeh 1965; 1973).

Related concepts such as page ranking and fuzzy sets and logic are briefly introduced in the next section. Then three experimental filters are presented along with their qualitative studies on performance for the following specific contexts:

¹In practice, massive web pages and their ranks are pre-compiled and key words are indexed by web crawlers, autonomous processes that explore links and URLs.

*Correspond via E-mail to inoueatsushij@gmail.com

narrowing down results, product specification and tutorial level classification. Some clear distinction from other similar works is made within the experimental setting.

Related Concepts

The following related concepts are briefly introduced: Page Ranking, Regular Grammar, Fuzzy Sets and Fuzzy Logic.

Page Ranking

Let $I(P)$ the significance of page P and B be the set of pages that refer to (i.e. have a link to) page P (Austin 2011). Suppose that page P_i has a link to page P and that P_i has l_i links. Then $I(P)$ is determined as follows:

$$I(P) = \sum_{P_i \in B} \frac{I(P_i)}{l_i} \quad (1)$$

To determine $I(P)$, we need $I(P_i) \forall P_i \in B$ and so do we for each and every one of those pages. This certainly causes "chicken and egg" situation. To resolve this situation, we use the *power method*. Let $H = [H_{ij}]$ be a square matrix² representing references (links) among all web pages P_1, \dots, P_n such that

$$H_{ij} = \begin{cases} \frac{1}{l_j} & \text{if } P_j \in B_j \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

and let $I = [I(P_i)]$ be a vector whose components are the significance of all the web pages. Then we observe $I = H \cdot I$. This means that I is an eigenvector (aka a stationary vector) with $\lambda = 1$. The *power method* iteratively computes

$$H \cdot I_i = v = \lambda_{i+1} \cdot I_{i+1} \quad (3)$$

until $I_i = I_{i+1}$ (i.e. practically $|I_i(P) - I_{i+1}(P)| < \epsilon$ for all components). The initial eigenvector usually has only one component whose value is 1, and the remaining components have 0s. The initial eigenvector is $\lambda_1 = 1$. The convergence of this iteration is determined whether $|\lambda_2| < 1$, and its speed (i.e. #iterations) is determined by the magnitude of $|\lambda_2|$ such that it gets slower as $|\lambda_2|$ is closer to 0.

Regular Grammar

Regular grammar can describe a set of strings by a collection of rules in the following patters: $A \rightarrow c$ and $A \rightarrow cB$ (Sipser 2005). Such rules yield only linear parsing trees. In practice such as scripting and programming, we use *regular expressions* that consists of the following:

- |: Boolean "or."
- (. . .): A regular expression within the parentheses.
- *: Zero or more repetition of the preceding element.
- +: One or more repetition of the preceding element.
- ?: Zero or one repetition of the preceding element.

For example, $ab?a$ yields aa and aba ; $ab * a$ yields aa, aba and $abba$; $a(cb)^+$ yields acb and $acbc$; and $a(c|b)^+$ yields ac, ab, acc, abb and abc .

² i -th row and j -th column.

Fuzzy Sets

In the most cases, fuzzy sets represent linguistic expressions that intrinsically contain fuzziness such as 'tall' on height and 'low' on leftover stipend (Zadeh 1965).

Definition. A *fuzzy subset* of a set U is defined by means of a membership function

$$\mu : U \rightarrow [0, 1] \quad (4)$$

The set U is so-called the *universal set*. In the above two examples, linguistic terms 'tall' and 'low' correspond to fuzzy sets defined over appropriate universal sets such as 'height' (i.e. a set (interval) of real numbers representing human height, e.g. [100, 220] in centimeters) and 'leftover stipend' (e.g. [0, 100] in USD). In case of a crisp set, the range of the membership function becomes $\{0, 1\}$ instead of $[0, 1]$.

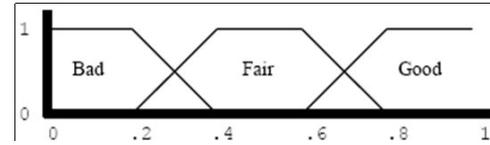


Figure 1: Fuzzy Sets (Fuzzy Partition)

Usually, fuzzy sets are defined in simple canonical shapes such as triangles and trapezoids (see fig. 1 as an example of three trapezoidal fuzzy sets). In this simplicity, you may easily see *the elasticity of fuzzy sets* such that every crisp (i.e. non-fuzzy) interval has only one fuzzy set i.e. $\mu(x \in U) > 0$ and $\mu(x) = 1$ (i.e. the complete membership), and every fuzzy interval has more than one fuzzy set i.e. $\mu(x) > 0$ and $\mu(x) < 1$ for all within that interval (i.e. the partial memberships). Further, a fuzzy partition is often considered for the sake of completeness in computational models.

Definition. A *fuzzy partition* of a set U is a set of normal (i.e. at least one element $x \in U$ s.t. $\mu(x) = 1$) fuzzy sets of U such that

$$\sum_i \mu_i(x) = 1 \forall x \in U \quad (5)$$

Fuzzy sets may be defined subjectively, unlike probability distributions. Appropriate fuzzy partitions with simple canonical shapes as shown in fig. 1 are often used in many cases. They are also dynamically generated or refined by applying some machine learning methods. In such cases, simple and smooth shapes of fuzzy sets should be maintained due to their elasticity and approximation nature.

Finally, their (*standard*) *set operations* are defined as follows:

$$\begin{aligned} \text{Set intersection: } & \mu_{A \cap B}(x) = \min[\mu_A(x), \mu_B(x)] \\ \text{Set union: } & \mu_{A \cup B}(x) = \max[\mu_A(x), \mu_B(x)] \\ \text{Set complement: } & \mu_{\bar{A}}(x) = 1 - \mu_A(x) \end{aligned} \quad (6)$$

Fuzzy Logic

Fuzzy logic is originally proposed by Zadeh as a qualitative, simplistic method for (especially complex) system analysis (Zadeh 1973). In this framework, Modus Ponens is generalized and formalized as a fuzzy relation (i.e. considered

as a partial truth maintenance system). Formally the generalized Modus Ponens can be written as

$$a \rightarrow b \wedge a' \Rightarrow b' \quad (7)$$

where a , b , a' and b are fuzzy (sub)sets representing fuzzy statements, e.g. 'temperature is high'. We may now rewrite this in fuzzy set theoretic, i.e. fuzzy relational, aspects such that

$$\mu_{b'}(y \in V) = \bigvee_x [R_f(x, y) \wedge \mu_{a'}(x \in U)] \quad (8)$$

where all the membership functions μ represent those fuzzy statements (e.g. μ_{high} in the above example), U and V represent their universal sets (e.g. 'temperature' in the above example), and R_f is the fuzzy relation that represents the (fuzzy) implication $a \rightarrow b$.

That fuzzy relation is further specified as a result of projecting material implication (i.e. $a \rightarrow b = \neg a \vee b$) such that

$$R_f(x, y) = (a \times b) \cup (\bar{a} \times V) = (\mu_a(x) \wedge \mu_b(y)) \vee \mu_{\bar{a}}(x) \quad (9)$$

where $a \subseteq U$ and $b \subseteq V$.

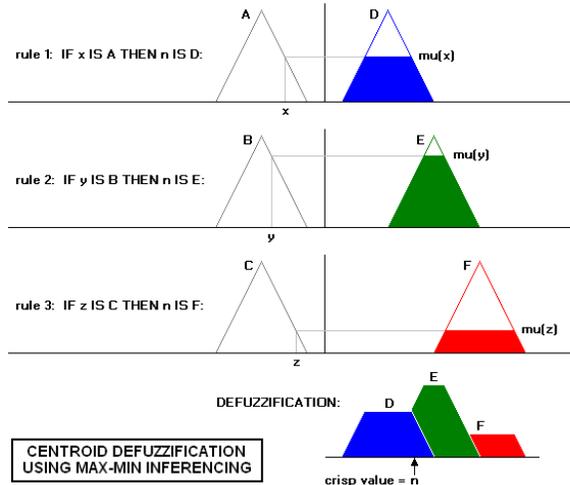


Figure 2: Fuzzy Control (Mamdani)

In fuzzy control, we only need to consider the special case³ such that

$$R_f(x, y) = (a \times b) \cup (\bar{a} \times \emptyset) = (\mu_a(x) \wedge \mu_b(y)) \quad (10)$$

This is indeed Mamdani's fuzzy control model when selecting $\mu_a(x) \wedge \mu_b(y) = \min[\mu_a(x), \mu_b(y)]$ (Mamdani and Assilian 1975) (see fig. 2).

When multiple fuzzy implications (aka fuzzy IF-THEN rules) exist in the system, we need to disjunctively combine all the results, i.e. partial truth such that $c = c_1 \vee \dots \vee c_n$, where $c_{1 \leq i \leq n}$ is a fuzzy set representing the result for the

³System control cannot specify outputs for complements of inputs, i.e. $\bar{a} \times \emptyset = \emptyset$.

i -th fuzzy implication and c is the one for all results combined. In case of fuzzy control, c represents all possible outputs with associated partial truth values. In order to determine a single output, we need to select one output such that $c(\subseteq V) \rightarrow y^*(\in V)$ (so-called *defuzzification*). Center-of-Gravity (CoG) method is proposed in Mamdani's model (see fig. 2).

$$y^* = \frac{\int \mu_c(y) \cdot y \, dy}{\int \mu_c(y) \, dy} \quad (11)$$

Takagi-Sugeno fuzzy control model better integrates fuzzy implications and defuzzification as a model-free regression (Takagi and Sugeno 1985). In this model, each fuzzy implication $R_f(x, y)$ is approximated as a function $y_i = f_i(x)$ for input x , and its output y_i is linearly combined based on the partial truth value of the hypothesis, i.e. $\mu_{a_i}(x)$, such that

$$y^* = \frac{\sum_i \mu_{a_i}(x) \cdot f_i(x)}{\sum_i \mu_{a_i}(x)} \quad (12)$$

In discrete problem domains such as classification, we need to identify a class as a result of disjunctively combining all those result fuzzy sets c_i . Since their membership functions are all constants (i.e. $\mu_{c_i}(y) = z_i \in [0, 1]$ s.t. $z_i = \mu_{a_i}(x)$ from fuzzy implication $a_i \rightarrow b_i$), the defuzzification is achieved simply as a result of the disjunctive combination with $\max[\cdot]$ in order to select the class label. This is corresponding to the definition of fuzzy classifier such that

$$C = \text{ARGMAX}_i[\mu_{a_i}(x)] \quad (13)$$

where C is a class label associated with μ_{b_i} from fuzzy implication $a_i \rightarrow b_i$, as well as its result μ_{c_i} .

Experimental Filters

Three experimental filters are presented along with brief studies on their performance for the following specific contexts: narrowing down results, product specification and tutorial level classification. The experimental setting is presented first together with some clear distinction from other similar works.

Experimental Setting

In general, we consider *simplicity* as the core of development. In particular, the following setting is followed in the development of experimental filters.

Add-on Filters. Given all possible bias on intentions and interpretations, we focus on development of *personalized add-on filters* on web browsers. Such filters are very likely implemented as extensions and other forms of modules according to architectural specifications of web browsers as well as application programming interfaces (APIs). Fuzzy Sets and Logic are used as the technical framework of those filters and are easily implemented in any forms of development environment, application framework and programming language. The *input* of each filter is a *list of web pages*, most likely that of URLs, generated as a result of using a web search engine such as Google and Bing. The filter then accesses to texts from that list. The *output* is a *modified list*

of those web pages, e.g. altered orders, selective lists and grouped lists.

In doing so, users can easily switch back and forth between the ordinary and this filtered search results. In addition, the inputs, i.e. keywords, remain the same in both options. This is different from other works that utilize fuzzy sets and logic in a similar manner. For instance, Choi’s work (Choi 2003) incorporates linguistic processing features (using fuzzy sets and logic) directly to a web search engine, thus demands a modification on the server as well as in inputs. This causes substantial overheads on the server including, but not necessarily limited to, configurations of various personalization and context dependencies. Such configurations may likely serve as very critical overheads when considering recent studies on bias and ideal usage of web search engines (Goldman 2008).

Recent search engines such as Google and Bing keep track of search results for various personalization and customization purpose. They are implemented as a part of server (i.e. search engine) functionalities. In contrast, ours are implemented as extensions of web browsers, thus are served as *additional personalization*.

Context Dependency. Each user has one’s own intention and bias in many different situations and none are likely identical to the others. In other words, it is ideal (i.e. the simplest) to facilitate a collection of add-on filters that cater such different situations with various intentions and biases.

While many works in intelligent systems tend to handle context dependencies by adaptive capacity on the server, this causes substantial overheads. Anari et. al. approach to context dependencies by incorporating capacities of adaptive behaviors and generalization (i.e. capacity of fuzzy sets and logic) directly in the page ranking method (Anari, Meybodi, and Anari 2009). Such sophistication very likely causes the overhead, thus is not feasible for extensions of conventional services such as Google and Bing (unlike their own retrieval system).

Linguistic Keyword Processing. Fuzzy sets and logic are well incorporated with the standard *statistical natural language processing* such that simple features are extracted from texts in order to apply various methods of machine learning and reasoning. Such simple features may include *word frequencies* and *word appearances* as the core. We then consider other variations within those features such as

- those on words that are linguistically related, e.g. synonyms, antonyms, acronyms, etc.
- those on words with simply conjugation processing, more generally processing morphological structures.
- those on a sequence of n words, i.e. n -grams.

The most significant advantages of such simple features are with regard to text stream processing. Texts are parsed only once (aka one-pass) in order to yield real-time responses. Morphological structures are mostly handled by regular grammars (expressions), and those linguistically related features demand lexicons such as thesauri. Fortunately, those are feasible within the text stream processing frame-

work. Conventional syntactic analysis is unlikely feasible within this framework; however, stochastic and probabilistic analysis on n -grams (e.g. Markov process) and simple parsing with regular grammar may often compensate for this shortcoming.

It is commonly known that the web search inputs, i.e. keywords, are short and simple—consisting only of a few keywords. As a consequence, a canonical structure such as a pair of an adjective and a noun can be expected. This very well fits within the frame work of fuzzy sets and logic such that the adjective (a word or a phrase) corresponds to a fuzzy subset on the universal set and the noun (a word or a phrase) corresponds to that universal set. The elements in that universal set are one of those simple linguistic features and the fuzzy set is generated accordingly on those.

Prototype. All the experimental filters presented in this paper are required to be rigorously prototyped. The web search results are indeed extracted from Google and Bing for specific queries, i.e. lists of keywords. The standard models of fuzzy sets and logic are deployed, e.g. the standard fuzzy set operations and Mamdani’s fuzzy control model, for implementation advantages such as simplicity and available tools. As a trade-off, performance studies became very limited at this time—i.e. a continuing work.

Filter 1: Narrowing Down Results

This experimental filter considers narrowing down and re-ordering search results from a web search engine about favorite music. The intention is exploration of relevant information, thus is broad and general. More technical details are as follows:

Queries. For this experiment, we only consider “good reggae songs.”

Features. The universal set is the normalized frequency $f = \frac{f(w \in t \wedge d)}{f(w \in t) - f(w \in t \wedge i)}$ of affirmative words in a text t , e.g. ‘good’, ‘favorite’, ‘love’, etc. Function $f(w)$ indicates the frequency of word w with a specified membership (e.g. t , $t \wedge d$ and $t \wedge i$). A thesaurus d is used in order to identify a set of words and their conjugations. We remove meaningless words such as prepositions and pronouns and they are maintained in the ignored word list i .

Fuzzy model. A fuzzy classifier consisting of three fuzzy sets in a fuzzy partition similar to those in fig. 1: $\mu_{\text{high}}(f(t))$, $\mu_{\text{medium}}(f(t))$ and $\mu_{\text{low}}(f(t))$.

Output. (1) a class of the word frequency as a fuzzy degree of significance, and (2) a defuzzification of those three membership functions as the degree in order to determine the rank.

Search engine. Bing.

A small and simple performance evaluation was conducted as follows:

Subjects to evaluation. Significance labels generated by the fuzzy classifier.

Examinee(s). One person who is familiar with reggae songs.

Table 1: Confusion Matrix

	H	High	Medium	Low
M		High	Medium	Low
High		1	2	0
Medium		0	3	0
Low		1	0	2

Procedure. Several web pages (texts) that are randomly selected from the filtered results are presented to the examinee and ask the one to put one of those three labels ('low', 'medium' and 'high') in order to indicate significance.

Results. Compiled as a confusion matrix (see table 1) where H indicates the human examinee and M indicates this add-on filter.

Despite its very simple linguistic feature, i.e. the normalized word frequency of single word, we noticed some improvements. Two out of three misclassification cases are classified in adjacent classes: 'high' where it should be 'medium'. One case is completely off; however, this was the web page containing a song list and hardly contained affirmative words.

Table 2: Top 5 Results from Bing

Bing Ordering	Fuzzy Value	Human Value	Reason
1	Medium	Medium	Forum suggestion.
2	Low	High	Long list of songs. Few other terms.
3	Medium	Medium	Forum suggestion.
4	Medium	Low	Short list of links. Non-familiar artists.
5	Medium	Medium	Forum suggestion.

Table 2 indicates the results of the top 5 from Bing search. Two pages are off, but both are song lists and hardly contain affirmative words. Needless to say, more extensive studies are necessary. Nevertheless, this small performance study positively suggested further study.

Filter 2: Product Specification

This experimental filter anticipates to enhance product search results. The key idea is that a certain specification is accounted to significance of products appeared in the search results. Such a specification is automatically identified as a result of mapping a set of key words to some fuzzy model. The technical details of this filter follows:

Queries. For this experiment, we only consider "energy efficient light bulbs."

Features. The universal set is the efficiency $e(b) = \frac{l(b)}{w(b)} \in [5, 100]$, where $l(b)$ is the lumen and $w(b)$ is #watts of a light bulb b . Product specification such as $l(b)$ and $e(b)$ are obtained from Google's product search results by parsing XML attributes and keywords.

Fuzzy model. A single fuzzy set representing 'energy efficiency' such that

$$\mu(e(b) \in [5, 100]) = \begin{cases} 0 & \text{if } e(b) \in [5, 15] \\ \frac{e(b)-15}{35} & \text{if } e(b) \in (15, 50) \\ 1 & \text{if } e(b) \in [50, 100] \end{cases} \quad (14)$$

This is determined based on the official chart of Energy Federation Incorporated (EFI).

Output. Reordered list of products according to the membership degree of $\mu(e(b))$.

Search engine. Google Shopping (product search service).

A qualitative performance study on this filter is conducted as follows:

Subjects to evaluation. Quality of product search results.

Examinee(s). A few people who are in need of light bulbs.

Procedure. Obtain testimonials by presenting both product lists: filtered and not-filtered.

Results. A collection of testimonies are listed below. See fig. 3&4 to compare top 8 products (i.e. energy efficient light bulbs). Five bulbs in the filtered results also appear in the not-filtered one. Among those, only one (the top bulb) appear in the same rank. Three new bulbs appear in the filtered list.

Here is the sample testimonies:

- "I think [the filtered result] is better it sticks more to sorting the wattage in order, making the *energy efficiency* stand out more if people are looking to cut costs by reducing wattage. This is not taking things like price or actual lumenage of the bulb into effect (if people are looking for a brighter, yet energy efficient bulb, which might actually be a higher wattage)."
- "I'd have to go with [the filtered result] mainly because it ranked the Satco Halogen among the lowest. The Satco Halogen uses the most energy out of all the bulbs with the least gain of lumen. 57 watts for 1100 lumens vs 13 watts for 900 lumens."

This filter, despite its simplicity in uncertainty management, turned out to be better effective than we expected. The current trends on ecology stimulated the bulb market so that manufacturers release new types of bulbs and users (i.e. consumers) are not yet familiar with those new types. More comprehensive implementation of such a filter is likely significant from marketing aspects and is certainly feasible from technical aspects.

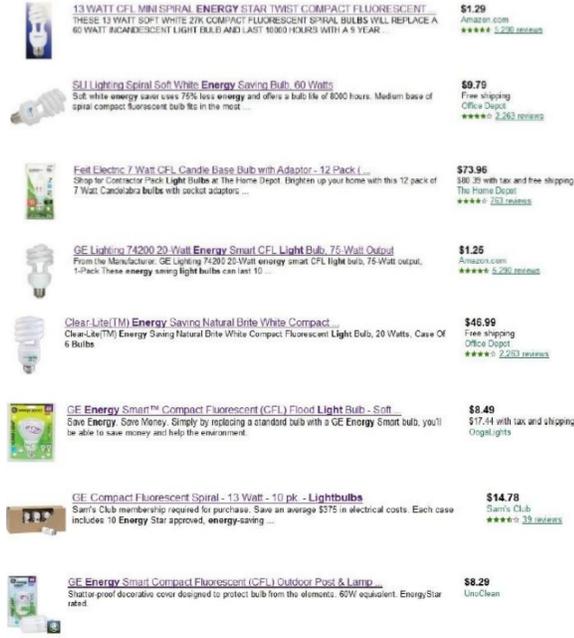


Figure 3: Product Search Results: not filtered

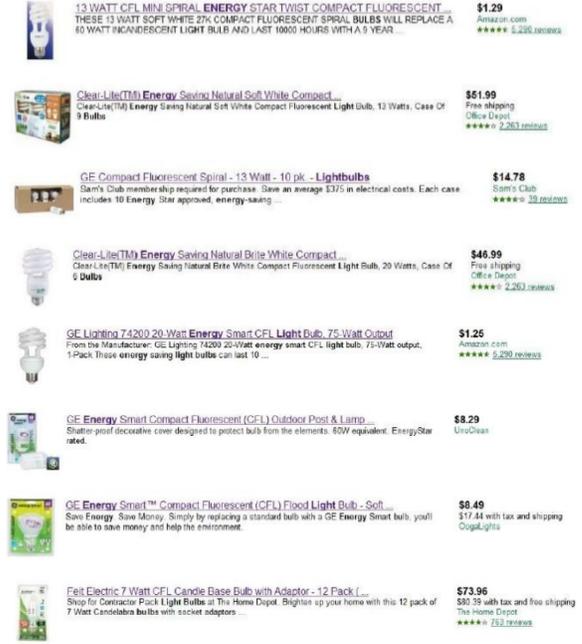


Figure 4: Product Search Results: filtered

Filter 3: Tutorial Level Classification

This experimental filter classifies tutorial sites into three classes: 'beginner', 'intermediate' and 'advanced'. Only a set of few general keywords are taken into account for this classification. In doing so, we maintain a high degree of generality (i.e. domain independence). Each of such keywords are associated with appropriate membership degrees per class. Mamdani's model is deployed for this classification as a result of defining singleton fuzzy sets over a shared universal set representing those classes.

Queries. For this experiment, we consider:
 $(c|ruby|python)(tutorials?|guides?)$

Features. We only use the following keywords and their conjugation (handled by regular expressions): $intro(a|...|z)*$, $novice$, $intermediate$, $advanc(a|...|z)*$, $experts?$. We also use their frequencies: $f_i(t)$ for $intro(a|...|z)*$, $f_n(t)$ for $novice$, $f_m(t)$ for $intermediate$, $f_a(t)$ for $advanc(a|...|z)*$ and $f_e(t)$ for $experts?$.

Fuzzy model. The following IF-THEN rules and fuzzy sets are defined:

1. $h(f_i(t)) \rightarrow c_i^h = \{1.0/0\}$
2. $h(f_n(t)) \rightarrow c_n^h = \{1.0/0.25\}$
3. $h(f_m(t)) \rightarrow c_m^h = \{1.0/0.5\}$
4. $h(f_a(t)) \rightarrow c_a^h = \{1.0/0.75\}$
5. $h(f_e(t)) \rightarrow c_e^h = \{1.0/1\}$
6. $m(f_i(t)) \rightarrow c_i^m = \{0.5/0\}$
7. $m(f_n(t)) \rightarrow c_n^m = \{0.5/0.25\}$
8. $m(f_m(t)) \rightarrow c_m^m = \{0.5/0.5\}$

9. $m(f_a(t)) \rightarrow c_a^m = \{0.5/0.75\}$
10. $m(f_e(t)) \rightarrow c_e^m = \{0.5/1\}$

where

$$h(f) = \begin{cases} 0 & \text{if } f \leq 5 \\ \frac{f-5}{5} & \text{if } 5 < f < 10 \\ 1 & \text{if } f \geq 10 \end{cases} \quad (15)$$

and

$$m(f) = \begin{cases} 0 & \text{if } f = 0 \vee f \geq 10 \\ \frac{f-5}{5} & \text{if } 0 < f < 5 \\ 1 & \text{if } f = 5 \\ \frac{10-f}{5} & \text{if } 5 < f < 10 \end{cases} \quad (16)$$

Output. The degree of expertise $e(t) \in [0, 1]$ as a result of defuzzification of this fuzzy model, where $e(t) = 1$ means 'expert' and $e(t) = 0$ means 'beginner'. The degree of each class is determined as follows:

Beginner: $b(t) = 1 - e(t)$

Intermediate: $i(t) = 1 - |e(t) - 0.5|$

Advanced: $e(t)$

Search engine. Bing and Google.

Fig. 5 and 6 show a search result and a classified result of Google with query "python tutorial." As you may notice, there are a few pages in unintentional context, i.e. Pokemon. This is a trade off with generality, that is caused due to a shallow linguistic analysis such as word frequencies of only a few keywords. Nevertheless, we received several positive feedbacks about this filter that mainly commends such a classification offers more efficient information browsing.

- ◆ [Ruby Basic Tutorial](#)
- ◆ [Ruby in Twenty Minutes](#)
- ◆ [Ruby Tutorial; Ruby Study Notes - Best Ruby Guide, Ruby Tutorial](#)
- ◆ [Ruby Tutorial - Learn Ruby](#)
- ◆ [Ruby Tutorial with Code Samples](#)
- ◆ ...

Figure 5: Search Results:Google

Beginning

- ◆ [Dive Into Python](#)
- ◆ [Python 101 – Introduction to Python](#)
- ◆ [BeginnersGuide/NonProgrammers - PythonInfo Wiki](#)
- ◆ [A Beginner's Python TutorialThe Python Tutorial — Python v2.7 documentation](#)

Intermediate

- ◆ [PyCon2007/Feedback/TutorialIdeas – PythonInfo Wiki](#)
- ◆ [PyBindGen Tutorial — PyBindGen v0.15.0 documentation](#)
- ◆ [Intermediate Python Programming – Learning Python, Linux, Java ...](#)
- ◆ [Intermediate Python Tutorial](#)
- ◆ [Intermediate Tutorial 6 – PyWiki](#)

Advanced

- ◆ [Advanced Python Programming](#)
- ◆ [Python 201 – \(Slightly\) Advanced Python Topics](#)
- ◆ [Python Tutorials, more than 300, updated March 2, 2009 and ...](#)
- ◆ [Advanced Applications of Python](#)
- ◆ [A little more advanced Python tutorial. – Ubuntu Forums](#)

Figure 6: Classified Results:Google

Concluding Summary

A preliminary study on page ranking refinement using fuzzy sets and logic is presented. Three simply experimental filters are presented in order to demonstrate their effectiveness. Their performance studies, despite their rough and mostly qualitative contents, positively suggest continuing works. In experimental setting, clear difference from similar works is discussed.

Among many future works, we will first conduct more extensive performance studies and prepare some development frameworks for popular web browsers such as Fire Fox and Chrome.

Acknowledgement

This research was conducted as a small course project (so-called *Intelligent Search Project*) in CSCD498/598 Fuzzy

Sets and Logic at EWU at the end of Fall 2010. The duration of this project was two weeks. The best three projects are presented as experimental filters in this paper and those students are recognized as its co-authors.

References

Anari, Z.; Meybodi, M. R.; and Anari, B. 2009. Web page ranking based on fuzzy and learning automata. In *Proceedings of the International Conference on Management of Emergent Digital EcoSystems*, MEDES '09, 24:162–24:166. New York, NY, USA: ACM.

Austin, D. 2011. How Google Finds Your Needle in the Web's Haystack. *AMS Feature Column: Monthly Essays on Mathematical Topics*.

Choi, D.-Y. 2003. Enhancing the power of Web search engines by means of fuzzy query. *Decis. Support Syst.* 35:31–44.

Franceschet, M. 2010. PageRank: Standing on the shoulders of giants. *ACM Computing Research Repository (CoRR)* abs/1002.2858v3.

Goldman, E. 2008. Search Engine Bias and the Demise of Search Engine Utopianism. In Spink, A., and Zimmer, M., eds., *Web Search*, volume 14 of *Information Science and Knowledge Management*. Springer Berlin Heidelberg. 121–133.

Mamdani, E., and Assilian, S. 1975. An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Man-Machine Studies* 7(1):1 – 13.

Sipser, M. 2005. *Introduction to the Theory of Computation (2nd eds.)*. Course Technology (Thomson).

Takagi, T., and Sugeno, M. 1985. Fuzzy identification of Systems and Its Applications to Modeling and Control. *IEEE Transactions on Systems, Man and Cybernetics*.

Zadeh, L. A. 1965. Fuzzy Sets. *Information and Control* 8:338–353.

Zadeh, L. A. 1973. Outline of a New Approach to the Analysis of Complex Systems and Decision Processes. *IEEE Transactions on Systems, Man, and Cybernetics* 3(1):28–44.

Computational Intelligence for Project Scope

Joseph M. McQuighan, PMP and Robert J. Hammell II, PhD

Towson University
8000 York Road
Towson, Maryland 21252

Abstract

Managing scope is a critical process in information technology (IT) project management. Reporting the status of scope requires both an understanding of the status of individual activities and the aggregation into an overall status for the project. Unlike cost and schedule which have the objective measures of currency spent or days passed, scope is subjective. Understanding the status of scope as a project moves forward is critical to success; however, many times IT projects fail due to mismanagement of scope constraints. Recent research has confirmed status reporting and analysis as a major problem in IT projects. Other research has looked at how computational intelligence (CI) techniques might be applied to the domain of project management for cost and time constraints. This study looks at scope, a third constraint of project management. Since scope has properties of imprecision and vagueness, fuzzy logic would be an appropriate tool from Computational Intelligence. This study focuses on using the recently proposed Z-mouse for the collection of status information, and then using fuzzy logic for the reporting of project status for the scope constraint.

Introduction

Project managers collect data on the performance of their projects in order to be able to report the status and to forecast future performance. The Standish Group recently surveyed 400 organizations and reported that only 32% of information technology projects were successful, with close to a quarter of the projects reported as failures (Levinson 2009a). Articles in CIO magazine point out that poor requirements and scope management contributes to these failures (Levinson 2009b, Levinson 2009c). Much has been written about how to manage scope, from improving business cases by establishing clear objectives, to ensuring requirements specify an acceptance criteria, to change management processes. The measuring of scope status has largely been ignored because of the difficulty of measuring requirements. This research looks at the fuzzy

nature of the inputs to status reports for the scope constraint to answer two questions:

- Can fuzzy systems offer a tool that can capture the status of the scope of an individual activity in an IT project?
- Can the scope status for activities be aggregated into a meaningful project scope status?

It is anticipated that from the first question it might be possible to determine if there is a common or generally accepted understanding amongst project managers as to how to report status when the inputs are vague or imprecise for an activity. The Z-mouse tool proposed by Lotfi Zadeh, which is an extension of Zadeh's work on fuzzy set theory (Zadeh 1973), is a leading edge data collection mechanism that will be used as the data collection tool in this study.

Project Status

Weill and Broadbent have stated that information technology (IT) is "very strongly project based" (Weill 1998). Understanding the status of a project is important to project managers, upper management, and executive sponsors of a project. There are many stakeholders outside of a project's organizational structure also interested in the status of a project (PMBOK Guide 2008). The overall project status many times is seen as an aggregation of the status of the three traditional project constraints: cost, schedule, and scope. Depending on the project, the fourth constraint of quality could also be present (PMBOK Guide 2008). Further, the overall status of each specific constraint is an aggregation of the status of each *activity* status considering that constraint. For example, each activity is examined to judge how it is meeting the cost constraint; the *project* status related to cost is an amalgamation of all the individual activity cost-related statuses. The project status for schedule is similarly determined by first considering each how each activity is performing with respect to that constraint. Finally, the overall project status is established by combining the individual constraint statuses. Thus, for the entire project, the constraints are dimensions that are evaluated independently and then later aggregated.

The Problem

As mentioned, the overall project status should be determined by aggregating the individual activity statuses. Instead, it is often reported as the opinion of the project manager, which is subjective. Recent postings on the LinkedIn internet site for professional project managers requesting help on project status met with a wide variety of rapid responses indicating the high interest level of practicing project management professionals. Since executive managers tend to focus on problem areas, this translates to projects in trouble and as a result, there is a tendency to under report status. Snow and Keil investigated variance between the true status of a software project from the reported status and found that accuracy was a major problem. "The intangible nature of software makes it difficult to obtain accurate estimates of the proportion of work completed, which may promote misperceptions regarding project status" (Snow 2001).

Snow and Keil found that in addition to misperceptions in the status of a software project, project managers might also censor the status reports of poorly performing projects. They cited an example of a project that lost \$125 million over 3 years, yet senior management did not have any insights into the problems. "The combined effects of project manager misperceptions (errors) and bias in reporting leads to what we call "distortion" in the project status information received by senior executives" (Snow 2001). Snow identified the need for better tools for understanding project status, and the necessity to automate the reporting of status to avoid project manager bias and reporting errors. With other research projects focused on schedule and cost constraints, this study investigates scope.

Project Status Background

Projects by definition are unique, and "because of the unique nature of projects, there may be uncertainties... The project team must be able to assess the situation and balance the demands in order to deliver a successful project" (PMBOK 2008). Assessments are the feedback during the execution of a project so that the project can be guided to a successful completion. As projects move forward, project managers are constantly gathering data on the status, converting that data into useful information to be reported, and then acting upon the information. Often the data is vague, or needs interpretation. An example of vague data is that it is difficult to determine to what extent the scope is being met. To label project scope as 67.35% met is recognized as impractical precision. The imprecision in the data is the subject of this study, and rather than using traditional methods to attempt to quantify scope, computational intelligence offers new tools and techniques for capturing vagueness. Computational intelligence tools can "identify semantically ambiguous concepts and convert them to fuzzy sets" (Cox, 1999) which can then be resolved into solutions that can be handled by project managers.

With over 300,000 members, the Project Management Institute (PMI) is recognized worldwide as an authority on project processes. Their Project Management Book of Knowledge (PMBOK) does not spell out the format of status reports, nor does it tell project managers specifically how to write a status report. Instead the PMBOK identifies processes, defines inputs, tools and techniques, and the data flows that tie the processes together (PMBOK 2008). The PMBOK, as stated in section 1.1, is an assembly of good practices that has the consensus and general agreement of project management professionals. The PMBOK "is a guide rather than a methodology. One can use different methodologies and tools to implement the framework" (PMBOK 2008). This gives practitioners the flexibility to choose techniques that work for their given situation.

The Project Management Institute's PMBOK identifies the performance reporting process as part of their Monitoring and Controlling process group (PMBOK 2008). The PMBOK lists three outputs from the performance reporting process: 1) Performance reports, 2) Organizational process assets updates, and 3) change requests. The purpose of the reports is to act as feedback into the processes that "track, review, and regulate the progress and performance of the project; identify any areas in which changes to the plan are required; and initiate the corresponding changes" (PMBOK 2008). To this extent, data is converted into actionable information guiding the project to completion. This process of reporting performance is crucial to initiating corrective actions and preventive actions, and becomes part of the organization's lessons learned historical database.

When reporting the status of projects, Dow and Taylor have found that project dashboards are often used by senior managers (Dow 2008). Dashboards are a graphical summary of the status of a project, many having a drill down capability. The purpose is to give a quick, high level overview of a project to upper management whose role is to prioritize, review, and make funding decisions (Benson 2004). Dow and Taylor state that two constraints of project management, cost and schedule, are evaluated independently and summarized. It is interesting to note that they make no mention of scope. They also found that to assist with quick problem identification sometimes a stoplight report is produced where each area is assigned a color to represent the status of that constraint. Typically the stoplight colors of red, yellow, and green are used to represent the status of each constraint (Dow 2008). These constraint statuses will be aggregated into a cumulative status for the project (Barnes 2009). Green-Yellow-Red traffic light status reporting is widely used because of its simplicity, and the quickness with which people can identify if there is a problem that needs addressing. This traffic light technique is in common use many projects, and especially popular in status reports to stakeholders who might have little time or inclination to understand the project details. Performance reports are essential inputs

necessary to monitor and control a project (PMBOK 2008), but the dashboards get the attention of the executives.

It would seem that numerical inputs into reports and dashboards should yield an objective status for reporting purposes. The ideal ought to be that for a given activity, the fixed numerical data goes in and a Green, Yellow, or Red project status comes out. The next stage in the process would be that the individual activities are then mechanically aggregated into an overall project status. The reality is that there are many factors that influence the decision to label a project status with a particular status value for a singular activity, and that the aggregation of those statuses for multiple activities of the critical constraints is open to interpretation as well.

When the project status is not a clear green or red, Barnes and Hammell found that "ambiguity is present in the scenario where the expert had to decide that the status of a project is Yellow" (Barnes 2009). Looking at the case of rating just one of the activities in a project, it is simple for status green. Most managers would look at a truly green activity and agree that the status is okay, or green. Beyond green status, it becomes questionable. Barnes has shown that yellow status can be misinterpreted or communicated as green.

The problem is much worse when the project is in serious trouble. Snow and Keil found that IT project status of red is frequently misreported (Snow 2002). Projects that are failing need the most attention from the executive management team; yet, without the knowledge that the status is red, the proper level of actions are not taken to bring a red project into compliance which often leads to financial disasters. The magnitude of project failures is alarming. For example, barely ten years ago *The San Francisco Chronicle* reported that the state of California wasted over \$1 billion on failed computer automation projects (Lucas 1999).

The second stage in reporting status, aggregation of the constraints into an overall project status, has been studied by a number of authors. But there are complexities that make the automatic summarization difficult. For example, a project that is ahead of schedule might also be significantly over cost at that point in time. What is the true status of that project? Just looking at the raw data might yield a green status on schedule, but a red status for cost. However, the costs might reflect that fact that the project is ahead of schedule, so it might be the case that the project will finish ahead of schedule, and ultimately within cost constraints. Ahead of schedule, and meeting cost constraints when completed would seem to mean the project is "green", in spite of a "red" cost. This implies that making status a simple mechanical output of numeric inputs can produce status errors. Human intervention is required to interpret the data into meaningful information.

Measuring the Constraints

The cost and schedule constraints of project management have numerical quantities that can be measured. The numbers have an element of objectivity which can be used in forecasts. Econometric methods such as regression analysis and autoregressive moving averages, or time series methods such as linear prediction, trend estimation, and moving averages have been used by practitioners of project management (PMBOK 2008). Currencies are tracked and reported using time series methods such as earned value (PMBOK 2008). Calendar dates and/or labor hours can be tracked for the time constraint. Depending on the project, quality might also be measurable and reportable.

Scope, however, is much more difficult to measure, and at the same time is the critical element from which the time and cost are derived. Richardson and Butler stated that "the concept of project scope is a foundation idea. It establishes the base for much of the subsequent management activities" (Richardson 2006). At a high level overview of the project management processes defined by the PMI, scope is derived from the project charter and requirements, the scope baseline then feeds into the Work Breakdown Structure (WBS). The WBS is the input to time management, which was an output of the scope definition being decomposed into activities. "Activities provide a basis for estimating, scheduling, executing, and monitoring and controlling the project work" (PMBOK 2008).

In a similar manner, scope and the WBS feed into cost estimates and cost management. This means that if the scope is wrong, the time and cost estimates will be wrong, or if the scope changes then time and cost can be severely impacted. Time and cost estimates are calculated by activity, but the list of activities comes from the scope definitions and WBS that were completed early in the life cycle of a project (Gido 2009). This implies that scope and requirements errors early in a project can carry over into constraints that are perceived to be more objective, such as cost.

Schwalbe states that managing scope is especially difficulty on IT projects. Scope can be relatively undefined at the beginning, can grow out of control due to creep, and suffer from an inability to verify (Schwalbe 2010). Textbooks on project management will point to cases such as the bankruptcy of FoxMeyer Drug in 1996 due to an IT project that had scope problems (James 1997 and Scott 1996). McDougall cites a \$170 million project failure by McDonalds Restaurants in 2001 due to scope problems (McDougall 2006).

Weill and Broadbent have stated that projects are late sometimes due to specification changes, or new business needs that occur during the project (Weill 1998). This event, called scope creep, impacts the other areas that management tracks for status reporting. The criteria that are more readily measured by objective criteria (time, cost, and resources) are directly impacted by scope creep

(PMBOK 2008). The uncontrolled changes of scope creep add costs of which a customer might not approve, delay schedules, and reroute critical resources.

The IT industry is full of examples of scope creep. A Google search of the term "project scope creep" produced over 4 million hits. A quick review of just a fraction of these web sites demonstrates a common assumption: that a project manager knows exactly and precisely the scope, and that the problem is that the scope changes or grows. This is a questionable assumption. Fleming and Koppelman, major advocates of the deterministic Earned Value model, admit that "earned value accurately measures project performance, but must assume that scope definition is adequate" (Fleming 2010). Many sites are devoted to advice about managing scope through a change control process, a respected technique, but this assumes that the scope is well defined, and that the changes are recognized. In reporting project status the ascertaining and reporting of scope status is critical, and yet lacks a clear and measureable standard. Stakeholders and executives have difficulty making decisions based on vague, subjective, and imprecise inputs. To put it simply, scope is fuzzy. Scope and the corresponding set of requirements are a collection of words describing an end product, and whether or not the deliverable meets the requirements can be open to interpretation.

Computational Intelligence Background

Computational intelligence (CI), implemented in a variety of soft computing techniques, has allowed the automation of the handling of vague and imprecise data. Computational intelligence offers a revolutionary set of tools capable of responding to fuzzy, inaccurate inputs. This research envisions that these tools and techniques can be effectively applied to project status assessment. This study concentrates on Information Technology (IT) projects, in particular the scope constraint, because of the inherent lack of a measure for scope. The IEEE Computational Intelligence Society defines CI as a number of core technologies, among them fuzzy systems, neural networks, evolutionary programming, and genetic algorithms (IEEE 2011). These technologies build intelligent systems to help with complex problems in which the information and data are vague, approximate, and uncertain. For this research computational intelligence will focus on fuzzy logic as applied to project status. In order to put a reasonable boundary around the subject, only project scope status will be evaluated.

Lotfi Zadeh proposed the concept of fuzzy variables that are linguistic in the 1960's. For project cost these linguistic variables might be (costs = {over, on cost, under}) (Li 2006). Fuzzy systems can replicate human decision making by handling vague data, to the point of coping with noisy and/or missing data (Yen 1999). McNeill in his text Fuzzy Logic explained the difference

between fuzzy logic and probability by asserting that with fuzzy logic "you have all the information you need. The situation itself makes either Yes or No inappropriate. ... Fuzzy answers...handle the actual ambiguity in descriptions or presentations of reality" (McNeill 1994). To this McNeill adds three characteristics of fuzziness: (McNeill 1994)

- Word based, not number based.
Example: "hot", not 85 degrees
- Nonlinear and changeable
- Analog (ambiguous), not digital (yes/no)

Zimmermann expanded upon Zadeh's description of fuzziness as that of possibility, with the idea of a possibility distribution. Zimmermann's example is that a fuzzy set $F_{\sim} = \{ (1,1), (2,1), (3,0.8) \}$ has a possibility distribution such that 0.8 is the possibility that X is 3 (Zimmermann 1996). The possibility distribution thus allows for something to be both "true" and "fairly false" at the same time. This concept is the basic question that will be asked of the experienced project managers in this research: is it possible that the measurement of scope is inherently fuzzy, and therefore does it make more sense to use tools and techniques that can capture the fuzziness associated with scope status.

Application of CI to Project Status

Some authors have suggested that in spite of objective and measureable numbers in cost and time constraints, there can be fuzziness in the interpretation of those numbers. Li, Moselhi, and Alkas proposed a forecasting method for cost and schedule constraints using Fuzzy Logic to compensate for the variability found on construction projects. They looked at four different, generalized methods to forecast project status (Li 2006). The first were stochastic methods that assumed each unit of work has a mean and standard deviation, but according to Li, et al, these methods are weakened by variability in costs per reporting period. The second methods were deterministic, such as earned value. The third method that they looked at was social judgment theory based, using human judgment in lieu of mathematic methods. The last method was their proposed use of fuzzy logic for project forecasting and status (Li 2008).

Other researchers have applied computational intelligence tools to project management for schedule and time control. Jin-Hsien Wang and Jongyun Hao proposed a Fuzzy Linguistic PERT (Program Evaluation & Review Technique) to replace stochastic methods that use means and standard deviations. They assert that too much data may be needed to obtain the random variable distribution, so fuzzy methods are more applicable (Wang 2007). Wang and Hao expanded PERT/CPM (Critical Path Method) by storing each activity duration as a fuzzy set.

Blakegg, et al, have already analyzed what should be measured in projects, and at the same time they

acknowledge that warning signs of problems are "often unclear and imprecise" (Klakegg 2010). They describe *what*, but not *how* to measure the subjective constraints. While other researchers have proposed how fuzzy set theory can be integrated into project management across the time and schedule constraints, this work focuses on the scope constraint. Additionally, we will use a CI tool to capture scope status directly from experts in a more realistic, human friendly form. Once the status is captured, it can then be aggregated into an overall scope status for a project. Without an objective criterion such as currency spent or elapsed time, scope is difficult to measure. Fuzzy systems allow the capturing of this subjective data, and then the aggregation using recognized fuzzy set mathematics.

Methodology for Collecting Status

This study proposes to use computational intelligence (CI) tools, in particular alternative tools like fuzzy logic, to understand the status of a project's scope. This use of CI is in contrast to the more conventional bivalent logic, which Zadeh described as working well with exact numbers, intervals, and probabilities. Rather than the hard, crisp nature of bivalent logic, these CI alternatives have been sometimes labeled "soft computing."

Zadeh stated that "it is a common practice to ignore imprecision, treating what is imprecise as if it were precise" (Zadeh 2009). The computing power available in the 21st century allows for the implementation of the concepts that Zadeh called computing with words (Zadeh 2009). Given the imprecise nature of project scope due to the linguistic nature of requirements, it makes more sense to use fuzzy intervals and fuzzy sets to capture the essence of the status of scope.

In his acceptance speech when receiving the Ben Franklin award at Villanova University in 2009, Zadeh provided an analogy for fuzzy logic:

In bivalent logic, the writing/drawing instrument is a ballpoint pen. In fuzzy logic, the writing/drawing instrument is a spray pen—a miniature spray can — with an adjustable, precisely specified spray pattern (Zadeh 2009).

Zadeh has stated that a valid application of fuzzy logic is in the handling of imperfect information. At Villanova Zadeh went on to say that "imperfect information is defined as information which in one or more respects is imprecise, uncertain, vague, incomplete, unreliable, partially true or partially possible" (Zadeh 2009). This leads to the core concept that membership in a fuzzy set is a matter of degree. For project managers, when looking at the status of a given line item's scope using fuzzy logic, that status is allowed to be a matter of degree. In practical terms this means that the scope of an item can be of status

mostly yellow, and at the same time that same scope item can be of status *a little red*.

This study uses a fuzzy data collection tool proposed by Zadeh, colloquially referred to as the Z-mouse. This tool is a spray paint web gadget that implements Zadeh's spray paint analogy. Jose Barranquero and Sergio Guadarrama created the Z-mouse to gather fuzzy opinions, or perceptions as they call it, from users (Barranquero 2010). In their work, they give users an English language word and ask the participant to rate that word on a scale using the Z-mouse.

This study builds upon their prototype by evaluating the fitness of their Z-mouse concepts when applied to project management. Project managers are asked to rate the scope for a WBS activity on a scale that is words, not numbers. It is anticipated that the non-numeric scale will be quickly recognized and easy to use by experienced project managers. Figure 1 illustrates the Z-mouse web gadget using a non-numeric, linguistic scale.

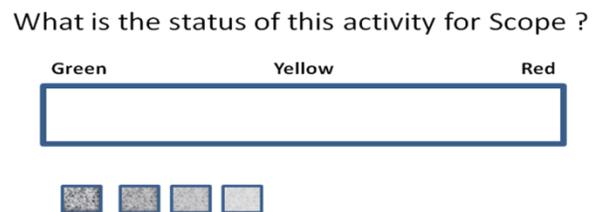


Figure 1. Linguistic scale for project scope

Barranquero and Guadarrama go on to state that the Z-mouse can be easily learned by non-expert end users (Barranquero 2010). This could lead to a design where the individuals doing the work of a WBS activity would input their opinions on the scope status, which would be passed on to the project manager and stakeholders. The scope status would be seen as a measurement that is analogous to cost and schedule measurements. Since errors in scope lead to errors in cost and schedule, the awareness of scope problems should contribute to early corrective actions, increasing project success.

In contrast to fuzzy systems, social scientists have used psychometric scales extensively in survey research. In Likert scales the survey participants are asked to select one number from a variety of choices. Many times these choices are an ordered scale, forcing the user to select one and only one value (Trochim 2006). The evaluator of a Likert scale survey can take advantage of the bipolar nature of this scheme, and apply conventional statistical tests, such as variance from a mean. Likert and other systems such as Thurstone scaling have strict rules.

One drawback from using Likert is that it cannot handle that people will perceive a given choice as falling into two categories simultaneously. Those models view this human tendency as a paradox, or a violation of the rules. A fuzzy

system allows that project managers might perceive the status as mostly yellow, with some modest amount of red. Having both statuses at the same time for one activity is an acceptable possibility in fuzzy systems. Another drawback to scaling systems is that a statistically valid number of participants are required in order to validate the data. In project management there might only be two or three participants working on a WBS activity, a number not amenable to conventional crisp probability.

This study gives participants a description of an activity and asks them to evaluate that activity for the status of the scope. Figure 2 gives an example that is illustrative of the types of questions in this survey.

Activity 1: Web Page Design

End users have requested changes to the web pages that should be relatively simple to accomplish. However, these end users are known to change their minds frequently.

- **Time Constraint:** the project is on schedule
- **Cost Constraint:** this activity is within the budget
- **Scope Constraint:** use the Z-mouse to input the status

What is the status of this activity for Scope ?

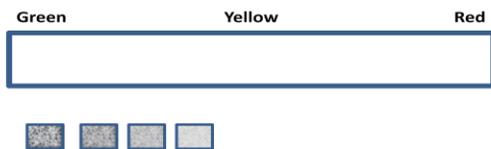


Figure 2. Sample project activity

Figure 3 is an example of a potential response using the Z-mouse. The individual inputting the status would select one of the four shades of grey from the pallet, and then paint the status bar where they think appropriate. If they want to indicate a lesser importance, then they would select a lighter shade of grey. Figure 3 would be an example of a project manager deciding that scope constraint was mostly yellow, yet leaning towards red.

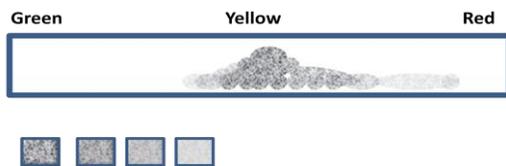


Figure 3. Project scope using the Z-mouse to spray paint the status of an activity as "mostly" yellow

It should be pointed out that these spray paint data points are converted to numeric values, and then evaluated using the strict mathematical rules of fuzzy sets.

Methodology for Aggregating Scope Status for an Entire Project

The next step is to aggregate the individual scope statuses for each activity into an overall status for a project. Since the origins are the fuzzy words (green-yellow-red), the proposed aggregation method would be an implementation of Zadeh's computing with words. Zimmermann offers three common methods to aggregate the individual inputs: COA (center of area), COS (center of sums), and MOM (mean of maxima) (Zimmermann 1996). This study will use COS to aggregate the fuzzy sets into the crisp value that will be reported at the overall status. Klir states that COS is the most common method to find a value that represents the overall conclusion. The COS calculation is based on recognized and accepted mathematics for fuzzy sets, which can be found in textbooks by authors such as Klir (Klir 1997). The COS solution finds the geometric centroid for the aggregated first moments, and then translates the solution value into a status.

Conclusion

Professional project managers have objective data for the time and cost constraints on their activities. With the introduction of an input tool to capture the status of scope, the measuring and reporting of subjective opinions of scope status can be done. The next step would be that each and every WBS activity would have a scope status that could be aggregated into an overall project scope status. Based on the gathering of this scope data it is expected that this would become the third constraint in a fuzzy system such as the one proposed by Li, Moselhi, and Alkas that only addresses cost and schedule. Since IT projects are unique and, thus have vague and imprecise scope requirements, it is believed that two questions will be answered in the positive by this study: 1) fuzzy logic can provide a tool for measuring scope of individual activities, and 2) the fuzzy scope can be aggregated into a meaningful project status.

References

- Barranquero, T. Guadarrama, S. 2010. IEEE World Congress on Computational Intelligence July, 18-23, 2010 Barcelona, Spain.
- Barnes, A. Hammell, R. 2008 "Determining Information Technology Project Status using Recognition-primed Decision Making Enabled Collaborative Agents for Simulating Teamwork (R-CAST)." Proceedings of CONISAR 2008, p.2.

- Benson, R. Bugnitz, T. Walton, W. 2004. *From Business Strategy to IT Action*. Hoboken, NJ: John Wiley & Sons. p.119.
- Cox, E. *The Fuzzy Systems Handbook, Second Edition*. Chestnut Hill, MA: Academic Press. p.15.
- Fleming, Q. Koppelman, J. 2010. *Earned value project management*. Newtown Square, PA: Project Management Institute, Inc. p.139.
- Dow, W. Taylor, B. 2008. *Project Management Communications Bible*. Indianapolis, IN: Wiley Publishing.
- Gido, J. Clements, J. 2009. *Successful Project Management, 4th edition*. Mason, OH: South-Western Cengage Learning.
- <http://www.google.com/#q=project+scope+creep&hl=en&prmd=ivns&ei=7NQjTei5HYK78gbXuqCRDg&start=280&sa=N&fp=7b989c6c17f79c85>, found 4 Jan 2011.
- IEEE Computational Intelligence Society statement of scope: http://iee-cis.org/about_cis/scope/
- James, G. 1997. "Information Technology Fiascos." *Datamation* vol:43 issue:11 p. 84.
- Klakegg, O. Williams, T. Walker, Andersen, B. D. Magnussen, O. 2010. *Early Warning Signs in Complex Projects*, Newtown Square, PA: Project Management Institute, Inc. p. 26-29.
- Levinson, M. 2009a. CIO Magazine "Recession Causes Rising IT Project Failure Rates." <http://www.cio.com/article/495306>.
- Levinson, M. 2009b. CIO Magazine "Common Project Management Metrics Doom IT Departments to Failure." <http://www.cio.com/article/440721>.
- Levinson, M. 2009c. CIO Magazine "Project Management The 14 Most Common Mistakes IT Departments Make " <http://www.cio.com/article/438930>.
- Li, J. Moselhi, O. Alkas, S. 2006. "Forecasting Project Status by Using Fuzzy Logic." *Journal of construction Engineering and Management*, Vol. 132, No. 11, November 2006. p. 1193
- Lucas, G. 1999. The San Francisco Chronicle "Computer Bumbling Costs State \$1 Billion" <http://www.sfgate.com/cgi-bin/article.cgi?f=/c/a/-1999/02/18/MN86384.DTL>
- McDougall, P. "8 Expensive I.T. Blunders," *InformationWeek* October 16, 2006
- McNeill, F. Martin. Thro, E. 1994. *Fuzzy Logic*. Boston, MA: AP Professional. p. 6.
- Project Management Institute. 2008. *A Guide to the Project Management body of Knowledge (PMBOK Guide)*, Fourth Edition. Newtown Square, PA: Project Management Institute, Inc.
- Richardson, G. 2006. *Readings in Information Technology Project Management*. Boston, MA: Course Technology. p. 115
- Schwalbe, K. 2010. *Information Technology Project Management*. Boston, MA: Course Technology. p.197.
- Scott, L. 1996. "Troubled FoxMeyer unit files Chapter 11." *Modern Healthcare*, Vol. 26, Issue 36.
- Snow, A. Keil, M. 2001. "The Challenge of Accurate Software Project Status Reporting: A Two Stage Model Incorporating Status Errors and Reporting Bias." *Proceedings of the 34th Hawaii International Conference on System Sciences*.
- Snow, A. Keil, M. 2002. "A Framework for Assessing the Reliability of Software Project Status Reports." *Engineering Management Journal*, June 2002. p. 21.
- Trochim, W. 2006. <http://www.socialresearch-methods.net/kb/scallik.php>.
- Weill, P. Broadbent, M. 1998. *Leveraging the New Infrastructure*. Boston, MA: Harvard Business School Press. p.208
- Yen, J. Langari, R. 1999. *Fuzzy Logic: Intelligence, Control, and Information*. Upper Saddle River, NJ: Prentice Hall.
- Zadeh, L.A. 1973. "Outline of a new approach to the analysis of complex systems and decision processes." *IEEE Transactions on Systems, Man, and Cybernetics*, 3, pp.28-44.
- Zimmermann, H.-J. 1996. *Fuzzy Set Theory and Its Applications, third edition*. Dordrecht, NL: Kluwer Academic Publishers.

Discovering Causality in Suicide Notes

Using Fuzzy Cognitive Maps

Ethan White

Applied Computational Intelligence Laboratory
University of Cincinnati
Cincinnati, Ohio 45221
whitee4@mail.uc.edu

Lawrence J. Mazlack

Applied Computational Intelligence Laboratory
University of Cincinnati
Cincinnati, Ohio 45221
mazlack@uc.edu

Abstract

An important question is how to determine if a person is exhibiting suicidal tendencies in behavior, speech, or writing. This paper demonstrates a method of analyzing written material to determine whether or not a person is suicidal or not. The method involves an analysis of word frequencies that are then translated into a fuzzy cognitive map that will be able to determine if the word frequency patterns are showing signs of suicidal tendencies. The method could have significant potential in suicide prevention as well as in other forms of sociological behavior studies that might exhibit their own identifying patterns.

Introduction

Computationally recognizing causality is a difficult task. However, discovered causality can be one of the most useful predictive tools. This is because understanding causality helps in understanding the underlying system that is driving the causal relationships [Steyvers, 2003]. One utilitarian outcome that causality provides is the prediction of human behavioral patterns either in a broad domain such as nations or religious groups or in groups of individuals. One such group of individuals that can be analyzed is those people who commit suicide. Suicide is one of the top three causes of death for 15-34 year olds [Pestian, 2010]. Therefore, suicide is a very pertinent topic for study. One of the ways to study suicide is to study the notes that were left behind by the ones who committed suicide [Leenaars, 1988]. Using these notes, a linguistic analysis can be performed that causal relationships can be extracted from. However, describing the causalities involved is difficult to do quantitatively, so previous causal analysis has mostly been qualitative. In contrast, this work considers causal suicide analysis using a quantitative method. This work uses fuzzy cognitive maps to discover and isolate root causal relationships based on words in suicide notes from people who take their own lives.

The long term goal of our work is to discover patterns within written material that may indicate causal relationships in human behavior. The focus of this work is based on patterns in the frequency of words as opposed to grammatical structure. The objective of this research, that will be the first step toward the long term goal, is to analyze a

specific human behavioral pattern, i.e. suicide, in the form of suicide notes in a way as to contrast it with non-suicide notes. The central hypothesis is that human behavioral patterns can be extracted from word frequencies in written material, and that these patterns can be represented using fuzzy cognitive maps.

To test the central hypothesis and accomplish the objective of this research, three specific aims are pursued:

Discover and extract patterns in written material in order to produce an initial fuzzy cognitive map to describe causality

The first step toward this aim is the analysis of suicide notes according to the working hypothesis that the causal patterns can be discovered by finding word frequency patterns. This will be done for both the original written material and a set of the same data with spelling corrections.

The reason for making the distinction between spelling errors and corrected errors is that misspellings in suicide notes could have patterns that are exclusive to such writings as opposed to other written material. If, on the other hand, it turns out that misspellings are not significantly tied in with either suicide or non-suicide notes then notes that have had their spelling corrected will not be considered in the analysis. Only the original notes will be used in developing the fuzzy cognitive map.

The second step is an analysis of non-suicide notes based on the same working hypothesis. Again this has to be done for both the original and corrected versions of the data. Once this analysis has been done, the frequency patterns of the data will be used to produce an initial fuzzy cognitive map for analysis in aim two.

Perform rigorous testing on the fuzzy cognitive map on the original data and make adjustments where necessary

Once the first aim has been accomplished and the patterns discovered are converted to a fuzzy cognitive map, then testing must be performed in order to ensure that the map will be able to tell the difference between the suicide notes and the non-suicide notes that were originally tested.

Again, as in the first aim, this must be broken up into testing the original data and the data with spelling corrections. These must be further divided up into testing groups of notes and testing individual notes. This will show how sensitive the fuzzy cognitive map is to the amount of data available. Once the map has been altered to a point where the results are acceptably reliable then aim three will be performed.

Perform rigorous testing on the fuzzy cognitive map based on different material

Once the cognitive map is able to distinguish between the two original data sets used to build it, the map must be able to find the patterns in different written sources to make sure that it can work on a variety of writing. This is also broken up into two steps as in aim one and aim two.

The first step is using the misspelled words as written, and the second step is the corrected words. Also, as in aim two, this must be tested for both individual notes and for groups of notes to determine if the amount of data affects the outcome. If satisfactory results have not been attained, then the new data must be factored into the fuzzy cognitive map until the results are reliably accurate. Then aim three must be performed again using a different source of data.

Creating the Initial Fuzzy Cognitive Map

Extracting patterns in general categories from written material

The first step to accomplishing the first aim and developing the initial fuzzy cognitive map was to analyze the written data of both suicide notes and non-suicide notes. The group of suicide notes that were studied consisted of notes written by those that successfully committed suicide. The non-suicide notes consist of three sets that are approximately the same size as the number of words used in the group of suicide notes.

All three sets are taken from informal sources, i.e., each source represents a natural human form of communication as opposed to magazine articles, professional journals, and other such written works. The first set is a collection of various product reviews extracted from www.Amazon.com. This sample set was taken from a number of different products over a range of different ratings that ranged from the highest rating of five stars to the lowest rating of one star. The second set is a collection of notes from a private blog at archbishop-cranmer.blogspot.com. This is different from the amazon.com data because it represents an individual instead of a group of people. The final set comes from a political website called www.biggovernment.com. This set contains more specific topics than are covered by random product reviews on amazon.com and random notes from an individual.

All of the words were grouped into abstract general categories and sorted in order from most frequently used to least frequently used words. Each grouping is defined by

how dense they are by percentage compared to the entire dataset, i.e., how frequently each group is used in a given set of data. The categories used are references to self, others, financial terms, medical terms, religious terms, negative and positive words, and misspelled words. The densities of these categories are shown in Fig. 1.

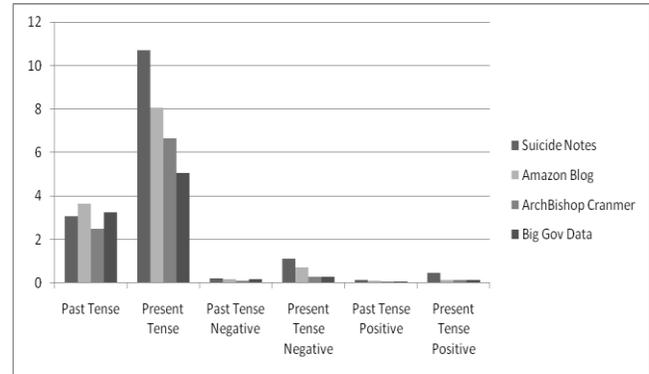


Figure 1. Word densities by percentage part 1

In addition to these categories, past tense and present tense words are also included along with their corresponding negative and positive references as shown in Fig. 2.

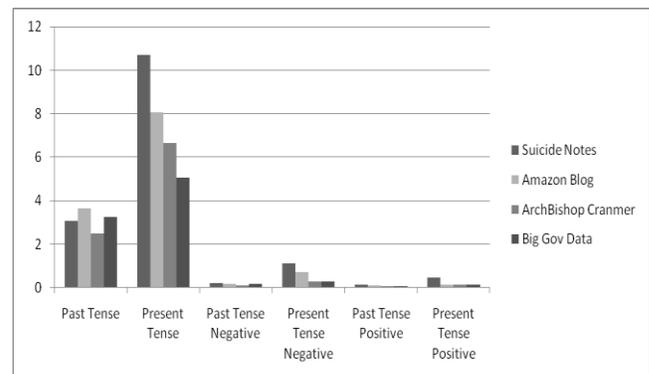


Figure 2. Word densities by percentage part 2

The results show that the greatest differentiation between the suicide notes and the non-suicide notes is found in three main categories that are references to self and others in fig. 1 and present tense in Fig. 2. Also, according to the data, there is not a significant amount of misspellings and even the small amount that is, does not show significant variation between suicide and non-suicide notes. Since the misspellings are not significant, they will not be considered in the analysis of the data. The three main categories are chiefly dominated by the set of suicide notes. This means that there would be no nodes in the fuzzy cognitive map that would push the final result toward a non-suicidal classification if it was analyzing a non-suicidal case. Therefore, the patterns have to be extracted on a word by word basis.

Extracting patterns from specific words in written material

The three best places to gather words that can provide varying reliable patterns are the groups for self references, references to others, and present tense. These groups contain the most references than any other kind and, therefore, the words in these categories are most likely to be found in a random set of notes to be analyzed and classified as suicidal or non-suicidal. The densities for these words, however, are not based on how many of each word is used in the entire dataset but rather on how many of each word is used in the group it occupies. Upon further analysis of the three groups, there were a number of words that proved to have either distinct suicidal influences or distinct non-suicidal influences. All words that had small percentages over all four datasets or did not vary significantly between suicide and non-suicide were removed from consideration. Fig. 3 shows the final results for references to self.

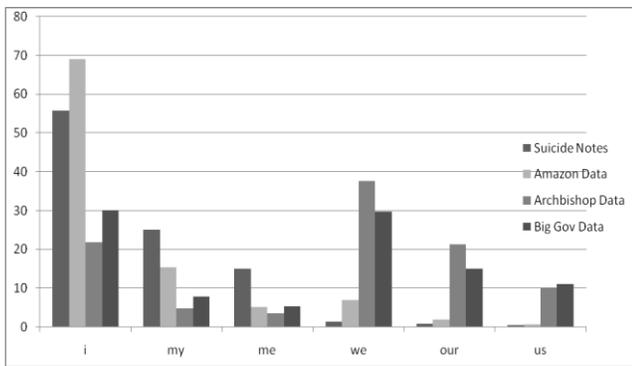


Figure 3. Word densities in self references by percentage

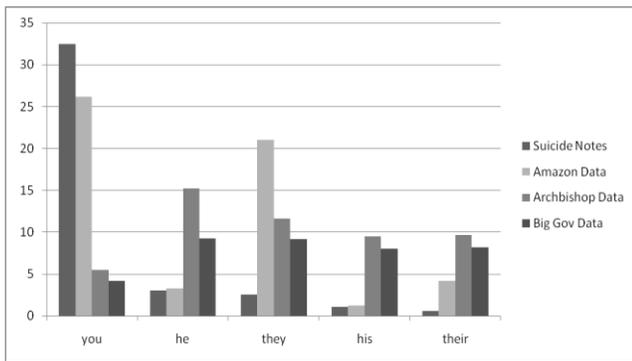


Figure 4. Word densities in others references by percentage

On average each word has a specific affiliation to either the suicide notes or non-suicide. However, the Amazon.com data shows definite anomalies in the words I, we, our, and us as compared with the other two non-suicide collection of notes. However, the apparent pattern is that suicide notes have more singular self references, i.e. I, my and me, while non-suicide notes seem to have more group self ref-

erences, i.e. we, our, and us. Fig. 4 shows the results for references to others.

Again, there is a definite pattern with suicide notes have a large amount of references to the word “you” and the non-suicide notes have larger references to “he”, “they”, “his”, and “their”. Again, the Amazon.com data shows anomalies being similar to the suicide data in the word “you” but showing a great deal more influence in the word “they”. The final results for present tense words is shown in Fig. 5.

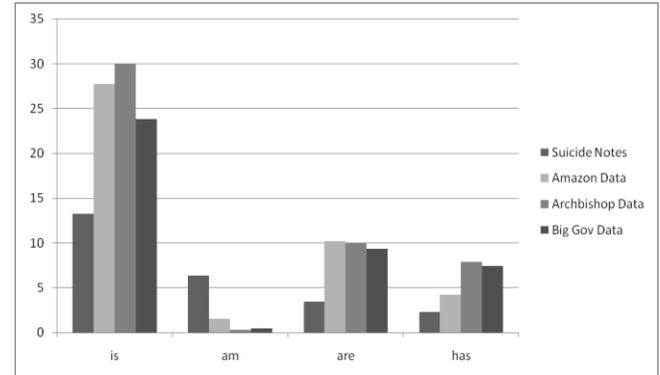


Figure 5. Word densities in present tense by percentage

In this group, the Amazon.com data acts similarly to the other non-suicide data except that the percentage for the word “has” is a little low, although not entirely problematic.

Developing the Initial Fuzzy Cognitive Map

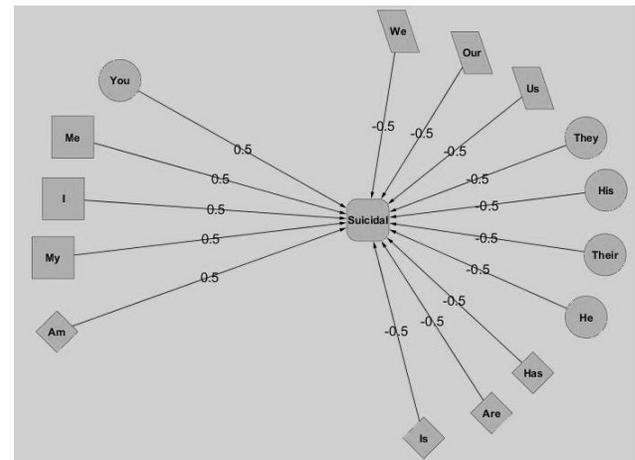


Figure 6. Initial fuzzy cognitive map

The fuzzy cognitive maps consist of a series of connected nodes that will represent the words being used from Fig. 3-5. These words will in some way connect to a suicidal node that will determine the classification of the dataset. The simplest graph that can be constructed, is for the

suicidal node to be central with all word nodes connected to only that one node as shown in Fig. 6.

The node roles in the graph are indicated by the shapes of the nodes. The square nodes are the words that are the singular self references. The parallelograms are words that are plural self references. The circles are words that are references to others. Finally, the diamond shaped nodes are present tense words.

Each of the edges has a weight between -1.00 and 1.00 that is attached to it to determine how much influence and what kind of influence a particular node has on the suicidal node. The nodes on the left of Fig. 6 are all the nodes that are associated with suicide notes and thus have a positive influence, while all the nodes on the right represent non-suicide notes and are therefore negative in their influence.

All of the initial edge weights are arbitrarily set to start at 0.5 or -0.5. This would be true if all nodes would have equal influences on the classification; these starting values are expected to change. However, by starting with these values, it can be determined whether or not the general structure of the map is good or bad.

Each of the nodes starts at a particular value between 0.00 and 1.00 and then the graph is allowed to iterate by a computer program until the graph reaches equilibrium or until enough time has shown that it will never reach equilibrium. If the graph has reached equilibrium, then the final value of the suicide node is examined. If the value is over 0.50, i.e. over 50%, then the graph has determined the dataset to be suicidal. If the value is under 50%, then the dataset would be non-suicidal, and if the value is at 50%, then the classification is uncertain.

The starting values of the nodes are determined by normalizing the data in the particular group, e.g. in Fig. 5, all four datasets would be normalized according to the archbishop result for the word "is". This means that about 30% is the new 100% which all other values are compared to within that group. Fig. 3 and 4 would have their own number for normalization. The starting number for the suicidal node is 0.00 because it is assumed that there is no initial influence from this node.

The final results for the fuzzy cognitive maps for each dataset were not entirely successful. The Amazon.com data was particularly unsuccessful because of its anomalies which made it similar to the suicide notes. This means that the nodes in the graph do not have the same influence. Therefore, in order to determine if this map structure can distinguish between the datasets correctly, a set of weights must be found that can find the dividing line. By using machine learning techniques (supervised learning), it was discovered that there is a set of weights which allows the fuzzy cognitive map to correctly classify each dataset. The graph with its final weights is shown in Fig. 7.

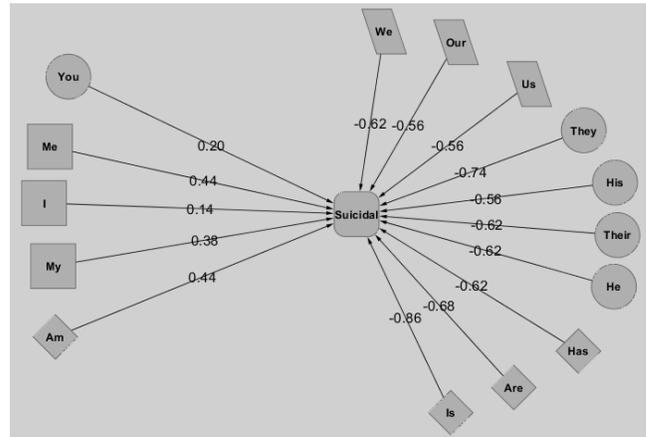


Figure 7. Final fuzzy cognitive map for testing

Testing the Fuzzy Cognitive Map

Now that a fuzzy cognitive map has been designed that can accurately classify the four datasets, this design must be tested against other collections of notes to see if the map can properly classify a random set of data.

General Category Testing

Three more datasets were used for testing. These consist of two sets of suicide notes and one non-suicide with each one only a fraction the size of the original four datasets. The first data set is a collection of suicide notes that contain some notes from the original suicide note collection as well as new ones. This was obtained from the website www.well.com/~art/suicidenotes.html and is labeled as suicide notes 2 in the analysis. The second set is a collection of suicide notes or the last words from famous actors, poets, and musicians labeled as suicide notes 3 in the analysis that was obtained from the website www.corsinet.com/braincandy/dying3.html. The final dataset is a collection of non-suicide notes from the private blog gregmankiw.blogspot.com.

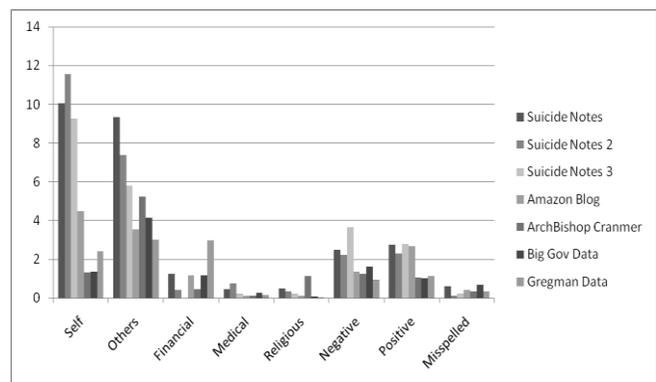


Figure 8. Results of all 7 datasets in general categories part 1

Before going straight into the word analysis, the results for the general categories should be compared with the original datasets. This is for the purpose of making sure that all of the datasets are following a predictable pattern. Fig. 8 and 9 show the results for each of the general categories.

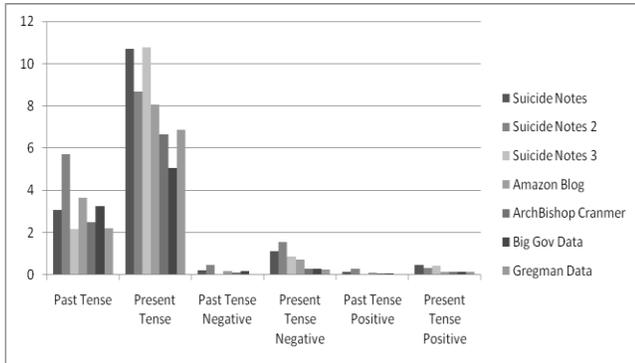


Figure 9. Results of all 7 datasets in general categories part 2

As can be seen from Fig. 8 and 9, the three new datasets follow similar patterns in both the suicide and non-suicide cases. Since there were no significant differences, then the specific word analysis could begin.

Specific Word Analysis

The final results for the word analysis are shown in Fig. 10, 11, and 12. As can be seen from the graphs, the three new datasets follow the same pattern for their respective classification with the exception of suicide notes 3 which produces some anomalies in the form of very large values in Fig. 12 for the words “is” and “are” which are very close to non-suicide patterns. Each of the new cases was normalized into the starting values for the nodes of Fig. 7. Each time, the fuzzy cognitive map accurately identified each dataset as either suicidal or non-suicidal. These findings suggest that this fuzzy cognitive map design is somewhat robust in that it was able to handle a random relatively small collection of suicide notes, i.e. suicide notes 3, and correctly identify them as such even with the non-suicidal like behavior that were found in Fig. 12.

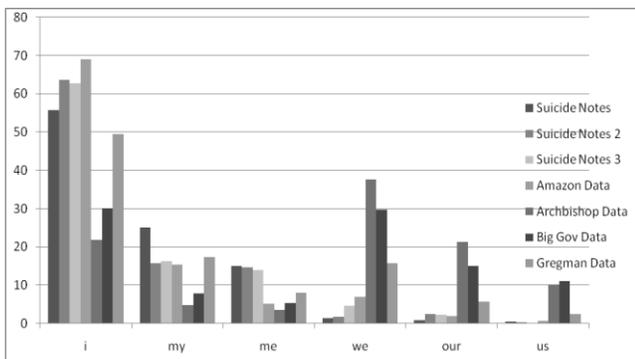


Figure 10. Word densities in self references by percentage

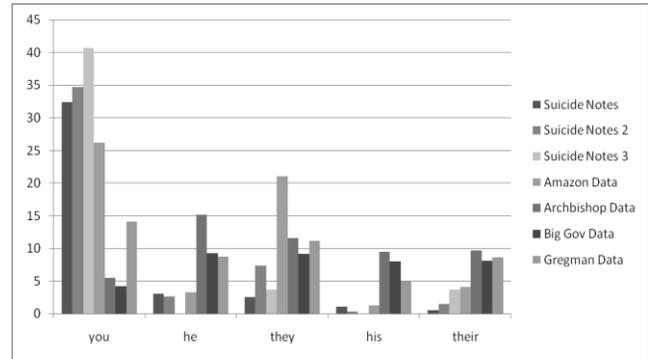


Figure 11. Word densities in others references by percentage

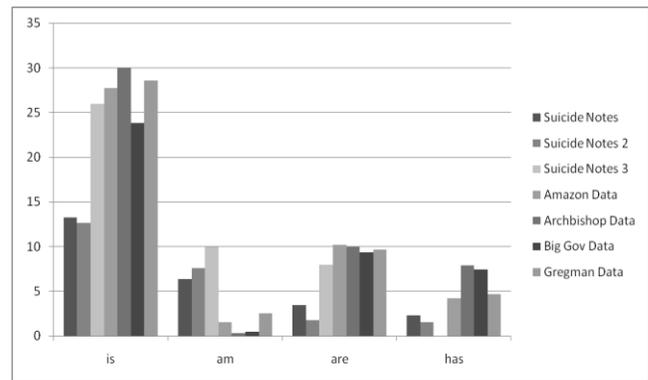


Figure 12. Word densities in present tense by percentage

Conclusion

The results of this research appear to provide strong evidence that it is possible to differentiate between suicidal behavioral patterns and non-suicidal patterns. Further testing must be done in order to ensure that this method can be used in all given situations. One such testing would be an analysis of suicidal ideation or intent to commit suicide [Barnow, 1997] which may or may not result in an attempted suicide. Also, further tests can be done from other collections of suicide notes as well as other sets of non-suicide notes.

This research is creative and original because it employs the use of fuzzy cognitive maps based on word frequencies in order to define human behavioral patterns. It is expected that the results of this research will further the understanding of causality and the prediction of human behavior. The broad application and positive impact of this work is a further development in the techniques for capturing causal relationships. Identification of causal relationships allows the ability to predict the consequences of actions from military strategies, governmental restructuring or societal rebuilding [Kosko, 1986] [Mazlack, 2010]. In the context of this research, fuzzy cognitive mapping is used to analyze writing and potentially to

predict suicide cases allowing possible intervention that could save lives.

References

Barnow, S. and Linden, M. 1997. Suicidality and tiredness of life among very old persons: Results from the Berlin Aging Study (BASE). *Archives of Suicide Research*: 171-182

Kosko, B. 1986. *Fuzzy Cognitive Maps*. Academic Press, Inc. vol. 24: 65-75

Leenaars, A. A. 1988. *Suicide Notes Predictive Clues and Patterns*. Human Sciences Press, Inc. Windsor Ontario, Canada.

Mazlack, L. August 31 – September 3, 2010. *Approximate Representations In The Medical Domain*. Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence.

Pestian J., Nasrallah H., Matykiewicz P., Bennett A., and Leenaars A. 2010. *Suicide Note Classification Using Natural Language Processing: A Content Analysis*. *Biomedical Informatics Insights*: 19-28.

Steyvers M., Tenenbaum J. B., Wagenmakers E., and Blum B. 2003. *Inferring causal networks from observations and interventions*. Cognitive Science Society, Inc.: 453-489.

This page is intentionally left blank.

Agent Systems and Evolutionary Algorithms

Chair: Alina Lazar

Robotic Dancing: Exploring Agents that use a Stratified Perceive- Decide-Act Cycle of Interaction

James Benze and Jennifer Seitzer

Department of Computer Science

University of Dayton, Dayton, OH 45469-2160

benzejaa@gmail.com, seitzer@udayton.edu

Abstract

An autonomous agent is any intelligent entity that engages in a perceive-decide-act cycle of interaction with its environment. In this paper we present a formalism using a stratified percept chain that renders an augmented interactive cycle. In particular, we identify two kinds of agents: a “lead” agent, who gains the percepts directly from the environment, and a “follow” agent, who gains its percepts from the lead agent (as well as the environment). In this work, the lead agent procures percepts from the environment, makes decisions based on this input, and passes these new orders in the form of secondary percepts onto the “follow”.

We exemplify this formalism in the application area of robotic dancing using models programmed in Alice, a programming IDE/language that facilitates the creation and visualization of autonomous agents.

Introduction

Autonomous agents achieve a level of intelligence by participating in a continual feedback loop to and from the environment. At the onset of the cycle, a percept is sent to the agent from the environment; the agent then internally makes a decision as to the action to perform next in the environment, and then lastly, the agent performs the action. In this work, I have expanded this cycle of interaction by involving two agent types: a “lead” who gets the percept directly from the environment, and a “follow” who gets its percept from the lead. This stratification of percept sources is new and affords many interesting challenges in multi-agent intelligent systems.

Stratified Percepts

In this paper we are defining “stratified percepts” to be a chain of percepts originating from the environment that are interpreted by an intelligent agent, and then received by another intelligent agent. We label the standard agent type receiving its percept from the environment as the “Lead”

agent. Additionally, we identify the “Follow” agent as the agent which gets its percepts from the Lead, and executes its actions in response to the Lead’s actions.

This kind of percept chain is applicable in many human interaction situations. For example, these percept chains are found in many kinds of partner dancing (Jitterbug, Lindy Hop, Waltz, Tango, etc.), however any chain-of-command domain houses many problems and applications that could benefit from the work and insights gleaned from studying percept chains. Standard organizational structure houses tiers of commands that are relayed down to lower members of the organization. For example, the Commander in Chief is the head of the American Armed Forces. Any decisions made by him must be passed down through the organization until it reaches the battlezone. Being able to simulate this effect would lead to more effective military simulations.

Obviously, when chaining stratified percepts together, one must be careful to keep the lines of communication clear. For example, consider the “Telephone game” played by children. One child at the head of a line of children whispers something in the ear of the next child. This child tries to whisper the same message to the next person, and so on until the entire line of children has been exhausted. Finally the last child announces the message aloud. This game becomes fun since rarely is message pronounced clearly between all children, and the message received by the last child is usually not even similar to original. In order to circumvent this problem, this project does not deal with a chain of intelligence agents, simply looking at the relationship between one “lead” and one “follow.”

Cycle of Interaction

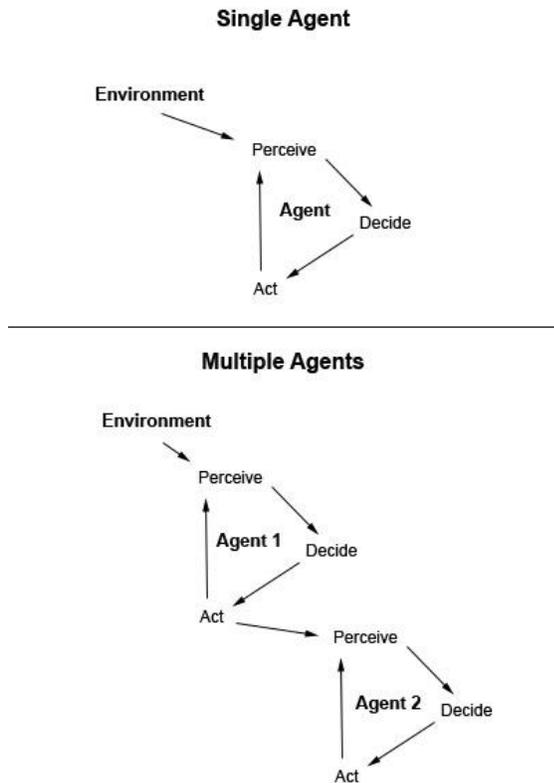


Figure 1: Single Agent vs. Multiple Agents

The actions of an intelligent agent are traditionally defined by a “Perceive-Decide-Act” cycle, as described above. When multiple agents are to be chained together, the cycles of one agent depends on the other, as demonstrated in Figure 1.

Implementation of System DANCER

We exemplify this formalism in the application area of robotic dancing using models programmed in Alice, a programming IDE/language that is geared towards an instructional computer programming and also facilitates the creation and visualization of autonomous agents. Two models of humans in Alice represent the Lead and the Follow. The two models are then made to dance in the style of East Coast Swing. The Lead interprets the “beat” given by the environment, and chooses dance moves to portray, and tries to demonstrate them to the Follow through body movement. The Follow then takes this body movement and interprets it so that she can do the correct move.

Alice

Alice was developed at Carnegie Mellon University in order to provide an easy environment to teach beginner computer science topics to introductory computer science students. However, Alice also contains an integrated graphics environment, allowing the easy placement of lighting, human models, etc with minimal effort from the developer. It was this factor that was crucial in our choice of Alice as the development environment.

One of the most challenging aspects in using Alice, however, is its lack of synchronization objects. Although Alice allows for many threads to run simultaneously (Alice is based on Java), it provides no method of protecting shared data. Because of this, creative ways had to be employed to circumvent race conditions.

East Coast Swing

East Coast Swing was chosen as our application area for several reasons: (1) the moves are relatively easy to model, (2) it is typically an introductory dance, and (3) the authors are both dancers and know East Coast Swing.

In the swing dances, such as East Coast Swing, one dancer is called the “Lead” (typically a male) and one dancer is called the “Follow” (typically a female). The Lead prepares the dance moves, and executes them at the given time. He should also make it obvious to the follow what he is doing. The Follow does her best to interpret the information given to her by the Lead (this information is also called a “lead”). This dance is therefore an excellent representative of a stratified percept chain, since it inherently requires a lead and a follow.

System Design of DANCER

In the Alice architecture, everything is contained within a global class called “World”. The World, in this project, acts as the environment, and provides the original source for the percepts.

Contained inside the world are the two intelligence agents, once again designated the Lead and the Follow. Each of these two classes are created with the Alice he/she builder, and is constructed of many smaller classes, each representing a body part (forearm, neck, etc). This allows for the animation of individual body parts, so that the follow and lead are each able to move in a dancing fashion.

The Directional Light and the Bedroom classes are relatively unimportant. The directional light exists to give a visual indication to the user of the beat created by the world. The Bedroom is pure decoration, and was added so that the Lead and the Follow would be on a wooden floor

(dancing is not often done on grass or sand, the two default ground textures for Alice worlds).

Finally, there are many “dummy objects” used in the system. These are invisible placemarkers that contain only a location and orientation (yaw, pitch, roll). These are used as markers for the Lead and the Follow, and provide points of reference for their movement.

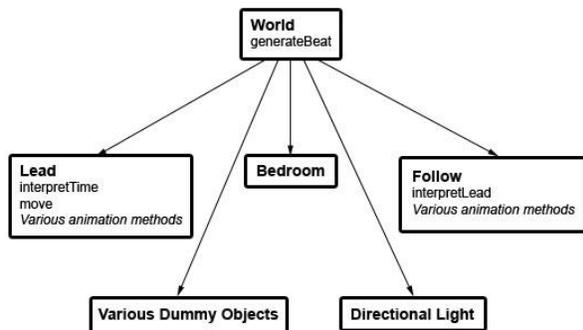


Figure 2: Classes in the East Coast Swing Environment

The East Coast Swing Environment

The Lexicon of Moves

In order to execute dance moves, both the lead and the follow agent had to be programmed with a vocabulary of movements that could be performed. Both the lead and the follow agents were programmed with the ability to perform the following dance moves:

- East Coast Basic
- Inside Turn
- Outside Turn
- Tuck Turn
- Repeaters (a variation on a tuck turn)

By programming these moves into the library each agent would be able to visually demonstrate its decision based on the percepts it received.

The System Algorithm and Environment

The purpose of the environment in this project is to provide the music for the lead to interpret. However, one of the limitations of Alice is a limited ability to interpret music in the project. Although an audio file could be played in the Alice environment, there was no way for any object to intelligently interact with it.

As such, a substitute for the music had to be found. Instead of playing music in the background, a flashing light

was used instead. This can serve the same purpose as a musical beat, since a musical beat is just a repeated auditory impulse. A repeated visual impulse creates the same effect but through a difference sense. Since one sense is as good as any other for this project, a flashing light was deemed to be an acceptable substitute.

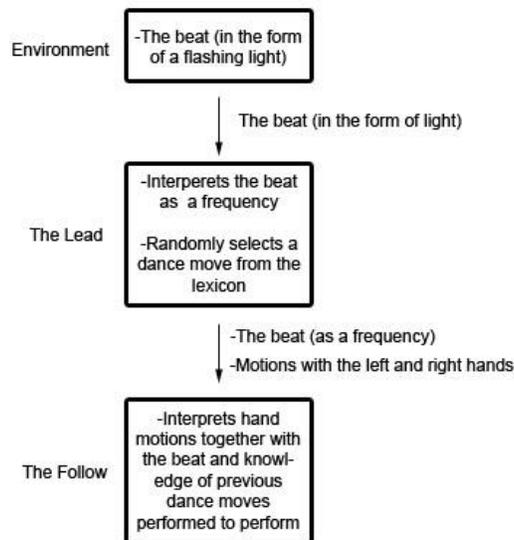


Figure 3: System Algorithm

Lead

The Lead agent has two primary responsibilities: to interpret and act upon the information from the environment, and to provide a clear percept to the Follow dictating her actions.

Clear interpretation of the beat of the music can be challenging for many humans. In our system, the Lead uses Alice’s internal timing function to record the time of each beat. By viewing the difference between each time, the Lead can accurately predict when the next beat would occur. This is essential in determining the speed in which to dance.

One problem was occurred due to the lack of synchronization ability in Alice. Due to other system processes running on the processor, there exists a discrepancy between the pulses of light and the time that they are recorded by the Lead agent. However, when no other projects were actively running on the system, the extra time variation was only about 0.06 to 0.09 seconds. By subtracting 0.1 seconds from each recorded beat, and by keeping the number of processes low, the recorded beat by the lead is at or slightly below the beat provided.

It was determined that having a recorded beat less than the provided beat was preferable than having the recorded beat to be greater than the provided beat. When the

recorded beat is less, the lead can simply pause, and wait for the action to “catch up”, whereas if the lead was too slow, he would simply be slower and slower until he has fallen noticeably behind.

To determine the specific move that the lead executes, the lead simply uses Alice’s random number generator, and randomly chooses a move. It then passes on this percept to the Follow.



Figure 4: DANCER preparing a turn

Follow

The follow program receives its variables through a function call. There are four different variables that are passed to the function as forms of “leads”: The lead from the left hand, the lead from the right hand, the beat of the music, and the distance between the two agents. The first two inputs should be obvious. These are the physical connect between the head and the follow. With is left and his right hand, he provides motions to direct the follow. The beat parameter is slightly more subtle, but in leading, the Lead pulses slightly, and in this way can transfer his knowledge of the beat to the Follow.

The distance parameter represents the translational velocity that the lead transfers to the follow. Doing certain moves the lead doesn’t just cause the follow to spin, but causes her to translate as well. The distance parameter represents this type of lead.

The follow uses a nested if statement to determine further course of action. She also takes her current momentum into account while determining further courses of action. The momentum is a variable set from previous moves.



Figure 5: DANCER during a turn

Lead-Environment Lag

As mentioned before, a lack of priority functions in Alice causes the timer function not be entirely accurate. As such, the timer between the lead and the environment had to be adjusted with a 0.1 second reduction in time to account for this. In addition, causing the processor to work a large number of tasks will cause this timer to displace even more. As such, the program should only be run in parallel with as few other programs as possible. No lag exists between the Lead and the Follow, since no system clock is used, only a function call.

Multiple Threads Between Lead-Follow

Originally, the plan was to have the Lead and the Follow run in separate threads, and for the Lead to send some sort of signal containing the leads to the Follow in order to start her motion. However, the lack of synchronization elements available in Alice made this unfortunately impossible. There was too much lag between the interactions between the two agents, which caused either too great of a delay in the Follow’s actions. As such, a simple function call was used to pass percepts between the two agents.



Figure 6: DANCER during the East Coast Basic

Conclusions and Future Work

An example of an interaction between two hierarchical agents was successfully modeled between the Lead and the Follow. However, many of the challenges have inspired us to probe further in future work as we describe here.

Clearer Communication between Agents

Currently, all percepts are passed directly and digitally between the Lead and the Follow, guaranteeing clear and precise communication. However, in real world scenarios, the communication between agents may become unclear. An example of this is perfectly clear in the Telephone game mentioned earlier. The specifics of a whispered message will become garbled over time, since one is difficult to hear.

Because of this potential problem methods should be implemented so that the Follow will approximate the closest action based on the decisions given. The other agents should be able to gracefully recover from a mistaken decision.

Recovery After a Delayed Response

One of the problems experienced in programming this project was that the Follow would sometimes have drastically delayed responses from the Lead. This could cause a desynchronization between the two Agents. Methods should be employed so that the Lead agent could gracefully recover from this separation.

Multiple Inputs

In this situation, each agent has a clear superior: The Lead's actions are governed by the environment and the Follow's action are governed by the Lead. However, in many situations, the Agents must receive input from multiple sources. For example, what if the Follow was able to receive input from the environment as well as the Lead? This input could be constructive (the Follow using the input musical beat in addition to the beat received by the lead to more accurately dance), or destructive (A fire alarm sounds, and the Lead ignores it. Does the follow obey input from the Lead and dance, or from the Environment, and leave the room. Both types of input will have to be considered.

Whisper-Down-the-Lane Agents

Here, an agent relationship was successfully demonstrated between a single Lead agent and a single Follow agent. This interaction could be expanded to chain of lead-follow Agents. Care would have to be taken to further reduce the risk of the other problems listed here.

Summary

Although this project accurately can portray a stratified percept chain, further testing and modeling is required should be performed in order to more accurately and effectively model this style of relationship between intelligent agents. The temporal aspect of decision making is much more important in hierarchical agent relationships, and methods of communication must remain remarkably clear in order for lower agents on the hierarchy to remain effective. Experimenting with ways to recover from these kinds of errors will vastly improve the robustness of hierarchical agents and will allow us to more accurately understand this style of interaction.

References

Alice.org, Available: <http://alice.org> [Accessed: March 22, 2010].

Alice—Project Kenai, Available: <http://kenai.com/projects/alice> [Accessed: March 22, 2010].

W. Dann, S. Cooper and R. Pausch, *Learning to Program with Alice*, Upper Saddle River, New Jersey: Pearson Education, Inc. 2006.

LEGO.com MINDSTORMS: Home, Available: <http://mindstorms.lego.com/en-us/Default.aspx> [Accessed: March 22, 2010].

Using a Genetic Algorithm to Evolve a D* Search Heuristic

Andrew Giese and Jennifer Seitzer

University of Dayton

gieseanw@gmail.com, seitzer@udayton.edu

Abstract

Evolutionary computation (EC) is the sub-discipline of artificial intelligence that iteratively derives solutions using techniques from genetics. In this work, we present a genetic algorithm that evolves a heuristic static evaluation function (SEF) function to be used in a real-time search navigation scheme of an autonomous agent. This coupling of algorithmic techniques (GAs with real time search by autonomous agents) makes for interesting formalistic and implementation challenges. Genetic evolution implies the need for a fitness function to guide a convergence in the solution being created. Thus, as part of this work, we present a fitness function that dictates the efficacy of a generated static evaluation function. In this work, we present algorithmic and formalistic designs, implementation details, and performance results of this multi-layered software endeavor.

Introduction

The A* Algorithm is a greedy best-first search algorithm that is used to calculate an optimal path between two points, or nodes, on a graph (Hart et. al 1968). The algorithm can be adapted to a run in real-time by way of restarting execution as new environment information becomes available, called a Dynamic A* (D*) search. The A* search uses a static evaluation function (SEF) that uses heuristics to find a path of state transformations from a start state to a goal state. The SEF assesses the merit of each child state as it is generated by assigning it a numeric value based on information about that state. The score allows the A* to direct its search by prioritizing the expansion of child nodes that could potentially expand into a goal state while neglecting child nodes that are less likely to lead to a goal.

In a real time environment, information about the actual goal state is unavailable and unobtainable for any given iteration of the search for an agent (because it is out of range of the agent's sensors). Therefore, the SEF must direct the search to the most appropriate state that anticipates system information as it becomes available. In our work, the SEF seeks to maximize some aspects of an agent's state while minimizing others. By evolving a weight on each "aspect-variable", we are able to create offline a highly effective SEF that can predict obstacles and challenges that occur in real time during the execution of D*. In this paper we present the offline pursuit of using a genetic algorithm as a mechanism to evolve an optimal SEF to be used in the real-time execution of A*.

Additionally, we use the simulator that will eventually benefit from this optimized SEF to provide feedback in the evolution process. That is, the simulator serves as the fitness function for the evolving SEF.

This work is novel in that it combines techniques of evolutionary computation using genetic algorithms and the use and refinement of a heuristic for the D* algorithm. There are many applications of genetic algorithms in diverse domains such as bioinformatics (Hill 2005), gaming (Lucas 2006), music composition (Weale 2004), and circuit optimization (Zhang 2006). Additionally, work in D* has been studied and developed in theory (Ramalingam 1996) as well as specific applications such as robotics (Koenig 2005). We are using the all-inclusive examination that genetic algorithms affords us to find the perfect (or near perfect) heuristic function for a derivative of the very traditional AI search, A*.

The A* and D* Algorithms

In this work, the evolution of a static evaluation function using a genetic algorithm is applied to an autonomous agent operating in an environment provided by Infinite Mario Bros., an open-source, faithful recreation of Nintendo's Super Mario World. The agent (Mario) uses a realtime execution of an A* search, called D*, to direct its movement through the environment to ultimately reach the goal (the end of the level). Mario may use information about what is currently visible onscreen, but beyond that nothing is known, making a calculation of an actual path to the goal impossible. Therefore, the SEF of the D* must direct Mario towards states that are on the path to the goal.

Mario has a total of thirteen distinct action combinations that allow him to negotiate the environment. These are move left, move right, duck, and two others—jump and speed—that can be used in combination with the other actions and each other. Jump allows movement along the y-axis, and can be used in combination with right, left, and duck. Speed allows for faster movement left or right, and higher jumps. This means that from any state, there could be up to thirteen child nodes. Since the agent must operate at 24 frames per second, the agent is allotted approximately 40 milliseconds to perceive its current state, decide what to do next, and return a chosen action. With up to thirteen child nodes from any node in the search tree, any algorithm that decides what Mario is going to do next must do so quickly and efficiently. A "brute force" approach that analyzes all possible children was infeasible given

available computing machinery, and therefore a dynamic D* search is more appropriate.

The SEF of a D* search uses information about a state to direct the search efficiently. For Mario, much information is immediately available from each percept provided by the environment. This information includes the position of Mario, the amount of damage Mario has taken, the positions and types of enemies onscreen, and position and types of obstacles onscreen. Other information can be tracked over time, like number of kills, X velocity, Y velocity, coins collected, time remaining, etc. The task of our system was to discover what effects, if any, the values of these variables should have on the valuation performed by the SEF of a node in the search graph. A high value for a variable might proportionally increase the cost to transition to that state, or conversely could proportionally decrease the transition cost.

The System

In 2009 and 2010 Julian Togelius of the ICE-GIC held a competition for entrants to create an autonomous agent (bot) that would play Markus Persson's Infinite Mario Bros. the best. "Best" in this sense means the distance a bot could travel within a given level and time limit. If two bots finished a level they were awarded equal scores, but if neither finished, the bot that travelled furthest was deemed better. In both iterations of the competition, the same bot was victorious. This bot was written by Robin Baumgarten (Baumgarten). Robin's bot used a D* search coupled with an accurate method for expanding child nodes, and a human-generated static evaluation function for the D*. Our system is a heavily modified version of Robin's, with the majority of the A* rewritten for legibility and efficiency while the means to produce child nodes was mostly preserved.

Every 24 frames, the environment provides the agent with a percept that includes the locations and types of all sprites on the screen, including Mario. The agent must return an action to the environment that the environment then effects upon Mario. For each percept received, the agent runs an A* search for 39ms or until the agent has planned out to 15 levels of the search tree. The agent keeps an internal representation of the world, and tracks Mario's x and y velocities among other things not provided by each percept.

After ensuring that its internal representation is consistent with the environment-provided one, the agent begins an A* search from Mario's current position and velocity. Children are generated by considering which actions are available to Mario at any node. That is, a child state reflects where Mario would be and how fast he would be moving if performed action A from node M. (Figure 1) A child state also informs the search of whether Mario would take damage, die, kill an enemy, collect a coin, etc. upon

performing action A from node M. A static evaluation function provides weights on Mario's X position, X velocity, Y position, Y velocity, Mario's damage taken, whether Mario is carrying a shell, Mario's X position multiplied by X velocity, and Mario's Y position multiplied by Y velocity. These weights are values between -1 and 1. After multiplying weights to their associated state variables, the sum of products forms the final SEF score for that node.

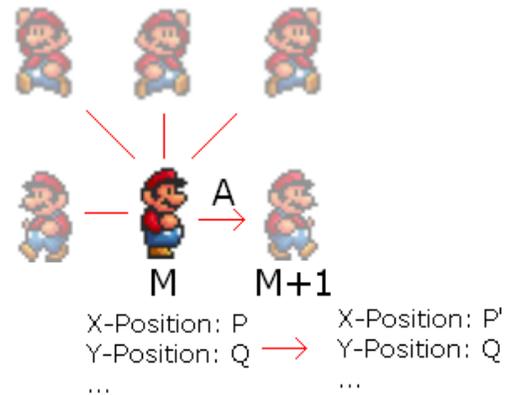


Figure 1

This SEF score is an estimation of the amount of work required to reach a goal state from the current node, and as such nodes with lower SEF scores are preferable. As an A* algorithm dictates, the level of the search tree at which the node was discovered is also added to the score. This is the "greedy" part of an A* search where not just a solution is desired, but the best solution. For Mario, the cost to transition from one state to another is uniform; all neighboring states have the same arc cost to travel to a neighbor. Adding the sum of arc costs into the SEF score for a node is a means by which the "work" required to reach a node in the graph is represented, so that if the same node is reached by two separate paths, the shortest path is favored. Since the D* search operates in a partially observable world, an admissible heuristic is difficult to discern, hence the motivation for a Genetic Algorithm to search for the optimal weights to apply in the SEF.

During the D* search, children nodes are generated from the current state of the agent. Generated children are placed on an open list sorted from lowest to highest scores. The child with the lowest score is taken from that list, and its children are generated. This process repeats until the agent has searched for 39ms or has searched 14 states (empirical number), at which point it returns the action that leads to the most optimal path for the current available information.

The values for the weights used in the agent's SEF mentioned above are deemed to be "unknown" to the system, and are provided via parameters supplied by an external entity, in this case a Genetic Algorithm. The

genetic algorithm is implemented as defined in (Russell and Norvig 2003). The chromosome being evolved is an array of 8 floating point values, each between -1.0 and 1.0. The mutation rate was 1%.

Each generation of chromosomes was tested for fitness by running a simulation on a training level where the agent used the chromosome's genes as the weights on state-variables evaluated by the SEF in a D* search. The fitness of the chromosome was a summation of Mario's distance travelled, and if he completed the level, also the remaining time Mario had to complete the level. A higher fitness score indicates a better, or more fit, chromosome. This is in contrast to the Static Evaluation function where a lower score indicates a more ideal state.

The test level that each bot was scored on had a variety of characteristics. The most important of these is that the level was short. As each chromosome needed to be used in an actual bot, a single fitness test could last upwards of a minute even if the bot could finish the level successfully. A short level guaranteed that if a bot was going to finish a level, it could do so without much time spent. The second characteristic of the level was an imposed time limit. This time limit places an upper bound on the possible time a bot could spend in a level. Slow bots, bots that stood still, or bots that got stuck therefore all required a maximum of N seconds to evaluate.

An ideal level must also contain challenges and obstacles that a full level will have on its maximum difficulty. These challenges include portions with a high volume of enemies, some which that cannot be destroyed by landing on their heads; portions with Bullet Bill towers of varying heights; gaps of varying width; pipes with Piranha Plants leaping out of them; and portions with mixtures of these scenarios.

Optimization to D* and GA

The D* search still performed sub-optimally given computing hardware, so the search tree needed to be pared down. Paring the tree followed a simple formula: if two child nodes generated the same score from the SEF, the first child node was kept and the other discarded. In a further endeavor to pare the tree, the maximum degree for a node was reduced from 13 to 11 by discounting nodes reachable through the action of ducking by the Agent. In an effort to avoid a bias in the reproduction phase of the genetic algorithm, a generated and tested chromosome was only added to the population if either its fitness score was unique or, failing that, the genes on the chromosome were unique among the chromosomes with the same fitness. If this precaution was not taken, a glut of identical chromosomes with the same score could skew the parental selection process unfairly.

The Experiment

The Experiment was conducted across two iterations of the Genetic Algorithm. For the initial one, a starting population of 10 chromosomes instantiated with random values was created. A total of 800 generations were iterated over, with five children produced per generation. The test level had a time limit of 36 in-game seconds (~26 seconds in realtime), and a length of 300 blocks (~4800 pixels). The level's "seed" used by the level generation engine was 4 and the difficulty was set to 15. The program execution lasted over 20 hours.

After this initial iteration completed, five of the top-scoring agents were used as the starting population for the second iteration of the Genetic Algorithm. The level length was increased three-fold to a length of 900 blocks (~14400 pixels), the time limit set to 100 in-game seconds, the "seed" to 65, and the difficulty retained at 15. 320 generations were evaluated, again with five children produced for each generation. As this test level's length and time were much larger than the first iteration of the GA, the execution time prolonged to about 30 hours.

Results

The technique of using a GA to evolve the SEF of a D* search allowed a system to generate an effective SEF in the absence of a priori knowledge about what makes one agent state more desirable than another. The results of this experiment demonstrate little to no direct correlations between individual weights and bot scores, implying a trial-and-error search for a human would be difficult and time-consuming.

For the initial iteration of the GA, over three thousand unique bots were evaluated over the course of 800 generations. 285 of those tied for the top score of 3955. An interesting note is that the first bot to score this amount was produced during the 11th generation of the GA.

The weights used in the bot's SEF that the GA iterated on varied greatly. Figures 2 and 3 show typical scatter plots for the values of weights over the course of the GA's execution.

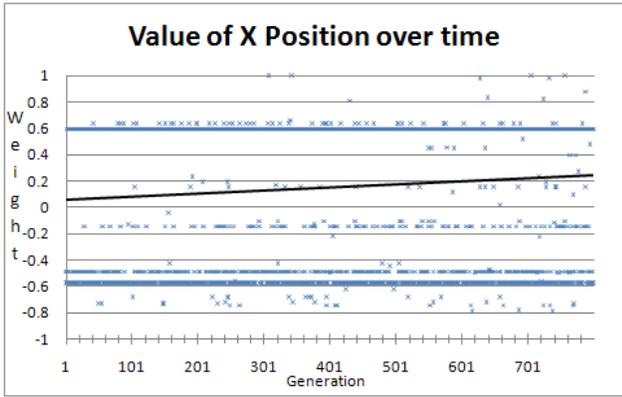


Figure 2

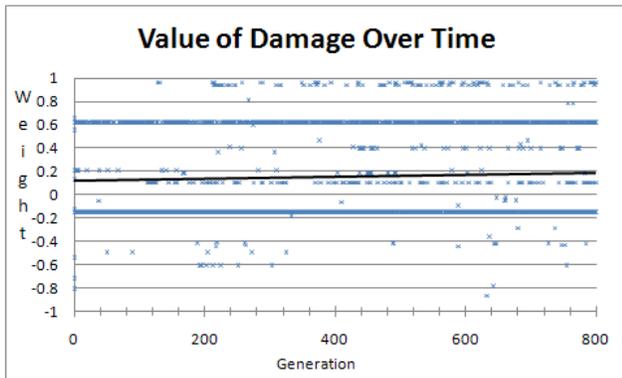


Figure 3

The weights on state-variables may appear to quickly converge to a few values and remain there. However, over time the amount of variance for any weight does not decline linearly. Figure 4 shows a plot of the amount of variation for each weight grouped by 50 generations. No decline of the standard deviation among populations of weight values is present.

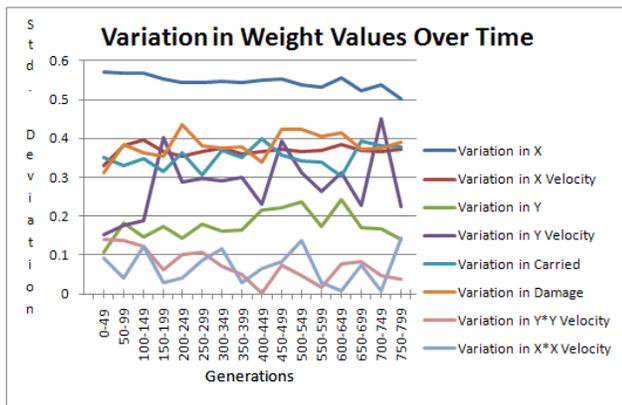


Figure 4

The scores that bots received likewise reached a local maximum early (generation 11), and were unable to improve thereafter (Figure 5).

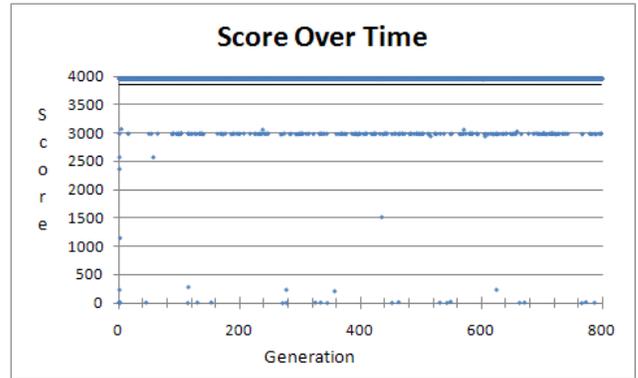


Figure 5

Upon examining the possible correlation between weight values and bot scores, a similar quandary is encountered where bots received top scores despite the weight values (Figures 6 and 7), save for the case of the weight on X Position multiplied by X Velocity (Figure 8). In the case of the value of the weight on the agent's X Position multiplied by the agent's X Velocity, a negative weight positively correlates to a higher score, and every single positive weight has a score of 0.0 or less (a negative score indicates the agent travelled backwards).

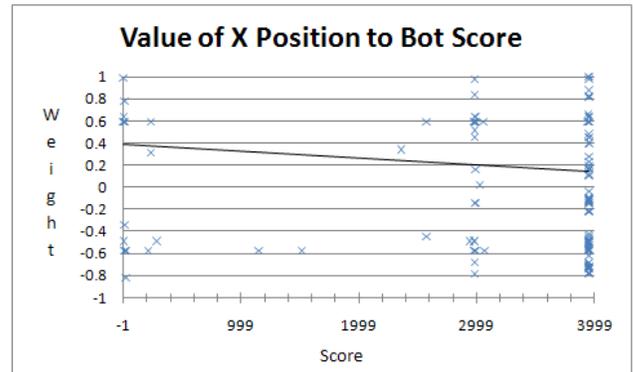


Figure 6

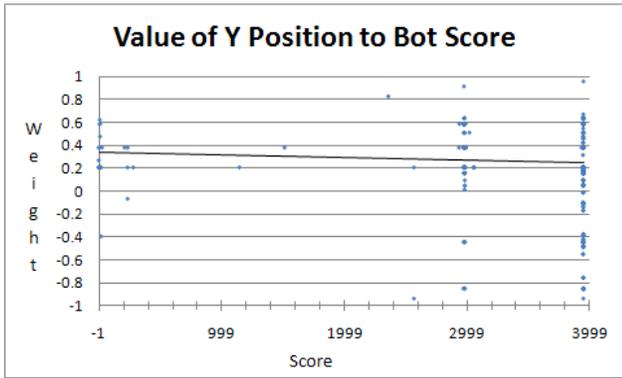


Figure 7

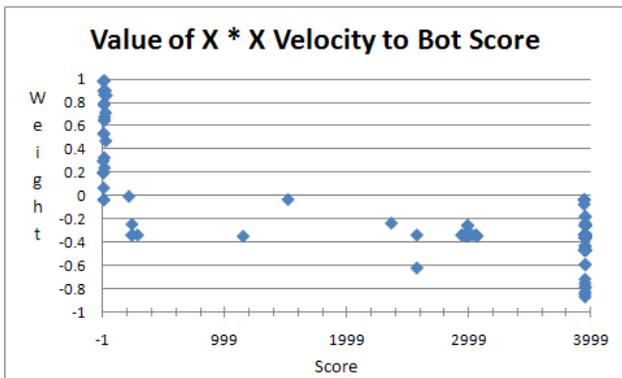


Figure 8

For the second iteration of the GA, similar results to the first iteration were obtained. However, only 3 bots out of a population of over a thousand shared the top score. Figure 9 presents the distribution of scores that bots received during the course of execution. Since the initial population of this iteration comprised top-scoring bots of the first iteration, it is understandable that so many bots scored so well so early, however a clear ceiling to the scores is visible, indicating the algorithm likely could not escape a local maxima.

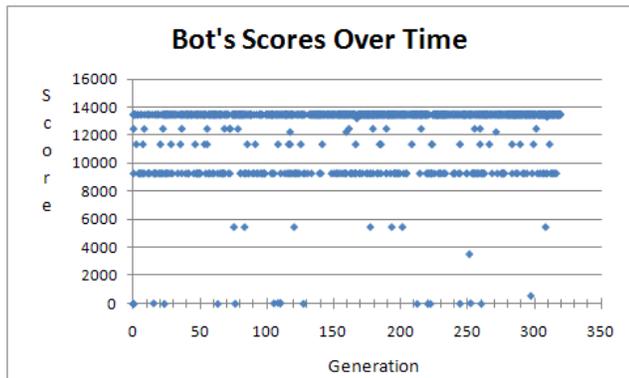


Figure 9

Figures 10 and 11 almost perfectly mirror Figures 6 and 7 in their distribution of scores for weight values on X Position and Y Position. Figure 12 likewise mirrors the data in Figure 8 that indicates negative weights on the agent's X Position multiplied by the Agent's X Velocity correlate to higher scores.

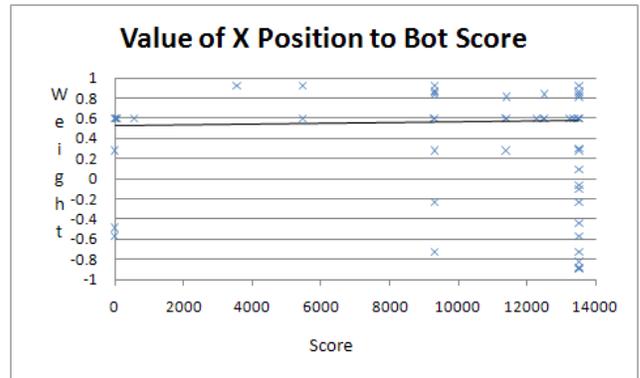


Figure 10

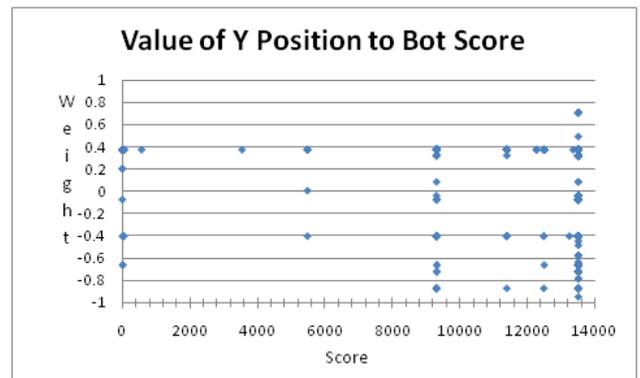


Figure 11

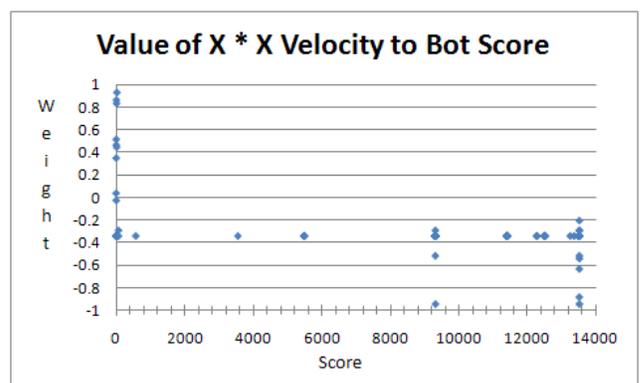


Figure 12

Although the weight values produced by the Genetic Algorithm for the top bots were distributed across the gamut of possible values, the end result was in fact bots

that performed roughly as well as Robin Baumgarten's bot that won the ICE-GIC competition two years in a row.

Table 1 displays a comparison of bot scores between Robin's Bot (AStarAgent) and our bot (AStarAgentEvolved) for a variety of levels whose difficulties were set to 15.

Agent	Level Seed	4	40	65	100	112	216	325	Total
AStarAgent		4450	4422	4420	4420	4470	4548	4470	31200
AStarAgentEvolved		4452	4421	4420	4419	4468	4548	4469	31197

Table 1

Conclusions

In this work, we presented the novel technique of using a genetic algorithm as an offline meta-search for an optimal static evaluation function to be used by the D* search of a real-time autonomous agent. The end results were Static Evaluation Function parameters that, upon use in the SEF for a real-time agent, enabled the agent to perform as well as the current best in its environment.

The fact that our agent performed as well as the current best is significant because we made very few assumptions about the valuation of agent states in a static evaluation function. That is, the algorithms presented in this paper automated this task. The implication is that similar techniques could be employed for autonomous agents in other, possibly real-world, environments with high confidence in the end result.

Future Work

The work presented here has much potential for expansion. Future work should include utilizing parallel computing clusters like Beowulf to take advantage of the natural independence between the analyses of members in a population by the GA's fitness function, as well as the evaluation of nodes in the open list of the D* algorithm by the algorithm's SEF. This sort of capability will allow for not only a deeper D* search, but shorter generations in the Genetic Algorithm and therefore the ability to run the algorithm for more generations in the same amount of time.

Potential future work could also include employing pattern matching techniques to identify a discrete set of distinct scenarios an agent would encounter. An agent could then utilize a separate SEF for each scenario.

Under the notion of pattern-matching, even further future research would focus on generating probability tables for the likelihood of scenarios occurring after each other. Knowing the probability of a scenario to occur next would

allow an agent to make an accurate prediction of an optimal path before receiving its next percept.

References

- Baumgarten, Robin. Infinite Super Mario AI. 9 September 2009. 8 February 2011
 <<http://www.doc.ic.ac.uk/~rb1006/projects:marioai>>.
- Hart, Peter E., Nils J. Nilsson and Bertram Raphael. "A Formal Basis for the Heuristic Determination of Minimum Cost Paths." IEEE Transaction of Systems Science and Cybernetics SSC-4, No. 2 (1968): 100-107.
- Hill T, Lundgren A, Fredriksson R, Schiöth HB (2005). "Genetic algorithm for large-scale maximum parsimony phylogenetic analysis of proteins". *Biochimica et Biophysica Acta* **1725** (1): 19–29.
- Koenig S. and Likhachev M. Fast Replanning for Navigation in Unknown Terrain. *Transactions on Robotics*, 21, (3), 354–363, 2005
- Lucas, S., and Kendell, G. (2006). Evolutionary computation and games. *IEEE Comput Intell Mag.*, pp. 10–18
- Mitchell, M. (1998). *An introduction to genetic algorithms*. MIT Press.
- Ramalingam G., Reps T., An incremental algorithm for a generalization of the shortest-path problem, *Journal of Algorithms* 21 (1996) 267–305.
- Russel, Stuart J. and Peter Norvig. Artificial Intelligence: A Modern Approach. Upper Saddle River: Prentice Hall/Pearson Education, 2003.
- Zhang, J., Lo, W.L., and Chung, H. (2006). Pseudocoevolutionary Genetic Algorithms for Power Electronic Circuits Optimization. *IEEE Trans Systems, Man, and Cybernetics*, **36C** (4), pp. 590–598.

Mining High Quality Association Rules Using Genetic Algorithms

Peter P. Wakabi-Waiswa and Venansius Baryamureeba

Faculty of Computing & Information Technology,
Makerere University, Kampala, Uganda
Email: pwakabi@hotmail.com, barya@cit.mak.ac.ug

Abstract

Association rule mining problem (ARM) is a structured mechanism for unearthing hidden facts in large data sets and drawing inferences on how a subset of items influences the presence of another subset. ARM is computationally very expensive because the number of rules grow exponentially as the number of items in the database increase. This exponential growth is exacerbated further when data dimensions increase. The association rule mining problem is even made more complex when the need to take the different rule quality metrics into account arises. In this paper, we propose a genetic algorithm (GA) to generate high quality association rules with five rule quality metrics. We study the performance of the algorithm and the experimental results show that the algorithm produces high quality rules in good computational times.

Keywords: confidence; support; interestingness; lift; J-Measure; genetic algorithms;

1 Introduction

Since the association rule mining problem was proposed by (Agrawal, Imielinski, & Swami 1993), several algorithms have been developed. In the most commonly used approach, the rule generation process is split into two separate steps. The first step includes applying the minimum support to find all frequent itemsets in a database and the second step includes applying the minimum confidence constraint on the frequent itemsets to form the rules. The first step is computationally very expensive because finding all frequent itemsets in a database involves searching all possible itemsets. The set of all possible itemsets grows exponentially with the number of items in the database. The exponential growth profoundly affects the performance of association rule mining algorithms (Agrawal, Imielinski, & Swami 1993).

Other than the aforementioned difficulties, most of the existing algorithms have been developed to produce simple, easy to understand association rules which measure the quality of generated rules by considering mainly only one or two evaluation criteria most especially confidence factor or predictive accuracy (Dehuri *et al.* 2008). These algorithms provide useful tools for descriptive data mining but there are several measures of rule quality such as comprehensibility, confidence, J-measure, surprise, gain, chi-squared value,

gini, entropy gain, Laplace, lift and conviction that can be used to evaluate the rules (Carvalho, Freitas, & Ebecken 2005), (Freitas 1999).

Quite a number of research works have been carried out in this arena but results indicate that more innovative approaches need to be introduced with a view of finding algorithms that can handle multiple and increasing rule quality metrics as well as improving algorithm efficiency (Hruschka *et al.* 1999), (Kotsiantis & Kanellopoulos 2006).

In this paper we propose a Multi-Objective Genetic Algorithm for Mining Association Rules (MOGAMAR), which generates association rules with *five* rule quality metrics: confidence, support, interestingness, lift and J-Measure which permit the user to evaluate association rules on these different quality metrics in a single algorithm run. The main motivation for using Genetic Algorithm (GA) is that a GA performs a global search and copes better with attribute interaction than the greedy rule induction algorithms often used in data mining tasks (Freitas 2007). Genetic Algorithms are robust with little likelihood of getting stuck in local optima and they are highly parallel in nature making them good candidates for distributed implementations (Liu & Kwok 2000a).

The rest of this paper is organized as follows. We present a detailed description of the proposed algorithm in Section 2. In Section 3 we evaluate the performance of the algorithm and in Section 4 we conclude the paper.

2 The Multi-Objective Genetic Algorithm for Mining Association Rules

In this work we use the underlying structure of the object-oriented genetic algorithm proposed in (Davis 1991) with modifications to the representation of the individuals. In the rest of this section we give a detailed description of the proposed algorithm.

2.1 Representation of the Rules

We adopted a modified Michigan approach proposed in (Ghosh & Nath 2004) whereby the encoding/decoding scheme associates two bits to each attribute in the database. If these two bits are 00 then the attribute next to these two bits appears in the antecedent part and if it is 11 then the attribute appears in the consequent part. And the other two

00	A	11	B	00	C	01	D	00	F
----	---	----	---	----	---	----	---	----	---

Figure 1: Modified Michigan Rule Representation

combinations, 01 and 10 will indicate the absence of the attribute in either of these parts. As an example, suppose there is a rule $ACF \rightarrow BE$. It will be represented as shown in Figure 1.

Following this approach the algorithm can handle variable length rules with more storage efficiency, adding only an overhead of $2k$ bits, where k is the number of attributes in the database. The downside of this approach is that it is not well suited for handling continuous valued attributes. For handling real-valued attributes we have incorporated the discretization method proposed in (Kwedlo & Kretowski 1999). The combination of these two approaches enabled us to uniformly represent the association rules using the Michigan approach. The decoding is performed as follows:

$$DV = mnv + (mxv - mnv) * \left(\frac{\sum (2^{i-1} * i^{th} \text{bit}_i)}{2^n - 1} \right) \quad (1)$$

where DV is the decoded value; $1 \leq i \leq n$ and n is the number of bits used for encoding; mnv and mxv are minimum and maximum values of the attribute, respectively; and bit_i is the value of the bit in position i . It is important to note with that the encoding of the rules in this algorithm, the consequent part of the rule is not important at the start of the run of the algorithm because the the consequent should not be randomly determined. The consequent part is determined when the fitness of the rule is computed.

2.2 Initialization

We avoid generation of the initial population purely randomly because it may result in rules that will cover no training data instance thereby having very low fitness. Furthermore, a population with rules that are guaranteed to cover at least one training instance can lead to over-fitting the data. It was shown in (Surry & Radcliffe 1996) that utilizing non-random initialization can lead to an improvement in the quality of the solution and can drastically reduce the the runtime. We, therefore, designed an initialization method which includes choosing a training instance to act as a “seed” for rule generation as proposed in (Freitas 2002). In our initialization approach, a seed should be a data instance lying in the middle of a cluster of examples with a common consequent. The training examples are stored in an array and iteratively for each example we calculate the fraction of those that cover the consequent against those that negate the coverage of that consequent (same-consequent/opposite-consequent), ρ , as in (2).

$$\rho = \frac{\text{Count}(\text{Same}_{consequent})}{\text{Count}(\text{Opposite}_{consequent})} \quad (2)$$

where $\text{Count}(\text{Same}_{consequent})$ denotes the number of same-consequent examples and $\text{Count}(\text{Opposite}_{consequent})$ denotes opposite-consequent examples. The training example with the highest ρ will be selected as the seed.

2.3 Reproduction

The reproduction mechanism involves rule selection and the application of the crossover operators. The rule selection method used by this algorithm follows the “universal suffrage” approach proposed in (Giordana *et al.* 1997). With this approach each association rule is represented by a single individual. The individuals to be mated are elected by training data instances. Each instance votes for a rule that it covers in a stochastic fitness-based way. Using an example, let us assume we have a set R of rules or chromosomes that cover a given instance i i.e. a rule whose antecedent and consequent are satisfied by the instance i . Then the instance i votes in one of the rules in R by using a roulette wheel selection scheme.

This means that each rule r in R is assigned a roulette-wheel slot whose size is proportional to the ratio of fitness of r divided by the sum of fitness of all rules in R . The better the fitness of a given rule r the higher its probability of being selected over the other rules covering the given instance i . In the event of absence of a rule covering i the algorithm automatically creates a new rule using the seeding operator used at the population initialization stage. Since it is only the rules that cover a given instance that do compete with each other, this results into some kind of niching. Niching is a mechanism through which evolutionary algorithms form and maintain subpopulations or *niches*. Niching fosters the evolution of several different rules each covering a different part of the data being mined. This assists avoid the convergence of the population to a single rule resulting in the discovery of a set of rules rather than a single one.

The actual reproduction takes place by performing a *multi-point crossover* and the *mutation* on the new individuals.

The Crossover Operator We modified the standard crossover operator to either generalize the crossover operator if the rule is too specific, or to specialize it if the rule is too general. A rule is considered too specific if it covers too few data instances i.e. when too few data instances satisfy both the antecedent and the consequent of the rule. In contrast, a rule is considered too general when it covers too many data instances. We make use of the bitwise *OR* and the bitwise *AND* to implement generalization and specialization, respectively. The bitwise *OR* and the bitwise *AND* are applied to the antecedent part of the rule since this determines the consequent part (Giordana *et al.* 1997).

The generalization/specialization crossover procedure first constructs an index, $I = \{i_1, \dots, i_n\}$, of pointers to the positions in the chromosome (bit-string) where the corresponding bits in the two parents have different values. Then, for every element $i_k \in I$ the following procedure is repeated. If the rule is too general, the algorithm replaces the value of the bits $b(i_k)$ with the logical *OR* of the corresponding bits in the parents, otherwise if the rule is too specific the algorithm replaces the value of the bit $b(i_k)$ with the logical *AND* of the corresponding bits in the parents. The crossover operator is equipped with a mechanism for detecting and eliminating invalid genotypes.

The Mutation Operator The mutation operator helps in maintaining the diversity within the population and also in preventing premature convergence to local optima (Liu & Kwok 2000b). The mutation operator needs to be designed in such a way that we avoid the population being dominated by a single highly fit individual. Our approach to cope with this problem is to use an adaptive mutation probability, where the value of the mutation probability is varied as the population becomes dominated by an individual. We adopted the *non-uniform-mutation* operator proposed in (Michalewicz 1999). The non-uniform mutation operator adapts to the environment by varying as the fitness of the individuals in the population changes. We made a modification to the non-uniform mutation operator to enable it to generalize and/or specialize a rule condition. The mutation operator randomly selects a condition from the rule. If that condition involves a nominal attribute, then the value will be randomly changed from one value to another. If the attribute is continuous, mutation will randomly change the conditions interval values. The specialization mutation operator works on a randomly selected condition in the rule. If the condition involves a continuous attribute, specialization shrinks the interval.

The mutation operator used here ensures that the high mutation probabilities don't cause the loss of the fittest individual of a generation. Furthermore, the mutation operator can cause the undesired effect of changing the rule's consequent if applied before generating the consequent but in our case the consequent is generated after the mutation operator has been applied.

2.4 Replacement

We use an elitist individual replacement approach that ensures that more fit genotypes are always introduced into the population.

Uniqueness testing The application of the genetic operators on the parent population may result in identical genotypes in the population. The algorithm first tests to ensure the new offspring do not duplicate any existing member of the population. There is, however, a computational overhead caused by the uniqueness testing process on the operation of the algorithm. The computational overhead is compensated by a reduction in the number of genotype evaluations required because the check for duplicates can be performed before fitness evaluation. The saving of number of fitness evaluations significantly increases efficiency.

Furthermore, the adoption of a replacement strategy with genotype uniqueness enforced preserves genetic diversity within the population (Lima *et al.* 2008). Genetic diversity is significant as crossover-type operators depend on recombining genetically dissimilar genotypes to make fitness gains. Uniqueness of the genotypes permits the algorithm to find not only the single best individual but the n best genotypes in a single run, where n is the population size. The process of finding the best n individuals imposes some computational load but it is compensated with the generation of n high-fitness individuals in a single program run.

Fitness Evaluation MOGAMAR performs the fitness evaluation of the generated rules using a set of five complementary metrics: confidence, support, interestingness, lift and J-Measure. These metrics are converted into an objective fitness function with user defined weights. The *support*, $\sigma(X)$, of an itemset X is defined as the proportion of transactions in the data set which contain the itemset. The *confidence factor* or predictive accuracy of the rule is the conditional probability of the consequent given the antecedent, calculated as in (3).

$$\text{confidence} = \sigma(X \cup Y) / \sigma(X) \quad (3)$$

We adopt the interestingness metric calculation proposed in (Freitas 1998). The algorithm first calculates the information gain, $\text{InfoGain}(A_i)$, of each attribute, A_i . Then the interestingness of the rule antecedent, RAI , is calculated by an information-theoretical measure as (4).

$$RAI = 1 - \left[\frac{\sum_{i=1}^n \text{InfoGain}(A_i)}{\frac{n}{\log_2(|G_k|)}} \right] \quad (4)$$

The degree of interestingness of the rule consequent (CAI) is as (5):

$$CAI = (1 - \Pr(G_{kl}))^{1/\beta} \quad (5)$$

where G_{kl} is the prior probability of the goal attribute value G_{kl} , β is a user-specified parameter, and $1/\beta$ is a measure for reducing the influence of the rule consequent interestingness in the value of the fitness function.

The *interestingness* of the rule is given by (6):

$$\text{interestingness} = \frac{RAI + CAI}{2} \quad (6)$$

J-Measure shows how dissimilar a priori and posteriori beliefs are about a rule meaning that useful rules imply a high degree of dissimilarity. In rule inference we are interested in the distribution of the rule "implication" variable Y , and its two events y and complement \bar{y} . The purpose is to measure the difference between the priori distribution $f(y)$, i.e. $f(Y = y)$ and $f(Y \neq y)$, and the posteriori distribution $f(Y | \vec{X})$. The *J-Measure* is calculated as (7):

$$J_M = f(x) \left(f(y|x) \cdot \ln \left(\frac{f(y|x)}{f(y)} \right) + (1 - f(y|x)) \cdot \ln \left(\frac{(1 - f(y|x))}{1 - f(y)} \right) \right) \quad (7)$$

Lift is equivalent to the ratio of the observed support to that expected if X and Y were statistically independent and it is defined as (8):

$$\text{lift}(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X) * \sigma(Y)} \quad (8)$$

Finally, the fitness function is calculated as the arithmetic weighted average of confidence, support, interestingness,

lift and J-Measure. The fitness function, $f(x)$, is given by (9):

$$f(x) = \frac{w_s * S + w_c * C + w_i * I + w_l * L + w_J * J_M}{w_s + w_c + w_i + w_l + w_j} \quad (9)$$

where S denotes support, C denotes confidence, I denotes interestingness, L denotes lift, J denotes J-Measure, and their respective user defined weights are w_s, w_c, w_i, w_l, w_j . The chromosomes are then ranked depending on their fitness.

Selection Fitness Calculation Using the rank-based linear normalization we convert the fitness values into selection-fitness values. With rank-based linear normalization a linear increment is used to set the selection-fitness values for the genotypes based on their rank. The main reason for using rank-based linear normalization is because it provides explicit control on the selection pressure applied to the population and reduces the likelihood that the population will converge prematurely on a sub-optimal solution (Metzler 2005). This assists in avoiding problems caused by super-individuals dominating the selection process.

2.5 Criteria for Termination

The algorithm terminates execution when the *Degeneracy condition is met* –i.e. when the best and worst performing chromosome in the population differs by less than 0.1%. It also terminates execution when the total number of generations specified by the user is reached.

3 Algorithm Performance

In this section, we evaluate the relative performance of the algorithms and compare its performance to that of the Data Mining by Evolutionary Learning (DMEL) algorithm (Au, Chan, & Yao 2003). DMEL is known to be one of the best performing evolutionary algorithms for association rule mining (Reynolds & de la Iglesia 2008). The performance characteristics studied included the quality of solutions found and CPU utilization.

3.1 Datasets

To evaluate our algorithm we used Adult, Connect-4, Chess, Diabetes, DNA and Mushroom datasets from UCI repository of machine learning databases (Frank & Asuncion 2010). We also used the Motor Vehicle Licensing System (MVLS) database from the Uganda Revenue Authority (URA) which we processed for use in experiments. The MVLS database contains data pertaining to motor vehicle licensing fees, vehicle ownership and registration, transfer of ownership, accidents, usage, country of origin, date or year of manufacture. The MVLS has over 4 million records but we randomly selected 441,327 transactions for these experiments. The summary of the datasets is given in Table 1.

3.2 Relative Performance of the Algorithms

The summary of the results of our experiments with the fixed parameters are given in Table 2. The results include the av-

Dataset	No. of Instances	Attributes
Adult	48,842	14
Chess	3,196	36
Connect-4	67,557	42
Diabetes	768	20
DNA	3190	62
Mushroom	8,124	22
MVLS	441,327	32

Table 1: Summary of the Datasets

erage rule quality metrics and CPU utilization. From Table 2, we observe that:

1. The discovered rules have a very high average quality value for all datasets implying that they are good algorithms
2. The algorithms are generally fast for smaller datasets with increasing average CPU times for larger datasets
3. The algorithms generally produce poorer quality rules for large datasets.

Our overall observation is that the quality of the generated rules decreases as the dataset dimension increase.

It can be observed that the algorithms consumed quite a bit of time with the Connect-4 and the MVLS datasets. It is quite evident that these two datasets have a much larger number of transactions and attributes as compared to the rest of the datasets. We also observed that the rules with high confidence within these datasets have very low support. For instance, the rules that have a ‘*tie*’ as their consequent and a confidence of 100% in the Connect-4 dataset, have a support of only 14 records. This profoundly affects the time it takes to process the records. When we set a low minimum support the algorithm response time greatly improves.

Dataset	MOGAMAR		DMEL	
	Average Quality	Time (Secs)	Average Quality	Time (Secs)
Adult	89.45	1.35	89.7	1.34
Connect-4	87.3	12	88.2	19
Chess	97.35	0.01	95.6	0.015
Diabetes	87.75	0.015	79.8	0.017
DNA	96.0	1.23	95.4	1.21
Mushroom	89.4	0.03	88.5	0.04
MVLS	82.4	900	83.6	903

Table 2: Quality of rules and run times

3.3 Sensitivity to Dataset Size

In Section 3.2, we used fixed datasets sizes to assess the performance of the algorithms. With a fixed dataset size it may not be possible to gauge how well the algorithm performs when the datasets grow. We now study the difference in performance of the algorithm with increasing dataset sizes. Figure 2 summarizes the trends in the performance of algorithms with varying dataset sizes. From Figure 2 we observe

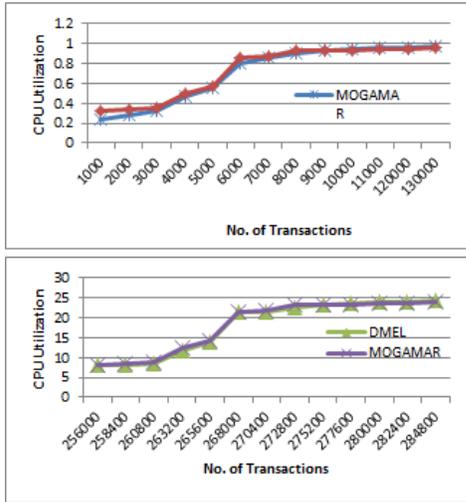


Figure 2: Relative Performance of the algorithms

that the increase in dataset size leads to an increase in the average response time of the algorithms. Furthermore, when the data are increased two-fold, there is a multiplier effect on the response time of the algorithms. This implies that the performance of the ARM algorithm highly depends on the size of the data being mined.

3.4 Sensitivity to Genetic Operators and Parameters

Experiments we carried out in Section 3.2 were done using specific parameter instances. It is possible that variations in the parameter values can lead to either an improvement or deterioration in performance of the algorithms. In this subsection we make a deeper study on the effect of the crossover and mutation operators on the performance of the algorithms.

Effect of Crossover Operator We studied the performance of MOGAMAR and DMEL with different values of the crossover rate. We observed that the overall performance of both algorithms with different crossover rates is slightly different from that shown when the values of the crossover rate is fixed. This implies that the crossover rate does not have a very profound effect on the performance of the algorithm.

Effect of Mutation Operator Figure 3 shows the performance of the algorithms with varying mutation rates. It can be seen that their performance was drastically affected with increasing mutation rates implying that the mutation operator has high impact on the performance of genetic algorithms. With the mutation rates increasing, the utilization of the CPU drastically increased showing a big reduction in the algorithm efficiency and disparity in the characteristics of the algorithms became more evident. The main cause of this phenomenon is that the mutation operator reintroduces useful genotypes into the population for a diverse pool of parents. This increases the diversity of the population be-

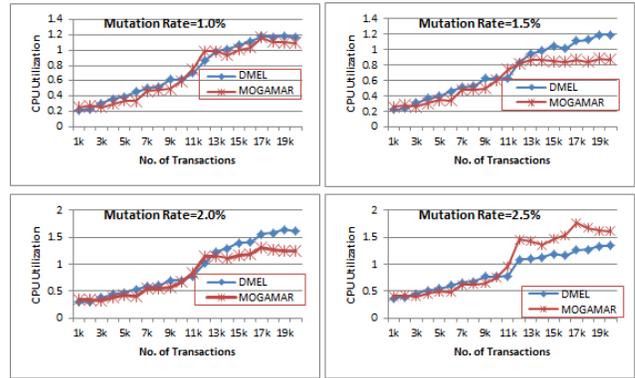


Figure 3: Relative Performance of MOGAMAR and DMEL for different mutation rates

cause each structure in the new population undergoes a random change with a probability equal to the mutation rate. This prevents members of the population from converging to a single very fit individual and as such increasing the required number of valuations. This implies that the mutation operator needs to be more thoroughly investigated to establish the most suitable rates.

4 Conclusion

In this paper we have proposed a new association rule mining algorithm, MOGAMAR. The approach proposed in this paper incorporates a novel population initialization technique that ensures the production of high quality individuals; specifically designed breeding operators that ensure the elimination of defective genotypes; an adaptive mutation probability to ensure genetic diversity of the population; and uniqueness testing. The performance of MOGAMAR has been compared with DMEL, an efficient ARM algorithm, on a number of benchmark datasets with experimental results showing that our proposed approach can yield equally as good performance with consistently high quality rules. MOGAMAR provides the user with rules according to five interestingness metrics, which can easily be increased if need be by modifying the fitness function. We studied the effect of parameters to the performance of the algorithm specifically dataset size, crossover and mutation rates. We observed that the algorithm performs poorly for large dataset sizes. The poor performance is more evident as the datasets increase in size. This has been identified as an area requiring more research efforts. It was further observed that the crossover rate does not have a big impact on the performance while the mutation rate does. This indicates that it is necessary to find methods for finding the right mutation rates that encourage the performance of the algorithm. This is another area we shall be researching in. We have used weighted fitness function for finding the best rules but this approach may not be the best when there are several solution-quality criteria to be evaluated. This is because these criteria may be non-commensurate or conflicting most especially when they evaluate different aspects of a candidate solution. In our future works we shall specifically look into the possibility of

differently modeling the ARM problem.

References

- Agrawal, R.; Imielinski, T.; and Swami, A. 1993. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C.*, 26–28.
- Au, W.; Chan, K.; and Yao, X. 2003. A novel evolutionary data mining algorithm with applications to churn prediction. *IEEE Transactions on Evolutionary Computation* 7(6).
- Carvalho, D. R.; Freitas, A. A.; and Ebecken, N. F. F. 2005. Evaluating the correlation between objective rule interestingness measures and real human interest. In *PKDD*, 453–461.
- Davis, L. 1991. *Handbook of Genetic Algorithms*. VNR Computer Library, Von Nostrand Reinhold, New York.
- Dehuri, S.; Patnaik, S.; Ghosh, A.; and Mall, R. 2008. Application of elitist multi-objective genetic algorithm in classification rule generation. *Applied Soft Computing Journal* 8:477–487.
- Frank, A., and Asuncion, A. 2010. UCI machine learning repository.
- Freitas, A. 1998. On objective measures of rule surprisingness. In *Principles of Data Mining and Knowledge Discovery, 2nd European Symp., PKDD98. Nantes, France*, 1083–1090.
- Freitas, A. A. 1999. On rule interestingness measures. *Knowledge-Based Systems* 12:309–3135.
- Freitas, A. A. 2002. *Data mining and knowledge discovery with evolutionary algorithms*. Springer-Verlag Berlin Heidelberg.
- Freitas, A. A. 2007. A review of evolutionary algorithms for data mining. *Soft Computing, Knowledge Discovery and Data Mining*.
- Ghosh, A., and Nath, B. 2004. Multi-objective rule mining using genetic algorithms. *Information Sciences* 163:123–133.
- Giordana, A.; Anglano, C.; Giordana, A.; Bello, G. L.; and Saitta, L. 1997. A network genetic algorithm for concept learning. In *Proceedings of the Sixth International Conference on Genetic Algorithms*, 436–443. Morgan Kaufmann.
- Hruschka, E. R.; Campello, R. J.; Freitas, A. A.; and de Carvalho, A. C. P. L. F. 1999. Survey of evolutionary algorithms for clustering. *IEEE Transactions on Systems, Man, and Cybernetics* 12:309–315.
- Kotsiantis, S., and Kanellopoulos, D. 2006. Association rules mining: A recent overview. *GESTS International Transactions on Computer Science and Engineering* 32(1):71–82.
- Kwedlo, W., and Kretowski, M. 1999. An evolutionary algorithm using multivariate discretization for decision rule induction. In *Proceedings of the 3rd European Conference on Principles and Practice of Knowledge Databases (PKDD '99)*.
- Lima, C. F.; M.Pelikan; Goldberg, D.; O G. Lobo, K. S.; and Hauschild, M. 2008. Influence of selection and replacement strategies on linkage learning in boa. In *Evolutionary Computation, 2007. CEC 2007. IEEE Congress*, 1083–1090.
- Liu, J., and Kwok, J. 2000a. An extended genetic rule induction algorithm. In *Proceeding of the 2000 Congress on Evolutionary Computation*.
- Liu, J., and Kwok, J. 2000b. An extended genetic rule induction algorithm. In *Proceedings of the 2000 Congress on Evolutionary Computation*, volume 1, 458–463.
- Metzler, D. 2005. Direct maximization of rank-based metrics. Technical report, University of Massachusetts, Amherst.
- Michalewicz, Z. 1999. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, New York.
- Reynolds, A., and de la Iglesia, B. 2008. A multi-objective grasp for partial classification. *Soft Comput.* 13.
- Surry, P., and Radcliffe, N. 1996. Inoculation to initialize evolutionary search. In T.C., F., ed., *Evolutionary Computing: AISB Workshop, Springer, Brighton, U.K.*

Support for Agent Based Simulation of Biomolecular Systems

Harika Korukonda and Carla Purdy

School of Electronic and Computing Systems, University of Cincinnati
contact: carla.purdy@uc.edu

Abstract

We describe our work developing GenericSystem, which contains the basic classes and functions necessary to model and simulate a typical biomolecular system. GenericSystem is user-friendly and extensible. It consists of well-defined functions which can be readily customized to reduce development time for a biomolecular simulation. Five systems-- bioluminescence in *Vibrio fischeri* bacteria, skin regulatory system, phage lambda in *E. coli*, epithelial cells growth cycle and Wnt signaling pathway--have been successfully implemented using the GenericSystem tool. We show some results for one of these systems and describe the process to translate a typical differential equation-based system description into a GenericSystem model.

Introduction

For molecular systems, traditional modeling techniques are known as equation based modeling (EBM) [2] and include ordinary differential Equations (ODE), partial differential equations (PDE) [3], stochastic differential equations (SDE), Petri nets [4] and Pi-calculus [5]. The main disadvantage of these approaches is their inability to consider the spatial dynamics and heterogeneity of the system. Petri nets and Pi-calculus are graph based techniques. Graph-based techniques have several additional shortcomings such as basic modeling constructs which are quite primitive and hence they fail to model complex systems successfully. Also spatial representation of the system can get very complex with respect to time and effort. Another main disadvantage is the inefficiency of representing priorities or ordering of events which is essential in systems modeling. Because of these shortcomings, graph-based modeling techniques are not well-developed and hence rarely used. The other class of models comprising of ODE, PDE and SDE form the equation based modeling class of techniques. This gamut of approaches essentially represents the system by identifying the system variables. These variables are integrated and sets of equations relating these variables are formed. Evaluation of these equations forms the basis of EBM. The fact that this approach has been in use for several decades essentially showcases its ability to model a system satisfactorily. But as the complexity of the system increases, this approach starts to fail. This happens mainly because the equations involved become too complex to handle. A clear comparison between agent based modeling and the equation based approach is given in [2].

The disadvantages of modeling using traditional techniques include: the spatial dynamics of the systems cannot be modeled; systems with both continuous and discrete behavior cannot be modeled; the high complexity and stochasticity of the system cannot be taken into account; and most of these methods tend to aggregate the values during modeling, which may lead to incorrect results [14]. ABM, on the other hand, has many advantages: ABM describes a system in a way which is closest to it in reality [1]; randomness is applied in the most appropriate way instead of just adding a noise term to an equation; ABM captures emergence phenomena of the system which are the result of interactions between individual entities of the system; and ABM is flexible, i.e., it provides a natural framework for tuning the complexity of the agents. The behavior, degree of rationality, ability to learn and evolve and rules of interactions are adjustable [1]; the levels of agglomeration can be varied. i.e., dealing with single agents and groups of agents simultaneously becomes easy; the interactions can be changed dynamically, since they are defined at the agent level; and positive and negative feedback can be modeled. For systems in which activities describe the system better than processes and in which stochasticity applies to an agent's behavior, ABM is often the most appropriate way of modeling [1]. It can also be applied to problems where the population is heterogeneous or the topology of the interactions is heterogeneous and complex. There are several situations when ABM is the only resort. When the behavior of individual entities of a system cannot be clearly defined through aggregate transition rates, ABM is especially useful. As the individual behavior grows in complexity, the complexity of differential equations modeling them also grows exponentially and thus becomes unmanageable. ABM has no such overhead and has proved successful in modeling several complex systems [7].

Project Goals

The goals of this project were:

- To design GenericSystem, a generic easy-to-use simulation model using the agent-based modeling technique which can efficiently model many of the commonly found biological systems.
- To implement GenericSystem using MASON [9,10] and make the right use of the advantages available in the tool. MASON was chosen as the basis for our system after a thorough analysis of the available tools. A summary of many of the tools we considered is available in [8].

- To incorporate as many features as possible into the generic system so that it can successfully be used to model systems with entities of various complex shapes.
- To provide a procedure for transforming a biomolecular system modeled traditionally into an ABM version using our tool
- To provide case studies of specific system models, including examples previously developed individually in our lab (bioluminescence in *Vibrio fischeri* [12], skin cell regulation (normal and wound conditions) [13] and phage-lambda in *E. coli* acting as a biological inverter [11]).
- To provide an example of translating a differential equations-based simulation to an ABM simulation in GenericSystem, based on the Wnt signaling pathway [16,17].

GenericSystem

GenericSystem was designed using AUML, an agent based extension of UML [15]. There are three main classes of agents, Stationary, Mobile, and Vibrating. Users can extend these classes or add new classes which are derived from the base class Agent. Initial subclasses included in the system are Rectangular Sheet, Rectangular Box, Spherical, Cylindrical, Sticky Rectangular Sheet (Stationary), Sphere, Dumbbell, Rectangular Box, Rectangular Sheet (Mobile), and Rectangular Box, Rectangular Sheet, U- Shaped (Vibrating). Figure 1 shows an AUML diagram for a GenericSystem class. Notice how the diagram facilitates modeling communication between the agent and its environment.

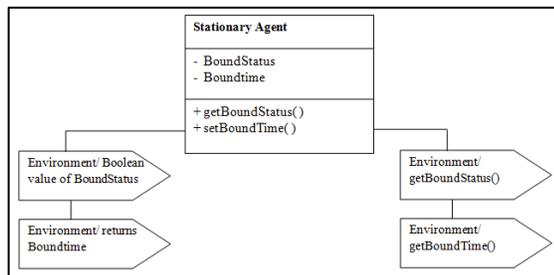


Figure 1. AUML diagram of Agent class.

Experimental Setup and Base Parameters of Simulations

The computer used to run the experiments has the 1.6GHz Intel Core 2 Duo processor. The operating system used is a 32bit Windows 7 OS. The RAM installed in the computer is 2GB.

The software versions used in the work are:

- MASON version 14
 - Java™ SDK 6 update 11
 - Java 3D version 1.5.1
 - Eclipse platform version 3.4.1 (IDE for Java)
- Model design parameters are:

- Size : Generic unit which can be interpreted as the user wishes. It can be specified by using the scale function of display class.
- Velocity: Can be interpreted as generic unit of time or generic unit of length as per user's convenience.
- Container : The large rectangular box which contains all the simulations elements. It can be interpreted as the entire simulation space where the reactions are taking place. It is assumed that all the reactions take place within the container.

For all the systems to be implemented, the chemical reactions and their reaction rates have been found from the literature. The lifetimes and binding times of the molecules are calculated from the respective reaction rates. In [6] a relation between rate constants and the reactions times has been established. The inverse of the rate constant is considered as the reaction time measure. This is because any two given reactions with the same initial concentration of reactants proceed with velocities that have the same ratio as their reaction rate constant ratio. If v_1 and v_2 are reaction velocities of reactions with rate constants K_1 and K_2 , then the relation between them defined in [6] is

$$v_1/K_1 = v_2/K_2..$$

Building a GenericSystem Model

We illustrate the use of GenericSystem through the model of the Wnt signaling pathway and some simulations of its behavior. The Wnt signaling pathway describes a set of proteins most commonly known for their effect on embryogenesis and cancer tumors. A protein called β -catenin acts as a transcriptional coactivator for cancer causing tumor cells. Other important proteins which form part of the Wnt signaling pathway are APC and Axin, which is required for degradation of β -catenin. The reactions taking place in the Wnt pathway are shown in Figure 2.

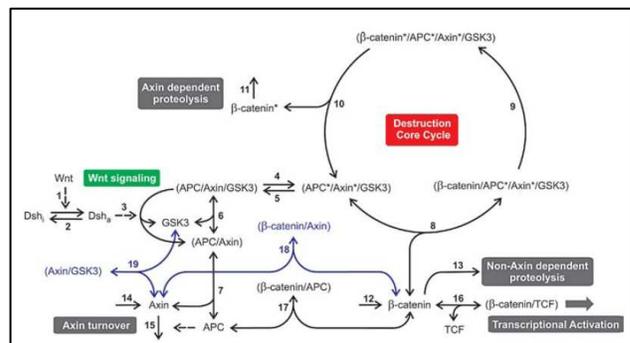
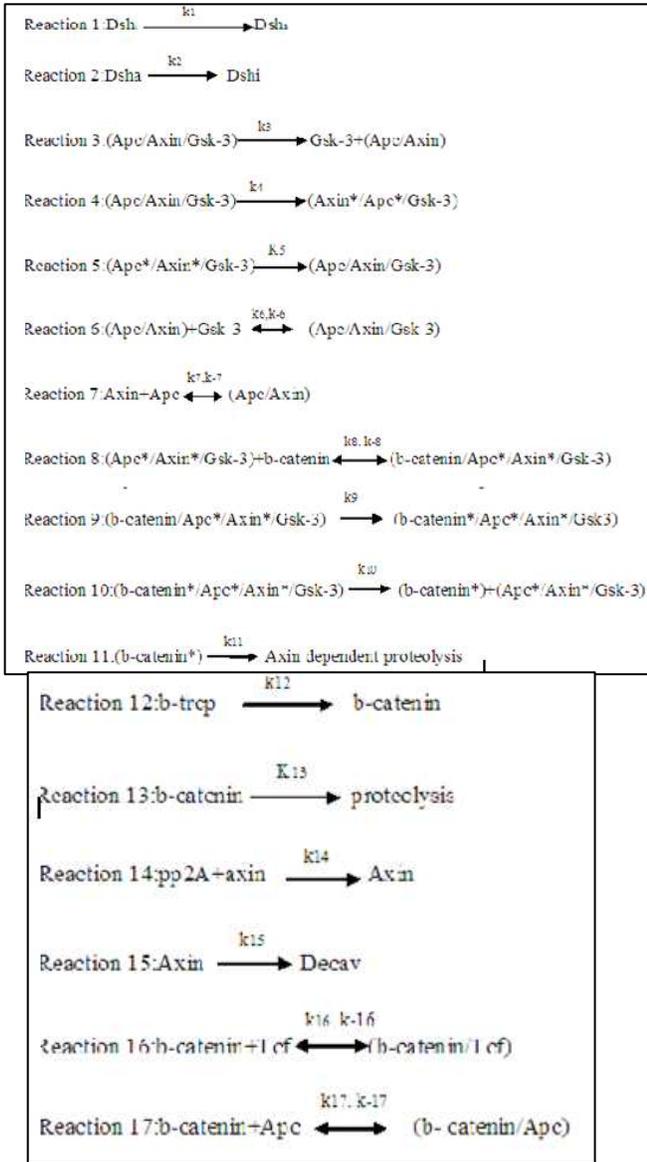


Figure 2. Reactions in Wnt pathway [16,17].

This forms a transcriptional regulation of Wnt genes. When Wnt signal is absent, APC directly associates with TCF/LEF binding site on Wnt target genes and mediates exchange between coactivator and corepressor complex proteins. This represses concentration of β -catenin. Also

when Wnt signal is absent, APC transports β -catenin from the nucleus to the destruction complex where it phosphorylates and is recognized by β -TrCP. This also results in further degradation of β -catenin protein. On the other hand, when Wnt signal is present, the phosphorylation of β -catenin is inhibited, leading to its dissociation from the Axin-assembled destruction complex [18]. The stabilized β -catenin reaches the nucleus and binds to the TCF/LEF resulting in activation of Wnt target genes. The chemical reactions taking place in the system are shown in Figure 3 and Figure 4. A differential equation based model of the Wnt pathway was previously developed in our lab [19]. Here we use the GenericSystem procedure to translate that model into an ABM model.



Figures 3 and 4. Wnt equations.

Now we categorize the molecules to match agent types available in the GenericSystem. The agents chosen to implement the system molecules are provided in Table 1.

Molecule	Agent
Dsh	Mobile Spherical Agent
Dsha	Mobile Spherical Agent
apc*/axin*/gsk3	Mobile Spherical Agent
apc/axin/gsk3	Mobile Spherical Agent
Gsk3	Mobile Spherical Agent
Apc/axin	Mobile Spherical Agent
Apc	Mobile Spherical Agent
β -catenin/apc*	Mobile Spherical Agent
β -catenin*/apc*/axin*/gsk3	Mobile Spherical Agent
β -catenin *	Mobile Spherical Agent
β -catenin	Mobile Spherical Agent
Axin	Mobile Spherical Agent
Tcf	Stationary Rectangular Box
β -catenin/tcf	Mobile Spherical Agent
β -catenin/apc	Mobile Spherical Agent
TCF/LEF Binding site	Stationary Rectangular Box

Table 1. Wnt molecules and corresponding agents.

The chemical reactions are interpreted as collisions between the corresponding molecules. The rate of reaction is modeled as the velocity of corresponding molecules. The functions used are listed in Table 2.

Chemical Reaction	Function of GenericSystem
Bonding to another molecule	Attach()
Unbond from another molecule	Detach()
Grow in size	GrowInSize()
Decay	DeleteAgent()
React with another agent	DetectCollision()
Move freely	Move()
React and form new molecule	DeleteAgent(), CreateAgent()

Table 2. Agent functions and chemical reactions.

The number of molecules and their lifetimes are based on the concentrations given in [16]. The rate constants K_i are given in Figure 5.

Rate	K1	K2	K3	K4	K5	K6	k7
value	0.182	0.0182	0.05	0.267	0.133	0.0909	0.0909
Rate	K8	K9	K10	K11	K12	K13	k14
Value	1000	12000	0.01	0.5	206	206	0.417
Rate	K15	K16	K17				
Value	0.423	0.000257	0.0000822				

Figure 5. Rate constants for Wnt equations.

The concentrations of molecules in this experiment are spread over a large range starting from 0.00049 to 100. Hence directly relating the concentration to the number of molecules is not possible in this case. So the number of molecules are selected according to their concentration levels. Taking the number of molecules to the limiting number, i.e., the maximum allowed by the tool, is not advisable since the movement of the molecules is hindered and this reduces the rate of reactions. Hence the maximum number is taken to be 100 and accordingly other molecule numbers are selected. Molecules which are very low in concentration are assigned number 1. A series of experiments are run with various combinations of numbers of molecules. The numbers of molecules in the base case and their lifetimes are given in Table 3 and Table 4.

Molecule	Number
Dshi	100
Dsha	0
apc*/axin*/gsk 3	5
apc/axin/gsk3	3
Gsk3	1
Apc/axin	3
Apc	100
β -catenin/apc*	1
β -catenin */apc*/axin*/g	1
β -catenin *	1
β -catenin	20
Axin	2

Table 3. Numbers of molecules.

The lifetimes of the molecules that decay are calculated by taking the rate of decay and relating it the total simulation time. Table 4 tabulates the lifetimes of the molecules.

Molecule	Lifetime
Dshi	1000
Dsha	10000
Axin	500
β -catenin*	300
β -catenin	100

Table 4. Molecule lifetimes.

Snapshots of simulation

The system was successfully modeled using GenericSystem. Snapshots of the simulation are shown in Figure 6, 7, 8, and 9.

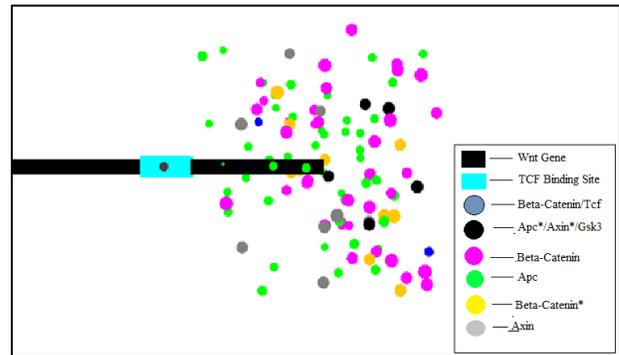


Figure 6. System at time step 500.

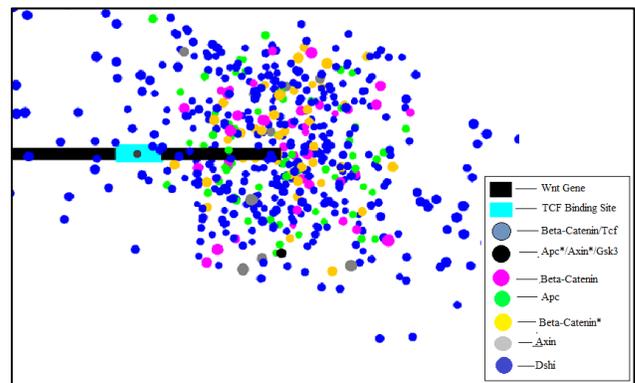


Figure 7. System at time step 900.

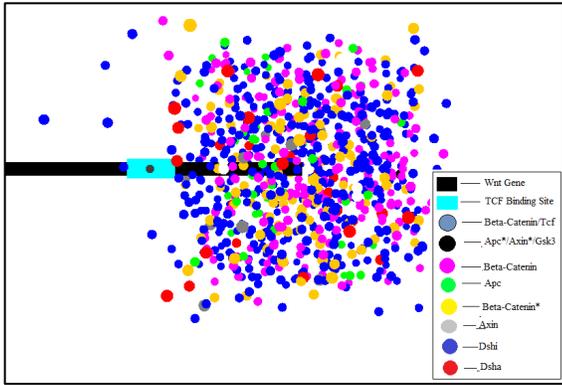


Figure 8. System at time step 1100.

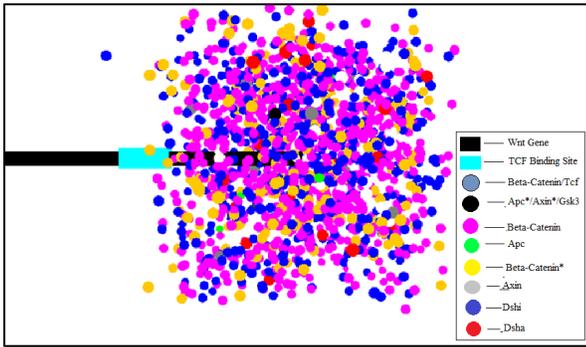


Figure 9. System at time stop 7000.

Description of Results and Comparison to Literature

The numbers of the molecules β -catenin and Apc*/Axin*/gsk3 have been observed throughout the simulation. As Wnt changes from 0 to 1, the number of β -catenin molecules increases from 20 to 500. The number of Apc*/Axin*/gsk3 molecules decreases from 5 to 1. These results are plotted in Figures 10 and 11

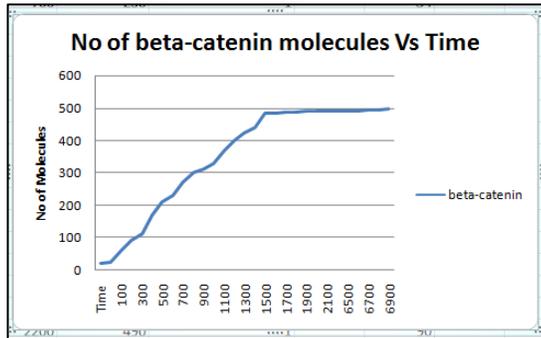


Figure 10. β -catenin molecules vs time.

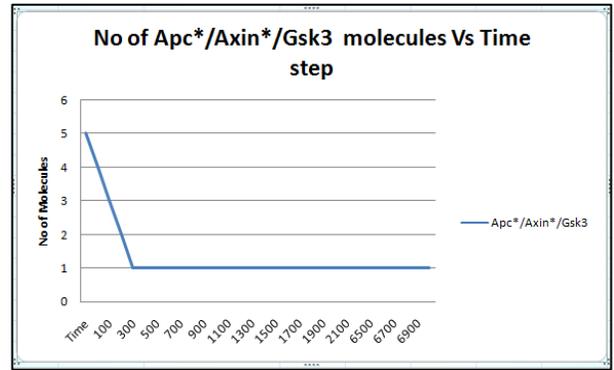


Figure 11. Apc*/Axin*/gsk3 molecules vs time.

The results obtained by the differential equations method in [19] are shown in Figures 12 and 13. We see that in this case the ABM simulation results match well with the results in [19].

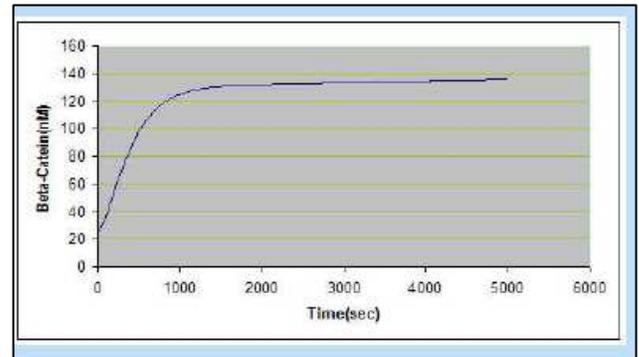


Figure 12. Concentration of β -catenin vs time [19].

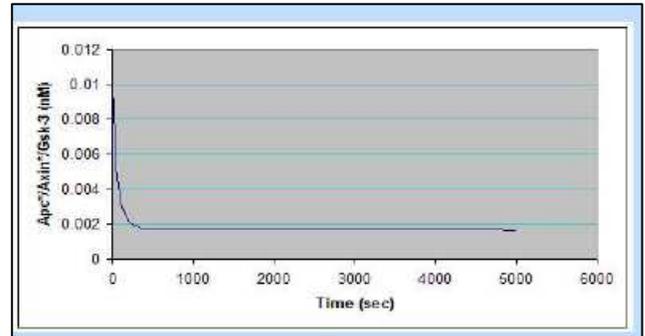


Figure 13. Concentration of Apc*/Axin*/Gsk3 vs time [19].

Conclusions and Future Work

Here we have compared our ABM model with a differential equations model. In the case we considered, we see that qualitatively we are getting the same behavior. Similar results were obtained for the other systems mentioned above. Details about these systems can be found in [21]. The agent based model and stochastic

modeling were compared by Karkutla in [20]. Karkutla also showed that ABM can be used to simulate nonhomogeneous systems, which cannot be simulated accurately by either stochastic or differential equations, and he demonstrated that for cases where both ABM and stochastic simulation can be applied, the results also compare well quantitatively.

GenericSystem can use the graphical display feature of MASON to produce animations of the models under consideration. Using this feature, we are working on developing realistic animations of a version of the skin cell example. We are also working on extending the system to model more complex dynamic behavior, for example DNA self-assembly and nanotube growth. A tool that could simulate such phenomena in a cost-effective way would be very useful in supporting virtual experiments involving novel materials for future generation computer elements. Accurate modeling of fine-grained dynamic behavior will require additional computational resources. Thus another question we are studying is how to accurately characterize the relative costs of ABM simulation versus stochastic or differential equation simulations. ABM methods are most effective for fine-grained behavior and low concentrations of molecules. A method to quantify this statement for a given example would be very useful.

References

1. E. Bonabeau, Agent-based modeling: Methods and techniques for simulating human systems, *Proceedings of the National Academy of Sciences*, vol. 99, May 2002, pp. 7280-7287.
2. H. Van Dyke Parunak, Robert Savit, and Rick L. Riolo, Agent-based modeling vs. equation-based modeling: a case study and users' guide, *Proceedings of the First International Workshop on Multi-Agent Systems and Agent-Based Simulation*, July 1998, pp. 10-25.
3. James W. Haefner, *Modeling Biological Systems: Principles And Applications*, Springer, 1996, pp. 129-132
4. Mor Peleg, Daniel Rubin, and Russ B. Altman, Using Petri net tools to study properties and dynamics of biological systems, *Journal of American Medical Information Association*, 12(2), Mar-Apr 2005, pp. 181-199.
5. A. Regev, W. Silverman, and E. Shapiro, Representation and simulation of biochemical processes using the pi-calculus process algebra, *Pacific Symposium on Biocomputing*, 6, 2001, pp. 459-470.
6. S. Khan, R. Makkena, F. McGeary, K. Decker, W. Gillis and C. Schmidt, A multi-agent system for the quantitative simulation of biological networks, *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*, Melbourne, Australia, July 14-18, 2003.
7. http://www.dcs.shef.ac.uk/~rod/Integrative_Systems_Biology.html. Accessed 06/02/2009
8. I. Politopoulos, *Review and Analysis Of Agent-Based Models In Biology*, 2007.
9. <http://cs.gmu.edu/~eclab/projects/mason/> Accessed 05/10/2009.
10. S. Luke, C. Cioffi-Revilla, L. Panait, and K. Sullivan MASON: a new multi-agent simulation toolkit, *SwarmFest Workshop*, 2004
11. V. Vallurupalli and C. Purdy, "Agent based modeling and simulation of biomolecular reactions", *Scalable Computing: Practice and Experience (SCPE)*, 8(2), 2006, pp. 185-196.
12. Krupa Sagar Mylavarapu, Agent based model of bioluminescence in vibrio fischeri, Master's Thesis, University of Cincinnati, Ohio, Oct 2008.
13. Balakumar Rajendran, 3D agent based model of cell growth, Master's Thesis, University of Cincinnati, Ohio, Dec 2008
14. Paul Davidsson, Agent based social simulation: a computer science view, *Journal of Artificial Societies and Social Simulation* 5 (1), 2002.
15. B. Bauer, J. P. Muller, and J. Odell, Agent UML: A formalism for specifying multiagent interaction, *Agent-Oriented Software Engineering*, Paolo Ciancarini and Michael Wooldridge ed., Springer-Verlag, Berlin, 2001, pp. 91-103.
16. E. Lee, A. Salic, R. Kruger, R. Heinrich, and M.W. Kirschner, The roles of APC and Axin derived from experimental and theoretical analysis of the Wnt pathway, *PLoS Biol* 1(1), 2003, pp. 116-132.
17. Wnt Signaling pathway, http://en.wikipedia.org/wiki/Wnt_signalling_pathway, accessed 10/15/2010.
18. Yue Xiong and Yojiro Kotake, No exit strategy? no problem: APC inhibits β -catenin inside the nucleus, *Genes Dev.* 20, March 15, 2006, pp. 637-642
19. Sravanthi Mailavaram, Database for the study of biological pathways, with Wnt signaling pathway use case, Master's Thesis, University of Cincinnati, Ohio, Nov 2008.
20. Raja Karkutla, Agent based and stochastic simulations for non-homogeneous systems, Master's Thesis, University of Cincinnati, Ohio, March 2010.
21. Harika Korukonda, A generic agent based modeling tool for simulating biomolecular systems, Master's Thesis, University of Cincinnati, Ohio, November 2010.

***Special Session: Artificial Intelligence in
Biometrics and Identity Sciences I***

Chair: Gerry Dozier

GEFeWS: A Hybrid Genetic-Based Feature Weighting and Selection Algorithm for Multi-Biometric Recognition

Aniesha Alford⁺, Khary Popplewell[#], Gerry Dozier[#], Kelvin Bryant[#], John Kelly⁺,
Josh Adams[#], Tamirat Abegaz[^], and Joseph Shelton[#]

Center for Advanced Studies in Identity Sciences (CASIS@A&T)

⁺Electrical and Computer Engineering Department,

[#]Computer Science Department

[^]Computational Science and Engineering Department

North Carolina A & T State University

1601 E Market St., Greensboro, NC 27411

aalford@ncat.edu, ktpopple@ncat.edu, gvdozier@ncat.edu, ksbyrant@ncat.edu, jck@ncat.edu,
jcadams2@ncat.edu, tamirat@programmer.net, jashelt1@ncat.edu

Abstract

In this paper, we investigate the use of a hybrid genetic feature weighting and selection (GEFeWS) algorithm for multi-biometric recognition. Our results show that GEFeWS is able to achieve higher recognition accuracies than using genetic-based feature selection (GEFeS) alone, while using significantly fewer features to achieve approximately the same accuracies as using genetic-based feature weighting (GEFeW).

Introduction

A biometric system is a pattern recognition system that uses physiological and behavioral traits, characteristics that are unique for every individual, to perform recognition (Jain, Ross, and Prabhakar 2004). The value of a biometric system depends largely on its ability to accurately authenticate an individual. Thus, the recognition accuracy is a major concern and is a key area of research for the biometrics community (Deepika and Kandaswamy 2009).

Researchers have shown that biometric systems that use only one biometric modality can produce highly accurate results (Adams et al. 2010; Dozier et al. 2009; Miller et al. 2010; Ross 2007). However, when these systems are applied to real-world applications, their performance can be affected by numerous factors such as noisy sensor data due to dust or lighting conditions and spoofing. Multi-biometric systems that fuse multiple biometric modalities have been shown to be more robust, able to counter many of the aforementioned limitations, and are also capable of achieving higher recognition accuracies (Jain, Nandakumar, Ross 2005; Ross 2007; Eshwarappa and Latte 2010).

Feature selection and weighting have also been proven as successful methods of improving the accuracy rates of biometric systems (Adams et al. 2010; Dozier et al. 2009;

Gentile, Ratha, and Connell 2009; Mumtazah and Ahmad 2007). The goal of feature selection is to reduce the dimensionality of a data set by discarding features that are inconsistent, irrelevant, or redundant; thus keeping those features that are more discriminative and contribute the most to recognition accuracy. Feature weighting is a more general case of feature selection, with each feature being assigned a weight based on its relevance (Yang and Honavar 1998).

Genetic and Evolutionary Computation (GEC) has been utilized by researchers to optimize feature selection and weighting (Hussein et al. 2001; Yang and Honavar 1998; Yu and Liu 2003; Tahir et al. 2006; Raymer et al. 2000) and has also been used by the biometrics community to optimize the recognition accuracy (Dozier et al. 2009; Adams et al. 2010; Giot, El-Abed, and Rosenberger 2010). The goal of GEC is to find the optimal or near optimal solution to a problem, and typically works as follows. A population of candidate solutions is generated randomly and assigned a fitness based on a user-defined function. Using this fitness, members of the population are chosen and reproduce. The resulting offspring are then evaluated and typically replace candidate solutions within the population that have a lower fitness. This evolutionary process is continued until the population converges, a user-specified number of evaluations have completed, or no solution can be found.

In this paper, we use a hybrid GEC-based feature weighting and selection (GEFeWS) technique for multi-biometric recognition. Our goal is to reduce the number of features necessary for biometric recognition and increase the recognition accuracy. The performance of GEFeWS is compared with the performances of genetic-based feature selection (GEFeS) and weighting (GEFeW) techniques individually. The modalities tested were face and periocular biometrics. The facial features were extracted using the Eigenface method (Turk and Pentland 1991; Lata et al. 2009), and the periocular features were extracted

using Local Binary Patterns (LBP) (Adams et al. 2010; Miller et al. 2010).

This research is inspired in part by the proposal of a hierarchical two-stage system, presented by Gentile et al. for iris recognition (2009). This system used a reduced feature set size in an effort to reduce the total number of feature checks required for an iris-based biometric recognition system. For a conventional biometric recognition system, a probe, p , is compared to every individual within a biometric database. The number of feature checks performed by a conventional biometric system, f_c , is:

$$f_c = nm$$

where n is the number of individuals in the database and m is the number of features used to represent an individual. A hierarchical biometric system reduces the number of feature checks performed by first using the reduced length biometric template to select a subset of the r closest matches to the probe p . The subset is then compared to p using all of the m features. The number of feature checks performed by a hierarchical system, f_h , is the summation of the calculations of the two stages, represented by:

$$f_h = nk + rm$$

where, once again, n represents the number of individuals in the database, k is the number of features in the reduced feature set, r is the subset of the closest r -individuals to the probe, p , and m is the number of features used to represent an individual. The savings gained by using the hierarchical biometric system, f_s , instead of the conventional biometric system is:

$$f_s = \frac{f_h}{f_c} = \frac{nk + rm}{nm} = \frac{k}{m} + \frac{r}{n}$$

The remainder of this paper is as follows. In the following section, a brief overview of the feature extractors used for our experiments is given. GEFeS, GEFeW, and GEFeWS are then described, followed by a description of our experiments, the presentation of our results, and finally, our conclusions and future work.

Feature Extraction

Feature extraction is one of the essential tasks performed by a biometric system. After a biometric sample is acquired from an individual, feature extraction is performed to extract a set of features, termed a feature template, which is used to represent the individual and is used in the comparisons to determine recognition (Jain, Ross, and Prabhakar 2004).

In this paper, we use two feature extraction schemes. The Eigenface method is used to extract features from the face (Turk and Pentland 1991; Lata et al. 2009). Local

Binary Patterns (LBP) is used to extract features from the periocular region (Adams et al. 2010; Miller et al. 2010).

Eigenface is based on the concept of Principal Component Analysis (PCA) and has been proven successful for facial recognition (Turk and Pentland 1991; Lata et al. 2009). PCA is a method used to reduce the dimensionality of a dataset while retaining most of the variation found among the data (Jolliffe 2005). For the Eigenface method, PCA is used to find the principal components, or eigenfaces, of the distribution of the face images within the entire image space, which is called the face space.

LBP is a method used for texture analysis that has been used in many biometric applications, including the extraction and analysis of periocular features for identification (Adams et al. 2010; Miller et al. 2010). LBP descriptors of each periocular region are formed by first segmenting the image into a grid of 24 evenly sized patches. Every internal pixel within the patch is used as a center pixel. The intensity change of the pixels around the center pixel is measured by subtracting the intensity value of the center pixel from each of the P neighboring pixels. For our experiments, the neighborhood size, P , was 8. If the resulting value is greater than or equal to 0, a 1 would be concatenated to the binary string representing the texture, otherwise a 0. The texture is then encoded into a histogram where each bin represents the number of times a particular binary string appears in a patch. For optimization purposes, only uniform patterns are considered. These are binary string patterns with at most two bitwise changes when the pattern is traversed circularly. Therefore, our histogram consisted of 59 bins (instead of $2^P=256$ bins), 58 for the possible uniform patterns and 1 for the non-uniform patterns.

GEFeS, GEFeW, and GEFeWS

The genetic and evolutionary techniques used within this paper are based on the eXploratory Toolset for the Optimization of Launch and Space Systems (X-TOOLSS) (Tinker, Dozier, and Garrett 2010), and are an instance of the X-TOOLSS Steady-State Genetic Algorithm (SSGA).

For GEFeS, a SSGA is used to evolve a feature mask that selects the most salient biometric features. For each real-valued candidate solution that is generated by the SSGA, a masking threshold of 0.5 is used to determine if the feature is used. If the values of the features within the mask are less than the masking threshold, the feature is turned off by setting the mask value to 0. Otherwise, the feature is turned on by setting the mask value to 1, resulting in a binary coded feature mask.

For GEFeW, a SSGA is used to evolve a real-valued feature mask composed of values between 0.0 and 1.0. The resulting feature mask value is multiplied by each feature value to provide the weighted feature.

GEFeWS is a hybrid of GEFeW and GEFeS. Like GEFeW, a SSGA is used to evolve the weight of the features. However, if the weight is less than the masking

threshold of 0.5, then the feature is not included, basically being turned off as done by GEFeS. Otherwise, the feature is weighted as done by GEFeW.

Associated with each candidate feature mask, i , there were two weights, w_{ip} and w_{if} , which are weights for the periocular and face feature submasks to allow for score-level fusion. The weights ranged from [0..1] and were co-evolved with the rest of the feature mask.

Experiment

To test our algorithms, we used a subset of 315 images taken from the first 105 subjects of the Face Recognition Grand Challenge (FRGC) dataset (Phillips et al. 2005). These images were used to form a probe set of 105 images (one of each subject) and a gallery set of 210 images (two of each subject). For each of the images in the probe and gallery set, the Eigenface method was used to extract 210 face features, and the LBP method was used to extract 2832 periocular features (1416 features for each eye).

Three biometric modalities were tested: face, periocular, and face plus periocular. For each of the three biometric modalities, GEFeS, GEFeW, and GEFeWS were used. The biometric modalities were also tested using all of the originally extracted features without the use of GECs. This served as a control/baseline for our experiments.

Results

For our experiments, the SSGA had a population size of 20 and a Gaussian mutation range of 0.2. The algorithm was run 30 times, and a maximum of 1000 evaluations were performed on each run.

In Table I, the average performance of the three experiments is shown. The first column represents the tested biometric modalities. The second column represents the type of algorithm that was used. The third column represents the average percentage of features used, and the last column represents the average accuracy of the 30 runs.

Table I shows the performance comparison of GEFeS, GEFeW, and GEFeWS. The results using the feature extractors without the GECs were also included to serve as a baseline for the experiments. When the face and periocular biometrics were fused, they both were weighted evenly.

For the Face-Only experiment, GEFeW performed the best in terms of accuracy, having an average accuracy of 87.59%. Based on the results of the ANOVA and t-test, GEFeWS was in the second equivalence class in terms of average accuracy, but there was only a 1.21% difference in the average accuracy for the two algorithms. In terms of the percentage of features used, GEFeWS was in the first equivalence class, along with GEFeS. GEFeWS was able to obtain an average accuracy of 86.38%, while using only 51.71% of the features.

For the Periocular-Only experiment, GEFeWS performed the best in terms of accuracy and the percentage of features used, having an average accuracy of 96.15% while using only 45.39% of the features. These results were confirmed using an ANOVA and t-test. GEFeW was in the second equivalence class in terms of average accuracy. In terms of the percentage of features used, GEFeS and GEFeW were in the second and third equivalence classes respectively.

For the Face + Periocular experiment, GEFeW performed the best in terms of accuracy, while GEFeWS was in the second equivalence class. However, in terms of the percentage of features used, GEFeWS was in the first equivalence class, using only 46.24% of the features to achieve an average accuracy of 98.48% (only a 0.5% difference when compared to GEFeW). GEFeS and GEFeW were in the second and third equivalence classes respectively.

The Face + Periocular experiment performed the best in terms of accuracy for all the algorithms used, followed by the Periocular-Only experiment and the Face-Only experiment.

Modalities Tested	Algorithms Used	Average % of Features Used	Average Accuracy
Face Only	Eigenface	100.00%	64.76%
	Eigenface + GEFeS	51.03%	77.87%
	Eigenface + GEFeW	87.71%	87.59%
	Eigenface + GEFeWS	51.71%	86.38%
Periocular Only	LBP	100.00%	94.29%
	LBP + GEFeS	48.03%	95.14%
	LBP + GEFeW	86.22%	95.46%
	LBP + GEFeWS	45.39%	96.15%
Face + Periocular	Eigenface + LBP [evenly fused]	100.00%	90.77%
	Eigenface + LBP + GEFeS	48.18%	97.40%
	Eigenface + LBP + GEFeW	87.59%	98.98%
	Eigenface + LBP + GEFeWS	46.24%	98.48%

Table 1. Comparison of the performances of GEFeS, GEFeW, and GEFeWS.

For the percentage of features used, GEFeWS used the least amount of features for the Periocular-Only and Face + Periocular experiments, and there was no statistical significance between GEFeWS and GEFeS for the Face-Only experiment. GEFeW used the highest percentage of features for all three experiments.

Conclusion

Our results show that the hybrid GEC, GEFeWS, is able to achieve higher recognition accuracies than GEFeS, while using about the same amount of features. GEFeWS is also able to use a significantly lesser amount of features than GEFeS while achieving approximately the same average recognition accuracy. Overall, the Face + Periocular performed better in terms of accuracy when compared to the Face-Only and Periocular-Only experiments. Our future work will include investigating additional multi-biometric fusion techniques as well as additional GECs in an effort to further improve the performance of multi-biometric recognition. In addition, we will investigate applying these algorithms to a larger dataset to see how well they generalize.

Acknowledgments

This research was funded by the Office of the Director of National Intelligence (ODNI), Center for Academic Excellence (CAE) for the multi-university Center for Advanced Studies in Identity Sciences (CASIS) and by the National Science Foundation (NSF) Science & Technology Center: Bio/computational Evolution in Action Consortium (BEACON). The authors would like to thank the ODNI and the NSF for their support of this research.

References

Adams, J.; Woodard, D.L.; Dozier, G.; Miller, P.; Bryant, K.; and Glenn, G. 2010. Genetic-Based Type II Feature Extraction for Periocular Biometric Recognition: Less is More. In *Proceedings of the 20th International Conference on Pattern Recognition*.

Deepika, C.L. and Kandaswamy, A. 2009. An Algorithm for Improved Accuracy in Unimodal Biometric Systems through Fusion of Multiple Feature Sets, In *ICGST International Journal on Graphics, Vision and Image Processing, (GVIP)*, Volume (9), Issue (III): pp. 33-40.

Dozier, G.; Frederiksen, K.; Meeks, R.; Savvides, M.; Bryant, K.; Hopes, D.; and Munemoto, T. 2009. Minimizing the Number of Bits Needed for Iris Recognition via Bit Inconsistency and GRIT. In *Proceedings of the IEEE Workshop on Computational Intelligence in Biometrics Theory, Algorithms, and Applications, (CIB)*.

Eshwarappa M.N. and Latte, M.V. 2010. Bimodal Biometric Person Authentication System Using Speech and Signature Features. In *Proceedings of the International Journal of Biometrics and Bioinformatics, (IJBB)*, Volume (4), Issue (4).

Gentile, J.E.; Ratha, N.; and Connell, J. 2009. SLIC: Short-Length Iris Codes. In *Proceedings of the IEEE 3rd International Conference on Biometrics: Theory, Applications, and System, (BTAS)*.

Gentile, J.E.; Ratha, N.; and Connell, J. 2009. An Efficient, Two-stage Iris Recognition System. In *Proceedings of the IEEE 3rd International Conference on Biometrics: Theory, Applications, and System, (BTAS)*.

Giot, R.; El-Abed, M; and Rosenberger, C. 2010. Fast Learning for Multibiometrics Systems Using Genetic Algorithms. In *Proceedings of the IEEE International Conference on High Performance Computing and Simulation (HPCS)*.

Hussein, F.; Kharma, N.; and Ward, R. 2001. Genetic Algorithms for Feature Selection and Weighting, a Review and Study. In *Proceedings of the Sixth International Conference on Document Analysis and Recognition*.

Jain, A.K.; Duin, R.P.W.; and Jianchang M. 2000. Statistical Pattern Recognition: A Review. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Volume (22), Issue (1): pp. 4-37.

Jain, A.K.; Ross, A.; and Prabhakar, S. 2004. An Introduction to Biometric Recognition. In *IEEE Transactions on Circuits and Systems for Video Technology*, Volume (14), Issue (1): pp. 4-20.

Jain, A.; Nandakumar, K.; and Ross, A. 2005. Score Normalization in Multimodal Biometric Systems. *Pattern Recognition*, Volume (38), Issue (12): pp. 2270-2285.

Jolliffe, I. 2005. Principal Component Analysis. *Encyclopedia of Statistics in Behavioral Science*.

Kohavi, R.; Langley, P.; and Yun, Y. 1997. The Utility of Feature Weighting in Nearest-Neighbor Algorithms. In *Proceedings of the 9th European Conference on Machine Learning (ECML)*.

Lata, Y.V.; Tungathurthi, C.; Rao, R., Govardhan, A.; and Reddy, L.P. 2009. Facial Recognition using Eigenfaces by PCA. In *International Journal of Recent Trends in Engineering*, Volume (1), Issue (1): pp. 587-590.

Miller, P.E.; Rawls, A.; Pundlik, S.; and Woodard, D. 2010. Personal Identification Using Periocular Skin Texture. In *Proceedings of the 2010 ACM Symposium on Applied Computing*.

Mumtazah, S. and Ahmad, S. 2007. A Hybrid Feature Weighting and Feature Selection Approach in an Attempt to Increase Signature Biometrics Accuracy. In *Proceedings of the International Conference on Electrical Engineering and Informatics*.

Phillips, P.J.; Flynn, P.J.; Scruggs, T.; Bowyer, K.W.; Chang, J.; Hoff, K.; Marques, J.; Min, J.; and Worek, W. 2005. Overview of the Face Recognition Grand Challenge. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Raymer, M.L.; Punch, W.F.; Goodman, E.D.; Kuhn, L.A.; Jain, A.K. 2000. Dimensionality Reduction Using Genetic Algorithm. In *IEEE Transactions on Evolutionary Computation*, Volume (4), Issue (2): pp. 164-171.

Ross, A. 2007. An Introduction to Multibiometrics. In *Proceedings of the 15th European Signal Processing Conference (EUSIPCO)*.

Tahir, M.A., et al., 2006. Simultaneous Feature Selection and Feature Weighting using Hybrid Tabu Search/K-Nearest Neighbor Classifier. *Pattern Recognition Letters*. Volume (28), Issue (4): pp. 438-446.

Tinker, M.L.; Dozier, G.; and Garrett, A. 2010. The eXploratory Toolset for the Optimization Of Launch and Space Systems X-TOOLSS). <http://xtoolss.msfc.nasa.gov/>.

Turk, M. and Pentland, A. 1991. Eigenfaces for Recognition. *Journal of Cognitive Neuroscience*. Volume (3), Issue (1): pp. 76-81.

Yang, J. and Honavar, V. 1998. Feature Subset Selection Using a Genetic Algorithm, In *Proceedings of the IEEE Intelligent Systems and their Applications*. Volume (13), Issue: (2): pp. 44-49.

Yu, L. and Liu, H. 2003. Feature Selection for High-Dimensional Data: A Fast Correlation-Based Filter Solution, In *Proceedings of the Twentieth International Conference on Machine Learning*.

Iris Quality in an Operational Context

James S. Doyle, Jr. and Patrick J. Flynn

Department of Computer Science and Engineering

University of Notre Dame

{jdoyle6, flynn}@nd.edu

Abstract

The accuracy of an iris biometrics system increases with the quality of the sample used for identification. All current iris biometrics systems capture data streams that must be processed to identify a single, ideal image to be used for identification. Many metrics exist to evaluate the quality of an iris image. This paper introduces a method for determining the ideal iris image from a set of iris images by using an iris-matching algorithm in a feedback loop to examine the set of true matches. This proposed method is shown to outperform other methods currently used for selecting an ideal image from a set of iris images.

Introduction

Biometrics is the use of one or more intrinsic characteristics to identify one person to the exclusion of others. There are many characteristics that can be used as biometrics, individually or combined in some manner to produce a hybrid biometric. Identification using the ear (Yan & Bowyer 2007), gait (L. Wang et al. 2003), and even body odor (Korotkaya) have been studied alongside more traditional biometrics such as face (Bowyer, Chang, & Flynn 2006), fingerprint (Chapel 1971), and iris (Bowyer, Hollingsworth, & Flynn 2008). The human iris is considered to be a mature biometric, with many real-world systems (LG 2010; Daugman & Malhas 2004; UK Border Agency 2010; Life Magazine 2010; Welte 2010) already deployed. The system designed by John Daugman (Daugman 2002) was documented to achieve a false match rate of zero percent in a particular application. In larger deployments there is still room for improvement.

Iris biometric systems can be used for identification purposes in which no claim of identity is presented, or for verification purposes where a sample and a claim of identification are supplied for the system to verify. For positive identification, both methods require that the subject be previously enrolled in the system, making him or her part of the gallery. Another sample, known as the probe, will be taken at the time of the identification attempt and compared to gallery samples. If the new probe sample closely resembles one of the gallery samples, the system reports a positive ID. If no gallery sample matches the probe sample close enough, the system will not report a match. For verification, an identity claim is also presented along with a probe sample, allowing

the system to only consider samples with the requested ID. If these probe-gallery comparisons meet the threshold set for the system, then the ID claim is verified.

One method that can increase the correct recognition rate of iris biometric systems is to select the highest quality sample of a subject to be used as the representative in the gallery. Some systems have an enrollment process that takes multiple samples and then uses the highest quality sample based on certain criteria, or quality metrics, for the gallery. Samples may also be enhanced after acquisition via contrast stretching, histogram normalization, image de-blurring, or by other methods.

The Iridian LG EOU 2200 used to capture biometric samples at the University of Notre Dame is capable of outputting video as well as still images when enrolling subjects. Multiple methods have been applied to the iris videos recorded with this camera in order to extract the frame of highest quality for enrollment. For instance, the IrisBEE (Liu, Bowyer, & Flynn 2005) software, discussed in Section 4, uses sharpness to determine the focus of a frame. It computes a focus score that can be used to rank the frames, and accordingly pick the sharpest frame. It will be shown later that this approach is not optimal.

This paper suggests a new method for quality-based ranking of a set of iris biometrics samples, shows that the new method outperforms the previous quality metric, and offers possible optimizations to reduce the run-time complexity of determining the sample ranking.

Related Work

The National Institute of Standards and Technology (Tabassi, Grother, & Salamon 2010) has tentatively defined the quality of an iris image over 12 dimensions, including contrast, scale, orientation, motion blur, and sharpness. However, the definitions of these quality metrics, as well as their implementation, are still undergoing debate.

Kalka et al. (Kalka *et al.* 2006) have examined multiple quality metrics for iris images, including defocus blur, motion blur, and off-angle gaze, as well as others. Using the Dempster-Shafer theory, a fused score was calculated for all tested quality metrics. This fused score was taken as the overall quality score for an image. Using a cutoff value, it was shown that the performance of a matching system was improved when only images with scores above the cutoff

were included in the experiment.

Kalka et al. (Kalka *et al.* 2010) have extended their work to a fully-automated program to evaluate iris quality. The program creates a single, fused score, based on multiple criteria. Using this program to remove poor-quality samples from multiple datasets, they were able to demonstrate an improvement in recognition performance.

Kang and Park (Kang & Park 2005) propose a method of restoring images of poor quality due to defocus blur. The experimental method discussed in the paper showed a decrease in Hamming distances when the original image was compared to the restored image versus when the original image was compared to an artificially defocused image.

Experimental Setup

Data Acquisition

The method proposed in this paper is general, and able to be applied to any set of iris samples from the same subject. Iris videos were used in this paper as a means of capturing large amounts of same-subject samples.

All iris samples were captured using the Iridian LG EOU 2200 video camera. The LG 2200 uses three near-infrared LEDs to illuminate the eye during acquisition: one above the eye, one to the bottom left of the eye and one to the bottom right of the eye. The LG 2200 uses only one illuminant at a time to reduce spectral highlights. Software on a workstation selects one candidate frame under each lighting scenario using the proprietary IrisAccess software, of which one would be enrolled. The LG 2200 also outputs an NTSC video stream, which is recorded and encoded into a high-bit-rate MPEG-4 video. All videos captured from this device were interlaced and recorded at a constant resolution of 720x480 pixels. Figure 1 depicts the details of the iris video acquisition setup.

All videos were captured in the same windowless indoor lab under consistent lighting conditions. Subjects were supervised during acquisition to ensure proper acquisition procedure was followed. Each video was captured while the subject was being enrolled by the IrisAccess software.

Data Manipulation

The conversion of the raw data stream to video has a noticeable effect on the quality of the images. Still frames from the videos are stretched slightly along the X-axis due to digitizer timing. This was corrected by shrinking the images 5% in the X direction before experimentation began. Images recorded at 720x480 became 684x480. Additionally, a contrast-stretching algorithm was applied to all the images such that 1% of the output pixels were white and 1% were black. These two steps were helpful in improving the quality of the input data set. Figure 2 shows an example frame in original, contrast stretched, and a resized states.

Data Selection

To evaluate the performance of quality metrics for iris biometrics in an operational environment, 1000 test videos were chosen from 11751 videos acquired from 2007-2009 and stored in the BXGRID biometrics database at University of

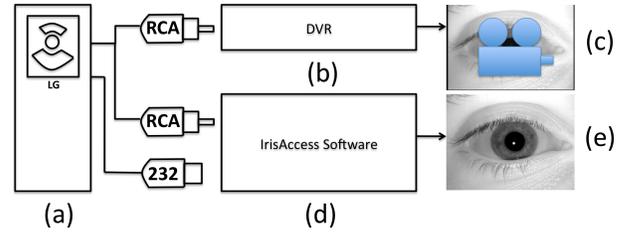


Figure 1: The LG 2200 iris camera (a) outputs one NTSC video data stream that is split and amplified using a powered splitter. The NTSC data stream is recorded by a DVR (b). This method is used to capture iris videos (c). The IrisAccess software (d) running on a workstation monitors the NTSC signal. This method is used to capture still images (e).

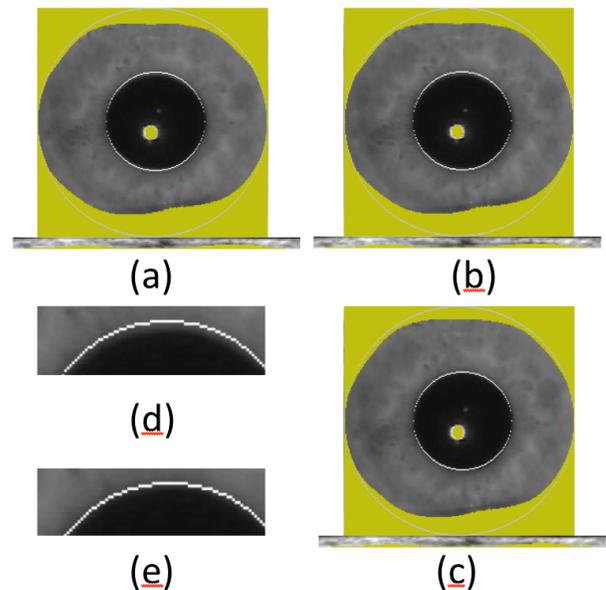


Figure 2: The three image types discussed in the paper are shown, with white circles indicating the segmentation and yellow regions representing spectral highlight removal and eyelash/eyelid masking. Image (a) is from video 05707d173 in its unmodified state. Image (b) is the contrast stretched version of (a). Image (c) is (b) scaled down by 5% in the X direction. Image (d) shows a close-up of image (a), demonstrating the segmentation error and necessity for resizing. Image (e) shows the same region after resizing.

Notre Dame (Bui *et al.* 2009). This video set included five separate videos for each of 200 subject irises. Because the time required to capture ideal frames illuminated by each light source was variable, the video lengths were not constant. The average video length was 620 frames, with a minimum of 148 frames and a maximum of 2727 frames. No effort was made to screen the data set to eliminate videos of especially poor quality, to keep the test system as close to real-world as possible.

Biometrics Software Packages

IrisBEE

IrisBEE, a modified version of the system originally developed by Masek (Masek 2003), modified by Liu (Liu, Bowyer, & Flynn 2006), released by NIST as part of the ICE challenge dataset, and further modified by Peters (Peters, Bowyer, & Flynn 2009), was used to identify iris boundaries as well as perform matching experiments. IrisBEE contains three executables, one for segmenting iris images and producing iris codes, one for performing matching experiments, and one for calculating rudimentary quality scores, based on image sharpness.

The IrisBEE IrisPreprocessor (Liu, Bowyer, & Flynn 2005) uses computer vision algorithms to detect an iris in an image. A Canny edge detector and Hough transform are used to identify the iris-pupil and iris-sclera boundaries. Active contours (Daugman 2007) are used to further refine the region borders. Two non-concentric circles are fitted to the contour to represent these two boundaries. The iris region formed by these circles is transformed into a rectangle through a polar-Cartesian conversion. Each row of the enrolled image is convolved with a one-dimensional log-Gabor filter. The complex filter response forms the iris code used for matching. A fragile bit mask (Hollingsworth, Bowyer, & Flynn 2007) is applied to allow the more stable regions of the iris code to be used in comparisons. Masking fragile bits improves the match rate, allowing for better match results than when comparing iris codes without masking fragile bits.

IrisBEE also supplies an executable for matching a gallery list of iris codes to a probe list of iris codes. The IrisMatcher outputs a fractional Hamming distance, using formula (1), for each individual comparison as well as the number of bits that were compared between the two codes, useful in normalization (Daugman 2007). All matching results from every experiment were normalized using equation (2) (Daugman 2007). Normalized fractional Hamming distances are referred to as “matching scores” throughout the rest of this paper.

For the purposes of predicting performance of a certain image in the IrisMatcher, IrisBEE provides a QualityModule executable that can rate images based on the sharpness of the whole image or just the iris region if it has been defined. The used by the QualityModule is described by Kang and Park (Kang & Park 2007). The sum of the response at each pixel of the input image is used as the image’s score. The higher the score, the better that image’s rank.

$$HD_{raw} = \frac{|(codeA \otimes codeB) \cap maskA \cap maskB|}{|maskA \cap maskB|} \quad (1)$$

$$HD_{norm} = 0.5 - (0.5 - HD_{raw}) \sqrt{\left(\frac{n}{900}\right)} \quad (2)$$

Neurotechnology VeriEye

A commercially-available biometrics package, Neurotechnology VeriEye (version 2.2), was also used for segmenting iris images and matching iris templates. Since VeriEye is a proprietary software package, details about the segmentation and matching algorithms are not available. The VeriEye matching algorithm reports match scores from 0 to 3235, higher scores indicating better matches. If the VeriEye matcher determines a pair of templates to be of different irises based on a threshold, it reports a match score of 0. For all experiments discussed here, this threshold was disabled to capture raw match scores, unlike the Hamming distance scores reported by IrisBEE. Input to the VeriEye matcher is order dependent, different match scores can be observed depending on the order of gallery and probe. For this paper, only one matching score was considered, with the older of the two images being the gallery image and the newer of the two being the probe image.

Smart Sensors MIRLIN

MIRLIN, another closed-source biometrics package was used to segment iris images and match iris templates, as well as to rate images based on four common quality metrics. Since MIRLIN is proprietary, specific details about its segmentation and matching algorithms as well as its quality metrics are not available. MIRLIN does provide matching scores as Hamming distances, but does not supply the number of bits used in the comparison, making normalization impossible. As a result, matching scores from MIRLIN can not be directly compared to those produced by IrisBEE. Matching scores are also symmetrical, so comparison order is not important. The four quality metrics that are discussed in this work are contrast, saturation, sharpness, and signal-to-noise ratio. MIRLIN also reports the average graylevel and occlusion percentage, but these quality metrics were not useful in classifying images since they had a very small range.

Quality Metric Experiments

Multiple quality metrics were considered: the IrisBEE QualityModule, the MIRLIN quality metrics, and the method proposed here, evaluated using IrisBEE, VeriEye, and MIRLIN. Each separate quality metric was evaluated in a similar manner so that results could be compared experimentally across the metrics.

IrisBEE Quality Module

The IrisBEE QualityModule is the current metric used at the University of Notre Dame to determine an ideal image or subset of images to be used in matching experiments from

an input set. Since the QualityModule processes individual images, the 1000 subject videos were split into individual images. The images were then segmented using the IrisPre-processor to identify the iris region of each image. Images that failed to segment were not included in the matching experiment. After segmentation, every frame f of a single video was given a quality score f_s by the QualityModule, higher scores indicating higher quality images. The frame scores were then sorted from highest to lowest such that $f_s[i] \geq f_s[i + 1] \forall i$.

To test whether this method is predictive of performance, the entire range of scores must be included in an all-vs-all matching. Due to the scale of this experiment, a subset of nine images was chosen from each video: the top-ranked image, the bottom-ranked image and seven equally spaced images in between. Selecting images in this manner controlled for the inconsistent video length of the data set. This reduced dataset was used in an all-vs-all matching experiment using the IrisMatcher. Receiver Operating Characteristic (ROC) curves were then created for each octile. These ROC curves can be found in Figure 3 and Figure 4.

With the exception of the top-ranked images, recognition performance was monotonically decreasing as QualityModule rank increased. The top-ranked images chosen by the QualityModule do not perform well in the matching experiment. As the QualityModule will recommend images with high sharpness, images with eyelid/eyelash occlusion will have artificially high scores. This causes some poor quality images to be highly ranked. Figure 5 shows a top-ranked frame with high occlusion and a lower-ranked frame more ideal for matching experiments, illustrating the drawbacks to reliance on the IrisBEE metric and other metrics that estimate image quality only.

MIRLIN Quality Metrics

The 1000 subject videos were split into individual images and segmented using MIRLIN to identify the iris region of each image. Images that failed to segment were not included in the matching experiment. After segmentation, every frame f of a single video was given four quality scores $f_{contrast}$, $f_{saturation}$, $f_{sharpness}$, and f_{snr} , by MIRLIN. Four rankings were determined for each video: $f_{contrast}[i] \geq f_{contrast}[i + 1] \forall i$, $f_{saturation}[i] \leq f_{saturation}[i + 1] \forall i$, $f_{sharpness}[i] \geq f_{sharpness}[i + 1] \forall i$, and $f_{snr}[i] \leq f_{snr}[i + 1] \forall i$.

The same experimental setup as was used in the IrisBEE Quality Module experiment was used here. The same phenomenon was noticed with all four of the MIRLIN quality metrics studied. In all cases, the Rank 0 frames were outperformed by the Rank $n/8$ frames. ROC curves for these experiments can be found in Figure 6.

IrisBEE IrisMatcher

Since the goal of this research is to find the image or set of images that performs best in matching experiments to represent a subject in a gallery, we investigated the use of the IrisMatcher itself to rate individual frames. To harness the IrisMatcher to pick an ideal representative sample, all frames of

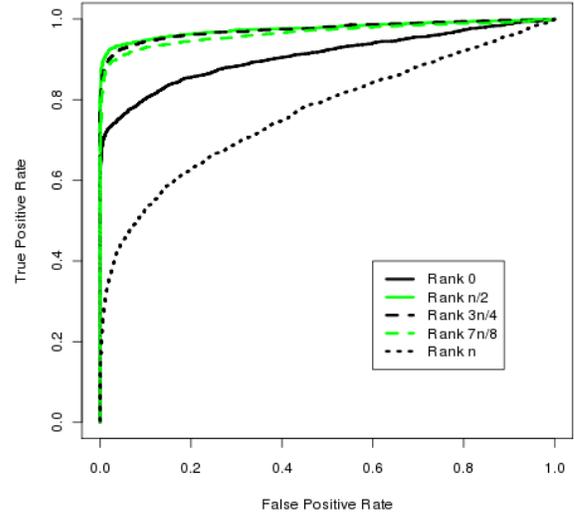


Figure 3: IrisBEE QualityModule experiment results as ROC curves, at select rank octiles. Normalized video length represented by n .

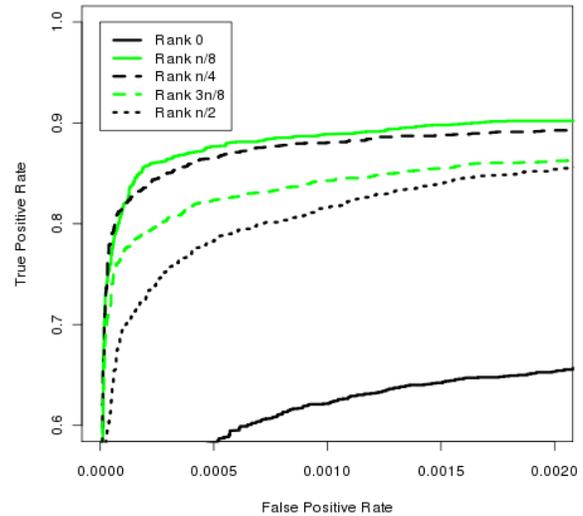


Figure 4: IrisBEE QualityModule experiment results as ROC curves, for selected octiles. Normalized video length represented by n .

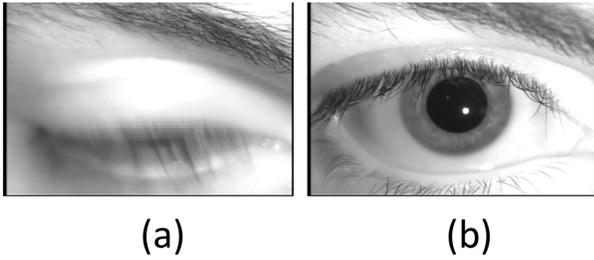


Figure 5: Sample images ranked by the QualityModule, illustrating non-intuitive quality scores from video 05697d222. Image (a) was the highest ranked image by the QualityModule. Image (b) was the 50th ranked frame of 1390 images in the same video.

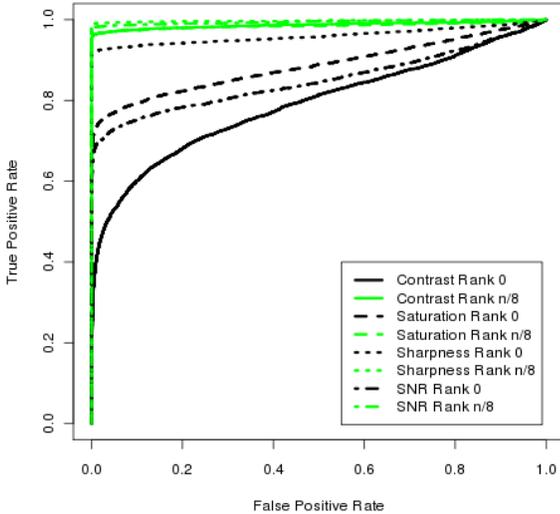


Figure 6: MIRLIN Quality Metrics experiments results as ROC curves, at select rank octiles. Normalized video length represented by n .

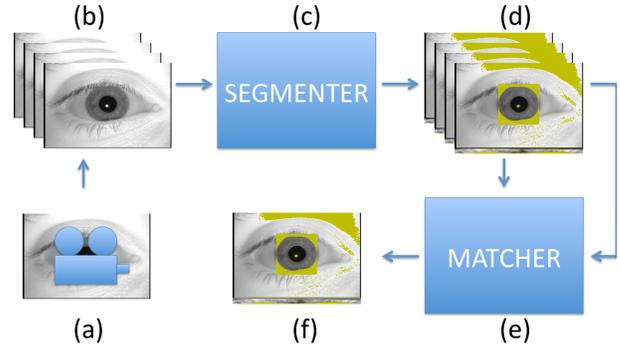


Figure 7: An iris video (a) is broken up into individual frames (b) and processed by the segmenter (c) to identify and mask out occluded regions of the iris. These iris segments (d) are then used as input into the matcher (e), which computes an all-pairs matching matrix. After some processing of the matcher results, an optimal frame can be determined (f).

the video are stored as images. The IrisBEE IrisPreprocessor was used to segment all images and to produce template files for matching. The IrisMatcher performs an all-vs-all matching of the input images for a single video, producing a fractional Hamming distance (1) for each unique pair of images. An average matching score f_s per frame f is found by averaging all matching scores resulting from comparisons involving that image. Since each iris video used in this experiment contained only one subject, all comparisons in this step were true matches. A separate matching is performed for each video. The average matching scores for each frame of a video are sorted lowest-to-highest such that $f_s[i] \leq f_s[i + 1] \forall i$, since low matching scores denote more similar items or better matches. This process is illustrated in Figure 7.

Please refer to Section 5.1 for frame selection method. ROC curves can be found in Figure 8 and Figure 9.

With no exceptions, recognition performance was monotonically decreasing as IrisBEE IrisMatcher rank increased. The amount of separation between ranks in the higher-ranked half of the set was orders of magnitude smaller than the separation of the lower-ranked half.

Neurotechnology VeriEye

The IrisBEE IrisMatcher experiment was repeated using the Neurotechnology VeriEye package. All images were segmented and templates were generated using the VeriEye segmenter. The VeriEye matcher performed an all-vs-all matching of the input images for a single video, producing a matching score between 0 and 3235 for each unique pair of images.

Please refer to Section 5.1 for frame selection method. ROC curves can be found in Figure 10 and Figure 11.

With no exceptions, recognition performance was monotonically decreasing as VeriEye matcher rank increased. The amount of separation between ranks in the higher-ranked

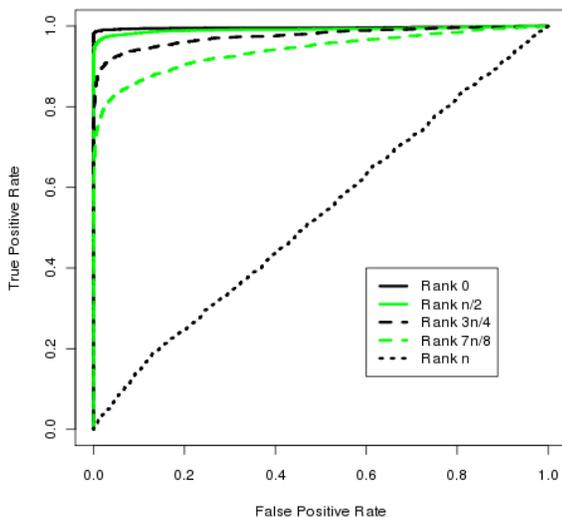


Figure 8: IrisBEE IrisMatcher experiment results as ROC curves, at selected octiles. Normalized video length represented by n .

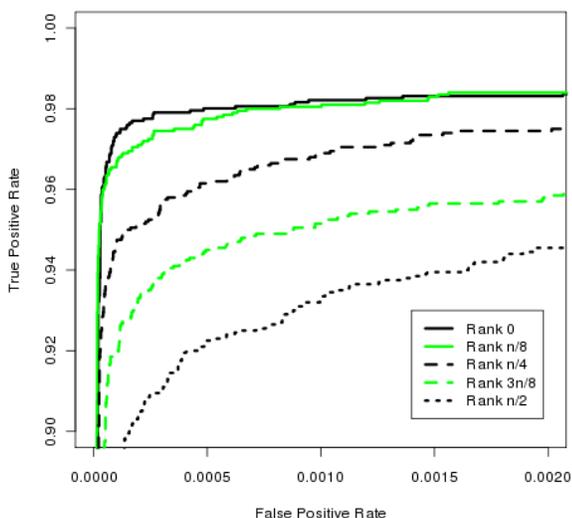


Figure 9: IrisBEE IrisMatcher experiment results as ROC curves, at selective octiles. Normalized video length represented by n .

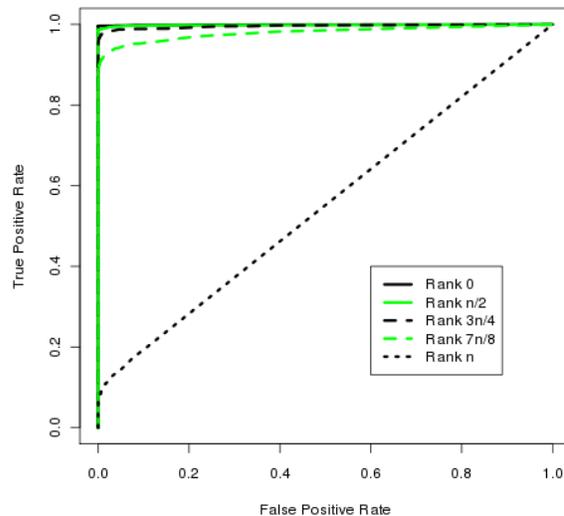


Figure 10: VeriEye matcher experiment results as ROC curves, select octiles. Normalized video length represented by n .

half of the set was orders of magnitude smaller than the separation of the lower-ranked half.

Smart Sensors MIRLIN

The experiment was again repeated using the MIRLIN package. All images were segmented and templates were generated using the MIRLIN `-get` command. The MIRLIN matcher performed an all-vs-all matching of the input images for a single video, using the MIRLIN `-compare` command, producing a matching score in the range $\{0,1\}$ for each unique pair of images.

Please refer to section 5.1 for frame selection method. ROC curves can be found in Figure 12 and Figure 13.

As was the case with IrisBEE and VeriEye, with no exceptions, recognition performance was monotonically decreasing as MIRLIN matcher rank increased. The amount of separation between ranks in the higher-ranked half of the set was orders of magnitude smaller than the separation of the lower-ranked half.

Quality Metric Comparison

For all experiments, there is an ordering, with higher ranked frames performing better in all cases except for the top frame reported by the IrisBEE and MIRLIN quality metrics. Poor performance of the top-ranked frame can be explained by the mechanism in which these quality metrics rank images. Since the quality metrics use image analysis techniques on the iris texture to rate a frame it can be heavily influenced by eyelashes or spectral highlights that were not properly masked, or other artifacts present in an image. These artifacts in turn produce noisy samples as they are blocking

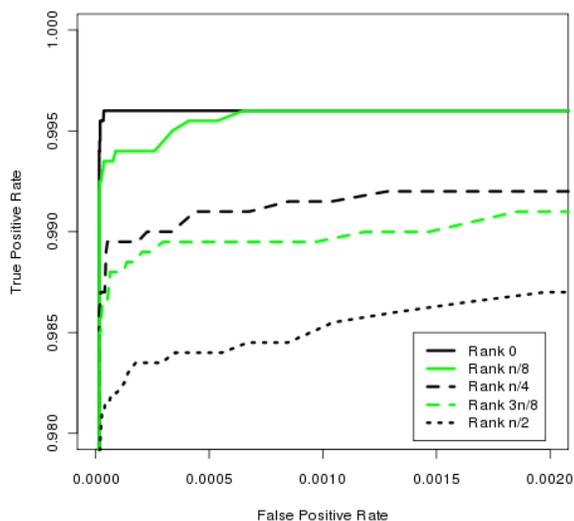


Figure 11: VeriEye matcher experiment results as ROC curves, top octiles. Normalized video length represented by n .

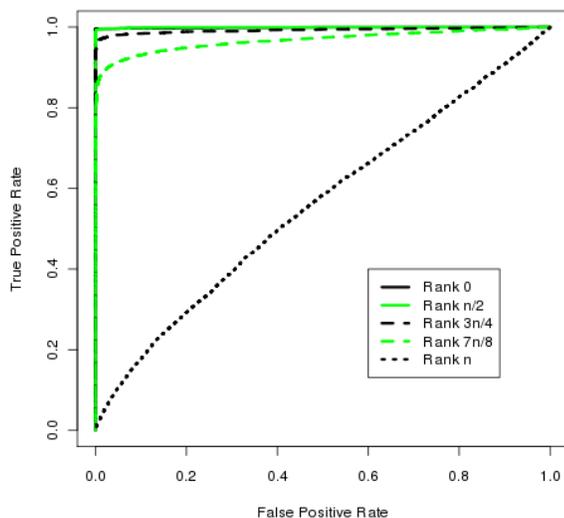


Figure 12: MIRLIN matcher experiment results as ROC curves, select octiles. Normalized video length represented by n .

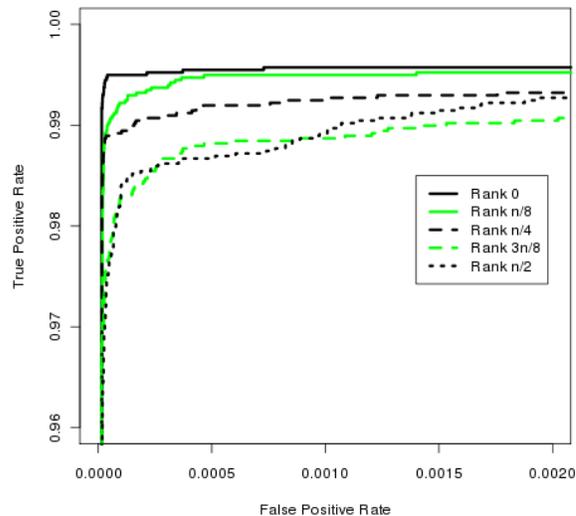


Figure 13: MIRLIN matcher experiment results as ROC curves, top octiles. Normalized video length represented by n .

parts of the iris texture from being compared, artificially skewing the match score higher than it should be for a known match (Bowyer, Hollingsworth, & Flynn 2008). However, even the best images from these quality metrics did not perform as well as the self-matching experiments. The self-matching image selection process can eliminate samples with these artifacts from being used in biometric comparisons by minimizing (or maximizing in the case of VeriEye) the average match scores.

The ordering seen in the ROC curves indicates that the intra-video IrisBEE, VeriEye, and MIRLIN match scores are predictive of inter-video matching performance. Figure 14 shows the ROC curve for the top-ranked frame from each of the metrics, as well as the $n/8$ -ranked frames from each of the quality metrics, as these were the highest performing ranks for the quality metrics.

Application

Data from the LG 2200 iris camera was used in this paper because it allows video streams to be captured easily. Other LG iris cameras, as well as iris cameras from other manufacturers, do not allow video information to be output from the device. However, the use of video in this paper was merely for convenience. The self-matching algorithm could be applied to any set of data captured by the same sensor, including a small set of still images captured by a newer iris camera.

Although production versions of most iris cameras do not allow video to be captured from the device, the proprietary software that interfaces with the camera does capture video information. This method could be applied to the data stream that is processed by the proprietary software. However, as

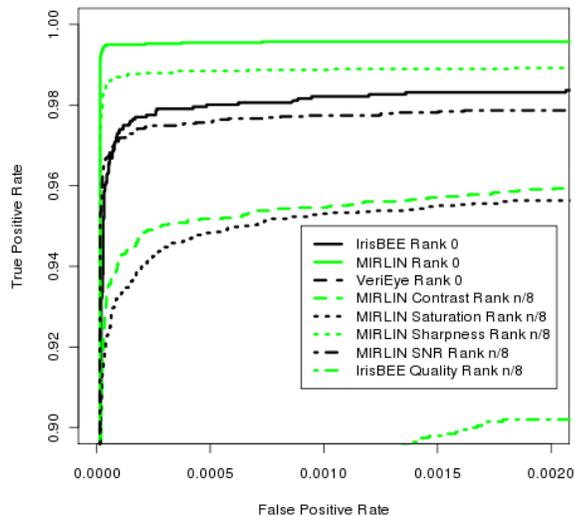


Figure 14: ROC curves of top octiles from all metrics, plus QualityModule and MIRLIN Quality octile n/8 which displayed best performance.

this method is somewhat time consuming, it may only be feasible to apply it during the enrollment phase. Performing this analysis for every probe would delay the response from the system by an unacceptably large amount.

Conclusions

It has been shown through empirical analysis that this method selects an ideal representative sample from a set of same-subject samples. This method outperforms a sharpness-based metric used currently and can be used with a commercially available system.

References

- [Bowyer, Chang, & Flynn 2006] Bowyer, K. W.; Chang, K.; and Flynn, P. 2006. A survey of approaches and challenges in 3d and multi-modal 3d + 2d face recognition. *Computer Vision and Image Understanding* 101(1):1 – 15.
- [Bowyer, Hollingsworth, & Flynn 2008] Bowyer, K. W.; Hollingsworth, K.; and Flynn, P. J. 2008. Image understanding for iris biometrics: A survey. *Computer Vision and Image Understanding* 110(2):281 – 307.
- [Bui et al. 2009] Bui, H.; Kelly, M.; Lyon, C.; Pasquier, M.; Thomas, D.; Flynn, P.; and Thain, D. 2009. Experience with BXGrid: a data repository and computing grid for biometrics research. *Cluster Computing* 12:373–386. 10.1007/s10586-009-0098-7.
- [Chapel 1971] Chapel, C. 1971. *Fingerprinting: A Manual of Identification*. Coward McCann.
- [Daugman & Malhas 2004] Daugman, J., and Malhas, I. 2004. Iris recognition border-crossing system in the UAE. <http://www.cl.cam.ac.uk/~jgd1000/UAEdeployment.pdf>.

- [Daugman 2002] Daugman, J. 2002. How iris recognition works. volume 1, I–33 – I–36 vol.1.
- [Daugman 2007] Daugman, J. 2007. New methods in iris recognition. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* 37(5):1167 –1175.
- [Hollingsworth, Bowyer, & Flynn 2007] Hollingsworth, K.; Bowyer, K.; and Flynn, P. 2007. All iris code bits are not created equal. 1 –6.
- [Kalka et al. 2006] Kalka, N. D.; Zuo, J.; Schmid, N. A.; and Cucic, B. 2006. Image quality assessment for iris biometric. volume 6202, 62020D. SPIE.
- [Kalka et al. 2010] Kalka, N.; Zuo, J.; Schmid, N.; and Cucic, B. 2010. Estimating and fusing quality factors for iris biometric images. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on* 40(3):509 –524.
- [Kang & Park 2005] Kang, B., and Park, K. 2005. A study on iris image restoration. In Kanade, T.; Jain, A.; and Ratha, N., eds., *Audio- and Video-Based Biometric Person Authentication*, volume 3546 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg. 31–40. 10.1007/115279234.
- [Kang & Park 2007] Kang, B. J., and Park, K. R. 2007. Real-time image restoration for iris recognition systems. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* 37(6):1555 –1566.
- [Korotkaya] Korotkaya, Z. Biometric person authentication: Odor. Lappeenranta University of Technology.
- [L. Wang et al. 2003] L. Wang et al. 2003. Silhouette analysis-based gait recognition for human identification. *IEEE Trans. Pattern Analysis and Machine Intelligence* 25(1).
- [LG 2010] LG. 2010. IrisID in Action. <http://irisid.com/ps/inaction/index.htm>.
- [Life Magazine 2010] Life Magazine. 2010. Iris identification used by parents and teachers to protect children in nj elementary school. <http://life.com/image/1965902>.
- [Liu, Bowyer, & Flynn 2005] Liu, X.; Bowyer, K.; and Flynn, P. 2005. Experiments with an improved iris segmentation algorithm. 118 – 123.
- [Liu, Bowyer, & Flynn 2006] Liu, X.; Bowyer, K. W.; and Flynn, P. J. 2006. *Optimizations in Iris Recognition*. Ph.D. Dissertation, University of Notre Dame.
- [Masek 2003] Masek, L. 2003. Recognition of human iris patterns for biometric identification. Technical report, The University of Western Australia.
- [Peters, Bowyer, & Flynn 2009] Peters, T.; Bowyer, K. W.; and Flynn, P. J. 2009. Effects of segmentation routine and acquisition environment on iris recognition. Master’s thesis, University of Notre Dame.
- [Tabassi, Grother, & Salamon 2010] Tabassi, E.; Grother, P.; and Salamon, W. 2010. Iris quality calibration and evaluation 2010. *IREX II IQCE*.
- [UK Border Agency 2010] UK Border Agency. 2010. Using the Iris Recognition Immigration System (IRIS). <http://www.ukba.homeoffice.gov.uk/travellingtotheuk/Enteringtheuk/usingiris>.
- [Welte 2010] Welte, M. S. 2010. Prison system looks to iris biometrics for inmate release. <http://www.securityinfowatch.com/Government+%2526+Public+Buildings/1314995>.
- [Yan & Bowyer 2007] Yan, P., and Bowyer, K. W. 2007. Biometric recognition using 3d ear shape. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29:1297–1308.

Fusion of Face and Iris Biometrics from a Stand-Off Video Sensor

Ryan Connaughton and Kevin W. Bowyer and Patrick Flynn

University of Notre Dame

Department of Computer Science and Engineering
Notre Dame, IN 46556

Abstract

Multi-biometrics, or the fusion of more than one biometric modality, sample, sensor, or algorithm, is quickly gaining popularity as a method of improving biometric system performance and robustness. Despite the recent growth in multi-biometrics research, little investigation has been done to explore the possibility of achieving multi-modal fusion from a single sensor. This approach to multi-biometrics has numerous advantages, including the potential for increased recognition rates, while still minimizing sensor cost and acquisition times. In this work, experiments are presented which successfully combine multiple samples of face and iris biometrics obtained from a single stand-off and on-the-move video sensor. Several fusion techniques are explored, with the best recognition rates achieved by using a weighted summation of face and iris match scores. The fusion results out-perform either single-modality approach, and the proposed multi-biometric framework represents a viable and natural extension to the stand-off iris sensor used to acquire subject data.

Introduction

The practice of using more than one biometric modality, sample, sensor, or algorithm to achieve recognition, commonly referred to as multi-biometrics, is a technique that is rapidly gaining popularity. By incorporating multi-biometrics into the recognition process, many of the shortcomings of traditional single-biometric systems can be alleviated and overall recognition accuracy can be improved. Multi-biometrics can inherently increase system robustness by removing the dependency on one particular biometric approach. Further, a system that utilizes more than one biometric feature or matcher may be more difficult to deliberately spoof (Ross, Nandakumar, & Jain 2006). Systems that make use of multiple biometric features can also provide redundancy that may lower failure-to-acquire rates.

While research into multi-biometrics has received a large increase in attention over recent years, the task of fusing multiple biometric modalities from a single sensor remains an under-studied challenge. Due to a lack of available multi-modal data, many current experiments in multi-biometrics create “chimeric” datasets, in which samples of one biometric modality from one set of subjects are arbitrarily paired with a second biometric modality from a separate set of subjects in order to simulate a multi-biometric scenario (Bowyer

et al. 2006). This approach, though useful for preliminary experimentation, may mask unknown dependencies between modalities. Further, chimeric datasets simulate a multi-biometric scenario in which samples of each modality are acquired independently. In practice, it is much more desirable to simultaneously acquire multiple modalities from a single sensor if possible for cost and usability reasons. This work presents a multi-biometric system which simultaneously acquires face and iris information under near-infrared (NIR) illumination using the Iris on the Move (IOM) sensor, which is composed of an array of three identical video cameras with timed NIR illumination (Matey *et al.* 2006). The face and iris information for each subject is combined to improve recognition rates beyond the observed recognition rates for either isolated biometric.

Background and Related Work

There are four general approaches to multi-biometric system design. In the *multi-sample* approach, multiple samples (e.g. images) of the same biometric modality are acquired and processed. In the *multi-sensor* approach, the same modality is sampled several times, using different sensors for each acquisition. In the *multi-algorithm* approach, each biometric sample is matched using multiple matching algorithms and the results are fused. Finally, *multi-modal* systems acquire samples of more than one biometric trait (e.g. iris and face) for matching. Additionally, some systems represent a hybrid of these approaches by adding redundancy at multiple stages of the recognition process.

There are several levels at which fusion can occur in a multi-biometric system. Using *signal-level* fusion, multiple samples may be combined together to create one superior sample (as in super-resolution techniques). Alternatively, features can be extracted from each biometric sample, and *feature-level* fusion can be used to condense all of the features into a single biometric signature. With *score-level* fusion, each sample is processed and matched separately, and the resulting match scores for each sample are combined into one final match score. *Rank-level* fusion combines match rankings, rather than the scores for each sample, into a final ranking to determine the best match. Finally, *decision-level* fusion applies a matcher to each biometric sample to determine whether or not each comparison is a match, and the response of each matcher is fused using Boolean operators,

a voting scheme, or some similar method.

The fusion of face and iris modalities is a biometric approach that has gained increasing attention over the past decade, likely due to the popularity of the individual modalities and the natural connection between them. Despite this recent trend, very few studies have been done on fusion of face and iris biometrics from a single sensor.

The most common method of multi-biometric fusion is score-level fusion. Zhang et al. approach the problem of fusing face and iris biometrics under near-infrared lighting using a single sensor (Zhang *et al.* 2007). Frontal face images are acquired using a 10 megapixel CCD camera. Eye detection and face alignment are performed using Local Bit Pattern histogram matching as described in Li et al. (Li *et al.* 2006). The eigenface algorithm and Daugman’s algorithm are used to perform face and iris recognition, respectively, and score-level fusion is accomplished via the sum and product rules after min-max normalization. Numerous other score-level fusion approaches have been tested on chimeric datasets. Chen and Te Chu use an unweighted average of the outputs of matchers based on neural networks (Chen & Te Chu 2005). Wang et al. test weighted average, linear discriminant analysis, and neural networks for score fusion (Wang, Tan, & Jain 2003).

Another common approach to biometric fusion is feature-level fusion through concatenation. Rattani and Tistarelli compute SIFT features for chimeric face and iris images and concatenate the resulting feature vectors (Rattani & Tistarelli 2009). The number of matching SIFT features between two vectors (measured by Euclidean distance) is used as a match score for that comparison. Son and Lee extract features for face and iris images based on a Daubechies wavelet transform (Son & Lee). Concatenation is used to form a joint feature vector, and Euclidean distance between feature vectors is used to generate match scores.

Approach

To facilitate the fusion of face and iris biometrics from a single sensor, we selected the Iris on the Move (IOM) sensor for data acquisition. The IOM is a sensor designed for high-throughput stand-off iris recognition (Matey *et al.* 2006). The IOM features a portal which subjects walk through at normal walking pace. As a subject passes through the portal, the subject is illuminated with near-infrared (NIR) LED’s, and frontal video is captured by an array of three vertically-arranged, fixed-focus cameras equipped with NIR filters. The presence of multiple cameras allows the system to handle a larger range of subject heights. Though the sensor is intended for iris image acquisition, the face is typically captured as well. While the sides of the portal help to direct subjects into the field of view of the cameras, it is possible for subjects to stray partially out of the video frames, leading to frames with partial faces or only one visible iris. Figure 1 shows corresponding frames from each of the three IOM cameras while a subject passes through the in-focus region of the IOM. Each frame captured by one of the IOM cameras is a 2048 by 2048 pixel grayscale image. A typical iris acquired by the system is approximately 120 pixels in diameter.



Figure 1: Example of corresponding frames from the IOM as the subject passes through the in-focus region of the portal. The left image shows a frame from the top camera, the middle image shows a frame from the middle camera, and the right shows a frame from the bottom camera.

The general steps used in this work to combine face and iris biometrics from the IOM sensor are outlined in Figure 2. As previously described, when a subject passes through the IOM portal, three videos are collected, with one video coming from each of the IOM cameras. In a preprocessing step, the corresponding frames of the three videos are stitched together to create one single video. Next, a series of detection phases are used to locate whole faces and eyes in each frame. Matching is then performed on each face and iris independently, and the results are fused using several different techniques.

Preprocessing

In order to increase the likelihood of a whole face being captured for each subject, the three videos from each IOM acquisition are “stitched” together to combine corresponding frames. As can be seen in Figure 1, there is significant vertical overlap between the top and middle cameras, as well as between the middle and bottom cameras. Due to imperfect calibration of the individual cameras, some horizontal misalignment between the cameras is also present.

A template-matching approach is taken to determine the desired translation to align frames from adjacent cameras. Specifically, the bottom portion of the top frame is cropped and used as a template. This template is then matched against the upper half of the middle frame, and the best match is selected as the desired alignment. This process is repeated for the bottom camera, where the template is created from the top portion of the bottom frame and matched against the lower half of the middle frame.

Finally, noticeable illumination differences were observed between corresponding frames from different cameras, likely due to mis-calibration. To account for this discrepancy, histogram matching is used to match the top and bottom frame to the illumination observed in the middle frame. Figure 3 shows the intermediate and final results of the stitching procedure for an example frame.

Face Detection

Once the frame stitching is completed, the next step in the preprocessing phase is to detect a face in each frame. To accomplish this task, the OpenCV implementation of the Viola-Jones cascade face detector is used (Bradski &

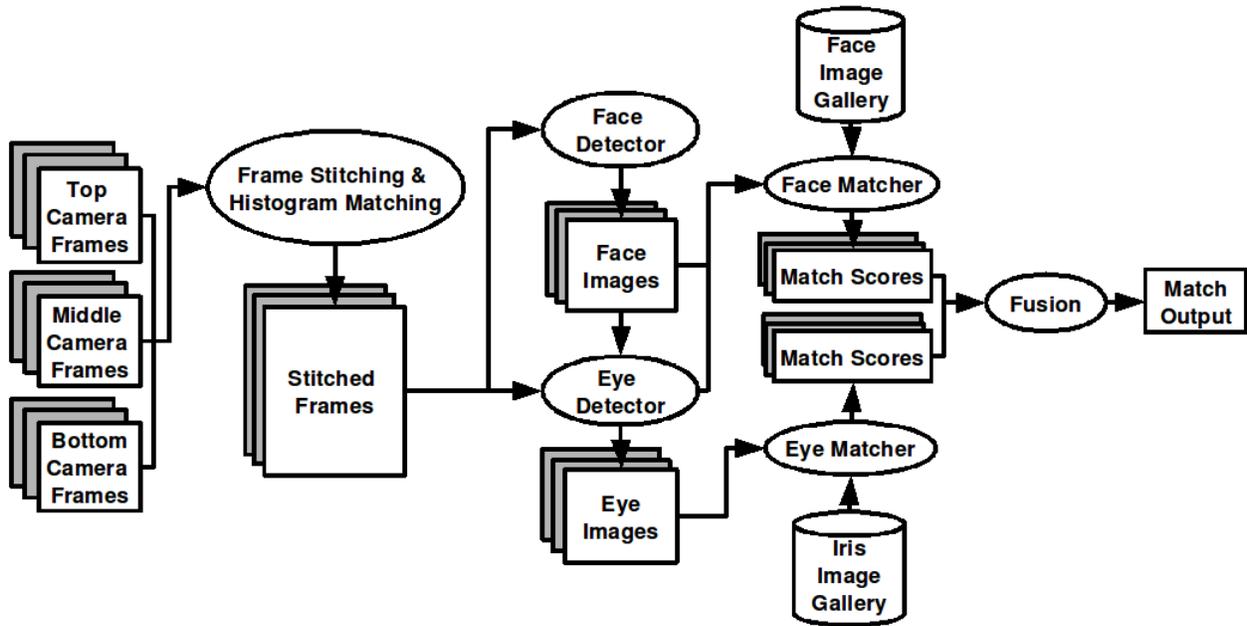


Figure 2: A diagram of the pipeline used in the proposed multi-biometric system.

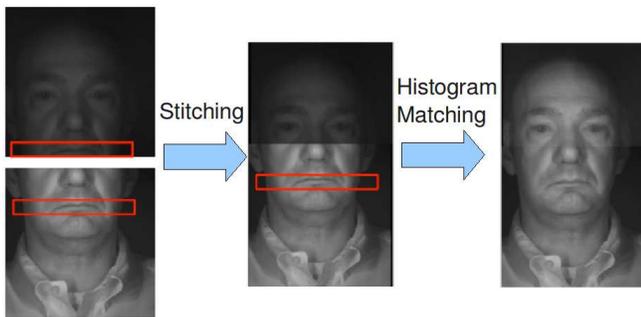


Figure 3: An example of the progression during alignment between corresponding frames from the top and middle camera. The top left image is the frame from the top camera with the template marked in red. The bottom left image is the frame from the middle camera, with the matched region marked in red. The middle image is the composite image, with the frame from the top camera cropped and padded. The overlapping region is indicated. The right image shows the final stitching results after histogram matching. A similar approach is used to stitch the frame from the bottom camera to the middle frame.

Kaehler 2008), (Viola & Jones 2001). The detector was trained on whole faces, and thus may or may not detect faces which lie only partially within the field of view of the camera.

Eye Detection

The purpose of the eye detection phase is twofold. The primary goal is to detect any eyes present in each frame for iris matching. However, the locations of the eyes that are detected in the faces produced by the face detector are also used for an alignment phase during face matching. A template matching approach is adopted for eye detection. The template used to search for eyes in each frame is based on the specular highlights generated by the reflection of the IOM LEDs.

The eye detection is completed in two phases. First, the template matching is performed on the upper left and upper right quadrants of each face detected by the face detector. This approach guarantees that each detected face will have two eye locations estimated as well.

Because it is possible for eyes to be detected in frames where whole faces were not present (or in frames where the face detector failed to detect the face), a second round of template matching is performed on any stitched frame where a face was not detected. In these frames, the location of the partial face can be crudely estimated by examining the sums of the rows and columns of the image. Once the partial face region has been estimated, the template matching is performed twice to identify the two best eye locations. Finally, the detected eyes are cropped from the corresponding location in the *original* frames to remove any possible artifacts caused by the histogram matching in the stitching phase. In cases where the detected eye is located in the overlapping

region between two cameras, the eye is cropped from *both* camera frames.

Face Matching

In this work, Colorado State University’s implementation of the eigenface algorithm is used for face matching (Colorado State University 2010), (Turk & Pentland 1991). To achieve alignment with the training set, the probe face images are normalized using the eye centers detected by the eye detector. The Mahalanobis cosine metric is used to compute the distance between two feature vectors. Using this metric, match scores can range from -1.0 to 1.0, with -1.0 being a perfect score. The output of the face matcher stage of the pipeline is a distance for every comparison between each probe face image and gallery face image.

Iris Matching

For the iris matcher, a modified version of Daugman’s algorithm is used to compare each probe iris image to the gallery (Daugman 2002). The normalized fractional Hamming distance, referred to simply as the Hamming distance in the rest of this work, ranges from 0.0 to 1.0, with 0.0 being a perfect match. The Hamming distance is normalized to adjust low Hamming distances that occur for comparisons that used relatively few bits. The output of the iris matcher stage of the pipeline is a Hamming distance for every comparison between each probe eye image and gallery iris image.

Fusion

In this framework, there is both a multi-sample (i.e. several faces from each video) and a multi-modal (i.e. both iris and face samples from each video) dimension to problem. Consequently, there are many methods which could be used to combine the face and iris biometrics from each video. Several fusion techniques are considered at both the score and rank-level.

The first method considers only one biometric modality in the fusion process, and makes use only of the multi-sample dimension of the problem by taking the minimum score for a given modality. For example, in the MinIris approach, the minimum score for all of the iris comparisons from a given video is reported as the best match. Similarly, the MinFace approach takes the minimum score for all of the face comparisons from a given video to determine the best match. Equations 1 and 2 express the MinIris and MinFace fusion rules, respectively, for a given probe video,

$$MinIris = Min\{I_{i,j} | i = 1...n, j = 1...G\} \quad (1)$$

$$MinFace = Min\{F_{i,j} | i = 1...m, j = 1...G\} \quad (2)$$

where n and m are the number of irises and faces detected in the video, respectively, G is the number of gallery subjects, $I_{i,j}$ is the Hamming distance between the i -th iris and the j -th gallery subject, and $F_{i,j}$ is the score for the comparison between the i -th face and the j -th gallery subject.

The next type of fusion method considered is rank-level fusion, and can incorporate face, iris, or both modalities into

the decision process. A Borda count is used to determine a best match across the desired biometric modalities. In a Borda count, the scores for all comparisons from a given sample are sorted such that the first rank corresponds to the best score for that sample. Each sample then casts votes for the top v ranked subjects, where the weight of each vote is inversely proportionate to rank number. Each sample votes in this manner, and the gallery subject with the most votes is taken to be the best match. In these experiments, the BordaIris method considers only the iris scores to perform fusion, and the BordaFace method considers only face scores. The BordaBoth method allows both face and iris samples to vote, with v votes being cast by each iris and face sample.

Two vote weighting schemes are tested for the BordaIris, BordaFace, and BordaBoth fusion methods. In the Linear approach, the vote weight is linearly proportional to the rank; specifically, the weight associated with the rank- n match is described by the equation

$$VoteWeight_n = v + 2 - n \quad (3)$$

and v represents the total number of votes cast by each biometric sample. In the Exponential approach, the weight of the vote is exponentially related to the rank. Specifically, the weight associated with the rank- n match is described by the equation

$$VoteWeight_n = 2^{v-n} \quad (4)$$

The third fusion method again uses score-level fusion, implementing a weighted summation of the iris and face scores. The summation rule can be expressed as Equation 5 for a given probe video,

$$SumScore_k = \frac{\alpha * \sum_{i=1}^n FNorm_{i,k} + \beta * \sum_{j=1}^m (1 - I_{j,k})}{\alpha * n + \beta * m} \quad (5)$$

where n and m are the number of irises and faces detected in the video, respectively, $I_{j,k}$ is the Hamming distance between the j -th iris and the k -th gallery subject, and $FNorm_{i,k}$ is the normalized score for the comparison between the i -th face and the k -th gallery subject. Each face score $F_{i,k}$ is normalized according to the expression

$$FNorm_{i,k} = 1 - \frac{F_{i,k} + 1}{2} \quad (6)$$

so that $0 \leq FNorm_{i,k} \leq 1$ and 1 is a perfect match. In Equation 5, α and β are coefficients used to weight the face and iris biometrics, respectively. In the presented work, $\alpha = 1 - \beta$ for simplicity. In Equation 5, $SumScore_k$ represents the final match score for the given probe video with gallery subject k ; the best match score can be determined by finding the maximum $SumScore_k$ for all k . SumIris is the special case where $\alpha = 0$ and $\beta = 1$, which corresponds to summing only the iris scores to determine the best match. Similarly, SumFace is the case where $\alpha = 1$ and $\beta = 0$, and equates to summing only the normalized face scores.

Table 1: DETAILED DETECTION RESULTS

Modalities Detected	Frame Count	Video Count
Left Iris	1,447 (5.1%)	35 (1.9%)
Right Iris	2,104 (7.4%)	46 (2.4%)
Face	900 (3.2%)	2 (0.1%)
Left & Right Irises	2,495 (8.8%)	209 (11.1%)
Face & Left Iris	1,411 (5.0%)	34 (1.8%)
Face & Right Iris	724 (2.6%)	27 (1.4%)
Face & Both Irises	6,798 (24.0%)	1,522 (80.6%)
None	12,502 (44.1%)	11 (0.6%)

Experiments

Dataset

The multi-biometric system being presented was tested on a probe dataset of 1,886 IOM video sets. Note that here a video “set” refers to the corresponding videos from each of the three IOM cameras, so the dataset is comprised of 5,658 videos in total. The 1,886 videos spanned 363 unique subjects, with an average of about five videos per subject. The most frequently occurring probe subject had 15 videos in the probe set, and the least frequently occurring had one probe video.

The iris gallery contained one left eye and one right eye for each of the 363 gallery subjects. The gallery images were acquired using the LG IrisAccess 4000 (LG Iris 2010), a high-quality iris acquisition camera, and the gallery was manually screened for good quality and segmentation.

The face gallery contained one full face image for each of the 363 subjects. The gallery images were acquired using the IOM. Each of the 363 subjects in the study had an additional IOM video set acquired in which the presence of a whole face was verified manually. The frames were stitched using the process previously described, and then the best frame was manually selected and the coordinates of the eye centers were manually annotated for alignment. The PCA training was performed on the face image gallery.

Detection Results

Across the entire dataset, 14,829 left irises and 14,711 right irises were detected and successfully segmented, and 9,833 faces were detected with valid eye locations for alignment. In this context, “successful segmentation” simply means that the iris segmentation routine returned pupil and limbic boundaries; it does *not* guarantee correctness. On average, 15.7 ($\sigma = 8.1$) irises, 5.2 ($\sigma = 3.7$) faces, and 20.9 ($\sigma = 20.9$) of either biometric samples were found in each video.

Table 1 provides a breakdown of the detection results by frame and video. The 1,886 videos were composed of a total of 28,381 frames. From Table 1 it can be seen that while a large number of frames (44.1%) contained no detected features, a much larger percentage of the probe *videos* (99.3%) had at least one biometric feature detected. Further, we see that the majority (80.6%) of the probe videos contained samples of face and both iris features.

Matching Results

Figure 4 shows the match and non-match score distributions for all 9,833 detected faces. The mean match score was -0.281 with a standard deviation of 0.213, while the mean non-match score was 0.000 with a standard deviation of 0.676. If each face were treated independently, the rank-one recognition achieved for the 9,833 probes faces would be 51.6% (5,073/9,833) recognition.

The results from the left and right irises were aggregated, and Figure 5 shows the match and non-match score distributions. The mean match score was 0.398 with a standard deviation of 0.053, while the mean non-match score was 0.449 with a standard deviation of 0.013. Figure 5 shows a significant number of match comparisons with fairly high scores. Upon examination of the data, it was found that most of these scores arise from incorrect segmentation. In some cases, these high match scores were caused by severe image defocus. Additionally, there are some false positives from the eye detector (non-eye regions) that contain features that resemble pupil and limbic boundaries according to the segmentation routine. If each iris image were treated independently, the rank-one recognition achieved for all of the probe irises would be 46.6% (13,556/29,112) recognition.

Fusion Results

The results of the iris and face matchers were combined using each of the methods previously described. The rank-one recognition rates achieved by each fusion approach are shown in Table 2. In the fusion methods based on Borda-counts, the number of votes given to each sample was varied between 1 and 363 (though all samples were given the same number of votes for any given fusion experiment), and the best results for each approach are presented. Similarly, results from the optimal tested values of α and β are presented.

Summarizing, the best single-modality fusion approach was the SumIris approach, which achieved an 87.8% rank-one recognition rate. The SumBoth approach achieved the overall highest recognition rate (93.2%), and all multi-modal fusion approaches achieved higher recognition rates than the fusion methods based on a single modality.

Figure 6 shows the ROC curves for the best SumBoth and BordaBoth approaches, as well as the MinIris, MinFace, SumFace, and SumIris results for comparison. From this graph, it is clear the the BordaBoth and SumBoth approaches out-perform the single-modality fusion methods. Interestingly, while SumBoth achieved the highest rank-one recognition rate, Figure 6 shows that the BordaBoth fusion technique performs better at false positive rates less than 0.06.

In general, the videos that failed to match correctly typically had relatively few face and iris features detected. While the iris proved to be the more accurate of the two modalities in the multi-sample fusion scenarios, Figure 5 indicates that many of the iris features detected are of poor quality, represent false detections from the eye detector, or failed to segment correctly. While the fusion techniques in these experiments were able to overcome these challenges when enough samples were present, videos in which a small

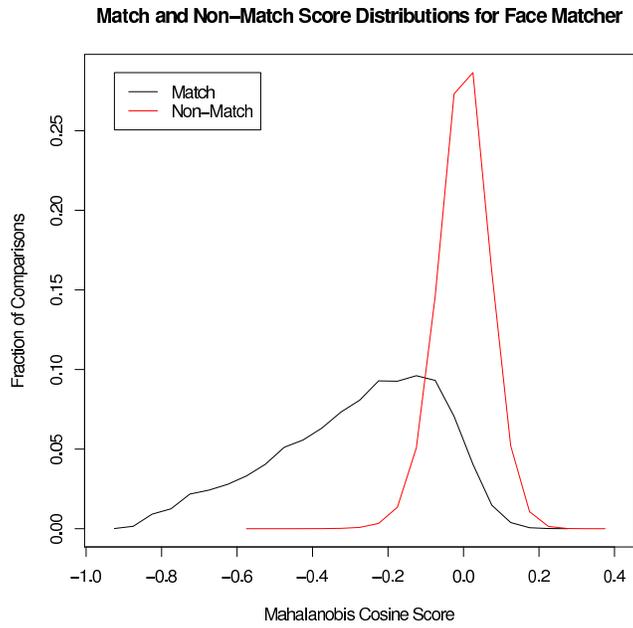


Figure 4: The match and non-match score distributions for the face features from the entire probe dataset.

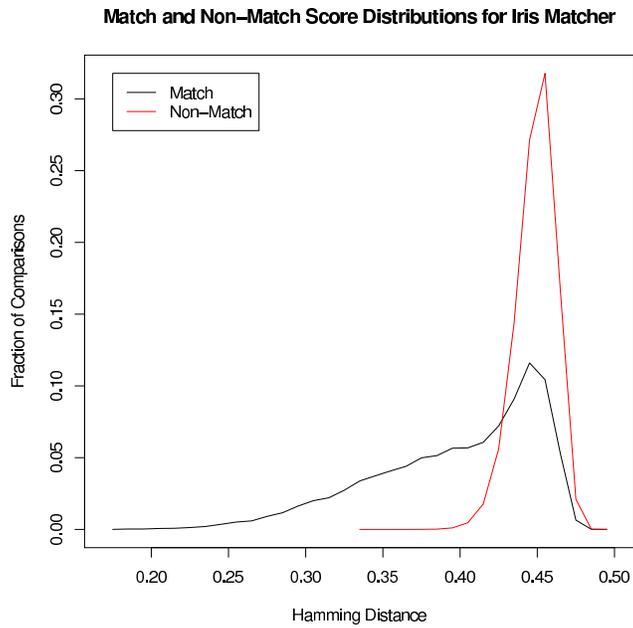


Figure 5: The match and non-match score distributions for the left and right iris features from the entire probe dataset.

Table 2: RANK ONE RECOGNITION RATES FOR FUSION APPROACHES

Approach	Fusion Parameters	Rank-One (Raw)
MinIris		86.7% (1,635/1,886)
MinFace		62.6% (1,180/1,886)
BordaIris-Linear	$v = 3$	86.4% (1,629/1,886)
BordaIris-Exponential	$v = 20$	86.8% (1,637/1,886)
BordaFace-Linear	$v = 3$	58.9% (1,110/1,886)
BordaFace-Exponential	$v = 5$	59.3% (1,118/1,886)
BordaBoth-Linear	$v = 10$	91.7% (1,729/1,886)
BordaBoth-Exponential	$v = 10$	92.0% (1,735/1,886)
SumIris	$\alpha = 0.0, \beta = 1.0$	87.8% (1,656/1,886)
SumFace	$\alpha = 1.0, \beta = 0.0$	61.3% (1,156/1,886)
SumBoth	$\alpha = 0.3, \beta = 0.7$	93.2% (1,757/1,886)

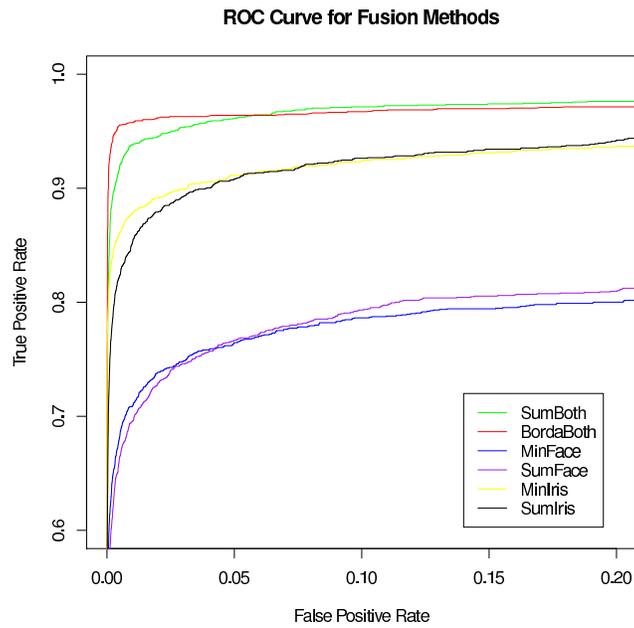


Figure 6: ROC curves for the various fusion methods using the optimal tested parameters for each. The BordaBoth method shown is the BordaBoth-Exponential method.

number of faces and iris are detected are much less likely to be correctly matched.

Conclusions and Future Work

This work presents an investigation into the fusion of face and iris biometrics from a single sensor, a surprisingly understudied problem in current literature. We present a multi-biometrics framework that utilizes both multi-sample and multi-modal fusion techniques to improve recognition rates from a single sensor. The multi-biometric system is tested on a non-chimeric dataset of over 1,886 videos spanning 363 subjects. This represents one of the largest genuine multi-modal experiments that has been conducted to date. Face and iris biometric samples extracted from videos produced from the Iris on the Move sensor were combined using several different fusion methods. In these experiments, the combination of face and iris biometrics via match score summation yielded a 5.4% increase in recognition rate over the best single-modality approach that was tested, while a modified Borda count approach performed best at lower false positive rates (< 0.06).

The multi-biometrics system we propose exploits the face information collected by the IOM, a sensor that is intended for iris recognition purposes, with no modifications to the sensor and no increase in probe data acquisition time. The resulting system is less likely to experience failures to acquire, and the use of multiple modalities could allow the system to identify subjects with incomplete gallery data. This approach could be extended to operate on other stand-off iris sensors, which often detect the face as a preliminary step to iris image acquisition.

In the future, new methods of fusion and matching will be explored, quality metrics and partial-face matching will be introduced, and run-time analysis will be conducted.

Acknowledgments

Datasets used in this work were acquired under funding from the National Science Foundation under grant CNS01-30839, by the Central Intelligence Agency, and by the Technical Support Working Group under US Army Contract W91CRB-08-C-0093. The authors are currently supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), through the Army Research Laboratory (ARL). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing official policies, either expressed or implied, of IARPA, the ODNI, the Army Research Laboratory, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

References

Bowyer, K. W.; Chang, K. I.; Yan, P.; Flynn, P. J.; Hansley, E.; and Sarkar, S. 2006. Multi-modal biometrics: An overview. Presented at the Second Workshop on Multi-Modal User Authentication (MMUA 2006).
Bradski, G., and Kaehler, A. 2008. *Learning OpenCV*. O'Reilly Media, Inc.

Chen, C.-H., and Te Chu, C. 2005. Fusion of face and iris features for multimodal biometrics. In Zhang, D., and Jain, A., eds., *Advances in Biometrics*, volume 3832 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg. 571–580.

Colorado State University. 2010. Evaluation of face recognition algorithms. <http://www.cs.colostate.edu/evalfacerec/algorithms5.html>.

Daugman, J. 2002. How iris recognition works. In *2002 International Conference on Image Processing*, volume 1, 33–36.

LG Iris. 2010. LG Iris products and solutions. <http://www.lgiris.com/ps/products/irisaccess4000.htm>.

Li, S. Z.; Chu, R.; Ao, M.; Zhang, L.; and He, R. 2006. Highly accurate and fast face recognition using near-infrared images. In *International Conference on Biometrics (ICB 2006)*, 151–158.

Matey, J.; Naroditsky, O.; Hanna, K.; Kolczynski, R.; Lolocono, D.; Mangru, S.; Tinker, M.; Zappia, T.; and Zhao, W. 2006. Iris on the move: Acquisition of images for iris recognition in less constrained environments. In *Proceedings of the IEEE*, volume 94. 1936–1947.

Rattani, A., and Tistarelli, M. 2009. Robust multi-modal and multi-unit feature level fusion of face and iris biometrics. In Tistarelli, M., and Nixon, M., eds., *Advances in Biometrics*, volume 5558 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg. 960–969.

Ross, A. A.; Nandakumar, K.; and Jain, A. K. 2006. *Handbook of Multibiometrics*. Springer Science and Business Media.

Son, B., and Lee, Y. Biometric authentication system using reduced joint feature vector of iris and face. In Kanade, T.; Jain, A.; and Ratha, N., eds., *6th International Conference on Audio- and Video-Based Biometric Person Authentication (AVBPA'03)*, Lecture Notes in Computer Science. Springer Berlin / Heidelberg.

Turk, M., and Pentland, A. 1991. Face recognition using eigenfaces. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '91)*, 586–591.

Viola, P., and Jones, M. 2001. Rapid object detection using a boosted cascade of simple features. In *2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2001)*, volume 1, 511–518.

Wang, Y.; Tan, T.; and Jain, A. K. 2003. Combining face and iris biometrics for identity verification. In *4th International Conference on Audio- and Video-Based Biometric Person Authentication (AVBPA'03)*, 805–813. Berlin, Heidelberg: Springer-Verlag.

Zhang, Z.; Wang, R.; Pan, K.; Li, S.; and Zhang, P. 2007. Fusion of near infrared face and iris biometrics. In Lee, S.-W., and Li, S., eds., *Advances in Biometrics*, volume 4642 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg. 172–180.

Learning and Classification

Chair: Carla Purdy

The Classification of Imbalanced Spatial Data

Alina Lazar
Department of Computer Science and Information Systems
Youngstown State University
Youngstown, OH 44555
alazar@ysu.edu

Bradley A. Shellito
Department of Geography
Youngstown State University
Youngstown, OH 44555
bshellito@ysu.edu

Abstract

This paper describes a method of improving the prediction of urbanization. The four datasets used in this study were extracted using Geographical Information Systems (GIS). Each dataset contains seven independent variables related to urban development and a class label which denotes the urban areas versus the rural areas. Two classification methods Support Vector Machines (SVM) and Neural Networks (NN) were used in previous studies to perform the two-class classification task. Previous results achieved high accuracies but low sensitivity, because of the imbalanced feature of the datasets. There are several ways to deal with imbalanced data, but two sampling methods are compared in this study.

Introduction

The aim of this paper is to show that class imbalance has a powerful impact on the performance of binary classification algorithms. Most machine learning algorithms provide models with better performances when trained using balanced training datasets. However, most of the real-world datasets from various domains like medical diagnosis, document classification, fraud and intrusion detection are highly imbalanced towards the positive or the minority class.

In general, classification algorithms are designed to optimize the overall accuracy performance. However, for imbalanced data, good accuracy does not mean that most examples from the minority class were correctly classified. Therefore, additional performance measures like recall, f-measure, g-means, AUC should be included when we study imbalanced problems.

One common approach to solve the imbalance problem is to sample the data to build an equally distributed training dataset. Several sampling techniques were proposed and analyzed in the literature (Van Hulse, Khoshgoftaar, and Napolitano 2007) including random under-sampling, random over-sampling and more intelligent sampling

techniques. A second class of methods uses meta-costs and assigns different penalties for the misclassified instances, depending on their true class. The problem with this type of methods is that it is hard to come up with a good penalty cost. The last type of methods is the algorithmic-based approach. They tweak the classifier to accommodate imbalanced datasets. The algorithm-based methods use meta-learning (Liu, An, and Huang 2006, Zhu 2007) or on-line active learning (Ertekin et al. 2007) to build better classifiers. Different combinations of these methods were also reported.

Real-world imbalanced datasets come from diverse application areas like medical diagnosis, fraud detection, intrusion detection, gene profiling, and object detection from satellite images (Kubat, Holte, and Matwin 1998). Our study investigates the effect of two sampling techniques when applied on four large GIS datasets with an imbalance ratio between 2.4 and 12.5. The four datasets contain over a million instances each, therefore there is no need to use over-sampling. Besides that, over-sampling is known to introduce excessive noise and ambiguity. Instead, the sampling methods considered were random sampling, under-sampling and the Wilson's editing algorithm in combination.

SVM and NN were used before in various studies to predict urbanization and land cover with almost similar results, but different prediction patterns (Lazar and Shellito 2005, Shellito and Lazar 2005). Even if SVM itself does not provide a mechanism to deal with imbalanced data, it can be easily modified. SVM builds the decision boundary on a limited number of instances that are close to the boundary, being unaffected by instances far away from the boundary. This observation can be used as an active learning selection strategy that provides a balanced training set for the early training stages of the SVM algorithm (Ertekin et al. 2007).

In the Background section we summarize related studies that deal with the problem of imbalanced datasets. The section Support Vector Machines and Multi-Layer Perceptrons presents the methods used, while the section describing our experiments presets a comparison between

random sampling, under-sampling and Wilson's editing. The last section presents the conclusions.

Background

Previous research (Lazar and Shellito 2005, Pijanowski et al. 2005, Pijanowski et al. 2002, Pijanowski et al. 2001, Shellito and Lazar 2005, Shellito and Pijanowski 2003) has shown that classification methods such as Support Vector Machines (SVM) and Neural Networks (NN) can be successfully used to predict patterns of urbanization in large datasets. SVM and NN can then be used as predictive tools to determine if grid cells can be accurately predicted as urban or non-urban cells. The effectiveness of the predictive capability of the SVM and NN can be measured through standard accuracy and other measures. The dataset generated for Mahoning County had over 1,000,000 instances and the imbalanced ratio was approximately 5:1. Even if the accuracy for both SVM and NN were over 90%, the recall was quite low 55%. Lately, several studies dealt with imbalanced datasets and their effect on classification performance; however none of the studies included datasets with over a million instances. Extensive experimental results using several sampling techniques combined with several classification methods applied on several datasets were reported by (Van Hulse, Khoshgoftaar, and Napolitano 2007). The sampling techniques considered were: random minority oversampling, random majority oversampling, one-side selection, Wilson's editing, SMOTE (Akbari, Kwek, and Japkowicz 2004), borderline SMOTE and cluster-based oversampling. They concluded that some of the more complicated sampling techniques especially one-side selection and cluster-based oversampling exhibit inferior performance in comparison with some of the simple sampling techniques.

Support Vector Machines

The machine learning algorithms named support vector machines proposed by (Vapnik 1999) consist of two important steps. Firstly, the dot product of the data points in the feature space, called the kernel, is computed. Secondly, a hyperplane learning algorithm is applied to the kernel.

Let (x_i, y_i) , $i = 1, \dots, l$, be the training set of examples. The decision $y_i \in \{-1, 1\}$ is associated with each input instance $x_i \in R^N$ for a binary classification task. In order to find a linear separating hyperplane with good generalization abilities, for the input data points, the set of hyperplanes $\langle w, x \rangle + b = 0$ is considered. The optimal hyperplane can be determined by maximizing the distance between the hyperplane and the closest input data points. The hyperplane is the solution of the following problem:

$$\min_{w \in R^l \times R^l, b \in R} \tau(w) = \frac{1}{2} \|w\|^2 \quad (1)$$

where $y_i (\langle w, x_i \rangle + b) \geq 1$ for all $i = 1, \dots, l$.

One challenge is that in practice an ideal separating hyperplane may not exist due to a large overlap between input data points from the two classes. In order to make the algorithm flexible a noise variable $\varepsilon_i \geq 0$ for all $i = 1, \dots, l$, is introduced in the objective function as follows:

$$\min_{w \in R^l \times R^l, b \in R} \tau(w, \varepsilon_i) = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^l \varepsilon_i \quad (2)$$

when $y_i (\langle w, x_i \rangle + b) \geq 1 - \varepsilon_i$ for all $i = 1, \dots, l$.

By using Lagrange multipliers the previous problem can be formulated as the following convex maximization problem (Liu, An, and Huang 2006):

$$W(\alpha) = \sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i,j=1}^l y_i y_j \alpha_i \alpha_j K(x_i, x_j) \quad (3)$$

when the following conditions are true, $0 \leq \alpha_i \leq C$ for all $i = 1, \dots, l$, and $\sum_{i=1}^l \alpha_i y_i = 0$. Here the positive constant C controls the trade-off between the maximization of (3) and the training error minimization, $\sum \varepsilon_i$.

From the optimal hyperplane equation the decision function for classification can be generated. For any unknown instance x the decision will be made based on:

$$f(x) = \text{sign}(\sum_{i=1}^l y_i \alpha_i K(x_i, x) + b) \quad (4)$$

which geometrically corresponds to the distance of the unknown instance to the hyperplane.

The method described until now works well on linear problems. Function K , the kernel from (4) enables good results for nonlinear decision problems. The dot product of the initial input space is called the new higher-dimensional feature space.

$$K : R^l \times R^l \rightarrow R, K(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle \quad (5)$$

A polynomial kernel, the radial basis and the sigmoid function are suitable kernels with similar behavior in terms of the resulting accuracy and they can be tuned by changing the values of the parameters. There is no good method to choose the best kernel function. The results reported in this paper were obtained by using the following radial basis function (Schölkopf and Smola 2002) as kernel.

$$K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\gamma^2}\right) \quad (6)$$

Multi-layer Perceptron Neural Networks

The multi-layer perceptron (MLP) (Witten and Frank 2000) is a popular technique because of its well-known ability to perform arbitrary mappings, not only classifications. Usually built out of three or four layers of neurons, the input layer, the hidden layers and the output layer, this network of neurons can be trained to identify almost any input-output function. The back-propagation algorithm used for the training process adjusts the synaptic weights of the neurons according with the error at the output. During the first step of the algorithm the predicted outputs are calculated using the input values and the network weights. Afterwards, in the backward pass the partial derivatives of the cost function are propagated back through the network and the weights are adjusted accordingly.

The problem with the MLP methods is that they are susceptible to converge towards local minimums. MLP methods are considered as “black box”, because it is impossible to obtain snap-shots of the process.

Sampling Methods

Since the datasets considered have over a million instances we decided to investigate under-sampling (US). This sampling technique discards random instances from the majority class until the two classes are equally represented. The other sampling method used in this study is called Wilson’s editing (Barandela et al. 2004) (WE). A k-means nearest neighbor classification procedure is used with k=3 to classify each instance in the training set using all the remaining instances. Afterwards, all the instances from the majority class that were misclassified are removed.

Performance Metrics

Especially in the case of imbalanced datasets, classification accuracy alone is not the best metric to evaluate a classifier. Several other performance metrics can be used in order to get a more comprehensive picture of the classifier’s capabilities.

Recall or sensitivity is the metric that measures the accuracy on the positive instances, It can be defined as $TruePositive / (TruePositive + FalseNegative)$. Specificity measures the accuracy on the negative instances and can be defined as $TrueNegative / (TrueNegative + FalsePositive)$. Both sensitivity and specificity are incorporated in the g-means measure (Ertekin et al. 2007), which is defined as square root from sensitivity * specificity.

Datasets

Seven broad predictor variables, which aid in describing the distribution of urbanization within the counties, were constructed using ESRI’s ArcGIS 9.2 software package. ArcGIS allows for modeling of a vast array of geospatial techniques, including the cell-by-cell raster models. These variables were chosen as they reflect large-scale factors that influence the patterns of urbanization and general urban trends for the region, as well as being similar to GIS variables for urban modeling within the Midwest (Pijanowski et al. 2005, Pijanowski et al. 2002, Pijanowski 2001, Shellito and Pijanowski 2003). The variables constructed were:

- a. Distance to City Centers
- b. Distance to Highways
- c. Distance to Interstates
- d. Distance to Railroads
- e. Distance to Lakes
- f. Distance to Rivers
- g. Density of Agriculture

For the county, a series of base layers was compiled to build the variables. The NLCD (National Land Cover Database) 2001 data was used for location of urban areas and as a source of agricultural data. Base layers for highways, interstates, and railways were drawn from US Census 2000 TIGER files. Lakes and rivers data was derived from Ohio Department of Transportation (ODOT) data. All base layers were projected into the UTM (Universal Transverse Mercator) projection and used to develop the predictor variables in raster format at 30m resolution. Distance variables were created by calculating the Euclidian distance of each cell from the closest feature in the base layers. The density variable was constructed by using a 3x3 moving window neighborhood operation and summing up the number of base layer grid cells in the neighborhood. Urban land was identified by selecting all grid cells with the “developed” classification in the NLCD dataset.

Predictor variables for each county were constructed by incorporating data from their bordering Ohio counties, to simulate the influence of nearby spatial factors outside the county borders (for instance, the proximity of a nearby city center in a bordering county could potentially effect the urban development within the target county). The resultant predictor variables created at this multi-county level were then clipped down to the boundaries of the chosen county and used in the analysis.

This type of data was extracted for four counties from the state of Ohio: Delaware, Holmes, Mahoning and Medina. All four resulting datasets contain more than a million instances each. Table 1 shows for each county dataset how many instances belong to the positive class, how many instance belong to the negative class and the ratio between the positive and the negative instances. All datasets are

mildly imbalanced from a 2.4:1 ratio for Mahoning County to a 12.5:1 ratio for Holmes County.

Table 1. Number of Training Instances and Ratios

	# Positive	# Negative	Ratio
Delaware	209,765	1,106,749	5.2761:1
Holmes	90,164	1,129,403	12.5260:1
Mahoning	353,411	868,423	2.4576:1
Medina	228,819	987,405	4.3152:1

For the first set of experiments we used two classifiers, the SVM and the Multi-Layer Perceptron (MLP). We used the libSVM (Chang and Lin 2001) software to run the parameter search, the training and the testing for SVM and Weka for the MLP. The experiments were similar with the experiments reported in (Lazar and Shellito 2005, Shellito and Lazar 2005) for Mahoning County.

Random stratified sampling, which maintains the ratio of positive versus negative instances in the datasets, was used to generate datasets of 10,000 instances for the parameter search and datasets of 50,000 for training sets.

A grid parameter search was performed for the SVM classifier and the values for the two parameters C and gamma are listed below in table 2.

Table 2. Parameters C and gamma for the LibSVM

	C	gamma
Delaware	8192	0.125
Holmes	2048	0.5
Mahoning	2	32
Medina	128	0.125

Next, both classifiers SVM and MLP were trained on the 50,000 instances datasets and the models obtained were tested using the entire datasets. The results obtained are reported in Table 3. For each dataset and for each classifier (SVM and MLP) three performance metrics are listed: accuracy, recall and g-means.

Table 3. Classification Performances for NN and SVM

		Del	Hol	Mah	Med
Accuracy	SVM	91.11	92.86	87.87	87.67
	MLP	80.36	93.8	85.72	87.53
Recall	SVM	57.35	4.19	70.44	47.13
	MLP	18.86	21.68	70.85	49.36
G-means	SVM	74.78	20.47	81.79	67.64
	MLP	40.02	46.46	80.63	68.97

The results show that even if SVM has higher accuracy for three of the datasets MLP has higher recall, so a better classification of the positive instances for three of the datasets. Recall has the largest values for the Mahoning

County dataset, which also has the lowest imbalanced ratio. Looking at the low recall values for the other three datasets, we need to investigate ways to better classify the instances from the positive class. Experiments using different sampling techniques are reported in the next section.

Experiments

We run experiments using RapidMiner (Mierswa et al. 2006) on the four datasets Delaware, Holmes, Mahoning and Medina as follows. For each dataset we performed 5 runs of a five-fold cross validation with the libSVM software. The rbf kernel was used. The two parameters C and gamma were changed to values previously found by running a grid parameter search.

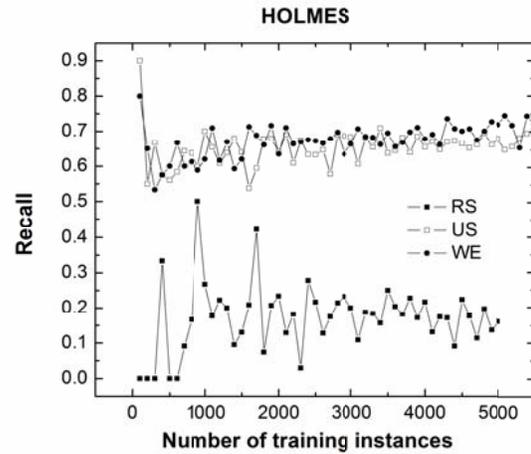


Figure 1. Recall for Holmes County Dataset

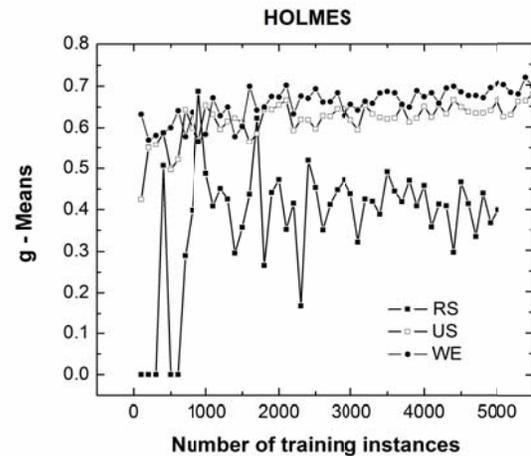


Figure 2. G-means for Holmes County Dataset

Three sampling techniques were used: random stratified sampling (RS), equal under-sampling (US) and Wilson's

editing sampling (WE). Each experiment was iterated through subsample datasets with sizes between 100 and 5000, with a step of 100.

The results are shown on two counties Holmes and Medina, due to space limitation. The Holmes County has the highest imbalanced ratio of approximately 12.5 and Medina has a 4.3 imbalanced ratio.

All four figures show that both under-sampling and Wilson's editing sampling have a great influence on the classification performance of the SVM learner. As accuracy is not relevant in the case of imbalanced datasets we looked at recall and g-means. The Wilson's editing worked only slightly better than the equal under-sampling, but required extensive preprocessing. The biggest difference in performance can be seen in Figure 1 with the recall for the Holmes County.

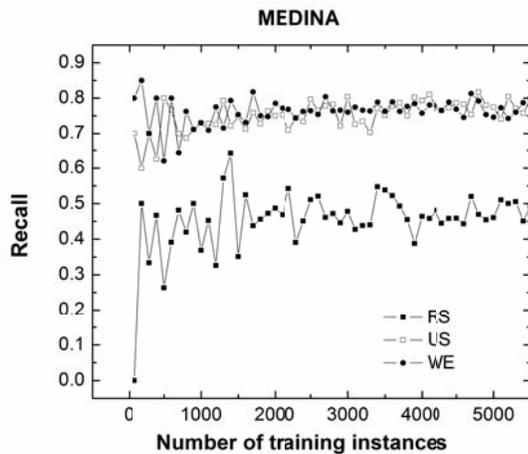


Figure 3. Recall for Medina County Dataset

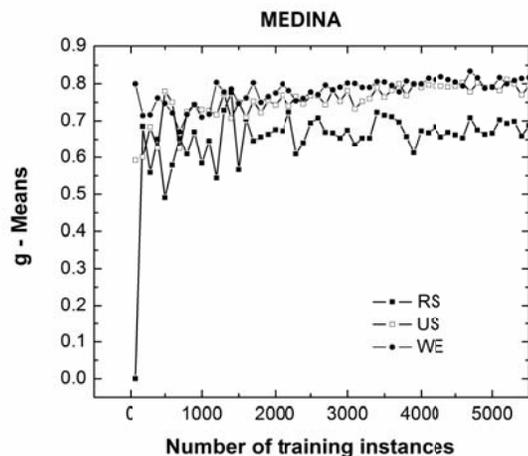


Figure 4. G-means for Medina County Dataset

Conclusions

We have presented an experimental analysis performed on large imbalanced GIS extracted datasets. The goal was to find what sampling techniques improve the classification performance especially for the minority class. It is important in the case of imbalanced datasets that additional performance measures like recall and g-means are compared in addition to the usual accuracy. We concluded that both equal under-sampling and Wilson's editing work better than just simple random stratified sampling, but there is no significant difference between the two.

Further research may investigate how other learners like Neural Networks or Decision Trees perform with under-sampling and Wilson's editing sampling. Over-sampling, cost-sensitive learning, and meta-learners are other alternatives that can be used to improve the performance for our datasets.

References

Akbani, R.; Kwek, S.; and Japkowicz N. 2004. Applying support vector machines to imbalanced datasets. Proceedings of European Conference on Machine Learning. 39-50. Pisa, Italy, Springer-Verlag, Germany.

Barandela, R.; Valdovinos, R. M.; Sanchez J. S.; and Ferri, F. J. 2004. The Imbalanced Training Sample Problem: Under or Over Sampling? In Joint IAPR International Workshops on Structural, Syntactic, and Statistical Pattern Recognition (SSPR/SPR'04), *Lecture Notes in Computer Science* 3138: 806-814.

Chang, C.; and Lin, C-J. 2001. LIBSVM : a library for support vector machines, 2001. Software at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>. Last accessed 01/15/2011.

Cristianini, N; and Shawe-Taylor, J. 2000. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*, Cambridge University Press, Cambridge, England.

Ertekin, S.; Huang, J.; Bottou, L.; and Lee Giles, C. 2007. Learning on the border: active learning in imbalanced data classification. In Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management (Lisbon, Portugal, November 06 - 10, 2007). CIKM '07. 127-136. ACM, New York, NY.

Koggalage, R.; and Halgamuge, S. 2004. "Reducing the Number of Training Samples for Fast Support Vector Machine Classification." *Neural Information Processing – Letters and Reviews* 2 (3): 57-65.

Kubat, M.; Holte, R. C.; and Matwin. S. 1998. Machine Learning for the detection of oil spills in satellite radar images. *Machine Learning*, 30(2-3): 195-215.

Lazar, A.; and Shellito, B. A. 2005. Comparing Machine Learning Classification Schemes – a GIS Approach. In Proceedings of

ICMLA'2005: The 2005 International Conference on Machine Learning and Applications, IEEE.

Liu, Y.; An A.; and Huang, X. 2006. Boosting Prediction Accuracy on Imbalanced Datasets with SVM Ensembles. *Lecture Notes in Artificial Intelligence*, vol. 3918: 107-118.

Mierswa, I.; Wurst, M.; Klinkenberg, R.; Scholz, M.; and Euler, T. 2006. YALE: Rapid Prototyping for Complex Data Mining Task. In Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-06).

Pijanowski, B.; Pithadia, S.; Shellito, B. A.; and Alexandridis, K. 2005. Calibrating a neural network based urban change model for two metropolitan areas of the upper Midwest of the United States. *International Journal of Geographical Information Science* 19 (2): 197-216.

Pijanowski, B.; Brown, D.; Shellito, B. A.; and Manik, G. 2002. Use of Neural Networks and GIS to Predict Land Use Change. *Computers, Environment, and Urban Systems* 26(6): 553-575.

Pijanowski, B.; Shellito, B. A.; Bauer, M. and Sawaya, K. 2001. "Using GIS, Artificial Neural Networks and Remote Sensing to Model Urban Change in the Minneapolis-St. Paul and Detroit Metropolitan Areas." In Proceedings of the ASPRS Annual Conference, St. Louis, MO.

Schölkopf, B.; and Smola, A. 2002. *Learning with Kernels*. MIT Press, Cambridge Massachusetts.

Shellito, B. A.; and Lazar, A. 2005. Applying Support Vector Machines and GIS to Urban Pattern Recognition. In Papers of the Applied Geography Conferences, volume 28.

Shellito, B. A.; and Pijanowski, B. 2003. "Using Neural Nets to Model the Spatial Distribution of Seasonal Homes." *Cartography and Geographic Information Science* 30 (3): 281-290.

Van Hulse, J.; Khoshgoftaar, T. M.; and Napolitano, A. 2007. Experimental perspectives on learning from imbalanced data. In Proceedings of the 24th International Conference on Machine Learning (Corvalis, Oregon, June 20 - 24, 2007). Z. Ghahramani, Ed. ICML '07, vol. 227. ACM, New York, NY, 935-942.

Vapnik, V. N. 1999. *The Nature of Statistical Learning Theory*, 2nd edition, Springer-Verlag, New York, NY.

Witten, I. H.; and Frank, E. 2000. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementation*, Morgan Kaufmann Publishers, Burlington, MA.

Zhu, X. 2007. Lazy Bagging for Classifying Imbalanced Data. In Seventh IEEE International Conference on Data Mining. 763-768. Omaha, NE.

Simplifying Probability Elicitation and Uncertainty Modeling in Bayesian Networks

Patrick Paulson* and Thomas E. Carroll and Chitra Sivaraman and Peter Neorr
Stephen D. Unwin and Shamina Hossain

Pacific Northwest National Laboratory
Richland, WA

Abstract

In this paper we contribute two novel methods that simplify the demands of knowledge elicitation for particular types of Bayesian networks. The first method simplifies the task of experts providing conditional probabilities when the states that a random variable takes can be described by a fully ordered set. In this order, each state's definition is inclusive of the preceding state's definition. Knowledge for the state is then elicited as a conditional probability of the preceding state. The second method leverages the Dempster-Shafer theory of evidence to provide a way for the expert to express the degree of ignorance that they feel about the estimates being provided.

Introduction

Currently, system administrators must be intimately familiar with their cyber assets and their organization's missions. But as the network of cyber resources continues to grow, it becomes exceedingly difficult to adequately prioritize time and resources across possible threats as the crucial tie between cyber assets and organizational missions is absent from most cyber monitoring tools. As business needs and market pressures are causing cyber systems to become more interconnected and thus more susceptible to cyber attacks, organizations require a tool that allows them to gauge risk exposure from multiple *risk perspectives*, such as public safety, environmental impact, and shareholder return.

This need motivated us to develop Carim, a decision-support methodology that provides an assessment of the consequences of threats to components of cyber systems so that security personnel can better allocate resources to protect key components. Because of the evolving nature of cyber attacks, we've relied on non-probabilistic techniques to allow us to characterize the completeness of the knowledge used to make risk assessments.

Carim models each asset in a system as a particular asset type. Asset types have known mitigating relationships with other asset types. The mitigating relationships are elicited from domain experts and best practices and encompass a consensus view on the types of actions that can be taken to reduce an asset's vulnerability. For example, a workstation might have mitigating relationships that include the installation of anti-virus software, a backup server and related

software, and a procedure for installing operating system patches. Each mitigating relationship involves other assets that might have mitigating relationships that require analysis. This network of mitigation relationships gives us a tool to elicit best practices from domain experts. It is similar to the causal mapping approach used for constructing Bayesian networks, where expert knowledge is represented as causal maps that are then, in turn, used to construct Bayesian networks (Nadkarni & Shenoy 2004). However, the elicitation of the conditional probabilities necessary for Bayesian networks proved difficult. This drove us to develop new methods for eliciting knowledge from experts.

Our Contributions In this paper we present two novel methods to simplify the demands of knowledge elicitation on certain types of Bayesian networks. The first method describes the values of a random variable using a fully ordered set of states. In this order, each state's definition is inclusive of the preceding state's definition. The second method uses Dempster-Shafer theory of evidence to provide a way for experts to express uncertainty in the estimates being provided.

Related Work Domain experts are heavily relied on to provide information about probabilistic networks. Yet, these experts often struggle with the complexity of this responsibility. The problem of eliciting probabilities attracted the attention of many Bayesian network community (Druzdzel & van der Gaag 1995; van der Gaag *et al.* 1999; Olumuş & Erbaş 2004). The elicitation of the probabilistic values for reasoning under uncertainty is a critical obstacle (Druzdzel & van der Gaag 1995; van der Gaag *et al.* 1999; Olumuş & Erbaş 2004). Various methods have been designed to elicit probabilistic relations. But these methods tend to be very time consuming and are difficult to apply when many thousands of probabilities must be assessed.

While Bayesian networks have been used previously to reason about beliefs (see, for example, (Simon & Weber 2006; Simon, Weber, & Levrat 2007)), we generalize these methods and formally tie them to Dempster-Shafer (DS) theory of evidence. We simplify DS theory such that the focal elements of a node (i.e., the subsets with non-zero mass) are confined to the singleton plus a general "don't know." No other subset is assigned mass.

Organization The paper is structured as follow. In Sec-

*Corresponding author. E-mail: patrick.paulson@pnl.gov

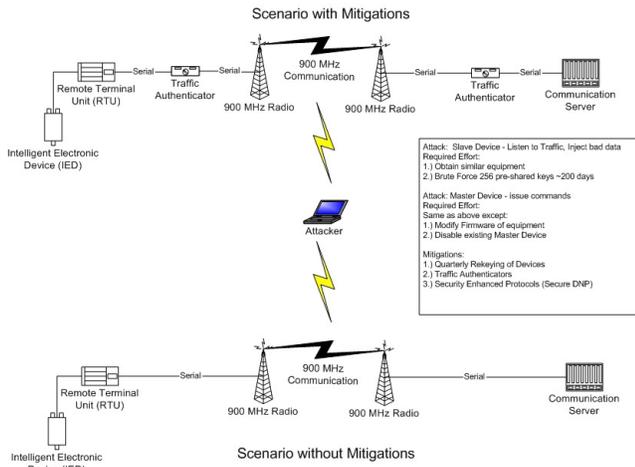


Figure 1: A man-in-the middle attack scenario in which the attacker can eaves drop on the wireless communication channels.

tion we describe a scenario for which the Carim methodology was applied and discuss approaches in eliciting expert knowledge and representing uncertainty in the estimates that they provide. In Section we provide the background necessary to understand our contributions. We discuss our contributions in Section , which are novel methods for eliciting knowledge for Bayesian networks. Finally, we summarize and conclude in Section .

Eliciting SCADA Domain Knowledge for Carim

Carim has been applied to security in the domain of *supervisory control and data acquisition* (SCADA) networks, the networks used to control industrial processes. In this domain, we were particularly concerned with the possibility of a *man-in-the-middle* attack when a SCADA network included unsecured links between nodes. Figure 1 is a representation of this scenario. Our resident expert suggested two technical fixes that could be used, either independently or together, to protect against such an attack: SecureDNP, an encrypted wire protocol, and SSCP, a protocol that ensures data integrity. The effectiveness of these techniques depends on the “Rekey” policy used: how often the encryption and authentication keys are changed. Finally, the vulnerability to a man-in-the-middle attack depends on the capabilities of the attacker: an insider might have access to the required keys, and a state-backed attacker may have access to enough computing power to break the encryption scheme. The factors considered by Carim in assessing vulnerability to this attack are summarized in Figure 2.

In order to assess the vulnerability of the SCADA network using traditional Bayesian techniques, we would be required to elicit from our expert conditional probabilities for each combination of mitigation states and attacker expertise. In approaching our expert with this task, we realized that the expert was much more comfortable providing some val-

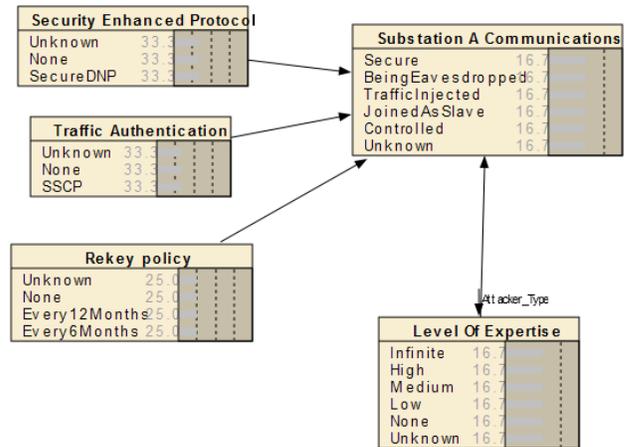


Figure 2: The elements for assessing the vulnerability of substation communications to an attack.

ues than others—for example, the relative effectiveness of rekeying policies was unknown, but the traffic authentication were clearer. When encoding these values into a Bayesian network however, the uncertainty of our expert disappears. While there is a good deal of controversy on the subject, the traditional approach of handling this problem with Bayesian networks is to ensure that the elicited probabilities encompass the doubts of the expert, and to not support additional “second order probabilities” (Pearl 1988, p. 360–363). We desired, however, to explicitly model uncertainty so that the end-user would have a measure of the applicability of the results. The method described here allows the expert to express uncertainty without forcing them to further analyze the factors causing their uncertainty so they can be expressed in one probability distribution.

We also realized that we were putting undue burden on the expert by requiring them to state probabilities that had to match the constraints of the problem: it is always required, for example, that the vulnerability not decrease when the only change is that the expertise of the attacker increases.

We are charged then, with the following requirements:

1. Devise procedures to simplify the elicitation of probabilities that are constrained by additional factors
2. Model the uncertainty of the knowledge used to provide a measure of applicability of the model

A Technique to Simplify Elicitation

In order to apply a Bayesian network to this problem, the expert was required to provide *conditional probabilities* for each state of compromise of the asset for each combination

Rekey policy	Security Enhanced ...	Traffic Authentication	Level Of Expertise	Secure	BeingEavesdr...	TrafficInjected	JoinedAsSlave	Controlled	Unknown
Every12Months	SecureDNP	None	Unknown	0	0	0	0	100	0
Every12Months	SecureDNP	SSCP	Infinite	0	0	0	0	100	0
Every12Months	SecureDNP	SSCP	High	24	32	8	7	4	25
Every12Months	SecureDNP	SSCP	Medium	44	27	3	1	0	25
Every12Months	SecureDNP	SSCP	Low	51	22	2	0	0	25
Every12Months	SecureDNP	SSCP	None	100	0	0	0	0	0
Every12Months	SecureDNP	SSCP	Unknown	49	19	2	0	0	30
Every6Months	Unknown	Unknown	Infinite	0	0	0	0	100	0

Figure 3: Elicitation requirements for man-in-the-middle attack on substation communications. The left pane is the state space; the right is the sample space of the variable.

Table 1: The reduced elicitation requirements for substation communication security. These values are for the case when keys are changed every 12 months, both SecureDNP and SSCP are enabled, and the attacker has medium expertise.

Secure	1
Eavesdrop	60%
Inject	10%
Join	50%
Control	10%
Unknown	0.25

of the random variables that can affect the asset’s state, as illustrated in Figure 3.

As described above, the expert is also allowed to specify a probability for the special state *Unknown*, which is probability they do not feel comfortable assigning to any particular state.

As can be seen in the Figure 3, the elicited probabilities in this problem have some interesting characteristics because of additional constraints on the state spaces of the variable. In particular, it is assumed that some states of compromise are “more difficult” to achieve than others; attackers with “higher” levels of expertise are accorded more probability of moving the asset into the more difficult states.

Because of these considerations we simplified our elicitation technique. For given states of the values of the mitigations and a specific level of expertise, we first have the expert give a estimate of the “uncertainty” they have in assessing the hypothesized situation. They are then asked to give, for the given level of expertise l , an estimate of the percentage of attackers with expertise l that can achieve the *lowest* level of compromise c_0 on the asset. (Since the lowest level of compromise is “completely secure”, this value is 100 percent). Then, for each succeeding level of compromise c_i , they are then asked to estimate what percentage of attackers with expertise l who can achieve level of compromise c_{i-1} can also achieve level of compromise c_i . Section describes how we then convert these elicited values to probabilities used in a Bayesian network.

Using Figure 3 as an example, we are eliciting values for when keys are changed every 12 months, and both SecureDNP (encryption) and SSCP (authentication) are used. Using our technique, we elicited the values given in Table 1

for the case when the attacker has medium expertise. The values are elicited as percentages of the potential attackers with the given level of expertise that can move an asset to a more compromised level given the state of mitigations. Since we assume that all such attackers can leave the asset in the “Secure” state, the first elicited value is the percentage of attackers that can change the state to “being eavesdropped”. In our example, the expert asserts a value of 60 percent. The next value we elicit is the percentage of attackers who can change the state to “inject messages.” An attacker who can effect this change also has the expertise to eavesdrop. The expert testifies that 10 percent of all attackers who can eavesdrop can also inject messages. We continue eliciting values in this fashion in the order that the states are specified. Finally, we ask the expert to quantify her confidence in the values she provided. If the expert feels the amount of information given in the constraints is sufficient to determine the elicited values, then the “unknown” value would be zero. If the expert feels that they have no basis for their judgments, then “unknown” would be one. Viewed this way, the “unknown” value is the portion of information required to make a judgment that is missing.

Background

In the following we briefly describe the foundations, Bayesian networks and Dempster-Shafer theory of evidence, on which we build our contributions.

Bayesian Networks *Qualitative Bayesian Networks* (Halpern 2003, p. 135), as a special case of *discrete influence diagrams* (Kjaerulff & Madsen 2008, p. ix), are convenient to elicit and encode an expert’s impressions of factors that influence values in their domain of expertise. In order to be operational, *quantitative* Bayesian network requires a myriad conditional probabilities to be specified for each combination of values in an expert that they may not feel comfortable in estimating.

A Bayesian network $N = (G, P)$ over a set of random variables $\mathcal{X} = \{X_1, \dots, X_n\}$ consists of a directed acyclic graph (DAG) G that encodes a set of conditional independence assertions and local probability distributions P for each variable. Together, G and P form a joint probability distribution over \mathcal{X} .

To be a Bayesian network, N must possess the local Markov property. Denote by $\text{pa}(X_i)$ and $\text{nond}(x_i)$ the set of parents and non-descendants, respectively, of X_i . A network possess the *local Markov property* if, for each $X_i \in \mathcal{X}$,

$X_{pa} \in pa(X_i)$, and $X_{nond} \in nond(X_i)$, the proposition (Neapolitan 2004, p. 37)

$$P(x_i) = 0 \vee P(x_{pa}|x_{nond}) = 0 \vee P(x_i|(x_{pa}|X_{nond})) = P(x_i)$$

evaluates to true. In words, the local Markov property states that each variable is conditionally independent of its non-descendants given its parent variables.

The local Markov property makes Bayesian networks an effective technique for eliciting knowledge: by viewing the network, an expert can determine if all factors are being considered when determining the probability of an event.

Dempster-Shafer Theory The inability to express uncertainty is a drawback of the approaches based on probability theory (Halpern 2003, p. 24). However, expressing uncertainty is a necessity when attempting to elicit understanding in knowledge-poor domains (see, for example, (Forester *et al.* 2004; Donell & Lehner 1993; O’Hagan & Oakley 2004)). In contrast to purely probabilistic methods for capturing domain knowledge, Dempster-Shafer theory (DS) provides a rich mechanism for describing the range of beliefs about a result (Gordon & Shortliffe 1990). This richness comes at the expense of complexity in both eliciting the values for expressing the different types of ignorance and in the combination of multiple pieces of evidence (Ai, Wang, & Wang 2008).

In the following we summarize DS theory. We refer the reader to (Gordon & Shortliffe 1990) for a reference on DS theory. Let X be a random variable specified by the finite set \mathbf{X} of its values. Set \mathbf{X} is also called the *frame of discernment*. A *basic probability assignment* (BPA) $m_{\mathbf{X}}$ over \mathbf{X} is a function

$$m_{\mathbf{X}} : 2^{\mathbf{X}} \rightarrow [0, 1],$$

where $2^{\mathbf{X}}$ is the power set of \mathbf{X} , for which

$$m_{\mathbf{X}}(\emptyset) = 0 \quad \text{and} \quad \sum_{S \subseteq \mathbf{X}} m_{\mathbf{X}}(S) = 1.$$

The *mass* or *degree of belief* $m_{\mathbf{X}}(S)$ of S is a measure of that portion of belief that is committed exactly to S by $m_{\mathbf{X}}$ and not to any particular subset of S . Each subset S such that $m(S) > 0$ is called a *focal element*. There are two measures that bound the interval that $m(S)$ resides. The function

$$bel_{\mathbf{X}}(S) = \sum_{T \subseteq S} m(T)$$

computes the *belief* (or *support*) for all $S \subseteq \mathbf{X}$. The *plausibility* of each $S \subseteq \mathbf{X}$ is given by

$$pl_{\mathbf{X}}(S) = \sum_{T \cap S \neq \emptyset} m(T).$$

Belief measures the total mass that is constrained to move within the set of interest, while plausibility measures the total mass that can visit somewhere in the set of interest but can also move outside it. From the definitions, we see that $bel_{\mathbf{X}}(S) \leq m_{\mathbf{X}}(S) \leq pl_{\mathbf{X}}(S)$.

In the next section, we discuss our methods for eliciting knowledge from experts.

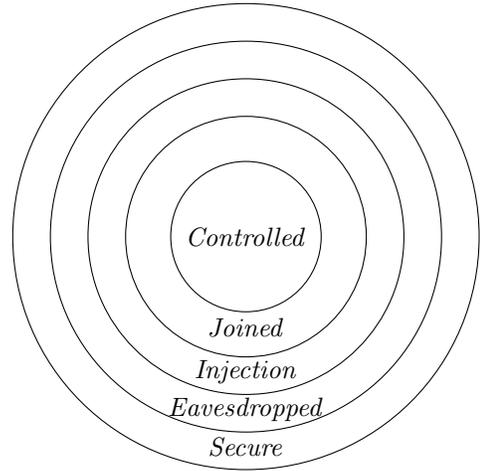


Figure 4: The inclusive compromise states of the substation communications using set theory.

Method

We next discuss our contributions to eliciting knowledge from experts. The first contribution is when the states of a variable can be described by a fully ordered set. In this set, a state implies all the preceding states. Our second contribution is using Dempster-Shafer theory of evidence to allow experts to express uncertainty of the estimates that they provide.

Simplifying Conditional Probability Elicitation in Bayesian Networks

Our goal is to elicit values in the form shown in Table 1 and calculate conditional probabilities that can be used in a Bayesian network. As an example, consider Figure 3 in which we need to elicit the probability of compromise conditioned on the Rekey policy, wire protocol, data integrity protocol, and the attacker’s level of expertise.

The elements of the sample space of the random variables in a Carim model often belong to a simple order. For example, when considered in terms of the progression of an attack, the states of compromise in Figure 2 can be ordered as *Secure* \prec *Eavesdropped* \prec *Injection* \prec *Joined* \prec *Controlled*. What this says is that for the attacker to control devices, she must have joined the network, and to join, she must have the ability to inject traffic, and so on. The implication of states can be represented as inclusive sets as we have done in Figure 4.

An advantage of this constraint when eliciting knowledge is that we can state our elicitation in terms of an already elicited value, which eases the cognitive load on the subject matter expert. For example, instead of asking: “What is the likelihood that that a person with high skill level can eavesdrop on the network”, and then separately asking “What is the likelihood that a person with high skill level can inject traffic into the network”, we can ask “What is the likelihood that a person with high skill level who can eavesdrop the network can also inject traffic into it?”

Let s_1, \dots, s_n be states of X such that state s_{i+1} implies s_i (i.e., s_{i+1} is a subset of s_i). We elicit beliefs $P(s_1), P(s_2|s_1), \dots, P(s_n|s_{n-1})$ from experts given the parents of X . But in probability theory, the elements of the sample space of a random variable must be disjoint. We obtain the disjoint sample space by defining x_i to mean for $s_i \wedge \neg s_{i+1}$. Treating the beliefs as probabilities, the probability $P(x_i)$ of X taking the value x_i given X 's parents is:

$$P(x_i) = (1 - P(s_{i+1}|s_i))P(s_1) \prod_{j=2}^i P(s_j|s_{j-1}), \quad (1)$$

for $i = 1, \dots, n - 1$, and

$$P(x_n) = P(s_1) \prod_{j=2}^n P(s_j|s_{j-1}). \quad (2)$$

If X is conditionally dependent on other variables, we have all the necessary values to construct a Bayesian network to compute $P(x_i)$.

Implementing Subset of Dempster-Shafer Theory with Bayesian Networks

The greatest disadvantage of DS theory is that in contrast to probabilistic models, which are described by their respective density functions, DS models must be described by a set, which grows exponentially with the number of variable values. It would be difficult to elicit degree of belief for each and every set. If we can represent that problem with a graph that satisfies the Markov property, we then can use the computational efficiency of Bayesian networks to compute degrees of belief.

Beliefs are elicited from experts for each value x of variable X and also the element *Unknown*, which is equivalent in DS theory to the set \mathbf{X} . All other sets have no mass. Given these conditions, $P(x)$ satisfies the requirements of $m_{\mathbf{x}}$ as $\sum_{\bar{x} \in \mathbf{X} \cup \{\text{Unknown}\}} P(\bar{x}) = 1$. A Bayesian network can be constructed such that the node that represents X has a state for each of its focal elements. The node's conditional probability table comprises the elicited conditional probabilities of X given its parents. The network output for the node computes $P(\bar{x})$, for each $\bar{x} \in \mathbf{X} \cup \{\text{Unknown}\}$. The belief in x is simply $\text{bel}_{\mathbf{X}}(x) = P(x)$ and the plausibility is $\text{pl}_{\mathbf{X}}(x) = P(x) + P(\text{Unknown})$.

We now consider an example. There are three variables X, Y , and Z , where X conditionally depends on Y and Z and Y and Z are conditionally independent. From the definition of joint probability, the probability $P(\bar{x})$ of $\mathbf{X} \cup \{\text{Unknown}\}$ is

$$P(\bar{x}) = \sum_{\substack{\bar{y} \in Y \cup \{\text{Unknown}\} \\ \bar{z} \in Z \cup \{\text{Unknown}\}}} P(\bar{x}|\bar{y}, \bar{z})P(\bar{y})P(\bar{z}).$$

This is the probability computed by the Bayesian network.

Conclusion

In eliciting knowledge for Carim, we frequently came upon the situation where we needed to determine the subjective

probability of a member of a simple order according to a domain expert. For example, the probability that a threat will compromise an asset at a particular level of compromise. Additionally, the domain expert may have the ability to know when their knowledge about an area is incomplete, but be unable to further describe the characteristics of the incomplete knowledge. For these reasons, we wanted our users to be aware of the completeness of the knowledge in decisions.

We solved these problems by using Bayesian networks constructed using knowledge gained via our elicitation methods described in this paper. The first method simplifies the elicitation of conditional probabilities when the sample space of a random variable can be described by a fully ordered set of inclusive states. The conditional probability of a state is dependent only on its predecessor. The second method implements a subset of Dempster-Shafer theory using a Bayesian network. This allows the network to provide a measure of uncertainty along with its output.

References

- Ai, L.; Wang, J.; and Wang, X. 2008. Multi-features fusion diagnosis of tremor based on artificial neural network and D-S evidence theory. *Signal Processing* 88(12):2927–2935.
- Carley, K. M., and Palmquist, M. 1992. Extracting, representing and analyzing mental models. *Social Forces* 70(3):601–636.
- Davey, B. A., and Priestley, H. A. 2002. *Introduction to Lattices and Order*. Cambridge University Press, 2 edition.
- Donell, M. L., and Lehner, P. E. 1993. Uncertainty handling and cognitive biases in knowledge engineering. *Systems, Man and Cybernetics, IEEE Transactions on* 23(2):563–570.
- Druzdzel, M., and van der Gaag, L. 1995. Elicitation of probabilities for belief networks: Combining qualitative and quantitative information. In *In Uncertainty in Artificial Intelligence (95): Proceedings of the 11th conference, Los Altos CA*, 141–148. Morgan Kaufmann.
- Fiot, C.; Saptawati, G.; Laurent, A.; and Teisseire, M. 2008. Learning bayesian network structure from incomplete data without any assumption. In Haritsa, J.; Kotagiri, R.; and Pudi, V., eds., *Database Systems for Advanced Applications*, volume 4947 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg. 408–423. 10.1007/978-3-540-78568-2_30.
- Forester, J.; Bley, D.; Cooper, S.; Lois, E.; Siu, N.; Kocaczkowski, A.; and Wreathall, J. 2004. Expert elicitation approach for performing atheana quantification. *Reliability Engineering & System Safety* 83(2):207–220.
- Gordon, J., and Shortliffe, E. H. 1990. The Dempster-Shafer theory of evidence. In Shafer, G., and Pearl, J., eds., *Readings in Uncertain Reasoning*. San Mateo, California: Morgan Kaufmann Publishers. 529–539.
- Halpern, J. Y. 2003. *Reasoning about Uncertainty*. Cambridge, Massachusetts: The MIT Press.

- Heckerman, D. 1997. Bayesian networks for data mining. *Data Mining and Knowledge Discovery* 1:79–119.
- Kjærulff, U. B., and Madsen, A. L. 2008. *Bayesian Networks and Influence Diagrams: A Guide to Construction and Analysis*. Springer.
- Nadkarni, S., and Shenoy, P. 2004. A causal mapping approach to constructing bayesian networks. *Decision Support Systems* 38:259–281.
- Neapolitan, R. E. 2004. *Learning Bayesian Networks*. Upper Saddle River, NJ: Prentice Hall.
- O'Hagan, A., and Oakley, J. E. 2004. Probability is perfect, but we can't elicit it perfectly. *Reliability Engineering & System Safety* 85(1–3):239–248.
- Olumuş, H., and Erbaş, S. O. 2004. Determining the conditional probabilities in Bayesian networks. *Hacettepe Journal of Mathematics and Statistics* 33:69–76.
- Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann.
- Simon, C., and Weber, P. 2006. Bayesian networks implementation of Dempster Shafer theory to model reliability uncertainty. In *Proc. of the 1st International Conference on Availability, Reliability and Security (ARES '06)*.
- Simon, C.; Weber, P.; and Levrat, E. 2007. Bayesian networks and evidence theory to model complex systems reliability. *Journal of Computers* 2(1):33–43.
- van der Gaag, L.; Renooij, S.; Witteman, C.; Aleman, B.; and Taal, B. 1999. How to elicit many probabilities. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*, 647–654. Morgan Kaufmann Publishers.

Confusion Matrix-based Feature Selection

Sofia Visa

Computer Science Department
College of Wooster
Wooster, OH
svisa@wooster.edu

Brian Ramsay

Sentry Data Systems, Inc.
Fort Lauderdale, FL
brian.ramsay@gmail.com

Anca Ralescu

Computer Science Department
University of Cincinnati
Cincinnati, OH
anca.alescu@uc.edu

Esther van der Knaap

Ohio Agricultural Research
and Development Center
The Ohio State University
Wooster, OH
vanderknaap.1@osu.edu

Abstract

This paper introduces a new technique for feature selection and illustrates it on a real data set. Namely, the proposed approach creates subsets of attributes based on two criteria: (1) individual attributes have high discrimination (classification) power; and (2) the attributes in the subset are complementary - that is, they misclassify different classes. The method uses information from a confusion matrix and evaluates one attribute at a time. **Keywords:** classification, attribute selection, confusion matrix, k-nearest neighbors;

Background

In classification problems, good accuracy in classification is the primary concern; however, the identification of the attributes (or features) having the largest separation power is also of interest. Even more, for very large data sets (such as MRI images of brain), the classification is highly dependent on feature selection. This is mainly because the larger the number of attributes, the more sparse the data become and thus many more (exponential growth) training data are necessary to accurately sample such a large domain. In this sense, the high dimensional data sets are almost always under represented. This problem is also known in literature as "the curse of dimensionality". For example, a 2-attribute data set having 10 examples in the square defined by the corners (0,0) and (1,1) covers the domain acceptably. If the domain to be learned is the cube defined by the corners (0,0,0) and (1,1,1), 10 points will not cover this 3-D domain as effectively.

Reducing the number of attributes for a classification problem is a much researched field. The brute force approach in finding the best combination of attributes for classification requires the trial of all possible combinations of the available n attributes. That is, consider one attribute at a time, then investigate all combinations of two attributes, three attributes, etc. However, this approach is unfeasible because there are $2^n - 1$ such possible combinations for n attributes and, for example, even for $n=10$ there are 1,023 different attribute combinations to be investigated. Additionally, feature selection is especially needed for data sets having large

numbers of attributes (e.g. thousands). Examples of such data domains with many features include text categorization and gene expression analysis. In the first example, each document is described by the most frequent words, leading to 20,000 words or more. In working with expressed genes in order to separate healthy from cancer patients, for example, the number of attributes may grow as high as 30,000 (Guyon and Elisseeff 2003). Another example of a challenging domain is the microarray data found in (Xin, Jordan, and Karp 2001), where a hybrid of filter and wrapper approaches is employed to successfully select relevant features to classify 72 data examples in a 7,130 dimensional space.

In addition to reducing the data dimensionality, selecting fewer attributes may improve classification and may give a better understanding of the underlying process that generated that data. Here we propose an attribute-selection technique based on a confusion matrix with the two-fold objective of better classification and better data understanding.

Depending on where the feature selection module is placed in relation to the classification module, there are two classes of methods for feature selections (Jain and Zongker 1997):

- **Filter methods** (Pudil, Novovicova, and Kittler 1994) rank features (or feature subsets) independently of the predictor. These methods investigate irrelevant features to be eliminated by looking at correlation or underlying distribution. For example, if two attributes have the same probability distribution, then they are redundant and one of them can be dropped. Such analysis is performed regardless of the classification method. Another filtering method ranks attributes based on the notion of nearest hit (closest example of same the class) and nearest miss (closest example of a different class) (Kira and Rendell 1992). The i^{th} feature ranking is given by the score computed as the average (over all examples) of the difference between the distance to the nearest hit and the distance to the nearest miss, in the projection of the i^{th} dimension (Guyon and Elisseeff 2003).
- **Wrapper methods** (Kohavi and John 1997) use a classifier to assess features (or feature subsets). For example, the decision tree algorithm selects the attributes having

Table 1: The confusion matrix for two-class classification problem.

	PREDICTED NEGATIVE	PREDICTED POSITIVE
ACTUAL NEGATIVE	a	b
ACTUAL POSITIVE	c	d

the best discriminatory power and places them closer to the root. Hence, besides the classification tree, a ranking of attributes results.

Another classification of attribute selection methods considers the search technique of the feature subsets. There are two main greedy search strategies: forward selection and backward elimination. Both techniques yield nested subsets of features. The forward selection starts with one attribute and continues adding one attribute at a time if the newly formed subset gives better classification. During backward elimination, unpromising attributes are progressively eliminated (Guyon and Elisseeff 2003). This greedy search technique is often used in system identification. The result, in either case, is not guaranteed to yield the optimal attribute subset (Sugeno and Yasukawa 1993).

In this research we investigate the use of the confusion matrix (Kohavi and Provost 1998) (which contains information about actual and predicted classifications) for attribute selection. In the context described above, this approach is a wrapper method because it uses a classifier to estimate the classification power of an attribute (or subset of attributes).

The Confusion Matrix and Disagreement Score

A confusion matrix of size $n \times n$ associated with a classifier shows the predicted and actual classification, where n is the number of different classes. Table 1 shows a confusion matrix for $n = 2$, whose entries have the following meanings:

- a is the number of correct negative predictions;
- b is the number of incorrect positive predictions;
- c is the number of incorrect negative predictions;
- d is the number of correct positive predictions.

The prediction accuracy and classification error can be obtained from this matrix as follows:

$$Accuracy = \frac{a + d}{a + b + c + d} \quad (1)$$

$$Error = \frac{b + c}{a + b + c + d} \quad (2)$$

We define the disagreement score associated with a confusion matrix in equation (3). According to this equation the disagreement is 1 when one of the quantities b or c is 0 (in this case the classifier misclassifies examples of one class only), and is 0 when b and c are the same.

$$D = \begin{cases} 0 & \text{if } b = c = 0; \\ \frac{|b-c|}{\max\{b,c\}} & \text{otherwise.} \end{cases} \quad (3)$$

The attribute selection methodology proposed here selects attributes that not only have good discrimination power on their own, but more importantly are complementary to each other. For example, consider two attributes A_1 and A_2 , having similar classification accuracy. Our approach will select them as a good subset of attributes if they have a large disagreement in terms of what examples they misclassify. A large disagreement is indicated by D values closer to 1 for both attributes, but distinct denominators in equation (3).

Algorithm for Confusion Matrix-based Attribute Selection

The pseudocode outlined below shows the steps to perform confusion matrix-based attribute selection for a 2-class classification problem. This method basically constructs attribute-subsets that: (1) have attributes with good individual classification power, and (2) have attributes that are complementary (i.e. they disagree in their misclassifications).

Note that the algorithm may lead to several subsets of attributes to be further investigated, i.e. further the subset yielding higher classification accuracy may be selected.

Also, the algorithm does not account for the possibility that two individually lower ranked attributes may combine in a high classification accuracy subset due to their high complementarity.

Algorithm 1 Pseudocode for Confusion Matrix-based Attribute Selection Algorithm

Require: 2-class data of n attributes

Require: classification technique

Require: k - number of member subset

Ensure: Output k -attribute subset as tuple $S_k = (A_1, A_2, \dots, A_k)$

Compute classifier C_i based on feature $A_i, i = 1..n$

Obtain: $Accuracy(C_i)$ and $ConfMatrix(C_i)$

Rank A_i according to $Accuracy(C_i) \Rightarrow R_A$

for $i = 1 \dots n$ **do**

 Compute disagreement based on $ConfMatrix(C_i)$

 as: $D_i = \frac{|b-c|}{\max\{b,c\}}$

end for

Rank A_i according to $D_i \Rightarrow R_D$

Select top k (according to R_A) attributes having large

D (according to R_D) but in different classes: $\Rightarrow S_k =$

(A_1, A_2, \dots, A_k)

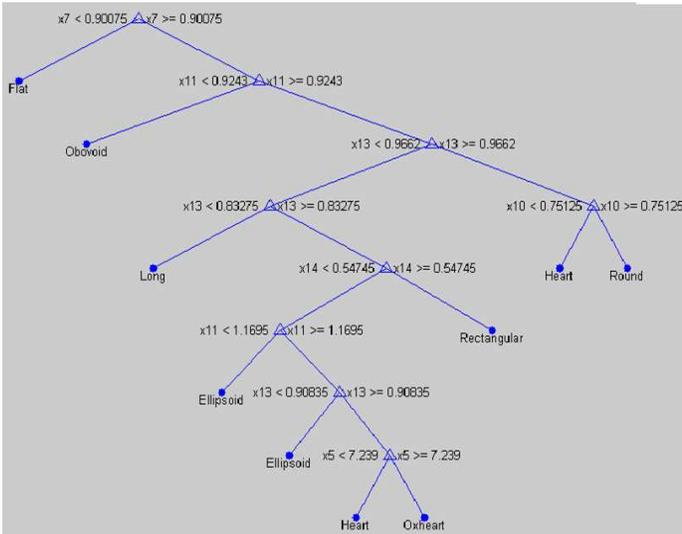


Figure 1: Decision tree obtained with CART for all data and all attributes.

Table 2: Data distribution across classes. In total, there are 416 examples each having 34 attributes.

Class	Class label	No. of examples
1	Ellipse	110
2	Flat	115
3	Heart	29
4	Long	36
5	Obvoid	32
6	Oxheart	12
7	Rectangular	34
8	Round	48

The Tomato Fruit Data Set

The data set used in the experimental part of this research consists of 416 examples having 34 attributes and distributed in 8 classes (the class-distribution is shown in Table 2). This set was obtained from the Ohio Agricultural Research and Development Center (OARDC) research group led by E. Van Der Knaap (Rodriguez et al. 2010) and the classification task is to correctly label a tomato fruit based on morphological measurements such as width, length, perimeter, circularity (i.e. how well a transversal cut of a tomato fits a circle), angle at the tip of the tomato, etc.

The data set was collected as follows: from the scanned image of a longitudinally section of a tomato fruit the 34 measurements are extracted by the Tomato Analyzer Software (TA) (Rodriguez et al. 2010) developed by the same group. For a complete description of the 34 tomato fruit measurements and the TA software see (Gonzalo et al. 2009).

In addition to tomato classification, of interest here is to find which attributes have more discriminative power and further to find a ranking of the attributes.

Data Classification and Attribute Selection

In this section we show the decision tree classification of the tomato data set, then we illustrate our attribute selection algorithm (in combination with a k-nearest neighbor classifier) on two (out of 8) classes. These two classes (1 and 7) are identified by both, decision trees and k-nearest neighbors, as highly overlapping.

Classification with Decision Trees - CART

We used the Classification and Regression Trees (CART) method (Breiman et al. 1984) because it generates rules that can be easily understood and explained. At the same time, classification trees have a built-in mechanism to perform attribute selection (Breiman et al. 1984) and we can compare our set of selected attributes, obtained from the confusion matrix and k-nearest neighbors analysis, with the set of attributes identified by CART. However, we anticipate that these sets will not perfectly coincide, which only means that the two approaches quantify the importance of a given attribute (or subset of attributes) differently and that the two methods learn the data differently.

The pruned decision tree obtained using CART is shown in Figure 1. The train and test error associated with this tree are 11.54% and 18.27%, respectively. As it can be seen from this figure, 10 rules can be extracted. In addition, CART selects the following 8 attribute as best in classification (listed in decreasing order of their importance):

- 7 - Fruit Shape Idx Ext1
- 13 - Circular
- 12 - Ellipsoid
- 11 - Fruit Shape Triangle
- 14 - Rectangular
- 10 - Distal Fruit Blockiness
- 8 - Fruit Shape Idx Ext2
- 1 - Perimeter

We also investigate the k-nearest neighbors classifier as we will use this classification method in combination with our attribute selection approach. Figure 2 shows the k - nearest neighbors classification error for k = 2,...,15. The top (blue) line corresponds to runs that include all 34 attributes and the bottom (red) line shows the error when only the best five attributes (identified by CART classification) are used.

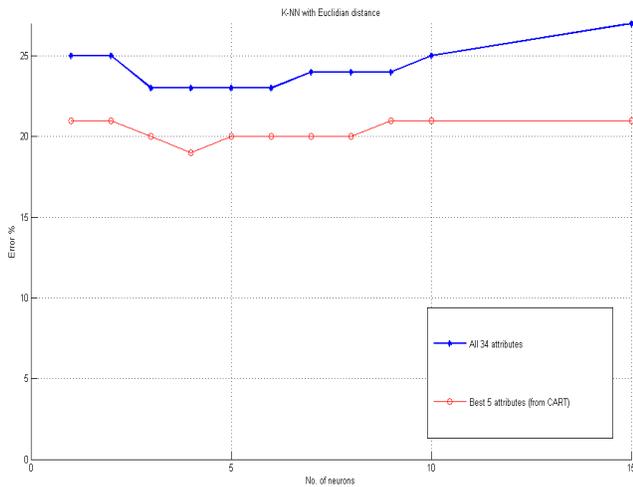


Figure 2: K - nearest neighbors classification error for $k = 2, \dots, 15$. The top (blue) line corresponds to runs that include all 34 attributes and the bottom (red) line shows the error when only the best five attributes are used (these attributes were identified through CART classification).

As shown in Figure 2, the k-nearest neighbors classification technique consistently scores lower error when using only 8 attributes (the one selected by CART), rather than all 34. Thus, a natural question arising here is: is there a better combination of attributes than the one selected by CART for classification? For $k = 4$ the k-nearest neighbors technique yields the lowest error, which justifies our choice of using $k = 4$ in the next experiments.

The Confusion Matrix for the Tomato Data Set

When using all 34 attributes and all 8 classes, the confusion matrix obtained from the k-nearest neighbors clustering with $k=4$ is shown in Figure 3, where along the x-axis are listed the true class labels and along the y-axis are the k-nearest neighbors class predictions. Along the first diagonal are the correct classifications, whereas all the other entries show misclassifications. The bottom right cell shows the overall accuracy.

In this confusion matrix it can be seen that 8 examples of class 7 are wrongly predicted as class 1. Additionally, from the CART classification, the classes 1 and 7 are identified as the two classes overlapping the most. Thus the experiment presented next uses the confusion matrix attribute selection to better separate these two classes. Namely, we search for a subset of the 34 attributes such that the attributes are complementary in the sense described above and quantified in equation (3).

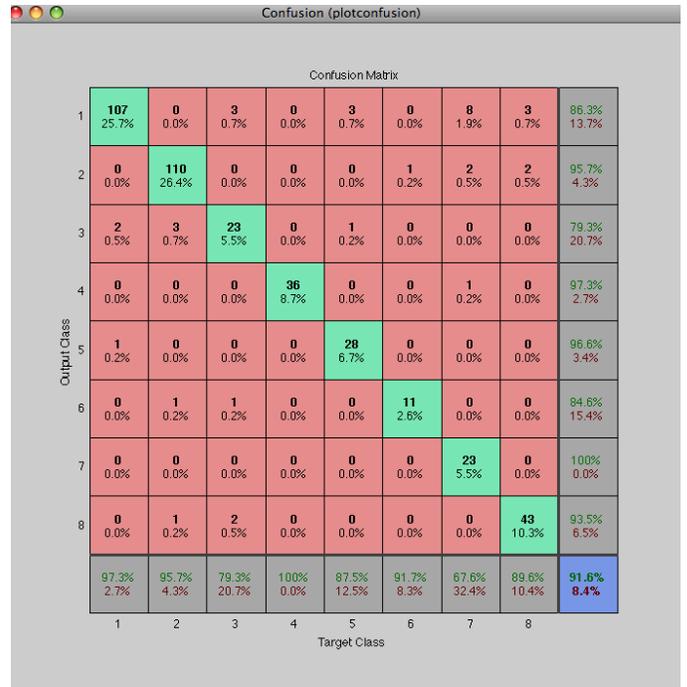


Figure 3: Confusion matrix for all classes and all attributes. Class 7 has 8 examples wrongly predicted as class 1 (see top row).

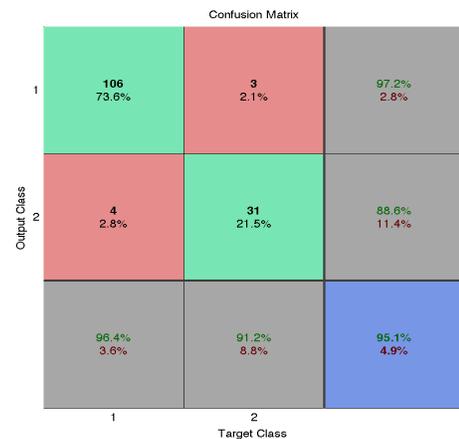


Figure 4: Confusion matrix for class 1 and 7 along attribute 14. Four examples of class 1 are misclassified as class 7, and 3 examples of class 7 belong to class 1.

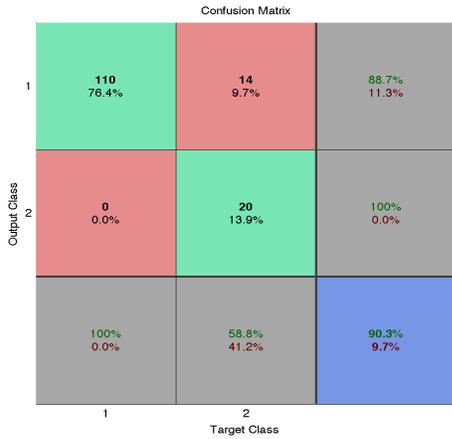


Figure 5: Confusion matrix for class 1 and 7 along attribute 20. Fourteen examples of class 7 are misclassified as class 1.

Confusion Matrix-based Attribute Selection for Classes 1 and 7

When using the data from classes 1 (Ellipse) and 7 (Rectangular), a data set of size 145 is obtained (110 in class 1 and 34 from class 7).

As illustrated in Algorithm 1, for each of the 34 attributes, the k-nearest neighbor algorithm (with $k = 4$) is used for classification and the corresponding classification accuracy and confusion matrix are obtained (for each attribute). Further, the 34 attributes are ranked in the order of their individual performance in distinguishing between class 1 and 7, leading to the ranking set $R = 14, 7, 8, 17, 1, 3, 6, 12, 30, 4, 9, 20, 29, 18, 26, 2, 10, 21, 34, 32, 11, 33, 5, 13, 19, 16, 15, 31, 25, 28, 24, 27, 22, 23$.

We first create growing nested subsets of attributes in the order specified by their individual classification abilities. Note that this particular choice of subsets is not part of Algorithm 1 and makes no use of the complementarity. We simply introduce it as a comparative model for our selection approach, which, besides the accuracy ranking, incorporates complementarity information as well.

Figure 6 shows the classification accuracy for subsets of attributes consisting of the top 1, top 2, top 3, etc. attributes from R (the subsets are shown on x-axis, while the y-axis shows classification accuracy). From Figure 6 it can be seen that the highest accuracy is achieved when the top 3 attributes are used together (i.e. attributes 14, 7, and 8),

Table 3: Attribute ranking based on disagreement score. The best classification attributes found by CART are shown in bold (they are also underlined). The attributes marked by (*) are the ones identified by our selection algorithm.

Attr. number	Disagreement score	Class of largest error
20	1	7
22	1	7
24	1	7
25	1	7
26	1	7
27	1	7
31	1	7
23	0.9655	7
28	0.9565	7
15	0.9524	7
2	0.9375	7
21	0.9375	7
29	0.9231	7
12	0.9167	7
30	0.9167	7
1	0.9091	7
3	0.9091	7
7	0.9000	7
17	0.9000	7
5	0.8889	7
11	0.8824	7
32	0.8750	7
34	0.8667	7
18	0.8462	7
13	0.8235	7
19	0.8235	7
6	0.8182	7
33	0.8125	7
4	0.7273	7
9*	0.7273	7
16*	0.6875	7
8*	0.6250	7
10*	0.3000	7
14*	0.2500	1

yielding a 97.2% correct classification.

The above approach is a (greedy) forward selection method, i.e. the attributes are progressively incorporated in larger and larger subsets. However, here we incorporate the attributes in the order dictated by their individual performance, yielding nested subsets of attributes. For example, we do not evaluate the performance of the subset consisting of first and third attribute. Indeed, it may be that this subset can perform better than considering all top three attributes. However, as indicated earlier in this paper, evaluating all possible combination is not a feasible approach. Thus, we propose to **combine the attributes that are complementary, i.e. two (or more) attributes that may achieve individually similar classification accuracy but they have the largest disagreement (this information is extracted from the confusion matrix).**

The disagreement scores for all 34 attributes when classify-

ing data from classes 1 and 7 are listed in Table 3, column 2. As column 3 of the same table shows, only attribute 14 misclassifies more examples of class 1 than of class 7 (see bottom row). All other 33 attributes have the largest number of misclassifications attributed to class 7. Thus, for this particular data set, we will consider subsets that combine attribute 14 with one or more other attributes having the largest disagreement.

For example, Figures 4 and 5 show the confusion matrix for classes 1 and 7 when using attribute 14 and 20, respectively. These two attributes disagree the most as the above figures and Table 3 show.

Algorithm 1 is illustrated here for $k = 2, 3, 4$, and 5 only (note for $k = 1$ the classification results are the same as in Figure 6). The classification accuracy for these experiments is shown in Figures 7, 8, 9, and 10, respectively. In addition to selecting the top k attributes in Table 3 (this combination yields the first star in the above plots), we also plot the classification accuracy of the sliding (moving from top to bottom in Table 3) window of k attributes.

Figures 7, 8, 9, and 10 show the classification accuracy of the k -nearest neighbor algorithm when attribute 14 is combined with all the other attributes in decreasing order of their disagreement scores (see Table 3). Figure 7 shows results for 2-member subsets and the results from Figure 8 are obtained for 3-member subsets: attribute 14 and two consecutive attributes from Table 3.

As Figure 10 illustrates, simply selecting the top attributes from Table 3 (having the largest disagreement) does not ensure a better classification, nor is an increasing or decreasing trend observed when sliding down the table. This is because classification ability is not additive and opens up the question of whether a better k -subset of attributes can be obtained by mixing attributes across the table, not only the k -neighbors selection used here.

Among the k -member subsets investigated here (note, there are more sliding window subsets for $k > 5$), the largest classification accuracy (98%) is achieved for a 5-member subset, namely for the attribute-set 14, 9, 16, 8, and 10. The CART classifier recognizes these two classes with 93% accuracy (using attributes 7, 13, 12, 14, 11, and 10), and the accuracy-ranking only (no complementarity information incorporated) selection achieves 97.3% (using top 3 attributes: 14, 7 and 8). The attribute subset with the largest discriminating power (when using a k -nearest neighbors clustering, $k = 4$) is obtained with the confusion matrix-based attribute selection; however, it is not a large improvement as the classes are pretty well separated to begin with.

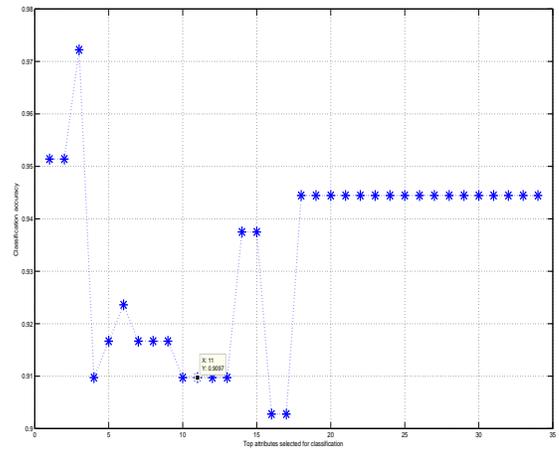


Figure 6: K - nearest neighbors classification accuracy for $k = 4$ when using data from classes 1 and 7 only. On x-axis are listed the nested subsets of attributes having top 1,2,3,...,34 attributes. The highest accuracy (97.2%) is obtained for the subset having the top 3 attributes: 14,7, and 8.

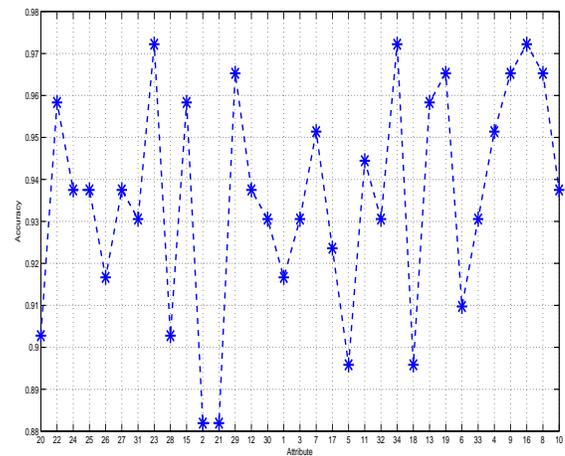


Figure 7: K - nearest neighbors classification accuracy for $k = 4$ when using 2-member subsets of attributes. Each subset contains attribute 14 and one of the remaining attributes; x-axis shows these subsets listed in the order of their complementarity - see Table 3. Largest accuracy is 97.3%.

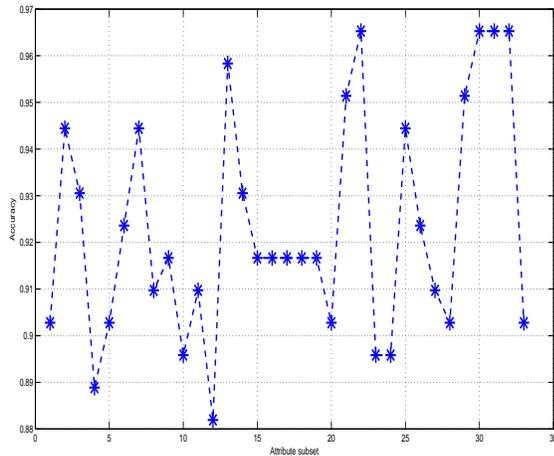


Figure 8: K - nearest neighbors classification accuracy for $k = 4$ when using 3-member subsets of attributes. Each subset contains attribute 14 and two consecutive attributes (in the order of their complementarity - see Table 3). Largest accuracy is 97.3%.

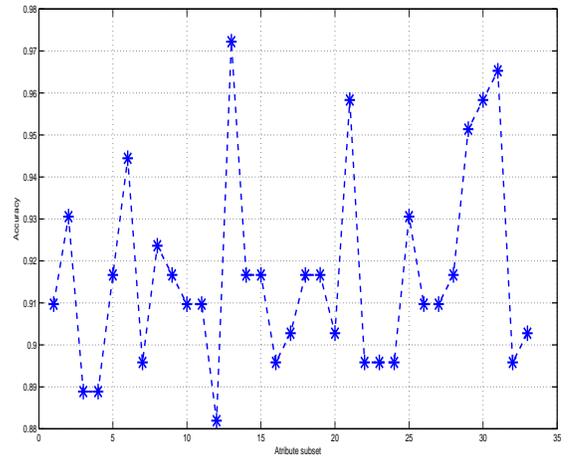


Figure 9: K - nearest neighbors classification accuracy for $k = 4$ when using 4-member subsets of attributes. Each subset contains attribute 14 and three consecutive attributes (in the order of their complementarity - see Table 3). Largest accuracy is 96.7%.

Conclusions and Future Work

A new technique for attribute selection is proposed here. The method selects attributes that are complementary to each other, in the sense that they misclassify different classes, and favors attributes that have good classification abilities by themselves. This new approach is illustrated on a real data set. For two classes of interest within this data set, this technique found a better (i.e. yielding higher classification accuracy) subset of attributes, than using all attributes or even using the 8 attributes identified by CART. However, we must investigate this new approach in more data sets and in combination with other classification techniques (here only the k-nearest neighbor classifier was investigated). Another future direction is to investigate the use of subsets that combine complementary attributes, even if these attributes are weak classifiers by themselves. The challenging factor for this approach is the large number of subsets that must be investigated. Depending on the data set, if this search space is very large, then genetic algorithms can be used to explore the version space. We must also extrapolate this method to multi-class data sets and investigate its scalability factor.

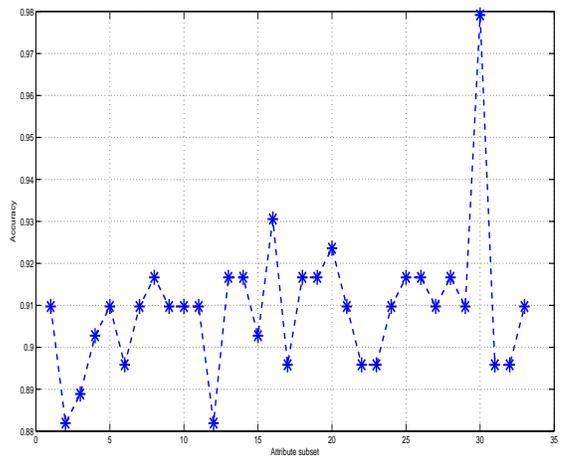


Figure 10: K - nearest neighbors classification accuracy for $k = 4$ when using 5-member subsets of attributes. Each subset contains attribute 14 and four consecutive attributes (in the order of their complementarity - see Table 3). Largest accuracy is 98%.

Acknowledgments

Esther van der Knaap acknowledges support from the NSF grant NSF DBI-0922661. Sofia Visa was partially supported by the NSF grant DBI-0922661(60020128) and by the College of Wooster Faculty Start-up Fund.

References

- Breiman, L.; Friedman, J.; Olshen, R.; and Stone, C., eds. 1984. *Classification and Regression Trees*. CRC Press, Boca Raton, FL.
- Gonzalo, M.; Brewer, M.; Anderson, C.; Sullivan, D.; Gray, S.; and van der Knaap, E. 2009. Tomato Fruit Shape Analysis Using Morphometric and Morphology Attributes Implemented in Tomato Analyzer Software Program. *Journal of American Society of Horticulture* 134:77–87.
- Guyon, I., and Elisseeff, A. 2003. An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research* 3:1157–1182.
- Jain, A., and Zongker, D. 1997. Feature selection: evaluation, application, and small sample performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19(2):153–158.
- Kira, K., and Rendell, L. 1992. A practical approach to feature selection. In *International Conference on Machine Learning*, 368–377.
- Kohavi, R., and John, G. 1997. Wrappers for features subset selection. *Artificial Intelligence* 97:273–324.
- Kohavi, R., and Provost, F. 1998. On Applied Research in Machine Learning. In *Editorial for the Special Issue on Applications of Machine Learning and the Knowledge Discovery Process*, Columbia University, New York, volume 30.
- Pudil, P.; Novovicova, J.; and Kittler, J. 1994. Floating search methods in feature selection. *Pattern Recognition Letters* 15(11):1119–1125.
- Rodriguez, S.; Moysenko, J.; Robbins, M.; Huarachi Morejn, N.; Francis, D.; and van der Knaap, E. 2010. Tomato Analyzer: A Useful Software Application to Collect Accurate and Detailed Morphological and Colorimetric Data from Two-dimensional Objects. *Journal of Visualized Experiments* 37.
- Sugeno, M., and Yasukawa, T. 1993. A fuzzy-logic-based approach to qualitative modeling. *IEEE Transactions on fuzzy systems* 1(1):7–31.
- Xin, E.; Jordan, M.; and Karp, R. 2001. Feature Selection for High-Dimensional Genomic Microarray Data. In *Proceedings of the 18 International Conference in Machine Learning ICML-2001*, 601–608.

A Preliminary Study on Clustering Student Learning Data

Haiyun Bian
hbian@mscd.edu

Department of Mathematical & Computer Sciences
Metropolitan State College of Denver

Abstract

Clustering techniques have been used on educational data to find groups of students who demonstrate similar learning patterns. Many educational data are relatively small in the sense that they contain less than a thousand student records. At the same time, each student may participate in dozens of activities, and this means that these datasets are high dimensional. Finding meaningful clusters from these datasets challenges traditional clustering algorithms. In this paper, we show a variety of ways to cluster student grade sheets using various clustering and subspace clustering algorithms. Our preliminary results suggest that each algorithm has its own strength and weakness, and can be used to find clusters of different properties. We also show that subspace clustering is well suited to identify meaningful patterns embedded in such data sets.

Introduction

Traditional data mining projects deal with datasets containing thousands or millions of records. Educational data mining tends to deal with much smaller datasets, normally in the range of several hundred student records. Even though a course may be offered multiple times, it is difficult to merge all these data records because every offering of the same course may involve different set of activities. On the other hand, many educational datasets are by nature high dimensional. For example, students' learning in a course may be assessed by utilizing an aggregation of several assignments, quizzes and tests. It is typical to have more than a dozen activities that contribute to a student's final grade. The log data from online teaching courses contain even more features describing the activities participated by each individual student.

Clustering is a very useful technique that finds groups of students demonstrating similar performance patterns. Since the number of students for each dataset rarely goes beyond a thousand and the number of features tends to be comparable to the number of students, finding coherent and compact clusters becomes difficult for this type of data. It is difficult because the pair-wise distance between students using the full-dimensional space becomes indistinguishable when the number of features becomes high. This problem is described as the curse of dimensionality, and it makes traditional clustering algorithms, such as k-means, unsuitable to be directly applied to high dimensional datasets.

Subspace clustering was proposed as a solution to this problem (Agrawal *et al.* 1998). Subspace clustering algorithms search for compact clusters embedded within subsets of features, and they have shown their effectiveness in domains that have high dimensional datasets similar to educational data. One specific example is its application to microarray data analysis. Microarray datasets tend to have similar sizes as educational datasets, mostly in the range of several hundred instances (genes or students) and several hundred features (samples or activities). Subspace clustering algorithms find subsets of genes that show similar expression levels under subsets of samples (Cheng *et al.* 2000; Madeira *et al.* 2004).

In this paper, we present some preliminary results from applying a variety of different clustering techniques, including subspace clustering, to student grade sheets. We show that clustering this type of datasets can provide the instructor a tool to predict who are likely to fail the course at very early stage as well as a possible explanation why they are failing.

Related Research

Over the last decade, many data mining techniques have been applied to educational data (Bravo *et al.* 2009; Dekker *et al.* 2009; Merceron *et al.* 2009). Research has shown that some techniques are more suitable for educational data than others, mainly because of the inherent characteristics of the datasets in this domain. For example, support and confidence, the two commonly used interestingness measurements for association rules, are not suitable for pruning off association rules when applied to educational data (Merceron *et al.* 2009). Instead, the authors have found that cosine and added value (or equivalently lift) are better measurements for educational data. One possible reason is that educational data have much smaller number of instances than the market basket data. Therefore, support and confidence tend to fall short in catching the real value of a good association rule in educational context.

Subspace clustering was first introduced to cluster students skill sets in (Nugent *et al.* 2009). The authors proposed to start with a "value-hunting" scanning for each individual feature to find out all features that contain meaningful and well-separated single-dimensional clusters. Those features that contain no good clusters were disregarded from further consideration. Then using all remaining features, conventional clustering algorithms such as hierarchical clustering and k-means were applied to identify clusters

resided in higher-dimensional spaces. In their research, subspace is used to prune off uninteresting features before the actual clustering process starts, and it is very similar to a feature selection procedure.

In general, a subspace cluster is a group of similar instances within their own subset of features. After the first subspace clustering algorithm for data mining was proposed (Agrawal *et al.* 1998), many different algorithms have been proposed. These algorithms can be classified into two categories: partition based approaches (Agrawal *et al.* 1999; Agrawal *et al.* 2000) and grid based approaches (or density-based approaches) (Agrawal *et al.* 1998; Cheng *et al.* 2000; Kriegel *et al.* 2009).

Partition-based algorithms partition all instances into mutually exclusive groups. Each group, as well as the subset of features where this group of instances show the greatest similarity is reported as a subspace cluster. Similar to k-means, most algorithms in this category define an objective function to guide the search. The major difference between these algorithms and the k-means algorithm is that the objective functions of subspace clustering algorithms are related to the subspaces where each cluster resides in. Notice that in subspace clustering, the search is not only on a partition on the instance set, but also on subspaces for each instance group. For example, PROCLUS (Agrawal *et al.* 1999) is a variation of the k-medoid algorithm. In PROCLUS, the number of clusters k and the average number of dimensions of clusters are specified before the running of the algorithm. This algorithm also assumes that one instance can be assigned to at most one subspace cluster or classified as an outlier, while a feature can belong to multiple clusters. Unlike PROCLUS that finds only axis-parallel subspace clusters, ORCLUS (Agrawal *et al.* 2000) finds clusters in arbitrarily oriented subspaces.

Grid-based (density-based) algorithms consider the data matrix as a high dimensional grid, and the clustering process is a search for dense regions in the grid. In CLIQUE (Agrawal *et al.* 1998), each dimension is partitioned into intervals of equal-length, and an n -dimensional unit is the intersection of intervals from n distinct dimensions. An instance is contained in a unit if the values of all its features fall in the intervals of the unit for all dimensions of the unit. A unit is dense if the fraction of the total instances contained in it exceeds an input parameter δ . CLIQUE starts the search for dense units from single dimensions. Candidate of n -dimensional dense units are generated using the downward closure property: if a unit is dense in k dimensions, all its $k-1$ dimensional projection units must all be dense. This downward closure property dramatically reduces the search space. Since the number of candidate dense units grows exponentially in the highest dimensionality of the dense units, this algorithm becomes very inefficient when there are clusters in subspaces of high dimensionality. Research has been done to extend CLIQUE by using adaptive units instead of rigid grids (Kriegel *et al.* 2009), as well as to use other

parameters such as entropy in addition to density to prune away uninteresting subspaces (Cheng *et al.* 2000).

Clustering Student Grade Sheets

We assume that datasets are in the following format: each row represents one student record, and each column measures one activity that students participate in the course. An example is shown in Table 1, where d_{ij} denotes the i th student's performance score in the j th activity. Most clustering and subspace clustering algorithms allow d_{ij} to take real values.

	Activity 1	Activity m
Stu 1	d_{11}	d_{1m}
Stu 2	d_{21}	d_{2m}
.....
Stu n	d_{n1}	d_{nm}

Table 1 Dataset Format

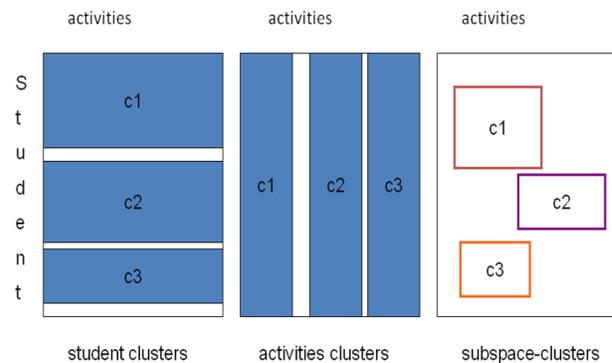


Figure 1. Clusters on Student Activity Data

Figure 1 shows three different clusters and subspace clusters that can be identified from the above data using different clustering and subspace clustering algorithms. Properties of each type of cluster as well as the process to find it will be presented in the following subsections.

The example dataset

We will use the grade sheet for a computer science service course as the example for this study. This dataset contains 30 students and 16 activities plus the final grade. The score of each activity as well as the final grade are in the range of $[0, 1]$. All students whose final composite grade is below .6 (60%) are marked as failing the course. In this dataset, 7 out of 30 students are marked as failed using this standard. Activities 1 through 12 are weekly in class labs in chronological order. Activities 13 and 14 are two large

projects due at mid semester and the end of the semester. Activities 15 and 16 are mid-term and the final examinations.

Each individual feature shows positive covariance with the final grade variable. Several features are highly correlated to the final grade, such as activities 6, 8, 9, 10 and 11. The least predictive features include activities 1, 13 and 15. It is not surprising for us to see that the first lab (activity 1) is not a good indicator of a student's performance in the course. But an interesting observation is that the mid-term exam (activity 15) and the mid-term project (activity 13) are both as bad as the first lab to tell whether a student will pass the course or not.

Another interesting observation is that activity 6 (lab 6) alone can predict with 100% accuracy about whether a student will pass the course or not. We later found out that the topic that was in that week was loops, which is considered challenging for most students. This suggest that if a student can grasp the concept of loop structure very well, he might as well be able to pass the course as a whole. Therefore, it would be worthwhile for the instructor to spend more time and effort on this subject matter.

Feature	Covariance	Feature	Covariance
<u>Activity 1</u>	<u>.5044</u>	Activity 9	.9075
Activity 2	.7758	Activity 10	.9089
Activity 3	.6097	Activity 11	.9067
Activity 4	.7415	Activity 12	.8137
Activity 5	.7125	<u>Activity 13</u>	<u>.5283</u>
Activity 6	.9787	Activity 14	.8427
Activity 7	.8670	<u>Activity 15</u>	<u>.5613</u>
Activity 8	.9065	Activity 16	.8046

Student clusters

Here we focus on identifying groups of students who demonstrate similar performances throughout the whole course. This type of clusters can be useful for the instructor to identify key activities that differentiate successful students from those who fail the course.

We have tried a wide variety of clustering algorithms' available from Weka (Weka URL) on the example dataset, and the results show that the simple k-means algorithm achieves at least comparable results as other more complicated algorithms in almost all cases.

Using the simple k-means algorithm, we started with k=2, that is, to find two clusters (Cluster0 and Cluster1) from this dataset. Cluster0 contains 6 out of 7 students who actually failed the course, and cluster1 contains 24 students among whom 23 are marked as passing the course. There is one failing student who is clustered into cluster1. We found out that this student's composite final score is .58, which lies right on the boundary of passing/failing

threshold. This suggests that choosing 0.6 as the passing/failing threshold seems rather arbitrary.

Figure 2 shows the centroids of the two clusters. We can see that some activities are better in differentiating the two clusters than others, such as activities 6, 8, 9, 10 and 11. This result is consistent with the result from individual feature's covariance with the final grade variable, suggesting that the clusters that were identified from the algorithm may have captured some real characteristics of the dataset.

We have also tried k=3 to find three clusters from this dataset. It resulted in cutting the failing cluster (cluster0) into two even smaller clusters, leaving cluster1 remain unchanged.

Features	Full Data(30)	cluster0(6)	cluster1(24)
Activity1	0.76	0.475	0.8313
Activity2	0.7533	0.3208	0.8615
Activity3	0.7908	0.3333	0.9052
Activity4	0.785	0.3333	0.8979
Activity5	0.815	0.3792	0.924
Activity6	0.7767	0.0708	0.9531
Activity7	0.79	0.3083	0.9104
Activity8	0.7983	0	0.9979
Activity9	0.7683	0	0.9604
Activity10	0.7308	0	0.9135
Activity11	0.7833	0.1667	0.9375
Activity12	0.7667	0.1667	0.9167
Activity13	0.7647	0.5767	0.8117
Activity14	0.667	0	0.8338
Activity15	0.7693	0.7844	0.7656
Activity16	0.5674	0.2278	0.6523

Figure 2. Centroids of Student Clusters (K = 2)

Activity clusters

In this section we take a different view on the same dataset. Here we focus on finding groups of activities in which all students demonstrate similar performance patterns. For example, we may find a group of activities on which all students demonstrate consistently high performance. This suggests that these activities involve relatively easy-to-grasp concepts. We may also find a group of activities where all students show worst than average performance. This suggests that the instructor may want to spend more time on these activities to cope with the difficulty.

To find this type of clusters, we would need to transpose the original data matrix as shown in Table 1 into Table 2.

	Student 1	Student n
--	-----------	-------	-----------

Activity 1	d11	dn1
Activity 2	d12	dn2
.....
Activity m	d1m	dnm

Table 2. Transposed Dataset

Similar as above, we applied the SimpleKMeans from Weka to the transposed example dataset. We tested five k values in the range of 2 to 6, and we tried to find the best value of k by looking at curve of within-group-variance as a function of k. The result is shown in Figure 3. As we can see, four clusters seems to be the best because the slope of the curve reduced significantly after k=4.

Among four clusters of activities, cluster3 is the most challenging group of activities because its cluster centroid is consistently lower than the other three clusters. Cluster3 contains three activities including activity 13, activity 15 and activity 16. Out of 30 students, 13 students showed significant lower than average performance on these three activities. An interesting observation is that in previous sections we have pointed out that activities 13 and 15 are also considered as insignificant in differentiating passing students from failing ones. This mean that these two activities maybe too hard to be used as criteria to predict the student overall performance of the course. On the other hand, activity 1, which is also considered as a bad feature to tell the difference between passing students from those who failed, might be too easy to be used as a criterion for that purpose.

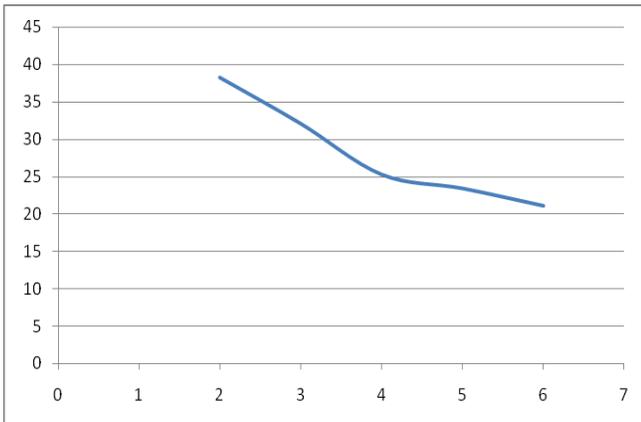


Figure 3. Within Group Variances

Subspace clusters

In this section, we show that subspace clustering algorithms can be used to find clusters embedded in subspaces. In earlier section, we have shown that student

clusters contain groups of students who demonstrate similar performance throughout the whole course. Here we relax the constraint to allow any groups of students who demonstrate similar learning patterns in any subsets of activities to become candidates for clusters.

We will first show the results from partition-based subspace clustering algorithm. We chose to use PROCLUS because it reports clusters in axis-parallel subspaces, which makes the final interpretation of the clusters easier. The PROCLUS implementation is from the open source subspace clustering package (OpenCluster URL).

Similar to K-means, PROCLUS needs a pre-determined number of clusters (k) before running. In addition, it also requires knowing the average subspace dimensionality (l). We set k=4, and tried several values of l between 2 and 5, and found out that the results are highly similar for all cases. For the example dataset, PROCLUS finds the following four clusters when we set k = 4 and l = 3:

```
SC_0: [0 0 0 0 0 0 1 0 0 0 1 0 0 0 0] #13 {2 5 7 8 10 13
14 15 17 21 23 27 29 }
SC_1: [0 0 1 0 0 0 0 1 1 1 0 0 0 0 0] #11 {0 1 4 12 16 19
20 24 25 26 28 }
SC_2: [0 0 0 0 0 1 1 0 0 0 0 0 0 0 0] #4 {3 6 9 22 }
SC_3: [0 1 0 0 0 0 0 1 1 1 0 0 0 0 0] #2 {11 18 }
```

Each line describes one subspace cluster. For example, the first subspace cluster (SC_0) lies in a subspace that contains two features: activity 8 and activity 12. SC_0 contains 13 students, and they are: stu2, stu5, stu7, and etc.

A simple investigation shows that SC_2 and SC_3 contain all failing students. In SC_2, 4 out of 6 students who fail this class have difficulty in doing activity 6 and activity 7, and SC_3 shows that the other two failing students showed difficulty in doing activity 2 and activities 8, 9 and 10. We later found out the activity 6 was a lab on loop structure and labs 8 and 9 are labs on Classes and Objects. This suggests that the majority of the students who failed this course started to fall behind when loops were introduced. The other half who failed the class failed to catch up when the concept of objected oriented programming were introduced. Therefore, the instructor may want to spend extra time to help students complete these three activities.

We can also see that SC_1 and SC_3 are two clusters that are best contrasted by activities 6, 7 and 8. Since all students in SC_1 passed the course while SC_3 students failed the course, these three activities may be crucial for students to pass the course.

We have also tried partition-based subspace clustering algorithm on the sample data. Grid-based algorithms produce more than a thousand subspace clusters, and the large number of reported clusters makes the interpretation of clusters very difficult. We will look into the possibility to prune off insignificant clusters based on domain knowledge. Similar research has been done in bio-medical

data analysis, where domain knowledge is used to measure the significance of each bi-cluster.

Comparisons between the three

Student clusters represent groups of students showing similar performance patterns throughout the whole course, while subspace clusters shows clusters of students who demonstrate similar performances in subsets of activities. Activity clusters is helpful in finding out difficult tasks for all students, while subspace clusters can identify subsets of activities that challenge different groups of students. Since not all students experience the same difficulty in all activities, subspace clustering seems to be well suited for this purpose. We can see from the example data that activity 6 may be a good feature to tell why some students failed this course, but it is not the only indicator. SC_3 suggests that there are some students who had no problem finishing activity 6 but still failed the course due to their unsatisfactory performance in activities 8, 9 and 10.

Conclusions and Future Work

This paper is our first attempt to adopt a rich collection of subspace clustering algorithms on educational data. Our preliminary results show that clustering and subspace clustering techniques can be used on high dimensional education data to find out interesting student learning patterns. These cluster patterns are helpful for the instructor to gain insights into the different learning behaviors and adapt the course to accommodate various students' needs. We will test and validate all presented clustering schemes on more educational data of larger size. We will also look into the possibility of applying grid-based subspace clustering algorithms to educational data guided by domain knowledge.

References

Aggarwal C. C., Wolf J. L., Yu P. S., Procopiuc C., and Park J. S. Fast Algorithms for Projected Clustering, *Proceedings of the 1999 ACM SIGMOD international conference on Management of data (SIGMOD'99)*, pp. 61-72, 1999

Aggarwal C. C., and Yu P. S. Finding Generalized Projected Clusters in High Dimensional Spaces, *Proceedings of the 2000 ACM SIGMOD international conference on Management of data (SIGMOD'00)*, pp. 70-81, 2000

Agrawal R., Gehrke J., Gunopulos D., and Raghavan P. Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications, *Proceedings of the 1998 ACM SIGMOD international conference on Management of data (SIGMOD'98)*, pp. 94-105, 1998

Bravo J., Ortigosa A. Detecting of Low Performance Using Production Rules, *Proceedings of the Second International Conference on Educational Data Mining*, 2009. p. 31-40

Cheng C. H., Fu A. W.-C., and Zhang Y. Entropy-based Subspace Clustering for Mining Numerical Data, *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'99)*, pp. 84-93, 1999

Cheng Y. and Church G. M. Biclustering of Expression Data, *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology (ISMB'00)*, pp. 93-103, 2000

Dekker G. W., Pechenizkiy M., and Vleeshouwers J. M. Predicting Students Drop Out: A Case Study, *Proceedings of the Second International Conference on Educational Data Mining*, 2009. p. 41-50

Kriegel H. P., Kroger P., and Zimek A. Clustering High-dimensional data: A Survey on Subspace Clustering, *Pattern-based Clustering, and Correlation Clustering, Transactions on Knowledge Discovery from Data (TKDD)*, Vol. 3, 2009

Madeira S., Oliveira A. Biclustering Algorithms for Biological Data Analysis: a survey, *IEEE Transactions on Computational Biology and Bioinformatics*, vol. 1, pp. 24-24, 2004

Merceron A., and Yacef K. Interestingness Measures for Association Rules in Educational Data, *Proceedings of the Second International Conference on Educational Data Mining*, 2009. p. 57-68

Nugent R. Ayers, E., Dean N. Conditional Subspace Clustering of Skill Mastery: Identifying Skills that Separate Students, *Proceedings of the Second International Conference on Educational Data Mining*, 2009. p. 101-110

OpenClusters:<http://dme.rwthachen.de/OpenSubspace/>

Weka: <http://www.cs.waikato.ac.nz/ml/weka>

Learning Morphological Data of Tomato Fruits

Joshua Thomas, Matthew Lambert, Benjamin Snyder,
Michael Janning, Jacob Haning, Yanglong Hu, Mohammad Ahmad, Sofia Visa

Computer Science Department

College of Wooster

jet4416@gmail.com, mlambert13@wooster.edu, benn.snyder@gmail.com,

mjanning13@wooster.edu, jacob.haning1@gmail.com,

yhul2@wooster.edu, mahmad12@wooster.edu, svisa@wooster.edu

Abstract

Three methods for attribute reduction in conjunction with Neural Networks, Naive Bayes, and k-Nearest Neighbor classifiers are investigated here when classifying a particularly challenging data set. The difficulty encountered with this data set is mainly due to the high dimensionality and to some imbalance between classes. As a result of this research, a subset of only 8 attributes (out of 34) is identified leading to a 92.7% classification accuracy. The confusion matrix analysis identifies class 7 as the one poorly learned across all combinations of attributes and classifiers. This information can be further used to upsample this underrepresented class or to investigate a classifier less sensitive to imbalance.

Keywords: classification, attribute selection, confusion matrix;

Introduction

Knowing (or choosing) the best machine learning algorithm for classifying a particular real world data set is still an ongoing research topic. Researchers have tackled this problem more as experimental studies, such as the ones shown in (Michie, Spiegelhalter, and Taylor 1999) and (Visa and Ralescu 2004), than as theoretical ones. Currently, it is difficult to study the problem of the best classification method given a particular data set (or the reverse problem for that matter), because data classification depends on many variables, e.g. number of attributes, number of examples and their distribution across classes, underlying distribution along each attribute, etc. Additionally, it is difficult to study classifier induction in general, because different classifiers learn in different ways, or stated differently, different classifiers may have different learning biases. Thus, this research focuses on finding the best classification method for a particular data set of interest through experimental means.

We investigate several machine learning techniques for learning a particular 8-class domain having 34 attributes and only 416 examples. We also combine these methods with various subsets of attributes selected based on their discriminating power. The main goal of this research is to find the

best classification algorithm (or ensemble of algorithms) for this particular data set.

The research presented here is part of a bigger project, with the classification of morphological tomato data (i.e. data describing the shape and size of tomato fruits such as the data set used here) being the first step. Namely, having the morphological and gene expression data, the dependencies between these two sets are to be investigated. The goal of such computational study is to reveal genes that affect particular shapes (e.g. elongated or round tomato) or sizes (e.g. cherry versus beef tomato) in the tomato fruit. However, as mentioned above, a high classification accuracy of tomatoes based on their morphological attributes is required first.

Table 1: Data distribution across classes. In total, there are 416 examples each having 34 attributes. (Table from (Visa et al. 2011))

Class	Class label	No. of examples
1	Ellipse	110
2	Flat	115
3	Heart	29
4	Long	36
5	Obvoid	32
6	Oxheart	12
7	Rectangular	34
8	Round	48

The 8 classes are illustrated in Figure 1 and the distribution of the 416 examples is shown in Table 1 (Visa et al. 2011).

The Tomato Fruit Morphological Data

The experimental data set was obtained from the Ohio Agricultural Research and Development Center (OARDC) research group led by E. Van Der Knaap (Rodriguez et al. 2010).

This morphological data of tomato fruits consists of 416 examples having 34 attributes and distributed in 8 classes. The

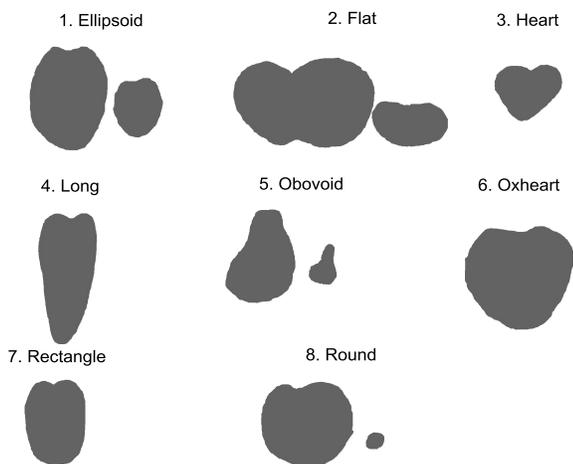


Figure 1: Sketch of the 8 morphological classes of the tomato fruits.

34 attributes numerically quantify morphological properties of the tomato fruits such as perimeter, width, length, circularity (i.e. how well a transversal cut of a tomato fits a circle), rectangle (similarly, how well it fits a rectangle), angle at the tip of the tomato, etc. A more detailed description of the 34 attributes and how they are calculated can be found in (Gonzalo et al. 2009).

Problem Description and Methodology

The focus of this research is to find the best (defined here as high classification accuracy, e.g. 90%) classification technique (or combination of classifiers) for the morphological tomato data set. In addition to tomato fruit classification, it concentrates on finding which attributes have more discriminative power and finding a ranking of these attributes.

As seen in Figure 1, the round class and several others may have smaller or much larger instances of tomato fruits. Thus, attributes 1 (perimeter) and 2 (area), for example, might have no positive influence in classifying these classes; at worst, it may hinder classification. The tomato data set of interest here has 34 attributes and only 416 examples available for learning and testing. One can argue that many more examples are needed to have effective learning in such high-dimensional space. Furthermore, the class-distribution is imbalanced with the largest and the smallest classes having 115 (class 2, Flat tomatoes) and 12 examples (class 6, Oxheart tomatoes), respectively (see Table 1).

For these reasons, our strategy is to investigate several machine learning classifiers on subsets of top-ranked attributes in an effort to reduce the data-dimensionality and to achieve better data classification. Finding if different classification algorithms make identical errors (for example, they all mis-

classify class 7 with class 1) is also of interest in this experimental study. Our hypothesis is that (some) different classifiers misclassify different data-examples and thus, by combining different classifiers, one can achieve better accuracy merely through their complementarity. The misclassification error for each individual class is tracked through the use of confusion matrices.

Attribute Selection Techniques

Two filter-methods (analysis of variance ANOVA (Hogg and Ledolter 1987) and the RELIEF method (Kira and Rendell 1992b), (Kira and Rendell 1992a)) and one wrapper-method are used in our experiments for attribute-ranking. The first two algorithms are independent of the choice of classifier (Guyon and Elisseeff 2003), whereas the third one is "wrapped" around a classifier - here the attributes selected by the CART decision tree are used (Breiman et al. 1984).

The first attribute ranking method considered here is based on the analysis of variance which estimates the mean value of each attribute by comparing the variation within the data (Hogg and Ledolter 1987).

The second ranking method we use is the RELIEF algorithm, introduced by (Kira and Rendell 1992b) and described and expanded upon by (Sun and Wu 2009). In short, the algorithm examines all instances of every attribute and calculates the distance to each instance's nearest hit (nearest instance that has the same classification) and nearest miss (nearest instance that has a different classification). It then calculates the differences of the nearest misses and the nearest hits over all instances of each attribute. as shown in equation (1).

$$d_n = \|x_n - NearestMiss(x_n)\| - \|x_n - NearestHit(x_n)\| \quad (1)$$

where d_n is an instance of an attribute, $NearestMiss(x_n)$ is the nearest miss of the instance, and $NearestHit(x_n)$ is the nearest hit of the instance. Then, the d-values are summed over all instances and the attributes are ranked from largest value to smallest value. Zero may provide an appropriate cut-off point when selecting attributes.

The third ranking is obtained as a result of decision trees classification (CART), which through a greedy approach places the most important attributes (based on information gain) closer to the root of the tree.

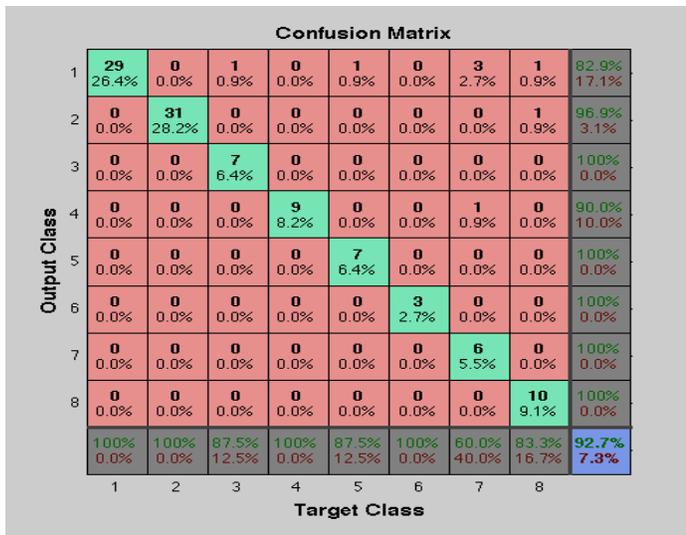


Figure 2: Confusion matrix of NN for top 8 CART attributes. This case achieved the highest classification accuracy when using NN (92.7%).

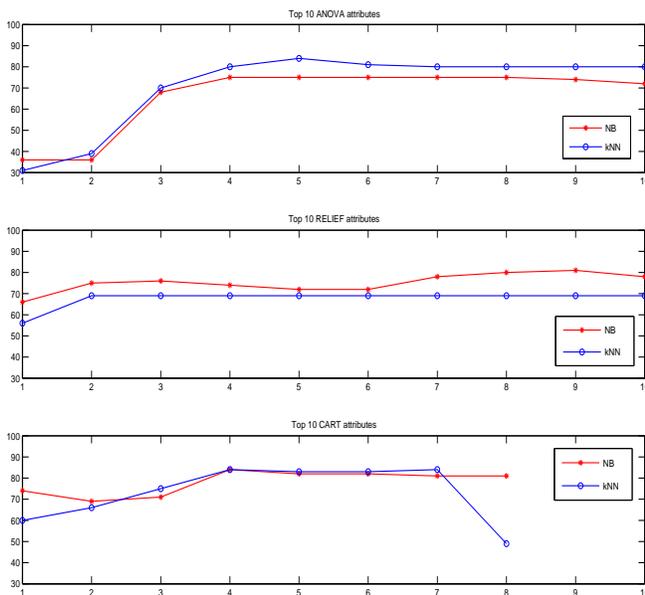


Figure 3: Accuracy of NB and kNN for top k (k=1,10) ANOVA attributes (top figure), top k (k=1,10) RELIEF attributes, and top k (k=1,8) CART attributes (k is shown on x-axis).

Table 2: Top 10 ANOVA and RELIEF attribute rankings. Column 3 shows the top 8 ranked attributes resulted from classification and regression trees (CART) (Visa et al. 2011)

	ANOVA	RELIEF	CART
17		21	7
20		18	13
18		7	12
21		8	11
2		33	14
1		13	10
28		11	8
26		9	1
6		19	-
5		22	-

Classification Techniques

We use Matlab to conduct these experiments. For each experiment 75% of data is randomly selected for training, and the remaining 25% of data is used for testing.

Matlab implementations of the Naive Bayes (NB), k-nearest Neighbors (kNN) for k=4, and various Artificial Neural Network (NN) configurations are tested in conjunction with the three reduced-attribute tomato data sets, as well as with the whole data sets (i.e. having all 34 attributes). For the latter case, the classifiers are ordered by their accuracies as follows: NN (89.1%), NB (80%), kNN (79.1%). Here, kNN is investigated for k=4 only because (Visa et al. 2011) shows that it achieves the lowest error over a larger range of k.

Results

The top 10 ANOVA and RELIEF attribute rankings are shown in the first two columns of Table 2. Column 3 shows the top 8 ranked attributes resulted from classification and regression trees.

NN Results

Many NN configurations (in terms of number of layers, number of neurons in each layer, training method, and activation function) for each of the three data sets obtained from selecting the subsets of attributes shown in Table 2 were tried. However, only the ones leading to the best results are reported in Table 3. Among the subsets of attributes studied here, the 8 attributes resulting from the decision tree classification lead to the best classification in the NN case (92.7%). The confusion matrix associated with this case is shown in Figure 2. From this matrix, it can be seen that the largest error comes from misclassifying 3 test data points of class 7 (Rectangle) as class 1 (Ellipsoid). Indeed, Figure 1 shows that these two classes are the most similar in terms of shape.

Table 3: Best NN configurations and their corresponding classification accuracies.

No. of attributes	No. of layers	No. of neurons	Accuracy
Top 10 ANOVA	1	10	84.5%
Top 10 RELIEF	2	25+15	88.2%
Top 8 CART	1	10	92.7%
All 34	2	25 +15	89.1%

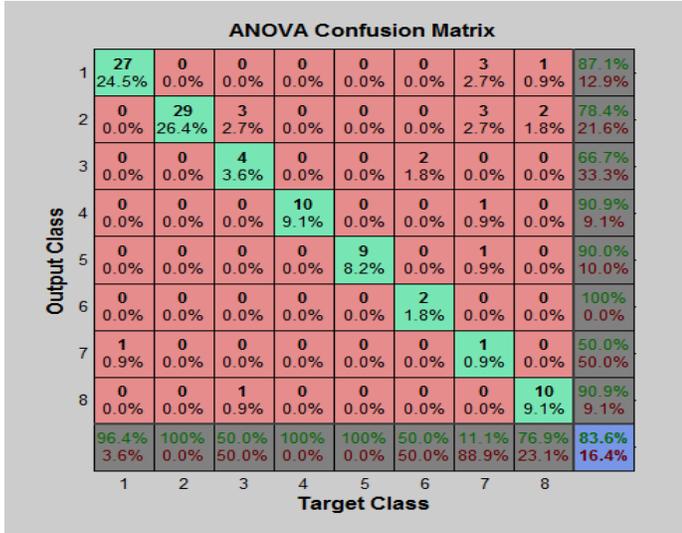


Figure 4: Confusion matrix of kNN for top 5 ANOVA attributes. This case achieved the highest classification accuracy when using kNN (83.6%).

NB and kNN Results

Figure 3 shows the accuracy of NB and kNN for top k (k=1,10) ANOVA attributes (top figure), top k (k=1,10) RELIEF attributes, and top k (k=1,8) CART attributes (k is shown on x-axis). The two largest accuracy values are obtained for kNN (83.6%) for the top 5 ANOVA attributes, and for NB (81.1%) in the case of top 9 RELIEF attributes. For these two cases, the confusion matrices showing the misclassifications across the 8 classes are shown in Figures 4 and 5, respectively. Similar to NN classifier, NB and kNN both misclassify class 7 as class 1 (by 4 and 3 examples, respectively). However, contrary to NN, NB and kNN carry some additional class confusions:

- NB misclassifies class 3 as class 1 (3 instances) and as class 8 (3 instances);
- Additional error for kNN comes from misclassifying class 3 as class 2 (3 examples) and class 7 as class 2 (3 examples).

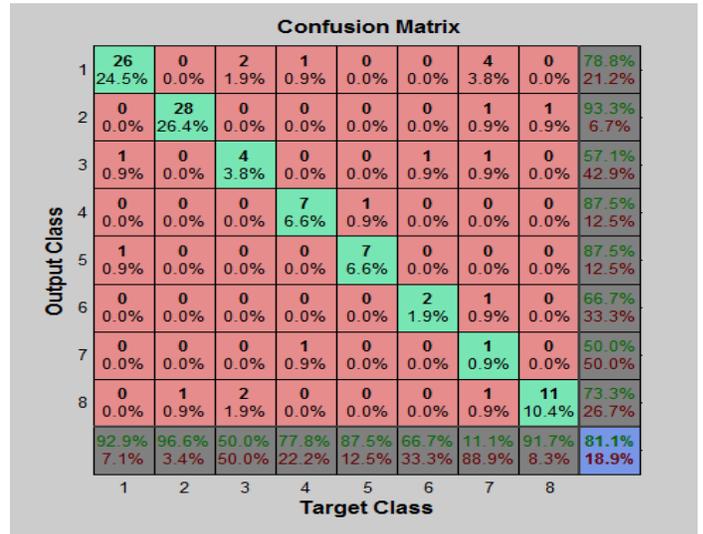


Figure 5: Confusion matrix of NB for top 9 RELIEF attributes. This case achieved best classification accuracy when using NB (81.1%).

Conclusions and Future Work

Several machine learning algorithms for classifying the 8-class tomato data are investigated here. In addition, 3 attribute selection strategies are combined with these learning algorithms to reduce the data set dimensionality. The best combination of attribute selection and classification method among the ones investigated here leads to a 92.7% classification accuracy (for the NN classifier on the 8 CART attributes).

The confusion matrix analysis points out that class 7 (Rectangle) is the one most frequently misclassified (or very poorly learned) across all three classifiers. It is more often misclassified as class 1. This is consistent with the observation that (1) based on Figure 1, these two classes are very similar, and (2) since class 1 is larger in terms of available examples (110 versus only 34 in class 7, see Table 1), we can conclude that the classifiers are biased toward the larger class. This situation is known in literature as learning with imbalanced data (Visa and Ralescu 2004). As a future direction, we point out that, for imbalanced data sets, classifiers less sensitive to the imbalance can be used such as the one proposed in (Visa and Ralescu 2004). Also, the imbalance can be corrected by intentional upsampling (if possible) of the underrepresented class.

A similar study that considers some additional classification techniques applied to a larger overall data set (the 416 examples in the current data sets poorly covers the 34-dimensional space) in which the classes are less imbalanced will provide more insight as to what attributes should be selected for better classification accuracy. Also, a more thorough analysis of the confusion matrices will identify com-

plementary classification techniques which can be subsequently combined to obtain a larger classification accuracy for the data set of interest.

Acknowledgments

This research was partially supported by the NSF grant DBI-0922661(60020128) (E. Van Der Knaap, PI and S. Visa, Co-PI) and by the College of Wooster Faculty Start-up Fund awarded to Sofia Visa in 2008.

References

Breiman, L.; Friedman, J.; Olshen, R.; and Stone, C., eds. 1984. *Classification and Regression Trees*. CRC Press, Boca Raton, FL.

Gonzalo, M.; Brewer, M.; Anderson, C.; Sullivan, D.; Gray, S.; and van der Knaap, E. 2009. Tomato Fruit Shape Analysis Using Morphometric and Morphology Attributes Implemented in Tomato Analyzer Software Program. *Journal of American Society of Horticulture* 134:77–87.

Guyon, I., and Elisseeff, A. 2003. An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research* 3:1157–1182.

Hogg, R., and Ledolter, J., eds. 1987. *Engineering Statistics*. New York:MacMillan.

Kira, K., and Rendell, L. 1992a. The Feature Selection Problem: Traditional Methods and a New Algorithm. In *Proceedings of AAAI*, 129–134.

Kira, K., and Rendell, L. 1992b. A practical approach to feature selection. In *International Conference on Machine Learning*, 368–377.

Michie, D.; Spiegelhalter, D.; and Taylor, C. e., eds. 1999. *Machine Learning, Neural and Statistical Classification*. <http://www.amsta.leeds.ac.uk/~charles/statlog/>.

Rodriguez, S.; Moysenko, J.; Robbins, M.; Huarachi Morejn, N.; Francis, D.; and van der Knaap, E. 2010. Tomato Analyzer: A Useful Software Application to Collect Accurate and Detailed Morphological and Colorimetric Data from Two-dimensional Objects. *Journal of Visualized Experiments* 37.

Sun, I., and Wu, D. 2009. Feature extraction through local learning. In *Statistical Analysis and Data Mining*, 34–47.

Visa, S., and Ralescu, A. 2004. Fuzzy Classifiers for Imbalanced, Complex Classes of Varying Size. In *Proceedings of the Information Processing and Management of Uncertainty in Knowledge-Based Systems Conference, Perugia, Italy*, 393–400.

Visa, S.; Ramsay, B.; Ralescu, A.; and Van der Knaap, E. 2011. Confusion Matrix-based Feature Selection. In *Proceedings of the 23rd Midwest Artificial Intelligence and Cognitive Science Conference, Cincinnati*.

A Qualitative Analysis of Edge Closure in Information Networks

Hareendra Munimadugu

School for Electronics & Computer Systems
Machine Learning & Computational Intelligence Lab
School of Computing Sciences and Informatics
University of Cincinnati
Cincinnati, OH 45221-0030
munimah@mail.uc.edu

Anca Ralescu

Machine Learning & Computational Intelligence Lab
School of Computing Sciences & Informatics
University of Cincinnati
Cincinnati, Oh 45221-0030
anca.ralescu@uc.edu

Abstract

Social Networks Analysis is one of the cutting edge research areas which finds applications in Information Retrieval and Data Mining and which deals with very large amounts of data. A considerable amount of research in this field has focused on mining useful information patterns in networks. Other applications have focused primarily on structure of networks. Several models have been proposed to address both issues. Existing models have been developed and replaced with better ones. One direction of this research has focused on how to implement one method of analysis on several data associations in order to understand how different data models behave. Responses to such questions account for the underlying differences in properties of the data considered. In its broadest sense, a community is a large set of nodes that have been collected over a period of time. In the present scenario efforts are being made to develop a complete model that can correctly explain and predict how links are formed in networks and how a network of nodes dynamically "progresses" into a bigger and more diverse one. If an idea is implemented on networks derived from different domains the results can account for the underlying differences in the properties of the data. Such an approach is useful because it helps to find not only differences based on domain but also identify similarity and therefore to correlate one domain with another.

Introduction

The ideas of human relations in a community have received particular attention since time immemorial. Therefore community structure and expansion are two areas that have been studied by philosophers, sociologists, and, more recently, by computer scientists. It is an exciting venture to try to develop a method that can account for the way in which people form and maintain relations with one another. This brings us to the community structure analysis and link prediction and it has become one of the important research problems in analysis of social networks. In this paper we take up one recent method of evaluating link formation in social communities and we compare its application to two different networks. In other words we do this by applying the same method on two different collections of data and comparing results. We then identify some possible criteria due to which significant

differences exist with the application of the same technique to the different networks. Pre-evaluating links is exciting; since it is interesting to be able to predict who someone's friends are before they actually form friendship in the network. In a similar context we can say that a particular user in a network will be able to efficiently develop and retain friendships because of his position in the network and the possibility of forming links in this manner. Many methods to evaluate community growth have concentrated on networks involving people and tried to explain about their relationships. However it is also quite interesting to see how some of these methods apply to similar networks not involving people, rather involving different aspects about people.

Keywords

Social Network, Information Network, Edge closure, Link Formation, Nodal Analysis, Closure ratio.

Problem Formulation

Computational analysis of Social Networks continues to be a subject of intense interest and. The problem of social ties is believed to be related to structure and function of social networks (Granovetter 1985). Positive and negative links play a major role in the same (Bogdanov, Larusso, and Singh 2010); this interplay has given foundation to many methods developed for efficient determination of influence and opinion (Leskovec, Huttenlocher, and Kleinberg 2010). The process of link copying which is implicit in all information networks and which explains the formation of new links is known as the directed closure process (Romero and Kleinberg 2010). The methodologies as mentioned above have been extensively used for understanding information networks consisting of people. Examples of such networks are Slashdot, twitter, Facebook and several others. Here we try to consider information networks not consisting exclusively of people though they contain relevant and important information about people. Here we are interested to see how this methodology applies to information networks having several different characteristics. We would also like to see how

differences in basic characteristics of networks account for observed differences in network formation. The idea of edge closure is one of the recent important developments in predicting link formation and community structure. Our objective here is to implement this method on different data sets or information networks with inherently distinct properties and concentrate upon what characteristics of the data sets might be influential in the difference of community structure by application of the same technique. As stated we focus our attention upon networks which involve not people but some practical aspects regarding people. One such practical aspect is to determine the interest and opinion of a group of people with respect to a certain trend or their attitude towards a topic. This type of treatment of networks can be useful to also determine closeness between various domains of knowledge.

Much research has been carried out on social networks consisting of undirected edges. Evaluation of an information network such as YouTube videos is comparatively challenging because closing of undirected triangles of nodes in a social network is relatively easier. It is instinctively known that the application of the same method will lead to different observation and result in different information networks. The short term goal of this research is to study the relevance and effect of directed closure on various information networks.

Towards Problem Resolution

It is a significant fact that in order to carry out such a research of an already developed technique, we have various kinds of online social communities available. Social networks (Facebook, Orkut, and connotation networks), Information networks (YouTube, Wikipedia, Web blogs, news blogs), Hybrid networks (twitter) and signed networks (Slashdot, Epinions) are among some examples of what communities we can possibly consider. However here we consider the information networks and try to provide a clear and concise analysis of such networks. We are interested in how the concept of edge closure is applicable to an information network. Some of the examples of such a network are E-book repositories, phishing corpus, Wikipedia pages, text corpus and YouTube. Because YouTube and Wikipedia are some of the largest networks available on the World Wide Web, and also because the information in these networks is freely available, we take these two information networks for the purpose of research in this paper. Therefore the data sets that we consider for this research are derived from YouTube videos and Wikipedia pages. These are essentially collection of tuples. In the data consisting of YouTube videos, each video is taken as a node in the information network. One node is directionally connected to another if the first video has the second in the list of top ten 'favorites' or 'suggestions'. Considering this order is important because the initial few videos from YouTube hits are going to be those that are closely associated with the main or desired result. Thus an edge is generated between nodes in a directed graph.

In the data set on Wikipedia, a directed edge exists between two nodes in the order of connection of one page and a page in the references list. We shall start with a test data collection of a hundred nodes where each node is a YouTube video. Two data sets are used simultaneously to test our hypothesis. The other data set consists of an equal number of Wikipedia links or pages.

Remark 1 *It is actually not essentially important that both data sets contain an equal number of data points because what we are primarily interested is the extent of closure or in other words the percentage of nodes exhibiting this phenomenon.*

For the sample data set of YouTube, we have a collection of tuples where each tuple has two nodes and the order in which they are connected is specified; it is the direction of the edge.

Data points for the Wikipedia information are also comprised of a collection of tuples. It is very clear that the number total of tuples in the flat file is the number of relations or the number of edges in the graph.

Experimental Design

As a starting example we begin with a specific context or theme in a query in YouTube.

An edge exists between two nodes in the graph if the source video has the destination video in its top ten favorites or suggested videos' list. This edge is going to be directional because it points towards one node starting from another as in the favorites list.

An edge exhibits closure if it completes a triangle between three nodes in order, or if it is going to be the edge that completes the triangle (Romero and Kleinberg 2010). As a working example, we shall consider "University of Cincinnati Engineering" as our theme or our concept for the data analysis. This means that is the query the user is interested in. Four of the related nodes in YouTube are "Tips to succeed in Engineering", "Is Engineering right for me?", "How an engineer folds a T-shirt" and "Advice for Engineering Students".

A similar query on Wikipedia yields four of the data points from Wikipedia data which are "University of Cincinnati College of Engineering", "University of Cincinnati", "graduate students" and "University of Cincinnati College of Design, Architecture, Art and Planning".

We can notice that, because the nodes are based on the same theme, they might be already linked, but what is more important is whether they exhibit closure.

When the results are taken as nodes in a directed graph, the nodes present in the graph formed from the YouTube results are as shown in Table while those present in the graph formed from the Wikipedia results are shown in Table .

Table 1: YouTube results

Node 1	:	Tips to succeed in engineering
Node 2	:	Is engineering right for me?
Node 3	:	How an engineering student folds a T-shirt
Node 4	:	Advice for engineering students

Table 2: Wikipedia results

Node 1	:	University of Cincinnati College of Engineering
Node 2	:	University of Cincinnati
Node 3	:	Graduate students
Node 4	:	University of Cincinnati College of Design, Architecture, Art and Planning

Directional connectivity exists in these networks in the following manner. In the first example, node 1 has nodes 2 and 3 in its top ten hits. Therefore the directed arrow in the graph exists from node 1 to node 2 and also from node 1 to node 3. Similarly node 2 and node 3 have node 4 in their hit list. The directed closure is satisfied if node 1 has node 4 in the list. In the example taken the closure does exist because node 4 is indeed present in the hit list of node 1. Similarly, in the second example directed connectivity exists between the nodes considered. A similar explanation holds for the second example also. In our testing phase over large data sets we make use of the idea that for large data sets nodal closure can be evaluated based upon analyzing ordered lists prepared for a node [4]. This is the additional information needed because we cannot always have time stamps for such data. We shall carry out this test of edge closure on both data sets taking an equal number of nodes at one time. For the testing example the result is that in both cases directional closure is satisfied. In the data sets we consider, the number of connections between nodes is fairly large. Many of the arrows are also bidirectional. The expected result of the experiment is that in the case of YouTube there will be a significantly more closure overall linkage when compared to Wikipedia pages. This, according to the hypothesis is based upon the search on a theme or idea. We can give the following reasoning to the expected result. In a search on YouTube the resultant videos or hits are based upon the overall query or collection of phrases or words. The videos will be displayed based on the theme of the query, which means combined meaning of words is important. In such a case there will be emphasis on certain words that are important. Unimportant words, though a part of query do not affect search results significantly. We might say that a change of these unimportant words does not cause the results to vary significantly. But results will change drastically should the main words change. However in case of the results displayed in Wikipedia pages

are retrieved based on words, meaning that a change of a word might result in a possible difference in context in the search. In Wikipedia even though combined meaning of words is important sometimes a change of any one word might result in a related but different result. This analysis applies to both the information networks, but significance in words or phrases varies for each network. These are information networks and therefore dynamic; when a new node is added to the network, new links form between the node and its neighbors. However in some cases this leads to the change in links between the existing nodes. One example of this is as follows. When the new node is very closely related to some of the neighbors there can be a change because the new node might be included in the top hits, thereby removing an already linked node from the top ten favorites list. However this is obviously not always possible. Hence within the same context the inclusion of a new node in YouTube signifies two things: that a new node has really arrived, or that a video moves up in the list of hits of another video, causing a replacement.

Conclusion

The experimentation on online data or text might identify the closeness between features of the data sets or networks in general. It is useful to determine the degree of difference in results for a same theme for different networks. This can be useful to explain the general behavior of a certain network with respect to an idea. In other words, this might account for the way a certain network behaves with respect to a theme or context. We might also be able to predict how a network is going to behave with respect to a certain theme, based on its behavior to similar themes. In order to extend this we can consider other information networks like phishing corpus, E-book repositories, personal, and news and web blogs. There are many networks available online which can be used for such research. After such experimentation it might be possible to say that a particular network has similar results with another in a given context. This can lead to saying that the related communities in such networks might behave similarly under similar given situations. One interpretation is that for a new incoming node the two networks might result in a similar number of links from a particular community of nodes. It might be possible to group different networks based on their similarity in behavior towards one particular context. Further work might include other information networks. One type of networks which offers scope for such research is networks with weighted edges (Kunegis, Lommatzsch, and Bauckhage 2009). It might be possible to extend the idea to networks with weights assigned to edges; it might be very interesting to see if closure is applicable to a network containing mixture of signed links It can also be extended to compare several networks at once when considering a common theme, or it is possible to compare other social networks. It will be a more challenging work for signed networks. It will aid to clarify the understanding of web communities and information networks. This can find useful

applications in related fields such as Sociology, Linguistics, Mathematics and others.

References

- Bogdanov, P.; Larusso, N.; and Singh, A. 2010. Towards Community Discovery in Signed Collaborative Interaction Networks. In *2010 IEEE International Conference on Data Mining Workshops*, 288–295. IEEE.
- Granovetter, M. 1985. Economic Action and Social Structure: The Problem of Embeddedness. *American journal of sociology* 91(3):481–510.
- Kunegis, J.; Lommatzsch, A.; and Bauckhage, C. 2009. The Slashdot Zoo: Mining a social network with negative edges. In *Proceedings of the 18th international conference on World wide web*, 741–750. ACM.
- Leskovec, J.; Huttenlocher, D.; and Kleinberg, J. 2010. Signed networks in social media. In *Proceedings of the 28th international conference on Human factors in computing systems*, 1361–1370. ACM.
- Romero, D., and Kleinberg, J. 2010. The Directed Closure Process in Hybrid Social-Information Networks, with an Analysis of Link Formation on Twitter. *Arxiv preprint arXiv:1003.2469*.

Identifying Interesting Postings on Social Media Sites

Swathi Seethakkagari

School for Electronics & Computer Systems
University of Cincinnati
Cincinnati, OH 45221-0030
seethasi@mail.uc.edu

Anca Ralescu

Machine Learning & Computational Intelligence Lab
School of Computing Sciences & Informatics
University of Cincinnati
Cincinnati, Oh 45221-0030
anca.ralescu@uc.edu

Abstract

This paper considers the classification of messages posted on social networking sites as a step towards identifying interesting/non-interesting messages. As a first approximation, a message is represented by two attributes – the *message length* (number of words), *posting frequency* (time difference between consecutive messages) for the same sender. A classifier, trained according to a user’s perception of whether a message is interesting or not, is used to label each message. *Facebook* is considered for illustration purposes.

Keywords

Social networks, classification, k-nearest neighbors.

Introduction

Social networking has long been an activity within social communities. Whether through relatives, friends, or acquaintances people are routinely using their social connections to further their careers, and improve and enjoy their lives. With the advent of online social networks and sites this type of activity has increased, making possible networking on a large scale between people at great physical distances. Friendship and contacts can now be maintained over longer period of time, idea can be exchanged between massive groups of people. Social networks have become an excellent communication source.

Analysis of the networking sites has led to many interesting research issues, in a field that is rapidly growing of social computing and cultural modeling. A natural, and often used way to represent a network is through graphs, in which a vertex corresponds to an entity in the network, usually an individual, and an edge connecting two vertices represents some form of relationship between the corresponding individuals (Al Hasan et al. 2006). *“Social network analysis provides a significant perspective on a range of social computing applications. The structure of networks arising in such applications offers insights into patterns of interac-*

tions, and reveals global phenomena at scales that may be hard to identify when looking at a finer-grained resolution” (Leskovec, Huttenlocher, and Kleinberg 2010).

Predicting the network evolution in time is central to such studies. In particular, detecting communities in a network, predicting the links between nodes in the network, have become much studied subjects in social computing, and other domains based on network representations. *“Prediction can be used to recommend new relationships such as friends in a social network or to uncover previously unknown links such as regulatory interactions among genes”* (Tan, Chen, and Esfahanian 2008).

By contrast with studies to reveal “global phenomena” one can consider the local, self-centric social network to which an individual belongs. The ability of anytime anywhere communication that online social networks provides to users has lead to an explosion of user generated data. Therefore, extracting patterns, global or local, from social networks, is necessary if we are to make sense of what a social network conveys about its users, and society at large. In this paper we consider the setting of a social network (such as Facebook, for example) where each user is free to post various messages (in Facebook this is done via the user *status* which the user updates. Some users are inclined to post frequent and often relatively uninteresting updates, others post them more rarely and their contents are more interesting. Then again, the evaluation “interesting/uninteresting” is subjective and varies from user to user. Therefore, to support a user whose community (friends) is large, filtering or classification of postings which can take into account this user’s preference is necessary.

In the remainder of this paper we explore the idea of classifying postings/updates in a user’s centered community based on the user’s perception of their contents as interesting or not.

Analysis of messages on the social network

In the setting of a Facebook-like network, let I denote a generic individual, and $\mathcal{F}(I)$ the collection of friends (direct or indirect) of I . A snapshot of such Facebook community is illustrated in Figure 1.

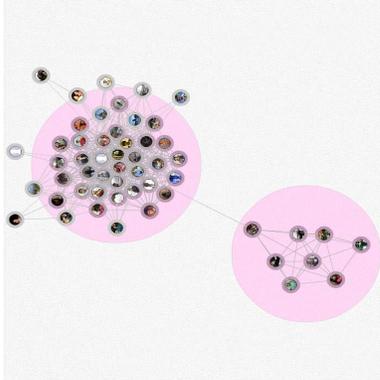


Figure 1: Snapshot from a Facebook friends network showing a group of clusters.

For each friend $f \in \mathcal{F}(I)$, $u_I(f)$ is an update of posted by f . The extent to which an update is interesting is, of course, a matter of its contents. This means that in principle, a text analysis of its contents should be done. However, while this would not doubt provide a deeper understanding of the actual content, other characteristics, such as message length, and frequency of messages from a particular f might give a close enough idea of how interesting a message is. For the purpose of this paper then, $u_I(f; n, t)$ denotes the update of length n (words), posted by the friend f at time interval t . The attributes, n and t are used to classify the update $u_I(f; n, t)$ as interesting or not.

k -Nearest Neighbors Classification of Postings

The well known k -nearest neighbor classifier (Hart 1967) is used to classify a newly posted message. The classification rule used by the k -nearest algorithm is very simple: using a set of labeled examples, a new example is classified according to its k -nearest neighbors, where k is a parameter of the algorithm. The k nearest neighbors "vote" each for the class with it has been labeled. Variations of the algorithm make possible to weigh a vote by the actual distance from each of these neighbors to the new example (Al Hasan et al. 2006): the vote of a closer neighbor counts more than that of a neighbor farther away. In the small experiment described below the simpler version of the algorithm is used. Algorithm 1 describes the steps for this classification.

Algorithm 1 Pseudo code for labeling a message using the k -NN Classifier

Require: 2-class training data set of size n

Require: Test data point and k (odd values to avoid ties)

Require: classification technique: k Nearest Neighbors Classifier

Ensure: The test data point is labeled with its class based on classifier output. Labels are set to -1 or +1.

Compute the classifier based on the distance of a test datum to the k nearest neighbors.

for $i = 1, \dots, n$ **do**

 Calculate the distance, $dist(i)$ with the i th data point in the training set

 Sort the distances

 Extract the k nearest neighbors

if the sum of the top k labels is positive **then**

 label of test data point is set to 1;

else

 label of test data point is set to -1;

end if

end for

A small real example

Table 1 shows a small set of updates posted on one of authors (S. Seethakkagari) Facebook page. The labels, \pm , are assigned according to her subjective evaluation of each posting.

Table 1: A small set of postings extracted from S. Seethakkagari's wall on facebook. N is the message length, T , the time from the last message of the same sender. The Label is assigned according her subjective evaluation of the posting content.

ID	1	2	3	4	5	6	7	8	9
N	23	16	26	20	30	22	32	16	12
T	272	81	149	287	10	26	4	1	36
Label	1	-1	1	1	1	1	-1	1	1
ID	10	11	12	13	14	15	16	17	18
N	6	4	7	1	32	4	17	15	3
T	0	64	39	558	199	72	52	32	216
Label	-1	-1	-1	1	-1	1	1	1	1
ID	19	20	21	22	23	24	25		
N	61	2	38	13	2	6	23		
T	27	594	63	0	80	39	57		
Label	-1	1	1	-1	-1	1	1		

Experimental Results

The data of labeled postings, shown in Table 1 are plotted in the $length \times time$ space as shown in Figure 2.

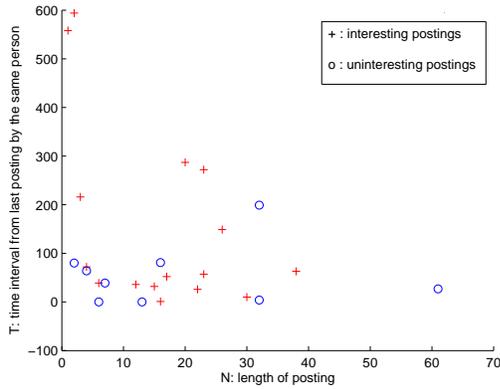


Figure 2: Plot of the 25 postings from S. Seethakkagari’s site.

The following experiment was carried out: all the possible test data sets of four postings were generated for a total of 12650 sets. The corresponding training sets were obtained by eliminating the test data sets from the set of postings. The number of neighbors was selected to be $k = 3$. Table 2 and Figure 3 show the classification results of all test postings.

Table 2: Results of classification for all postings of four messages with respect to 21 postings used as training data.

accuracy(%)	0	25	50	75	100
frequency	252	1865	4544	4455	1534
average	60.1858				
mode	50				
median	50				

Conclusion and future work

We explored the use of classification of postings on a social media site into two classes: interesting versus non-interesting. Each message was encoded using two attributes, length, expressed as the N the number of words in the message and the frequency with which its sender posts messages. Experiments were run for the set shown in Table 1, a small, but *real* data set, using a k -nearest neighbor classifier, with $k = 3$. We consider the results encouraging,

as the probability of classification accuracy greater than or equal to 50% is over 83%. As a future study, a larger attribute set may be used. For example, the comments (their length and/or contents) received for previous message, the ID of a message sender, can be considered. However, the tradeoff between classification accuracy and computational efficiency. For example, as the number of attributes, or the number of neighbors k increase, the complexity in calculation increases. Feedback from the user may be used to adapt

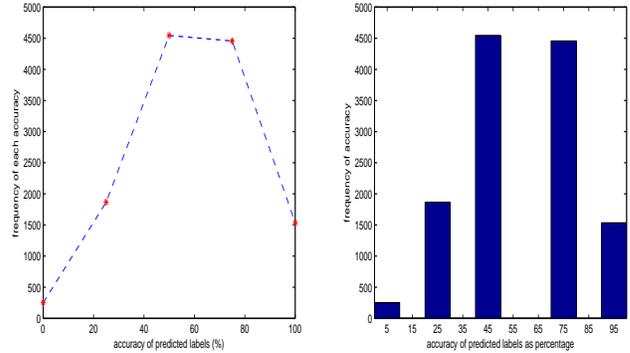


Figure 3: Accuracy of prediction when 21 messages are used to predict labels of a subset of four messages.

the classifier so as to achieve a better tradeoff between speed and accuracy.

References

- Al Hasan, M.; Chaoji, V.; Salem, S.; and Zaki, M. 2006. Link prediction using supervised learning. In *SDM06: Workshop on Link Analysis, Counter-terrorism and Security*.
- Hart, P. 1967. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory* 13(1):21–27.
- Leskovec, J.; Huttenlocher, D.; and Kleinberg, J. 2010. Signed networks in social media. In *Proceedings of the 28th international conference on Human factors in computing systems*, 1361–1370. ACM.
- Tan, P.; Chen, F.; and Esfahanian, A. 2008. A Matrix Alignment Approach for Link Prediction. In *Proceedings of ICPR 2008*.

Scientific Computing and Applications

Chair: Mihaela Malita

Evolutionary Computation on the Connex Architecture

István Lőrentz

Electronics and Computers Department
Transylvania University
Braşov, Romania
istvan@splash.ro

Mihaela Maliţa

Computer Science Department
Saint Anselm College Manchester
Manchester, NH, USA
mmalita@anselm.edu

Răzvan Andonie

Computer Science Department
Central Washington University
Ellensburg, WA, USA
and
Electronics and Computers Department
Transylvania University
Braşov, Romania
andonie@cwu.edu

Abstract

We discuss massively parallel implementation issues of the following heuristic optimization methods: Evolution Strategy, Genetic Algorithms, Harmony Search, and Simulated Annealing. For the first time, we implement these algorithms on the Connex architecture, a recently designed array of 1024 processing elements. We use the Vector-C programming environment, an extension of the C language adapted for Connex.

Introduction

Evolutionary Algorithms (EA) are a collection of optimization methods inspired from natural evolution (Bäck 1996), (Back, Fogel, & Michalewicz 1997), (Back, Fogel, & Michalewicz 1999). The problem is formulated as finding the minimum value of an evaluation function over a set of parameters defined on a search space. Well known evolutionary techniques are: Evolution Strategy (ES), Genetic Algorithms (GA), and Evolutionary Programming (EP). These techniques are also related to stochastic search (e.g., Simulated Annealing (SA)), and they share the following characteristics:

- Start with a random initial population.
- At each step, a set of new candidate solutions is generated based on the current population.
- Based on some criteria, the best candidates are selected to form a new generation.
- The algorithm is repeated until the solution is found, or a maximum number of iterations reached.

EAs are *meta-heuristic*, as they don't make many assumptions of the function being optimized (for example, they do not require known derivatives). From a meta-heuristic point of view, the function to be optimized is a 'black-box', only controlled by the input parameters and the output value. Meanwhile, EAs are parallel by their nature. Parallel implementations of optimization algorithms is generally a complex problem and this becomes more challenging on fine grained architectures with inter-processor communication burdens.

Our study focuses on implementation issues of EAs on a recent massively parallel architecture - the Connex Architecture (CA). The CA is a parallel programmable VLSI chip

consisting of an array of processors. Functionally, it is an array/vector processor. It is not a dedicated, custom-designed (ASIC) chip, but a general purpose architecture. The CA is now developed by Vebris¹. An older version was developed in silicon by BrightScale, a Silicon Valley start-up company in (see (Ştefan 2009)).

Several computational intensive applications have been already developed on the CA: data compression (Thiebaut & Ştefan), DNA sequences alignment (Thiebaut & Ştefan 2001), DNA search (Thiebaut, Ştefan, & Maliţa 2006), computation of polynomials (Thiebaut & Maliţa), frame rate conversion for HDTV (Ştefan 2006), real-time packet filtering for detection of illegal activities (Thiebaut & Maliţa 2006), neural computation (Andonie & Maliţa 2007), and Fast Fourier Transform (Lőrentz, Maliţa, & Andonie 2010).

We do not intend to compare the efficiency of different EAs on the CA, but to provide the implementation building blocks. The motivation and novelty of this work are to expose the CA's vector processing capability for meta-heuristic optimization algorithms. We will provide the resulted performance results (instructions/operators) for several optimization benchmarks. The code is written in C++, using Vector-C, available at (Maliţa 2007), a library of functions which simulate CA operations. We use simulation because the floating-point version of the chip is still under development.

Review of Evolutionary Algorithms

We will first summarize the following standard optimization algorithms: Evolution Strategy, Genetic Algorithms, and Harmony Search, and Simulated Annealing. We will describe them in a unified way, in accordance to the EA general scheme from the Introduction.

Genetic Algorithms

In the original introduction of the 'Genetic Algorithm' concept, described by (Holland 1975), the population of 'chromosomes' is encoded as binary strings. Inspired from biological evolution, every offspring is produced by selecting two parents (based on their fitness), the genetic operators are the cross-over and single-bit mutation. The theoretical

¹<http://www.vebris.com/>

foundation of GA is the Schema Theorem. Since the original formulation, GA evolved into many variants. We will consider here only the standard procedure:

Algorithm 1 Genetic Algorithm

Initialize population, as M vectors over the $\{0, 1\}$ alphabet, of length N .

repeat

Create M child vectors, based on:

1. Select 2 parents, proportionate to their fitness
2. Cross-over the parents, on random positions
3. Mutate (flip) bits, randomly

The created M child vectors will form the new population, the old population is discarded.

until termination criterion fulfilled (solution found or maximum number of iterations reached).

Evolution Strategy

Evolution Strategy is also a population based optimization method, with canonical form written as $(\mu/\rho + \lambda)$ -ES. Here μ denotes the number of parents, ρ the mixing number (number of parents selected for reproduction of an offspring), λ the number of offspring created in each iteration (Beyer & Schwefel 2002).

Algorithm 2 Evolution Strategy algorithm (μ, λ) -ES

Initialize population $\mathbf{V}_\mu = \{\mathbf{v}_1, \dots, \mathbf{v}_\mu\}$. Each individual \mathbf{v} of the parent population represents a vector of N numbers encoding the decision variables (the search space) of the problem. The population is initialized randomly.

repeat

Generate λ offspring $\tilde{\mathbf{v}}$ forming the offspring population $\{\tilde{\mathbf{v}}_1, \dots, \tilde{\mathbf{v}}_\lambda\}$ where each offspring $\tilde{\mathbf{v}}$ is generated by:

1. Select (randomly) ρ parents from \mathbf{V}_μ .
2. Recombine the selected parents \mathbf{a} to form a recombinant individual $\tilde{\mathbf{v}}$.
3. Mutate the parameter set \mathbf{s} of the recombinant.

Select new parent population (using deterministic truncation selection) from either

- the offspring population $\tilde{\mathbf{V}}_\lambda$ (this is referred to as comma-selection, usually denoted as (μ, λ) -selection), or
- the offspring $\tilde{\mathbf{V}}_\lambda$ and parent \mathbf{V}_μ population (this is referred to as plus-selection, usually denoted as $(\mu + \lambda)$ -selection)

until termination criterion fulfilled (solution found or maximum number of iterations reached).

The specific mutation and recombination operations will be presented later in this paper.

Harmony Search

Harmony Search (HS) is a meta-heuristic algorithm inspired by musical composition (Geem, Kim, & Loganathan 2001).

According to (Weyland 2010), HS is a particular case of the $(\mu + 1)$ ES algorithm. In HS, the population, encoded as vectors of real or integer numbers, is stored in a matrix. The population size (number of rows) is fixed. Each new candidate is created by a discrete recombination (identical to the recombination of ES), or as a random individual. Mutation is performed with given probability. A key parameter is the the mutation 'strength' (or bandwidth). The new individual will replace the worst individual in the actual population if it is 'better' than this one.

Simulated Annealing

Inspired from the physical process of annealing, SA allows unfavorable decisions, when a controlling parameter called 'temperature' is high.

Over the iterations, the temperature is decreased and the algorithm will asymptotically approach a stochastic hill climbing. SA (Kirkpatrick *et al.* 1983) can be implemented over a population of (1 parent + 1 descendant), using the uniform mutation presented later in this article.

Algorithm 3 Simulated Annealing

Initialize a random candidate solution V

Set initial temperature, $T = T_0$

repeat

mutate (perturb) the existing solution, to create V'

compute $\Delta = f(V') - f(V)$

if $\Delta < 0$ or $U(0, 1) < \exp(-\Delta/T)$ **then**

accept new candidate: $V = V'$

end if

Reduce T

until termination criterion fulfilled (Acceptable solution found or maximum iterations reached)

return $V, f(V)$

$U(0, 1)$ denotes an uniform random variable between $[0, 1]$.

The Connex-BA1024 chip

We implement the previous optimization algorithms on the CA, a massively parallel architecture known as the Connex BA1024 chip. In this section we briefly introduce some of the hardware characteristics of BA1024. As a first CA implementation example, we will describe a random number generator program. This generator will be used in our subsequent applications.

The CA is a Single Instruction Multiple Data (SIMD) device with 1024 parallel processing elements (PEs), as well as a sequential unit, which allows general purpose computations. It contains standard RAM circuitry at the higher level of the hierarchy, and a specialized memory circuit at the lower level, the Connex Memory, that allows parallel search at the memory-cell level and shift operations.

Several CA chips can be integrated on the same board, extending the length of processed vectors in increments of 1024, while receiving instructions and data from only one controller. A controller oversees the exchange of data between the two levels. Just as regular memory circuits, the

operations supported by the CA can be performed in well-defined cycles whose duration is controlled by the current memory technology, which in today's technology is in the 1.5 ns range.

The 1024 cells are individually addressable as in a regular RAM, but can also receive broadcast/instructions or data on which they operate in parallel at a peak rate of 1 operation per cycle. This general concept fits the Processor-In-Memory paradigm. The cells are connected by a linear chain network, allowing fast shifting of data between the cells, as well as the insertion or deletion of data from cells while maintaining the relative order of all the data. All these operations are performed in a single memory cycle.

The hardware performances of BA1024 are:

- Memory cycle: 1.5 ns.
- Computation: 400 GOPS at 400 MHz (peak performance)
- External bandwidth: 6.4 GB/sec (peak performance)
- Internal bandwidth: 800 GB/sec (peak performance)
- Power: ≈ 5 Watt
- Area: ≈ 50 mm² (1024-EU array, including 1Mbyte of memory and the two controllers).
- 65nm implementation

Using a 16-bit arithmetic, the BA1024 computes the scalar product of a 1024-tuple vector in 37.5 ns (26 million scalar products/sec), and performs 1024×1024 matrix multiplications in 40 ms (25 operations/sec). Adding up to 1024 numbers is done in 5 cycles. Multiplication is done in 10 cycles. The $P = 1024$ processing elements, each containing 512 registers, are interconnected in a ring. From an algorithmic point of view, the chip can be considered as an array of $P = 1024$ columns and $M = 512$ rows. By convention, we represent it as an array of horizontal vectors. In C-style row-major notation, $A[i][j]$ denotes the i 'th register inside the j -th processing element.

An important component of evolutionary algorithms is the pseudo-random number generator. An ideal random number generator should be (Quinn 2003): uniformly distributed, uncorrelated, cycle-free, satisfy statistical randomness tests, and reproducible (for debugging purposes). In addition, parallel generators must provide multiple independent streams of random numbers. We used the xorshift generator, introduced by (Marsaglia 2003), with period $2^{128} - 1$. The random seed needs 4 integer vectors $X[0], X[1], X[2], X[3]$ of 1024 elements each. Here is the C++ code of this pseudo-random generator, using the Vector-C library:

```
vector<uint> xor128(vector<uint> X[]) {
    vector<uint> T;
    T = x[0] ^ (X[0] << 11);
    T ^= (T ^ (T >> 8));
    T ^= X[3] ^ (X[3] >> 19);
    X[0] = X[1];
    X[1] = X[2];
    X[2] = X[3];
    X[3] = T;
    return T;
}
```

Vectors are in represented in uppercase and initialized with seed values from the host computer (in Linux, `/dev/urandom`). It is essential that each component of the seed vector has a different, independent value. Once initialized, the presented function generates 1024 independent pseudo-random streams.

On the CA, generating in parallel $N \leq 1024$ uniformly distributed random numbers results in a linear speedup: $S_{xor128} = T_{sequential}/T_{parallel} = N$, where $T_{sequential}$ is sequential execution time and $T_{parallel}$ is parallel execution time.

The `randvN(σ)` function returns a vector. Each component of this vector is an independent random variable with Gaussian distribution, 0 mean and σ standard deviation. The CA lacks trigonometric and logarithmic functions, used by the Box-Muller method for generating normal distributed random numbers. Therefore, we used an approximation method, based on the central limit theorem: $N(0, \sigma) \approx \sigma \left(\sum_{k=1}^{12} U(0, 1) - 6 \right)$, where $U(0, 1)$ is the uniform random number generator in the $[0, 1]$ interval.

Evolutionary operators on the CA

We present the building blocks of an evolutionary algorithm using the CA vector instructions. The control flow of the algorithm is still sequential, but mutation and evaluation operators are vectorized. The population is represented as a matrix. Rows (individuals) are mapped as CA vectors and use vectorial instructions for mutation, recombination, and evaluation. A population is evaluated sequentially. The vector length (max. number of decision variables of the search space) is limited to 1024, while the population size is limited by the number of CA rows. Horizontal mapping allows efficient computation of fitness functions via the parallel CA reduction operator.

Recombination

The recombination operator forms a new individual, based on a set of parents in the existing population. Typically the offspring will get a combination of the parents features. There are many variants for the recombination, we will present the commonly used ones in GA and ES: crossover and discrete recombination.

Crossover The crossover operation creates a new individual by combining the features of two parents. In one-point crossover, elements from the first parent vector are copied up to a random position. Continuing from that position, elements from the second parent vector are further copied. We implement this using a vector selection mask of random length (Fig. 1).

```
vector crossover(vector X, vector Y){
    int position = rand(VECTOR_SIZE);
    where( i < position )
        C = X; elsewhere C = Y;
    return C;
}
```

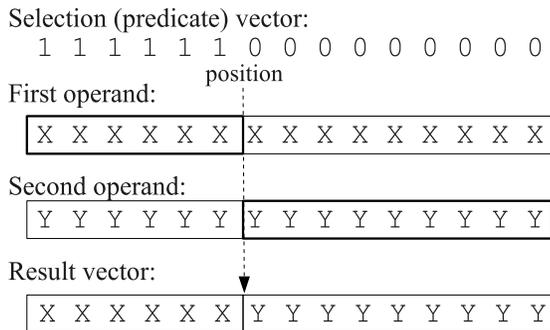


Figure 1: Parallel one-point crossover using predicate vector.

The `rand(n)` scalar function returns a random integer in the range $[0, n-1]$. The statement **where**(condition) ... **elsewhere** ... is a parallel-if construct available on CA. Index i denotes the processor element. The expression is evaluated in parallel on each PE_i , and a selection flag (predicate) is set, which conditions the execution of the statements inside the **where** block. The **elsewhere** block is executed after the selection predicates are negated. For brevity, we omit the vector element data type, which can be either integer or float.

To obtain a two-point crossover, we need to change the condition inside **where** to use 2 parameters, denoting the start and end splicing points:

```
where ( i >= a && i < b ) C = X;
elsewhere C = Y;
```

The above code can be generalized for uniform crossover (Sywerda 1989). In this case, for each position, a bit is randomly selected from one of the parents. Uniform crossover can be implemented by changing the condition to

```
where ( randvb(0.5) ) { ... }
```

where `randvb(p)` creates a Boolean vector, each bit having value '1' with probability p .

Discrete Recombination In ES, the recombination operator uses information from ρ individuals. In discrete recombination, each position of the candidate individual vector \mathbf{v}' is copied from the same position of a randomly chosen parent: $\mathbf{v}'(i) = \mathbf{v}_k(i)$. In this case, the HS algorithm uses a recombination of the entire population.

CA supports matrix-vector addressing (selecting a different cell from each column, to form a new vector), which is used for discrete-recombination.

For $N \leq 1024$, the parallel speedup of the two recombination operators is linear: $S_{crossover} = N$.

Mutation

Mutation involves changing a single, random position by a given amount. In horizontal mapping, first we create a selection mask, with a single '1' bit, on the k -th position, then perform a vector + scalar operation, which will add only the elements on the k -th position:

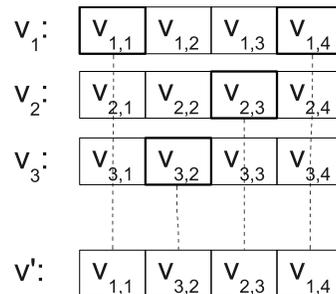


Figure 2: Discrete recombination. A new vector \mathbf{V}' is created from 3 parents.

```
vector mutate(vector X){
  int pos = rand(VECTOR.SIZE);
  float amount = rand11(); // [-1...1]
  where ( i == pos )
    X += amount;
  return X;
}
```

In ES, the mutation operator alters the vector by a random amount: $v' = v + N(0, \sigma^2)$, where $N(0, \sigma^2)$ denotes a random variable with normal distribution. Our Vector-C function name is `randvN(sigma)`. The σ^2 variance parameter controls the mutation strength:

```
vector mutateES(vector X){
  return X + randvN(sigma);
}
```

Since the single-bit mutation's serial execution time is constant, there is no speedup achieved by parallelization: $S_{mutate1bit} = 1$. On the other hand, the speedup for ES-mutation is linear: $S_{mutateES} = N$, since each vector element is affected.

Fitness Function Evaluation

In evolutionary techniques, evaluating the fitness functions usually consumes most of the time (compared to the mutation, selection), so it is crucial to implement it most efficiently. The class of functions that can be efficiently computed using vectorial instructions on the CA has the form:

$$f(x_1, x_2, \dots, x_N) = \bigoplus_{i=1}^N h_i(x_{i-k}, \dots, x_i, \dots, x_{i+k}) \quad (1)$$

where \bigoplus is the parallel-reduction operator, k defines a fixed-size neighborhood (independently of N). Currently, the CA supports parallel sum reduction. The $h_i()$ function should depend only on the i -th variable and optionally on a small local neighborhood, $i - k, \dots, i + k$. This is due to the constrain that processing elements (PEs) are interconnected by a ring bus, so efficient communication is done only by neighboring PEs (data-locality).

In (Malița & Ștefan 2009), it is described how to compose such a function on the CA, by combining data-parallel and time-parallel computations, illustrated in Fig. 3. Such

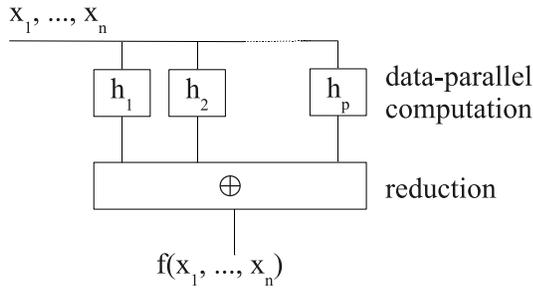


Figure 3: Parallel computation followed by reduction.

generic functions can be evaluated on P processors as follows (t_i are partial results):

```

for every  $i = 1 \dots P$  do in parallel
     $t_i = h_i(\dots)$ 
end for
result = reduce( $t_1, \dots, t_p$ )

```

For a one-to-one mapping of $h()$ invocations to processing elements, $f()$ is computed in $T_f = T_h + T_{red}$ time, where T_h is serial time to compute $h()$ and T_{red} is reduction time (which is a CA machine instruction).

Assuming data-parallel computation, for sequentially processing N items the speedup is $S = \frac{N(T_h + T_+)}{T_h + T_{red}}$, where T_+ is serial execution time of the associative operator used for reduction.

Due to the constant parallel evaluation time (up to the maximum vector size 1024), we use functions that can be expressed this way.

Selection

Given the 'horizontal' mapping of the population in the CA, after evaluation, the fitness value (a scalar) is available to the sequential unit. The selection decision operation is not vectorized, it is done by the sequential unit by comparing or sorting the scalar fitness values.

Selection in Simulated Annealing To implement SA on the CA, we use the `mutate()` and `evaluate()` functions already presented. The SA-specific selection operation (to choose between two solutions V_{old} , V_{new}) is:

```

vector selectSA(vector Vold, vector Vnew,
                float t)
{
    df = evaluate(Vnew) - evaluate(Vold);
    if ( df < 0 || randf() < exp(-df/t) )
        return Vnew;
    else
        return Vold;
}

```

The $\exp(-df/t)$ scalar function (Boltzmann factor) is evaluated by the CA's sequential unit. Function `randf()` returns an uniform random variable in the $[0,1)$ interval.

Experimental Results

In our experiments, we use two benchmark problems: the generalized Rosenbrock function and the geometric distance problem.

The generalized Rosenbrock function

This is a standard benchmark function used in optimization, illustrated in Fig. 4. The generalized N -dimensional form is (De Jong 1975):

$$f(\mathbf{x}) = \sum_{i=1}^{N-1} [(1 - x_i)^2 + 100(x_{i+1} - x_i^2)^2] \quad \forall x \in \mathbb{R}^N \quad (2)$$

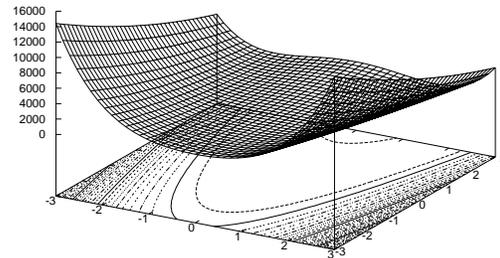


Figure 4: The Rosenbrock function of two variables.

The evaluation of the multi-dimensional Rosenbrock function can be performed using vector arithmetic, shifting and sum-reduction. The following code snippet shows the implementation:

```

float Rosenbrock(vector X) {
    vector A, X2, Xsh;
    Xsh = 0.0;
    where ( i < N )
        Xsh = rotateLeft(X, 1); // x1, x2, ...
    where ( i < (N-1) ) {
        X2 = X * X;           // x0^2, x1^2, ...
        Xsh -= X2;           // (x1-x0^2), ...
        Xsh *= Xsh * 100;
        X2 = 1 - X;          // 1-x0, 1-x1, ...
        X2 = X2 * X2;        // (1-x0)^2, ...
        X2 += Xsh;
    }
    return sumv(X2);
}

```

Geometric distance problem

The geometric distance problem arises in molecular geometry: given a set of distances between pairs of atoms space, determine each atom's (x,y,z) coordinate. Although various solutions exist, the problem can be tackled also as a global optimization problem (Grosso, Locatelli, & Schoen 2009).

We implement a simplified form of this problem, where each coordinate is assumed to take only *discrete* values inside a given bounding rectangle. The aim is to minimize

$$f(x_1, \dots, x_N) = \sum_{i \neq j} (||x_i - x_j|| - d_{ij})^2; \quad (3)$$

for all (i, j) pairs for which d_{ij} is known, where $x_i \in D \subset \mathbb{Z}^3$.

To parallelize the evaluation function, we notice that the list of distances must be distributed for each processing element, since the CA does not support random-access inter-processor communication. The pairs of points for which the distances are known (as input data) represent the edges of an undirected graph. We label the edges as $e_1 \dots e_N$ and the vertices as x_1, \dots, x_V . Each edge is mapped onto its own processor: $e_p \Leftrightarrow PE_p$.

To compute $f()$, we need for each pair the x_i, x_j, d_{ij} variables. The i, j vertex indexes for processor p are noted by i_p and j_p , ($p = 1 \dots N$).

Note that some of the vertices must be shared between processors. To implement this sharing, we use the following method: Each PE p will hold the distance d_p and the vertices of the two nodes it connects x_{i_p}, x_{j_p} . For example, in a simple triangle case with three vertices, we have three edges with labels e_0 : A - B, e_1 : B - C, e_2 : A - C (Fig. 5). To avoid inter-processor communication during the iterations, since each PE stores vertex data into private variables, we must assure that the variables which represents the same vertex on a different processor have identical values. We do this in the following way:

1. The vertices are initialized to random values, at the program initialization.
2. The vertices are distributed to each processor, each processor stores a private copy.
3. Each vertex x_i will have also associated a random number generator stream r_i .

This data representation allows parallel evaluation of the sum of the distances and parallel mutation of the vertex coordinates. We present the flowchart of the computation in Fig. 6.

For example, to load the graph represented in Fig. 5, we assign to each edge the corresponding *PE*. PE_0 will receive the data corresponding to edge 0: the coordinates of points A,B and the distance $d(A,B)$.

To evaluate the distances, no inter-processor communication is required. Each PE computes the distance between the vertices it holds and subtracts from the known, input distance. The parallel reduction step computes the sum of squared differences, resulting a scalar fitness value.

```
void evaluateDist(vector Xi, Yi, D)
{
    vector Dx, Dy;
    Dx=Xi[k]-Xj[k];
    Dy=Yi[k]-Yj[k];
    Dx *= dx;   Dy *= dy;
    Dx += dy;
    return sumAbsDiff(Dx,D);
}
```

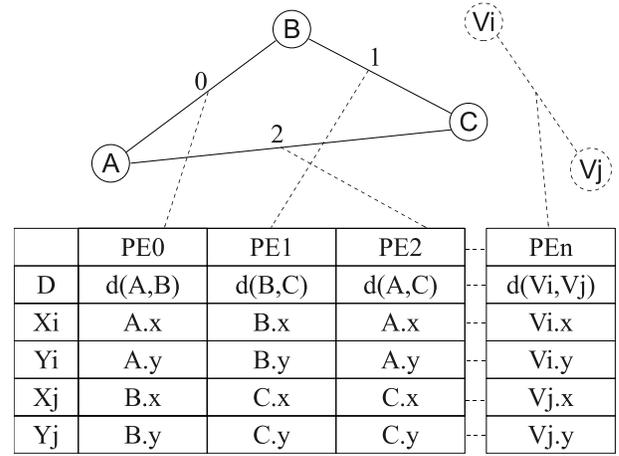


Figure 5: Example of a graph loaded into the Connex Array. The edge labels are the indexes, for which the distances are known. When new edges are added, the table extends horizontally, while the number of rows is kept constant. There are also two additional rows (Ri, Rj), not shown in the figure, which contain the seeds for the random generators

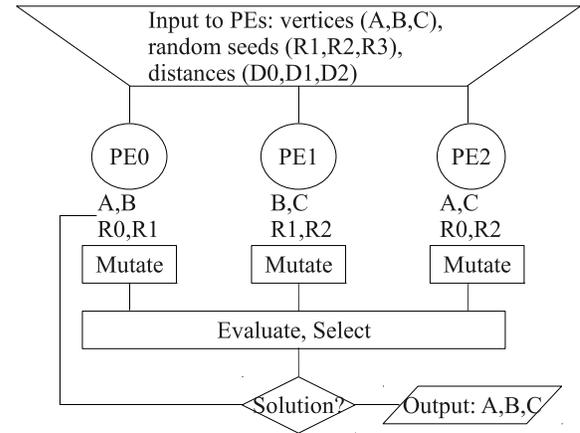


Figure 6: Flowchart of the parallel evolution of vertices. Note that apart from the evaluation (sum-reduction) there is no inter-communication between the processors

In the above listing, the input vectors are:

X_{i_p}, Y_{i_p} - first vertex belonging to edge p ,

X_{j_p}, Y_{j_p} - second vertex belonging to edge p ,

D_p = (known) length, squared, of edge p .

sumAbsDiff(Dx,D) sums the absolute differences of two vectors:

```
float sumAbsDiff(vector A, vector B) {
    vector V;
    V = A - B;
    where (V < 0)
        V = -V;
    return sumv(V);
}
```

Results

We measured the number of vectorial operations, for each specific evolutionary operator, as well as some test functions (see Table 1).

Operation	T_{Par}	T_{Seq}	S
A+=B	1	1024	N
xorshift 128	13	13312	N
sumAbsDiffs	7	4096	0.5 N
1-Point Crossover	3	2048	0.6 N
Uniform Crossover	15	14350	0.9 N
Uniform Mutation	33	21172	0.6 N
HS Mutation	107	71506	0.6 N
Rosenbrock	14	14325	N
evaluateDist	13	10240	0.7 N

Table 1: Vector instruction count by evolutionary operators

T_{par} is parallel execution time, measured in units of vectorial operations, T_{seq} is sequential execution time (number of sequential operations; we used the instruction count instead of physical time). The last column contains S , the speedup T_{seq}/T_{par} , running on $N \leq 1024$ processing elements. We use a one-to-one data element - PE mapping.

To accurately interpret these results, we have to emphasize that we used instruction counts instead of cycle counts simply because the floating-point version of the chip is still under development. The results give a theoretical achievable speedup when using the presented algorithms.

Conclusions

The meta-heuristic algorithms presented above are dependent on the way initial data is organized. We used horizontal mapping. Another choice is to map the population vertically, by loading the population data as columns in the CA. The vectorial instructions will operate in this case over the corresponding variables of the entire population. By this transposition, the previous parallel operations will become serial, and parallelism will operate over the entire population. However, in vertical mapping we cannot speed-up the evaluation function by using the parallel sum instruction. Since the evaluation function is the most time-critical, we did not explore further the vertical mapping method, to verify if there are benefits in other evolutionary blocks.

The CA offers vectorial computational facilities which are well suited for the implementation of evolutionary algorithms. We plan to continue our experimental work and test the efficiency of meta-heuristic optimization, including on the CA itself (not just on the simulator).

References

Andonie, R., and Malița, M. 2007. The Connex Array™ as a neural network accelerator. In *CI '07: Proceedings of the Third IASTED International Conference on Computational Intelligence*, 163–167. Anaheim, CA, USA: ACTA Press.

Back, T.; Fogel, D. B.; and Michalewicz, Z., eds. 1997. *Handbook of Evolutionary Computation*. Bristol, UK, UK: IOP Publishing Ltd., 1st edition.

Back, T.; Fogel, D. B.; and Michalewicz, Z., eds. 1999. *Basic Algorithms and Operators*. Bristol, UK, UK: IOP Publishing Ltd., 1st edition.

Bäck, T. 1996. *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*. Oxford, UK: Oxford University Press.

Beyer, H.-G., and Schwefel, H.-P. 2002. Evolution strategies A comprehensive introduction. *Natural Computing* 1:3–52.

Ștefan, G. 2009. One-Chip TeraArchitecture. In *Proceedings of the 8th Applications and Principles of Information Science Conference, Okinawa, Japan*.

De Jong, K. A. 1975. *An analysis of the behavior of a class of genetic adaptive systems*. Ph.D. Dissertation, Ann Arbor, MI, USA.

Geem, Z. W.; Kim, J. H.; and Loganathan, G. 2001. A New Heuristic Optimization Algorithm: Harmony Search. *SIMULATION* 76(2):60–68.

Grosso, A.; Locatelli, M.; and Schoen, F. 2009. Solving molecular distance geometry problems by global optimization algorithms. *Comput. Optim. Appl.* 43(1):23–37.

Holland, J. 1975. *Adaptation in natural and artificial systems*. University of Michigan Press.

Kirkpatrick, S.; Gelatt, C. D.; Jr.; and Vecchi, M. P. 1983. Optimization by Simulated Annealing. *Science* 220:671–680.

Lórentz, I.; Malița, M.; and Andonie, R. 2010. Fitting FFT onto an energy efficient massively parallel architecture. In *Proceedings of the Second International Forum on Next-Generation Multicore/Manycore Technologies, IFMT '10*, 8:1–8:11.

Malița, M., and Ștefan, G. 2009. Integral parallel architecture & Berkeley's Motifs. In *ASAP '09: Proceedings of the 2009 20th IEEE International Conference on Application-specific Systems, Architectures and Processors*, 191–194. IEEE Computer Society.

Malița, M. 2007. The Vector-C library on Connex (A software library for a Connex-like multiprocessing machine). http://www.anselm.edu/internet/compsci/Faculty_Staff/mmalita/HOMEPAGE/ResearchS07/WebsiteS07/.

Marsaglia, G. 2003. Xorshift RNGs. *Journal of Statistical Software* 8(14):1–6.

Ștefan, G. 2006. The CA1024: SoC with integral parallel architecture for HDTV processing. In *4th International System-on-Chip (SoC) Conference & Exhibit, November 1-2*.

Quinn, M. J. 2003. *Parallel Programming in C with MPI and OpenMP*. McGraw-Hill Education Group.

Sywerda, G. 1989. Uniform crossover in genetic algorithms. In *Proceedings of the third international conference on Genetic algorithms*, 2–9. Morgan Kaufmann Publishers Inc.

Thiebaut, M., and Ștefan, G. Ziv-Lempel compression with the Connex Engine. Tech. Rep. 077, Dept. Computer Sci-

ence, Smith College, Northampton, MA, 01063, January 2002.

Thiebaut, M., and Ştefan, G. 2001. Local alignment of DNA sequences with the Connex Engine. In *The First Workshop on Algorithms in BioInformatics WABI 2001*.

Thiebaut, D., and Maliţa, M. Fast polynomial computation on Connex Array. Technical Report 303, Smith College, November 2006.

Thiebaut, D., and Maliţa, M. 2006. Real-time packet filtering with the Connex Array. In *Proceedings of the Inter-*

national Conference on Complex Systems, 501–506.

Thiebaut, D.; Ştefan, G.; and Maliţa, M. 2006. DNA search and the Connex technology. In *Proceedings of the International Multi-Conference on Computing in the Global Information Technology (ICCGI'06)*.

Weyland, D. 2010. A rigorous analysis of the harmony search algorithm: How the research community can be misled by a "novel" methodology. *Int. J. of Applied Meta-heuristic Computing* 1(2):50–60.

Towards a Technique of Incorporating Domain Knowledge for Unit Conversion in Scientific Reasoning Systems

Joseph Phillips

De Paul University
School of Computing and Digital Media
243 S. Wabash Ave
Chicago, IL 60604, USA
jphillips@cdm.depaul.edu

Abstract

Unit conversion is often considered a straightforward task using analytical knowledge like the definition of centimeters in terms of meters. However, conversions like the computation of a photon's frequency from its wavelength implicitly use domain knowledge. We present an update on ongoing work on how a scientific reasoning system may intelligently convert between units using domain knowledge *and* tag data thus produced as dependent upon this domain knowledge. This is part of a project to intelligently use meta-data for scientific value manipulation (Phillips 2010).

1. Introduction

Computers should be made to understand scientists, not other way around! Unfortunately scientists are far from uniform in their notation, even within a single domain.

A good example of this non-uniformity is with unit usage. True, the metric system is widely used. However even here there are some applications where meters/kilograms/seconds (giving energy units Joules) are preferred, and others where centimeters/grams/seconds (giving energy units dynes) are common.

Beyond this we see some applications where different units are used, even for the same dimension, to keep the system on a common or intuitive scale and/or so that values naturally fall between ranges 0.1 to 1.0, or 1.0 to 10. Examples include the agricultural rainfall or irrigation unit hectare-mm (as opposed to liters), the electrical energy unit kilowatt-hours (as opposed to Joules), and the astronomical unit parsecs, from *par*allax *sec*onds (as opposed to meters).

A third class of units actually changes dimensions. Domain knowledge is needed implicitly to convert between the dimensions. This is often seen with light, where a photon's wavenumbers (in inverse centimeters), wavelength (in meters, millimeters, microns, or nanometers), or its energy (in electron-volts) all may be taken as stand-ins for its frequency.

Relying on domain knowledge is particularly tricky because it can change. For example, since the development of special relativity we believe that the speed

of light, the "c" in the equation $v=c/\lambda$ needed to convert from wavelength or wavenumbers to frequency, is constant for all observers in any reference frame. Prior to Einstein this would not have been a common belief.

Lastly, some units are actually for dimensionless values that have been normalized by being divided by some common standard. For example, the masses of planets in locations other than in orbit around our own Sun are commonly given in terms of how many "Jupiters" they are, rather than in kilograms. Also, the energy released in powerful events like large detonations is commonly given in kilotons of trinitrotoluene (TNT).

Herein we describe knowledge and algorithms to interconvert between pairs of units of the four cases given above. This work is a continuation and elaboration of on going work into writing a computer language which can reason about scientific meta-data intelligently (Phillips 2010).

The outline is as follows. We briefly discuss previous attempts to handle meta-data in the next section and follow with a brief description of the language we are building for scientific representation and reasoning. With this background we then present our approach and algorithms. We follow with a discussion of its limitations, and then conclude.

2. Prior Work

Several extensions exist to popular numeric and symbolic packages like *Mathematica* (Khanin 2001) and *Matlab* (deCarvalho 2006) enabling them to do dimensional analysis and unit conversion. Prior to that the Unix™ command *units* could do some multiplicative dimension and unit recognition, like converting 1000 cm³ to 1 liter (SunOS 1992) (Mariano 2004). Also, systems that do scientific discovery like the *Bacon* series of programs are able to invent new units to describe new phenomena (Langley *et al* 1987). To the best of our knowledge,

however, such systems do not tag the resulting calculations as dependent upon the correctness of the domain knowledge used to convert between dimensions (*e.g.* from wavenumbers to frequency), or are unable to do such dimension-changing conversions at all.

Fundamentally all these systems make the same mistake that Logical Positivists philosophers of science made when ignoring the extent to which data is theory-laden. They strive to carry forward only as much meta-data as needed to ensure the numeric or algebraic stability of the answer. They do not, however, even bother to capture, ask for, carry-forward, or exploit much domain knowledge¹.

3. A Brief Introduction to *StructProc*

StructProc is our language for a frame representation for scientific knowledge. Like other frame representations, *StructProc* knowledge is built around the <subject,attribute,value> triplet. *StructProc*, however, has a feature which may be less common. Numbers are not represented as being objects distinct from “symbols” (called “ideas” in *StructProc*) – rather numbers are represented as a special sort of idea. This allows numbers to be given properties (<attribute,value> pairs) like any other idea, for numbers to be queried on their properties like any other idea, *etc.*

The chief property to assign to numbers is their domain. A domain is an idea with corresponding dimensions, units, limits, *etc.* which is distinct from but obviously related to the attribute. For example, a building may be said to have a length, width and height. In general, all are distinct numbers corresponding to the attributes lengthA, widthA, and heightA (the postfix A is *StructProc*'s recommended way to designate attributes). Though they are three different numbers, all have the same domain (defaultMetersDomain) with the unit being meters, dimension being length, and limitation being that values less than 0 are illegal.

Besides being annotated by their domains, numbers (and any other idea) may have subjects, attributes and assumptions specified. The *StructProc* expression

```
45.4 {defaultMetersDomain,
    heightA,
    dePaulCenter,
    ^assumeS
    {measure1->angleOfElevationA =
      45.0 {defaultDegreesDomain},
    measure1->distanceA =
      41.0 {defaultMetersDomain},
    measure1->heightA =
      1.4 {defaultMetersDomain}
    }
```

¹ With the exception of perhaps boundary conditions for differential equations and simple notions of units and dimensions.

```
}
```

represents the number 45.4 which has been annotated as being the height of the building the DePaul Center in meters. Further, this value assumes the validity of three other measurements gathered under frame *measure1*: that 44.0 meters away from the building its top was sighted at an angle of 45.0 degrees above the horizon, and that the angle was measured from 1.4 meters above the ground.

4. Our Approach

The basic algorithm that underpins our approach is a straightforward conversion from one of unit to another of the same “fundamental” dimension as detailed in *sameSingleDimensionalConvert()*. Because both units share the same dimension, and because the compatibility of their subjects and attributes is checked elsewhere, no other checks need to be done or knowledge needs to be assumed.

The function *sameSingleDimensionalConvert()* returns linear conversion expression that converts from a value in *fromUnits* to one in *toUnits*. It does this by attempting to convert to the primary units of the “fundamental” dimension as an intermediate step. If it successfully finds a conversion then it saves it so it can be easily applied next time. (The expression *thisExpr.sub(expr)* takes the algebraic expressions *expr(x₁)* and *thisExpr(x₂)* and returns the new expression *thisExpr(expr(x₁))*). The expression *subject.get(attrA)* returns the first found value of subject's attribute *attrA*, or returns null if none are found. The expression *convertToA(units)* builds an attribute that represents a conversion to the specified units.)

This function allows the conversion from parsecs to kilometers by way of meters, and from Celsius to Kelvin with a slope of 1 and intercept of 273.15.

```
LinearExpression
    sameSingleDimensionalConvert
        (Units fromUnits, Units toUnits)
begin
LinearExpression expr, thisExpr;
Dimension dim;
Units    prime;

if (fromUnits == toUnits)
    Number slope    := 1;
    Number intercept := 0;
    return new LinearExpression(slope,intercept);
endif

expr := store.retrieve(fromUnits,toUnits)

if (expr != null)
    return(expr);
```

```

endif

dim := fromUnits.get(dimensionA);
prime := dim.get(primaryUnitsA);
expr := fromUnits.get(coverToA(prime))

if (expr == null)
    throw new InsufficientInformation();
endif

thisExpr := prime.get(coverToA(toUnits));

if (thisExpr == null)
    throw new InsufficientInformation();
endif

expr := thisExpr.sub(expr);
store.save(fromUnits,toUnits,expr);
return expr;
end;

```

Two functions are used to convert between multi-dimensional units. The function `incorporateUnits()` is given a list of dimension entries (`dimList`). Each entry tells a dimension and itself has a list of basic units of that dimension and the powers to which those units have been raised. The function adds to this list the new dimensions and basic units it finds for the `unit` parameter it has been given. The basic units of `unit` are incorporated into `dimList`, but their powers are multiplied by `sign`: either +1 or -1. This allows basic units that appear in both a `fromUnits` (line 2 with `sign -1`) and `toUnits` (line 3 with `sign +1`) to cancel each other at line 1.

```

DimList
    incorporateUnits
        (DimList dimList, Units unit, int sign)
begin
Units    basicU;
int      power;
DimEntry dimEntry;
UnitEntry unitEntry;
Dimension dimen;

forall basic unit pairs (basicU,power) in unit do
    dimen := basicU.get(dimension);
    dimEntry := dimList.find(dimen);

    if (dimEntry == null)
        dimEntry := new DimEntry(dimen);
        dimList.prepend(dimEntry);
    endif

unitEntry := dimEntry.unitList.find(basicU);
if (unitEntry == null)
    unitEntry := new UnitEntry(basicU);
    dimEntry.unitList.prepend(unitEntry);

```

```

endif;

unitEntry.addPower(power*sign); // Line 1
endfor

return(dimList);
end;

```

The function `sameMultiDimensionalConvert()` uses both `incorporateUnits()` and `sameSingleDimensionalConvert()` to return a conversion expression from a value in `fromUnits` to one in `toUnits`. It uses the former function to decompose (and hopefully partially cancel) both incoming units into their basic units, isolated by their dimensions (lines 2 and 3).

For each dimension it does the following. It finds the next occurrences of both a basic unit raised to a positive power (loop 4) and raised to a negative power (loop 5). They are different units of the same dimension, so they should be converted. If there are no more units to convert then it goes on to the next dimension (line 6).

When it finds units to convert it uses `sameSingleDimensionalConvert()`. Allowances are made when the units are raised to higher powers (lines 7, 8), those units are cancelled (lines 9, 10), and then it continues looking more units to convert. It also checks the expression returned from `sameSingleDimensionalConvert()` (not shown) to see if it has an added constant other than zero. It must do so because it builds an expression of multiplied (and divided terms): added terms would throw-off the computation. Also, they probably signify that the wrong units were used in the expression. For example, the Maxwell-Boltzmann distribution equation uses temperature in Kelvin (with an absolute 0) instead of in Celsius (with the additive term 273.15).

At the very end it returns an algebraic expression that uses the product of the slopes of all conversions.

```

Expression
    sameMultiDimensionalConvert
        (Units fromUnits, Units toUnits)
begin
DimList dimList := null;
double product := 1.0;
DimEntry dimEntry;
UnitEntry posEntry, negEntry;
int power;
Expression expr;

// Line 2
dimList:= incorporateUnits(dimList,fromUnits,-1);
// Line 3
dimList:= incorporateUnits(dimList, toUnits,+1);

forall dimEntry in dimList do
    posEntry := negEntry := dimEntry.unitList.head;

```

```

while (true)
  // Loop 4
  while (posEntry!=null AND posEntry.power<=0)
    posEntry := posEntry.next;
  endwhile

  // Loop 5
  while (negEntry!=null AND negEntry.power>=0)
    negEntry := negEntry.next;
  endwhile

  if (posEntry == negEntry == null) // Line 6
    break;
  endif

  expr := sameSingleDimensionalConvert
         (negEntry.unit,posEntry.unit);
  // Line 7
  power := min(posEntry.power,-negEntry.power);

  for (i := 0; i < power; i := i+1)
    product := product * expr.slope; //Line 8
  endfor

  posEntry.addPower(-power); // Line 9
  negEntry.addPower(+power); // Line 10
endwhile
endfor

expr := new LinearExpression(product,0);
store.save (fromUnits,toUnits,expr);
return(expr);
end;

```

This function allows the conversion from hectare-mm to liters by restating hectares as being hundred-meters squared, restating liters as decimeters cubed, and converting millimeters to decimeters (once) and hundred-meters to decimeters (twice). The second conversion from hundred-meters to decimeters benefits from the cached results generated by the first conversion.

Similarly, it can also convert kilowatt-hours to Joules. This particular conversion highlights the generality of the algorithm. Our system knows kilowatt-hours as:

$$\frac{(\text{kilograms}) \cdot (\text{meters}) \cdot (\text{kilometers}) \cdot (\text{hours})}{(\text{seconds}^3)}$$

The kilograms, meters, kilometers and inverse cubed seconds terms multiply to give kilowatts, and multiplying that by hours gives kilowatt-hours.

This is a natural way to define kilowatt-hours but it represents the time dimension in an atypical fashion as hours/seconds³. Hours and seconds do not need to be converted between fromUnits and toUnits but within fromUnits itself.

However, because our algorithm builds just one list

(dimEntry.unitList) of units to convert for each dimension, whether the units to convert are between fromUnits and toUnits or within either makes no difference.

Conversion between units of different dimensions necessarily uses domain knowledge. This knowledge includes the expression used to tie the two dimensions (and thus implicitly any knowledge on which the stating of that expression depends). This expression has some limited scope for which it holds, thus the search for a dimension converting expression starts with an ontological search from a most specific (among the smallest) ontological sets that encompass the subjects of both values and systematically considers increasingly broader sets.

At each set it considers all stated dimension conversions looking for one that converts from the from attribute to the to attribute. Unlike the algorithms for sameSingleDimensionalConvert() and sameMultiDimensionalConvert() which just used the units being converted, we now must use the attributes because we are trying to convert between different but specific aspects of the subject objects. Further, we assume the attributes imply specific dimensions.

Upon finding a candidate dimensional conversion we use sameMultiDimensionalConvert() to see if we can convert from the given fromUnits to the units expected by the expression, and from the units returned by the expression to the given toUnits. If we find such an expression and its required auxiliary conversions then we return that expression in which both auxiliary conversions have been substituted. We throw an exception otherwise. This algorithm is given as differentDimensionalConvert().

While differentDimensionalConvert() can use any expression for which it is clever enough to algebraically manipulate we anticipate many of its expressions will be simple linear or inverse linear 1/x functions. For example, conversion from either a photon's wavenumber in inverse centimeters (as is common in infrared spectroscopy) or its energy in electron volts (as is common in material science) to its frequency in Hertz (inverse seconds) would be built around a simple linear dimension conversion in which the from units would either be pre-converted to inverse meters (from cm⁻¹) or to joules (from eV). Conversion from wavenumbers would tag the result as depending on the assumptions related to v=c/λ, including those related to the speed of light c. Conversion from energy would tag the result as depending on assumptions related to v=E/h, including those related to Planck's constant h. Conversion from a photon's wavelength to its frequency would be very similar to its conversion from wavenumbers, except that it would use an inverse linear dimension conversion.

```

<Expression, AssumptionSet>
differentDimensionalConvert

```

```

    (Subject fromSubj,
     Attribute fromAttr,
     Units fromUnits,
     Subject toSubj,
     Attribute toAttr,
     Units toUnits)
begin
Conversion convert;
Units      givenFromUnits, givenToUnits;
Expression expr, exprTo, exprFrom;
Set s := mostSpecificCommonSet(fromSubj,toSubj);

while (s != null)
  for all convert := s.get(diffDimConvA
                          (fromAttr,toAttr)
                          ) do
    givenFromUnits := convert.get(fromUnitsA);
    givenToUnits   := convert.get(toUnitsA);
    expr := convert.get(exprA);

  try
    exprFrom :=
      sameMultiDimensionalConvert
        (fromUnits,givenFromUnits);
    exprTo :=
      sameMultiDimensionalConvert
        (givenToUnits,toUnits);
  catch InsufficientInformation
    continue;
  endCatch

  return <exprTo.sub(expr.sub(exprFrom)),
        expr.get(assumptionsA)>

endFor
s := s.get(nextMoreEncompassingSetA)
endWhile

throw new insufficientInformation();
end;

```

The `differentDimensionalConvert()` algorithm also lets us handle dimensionless values that have been normalized by some empirical standard. For example, since the mid-1990s astronomers have found about 500 or so “exo-planets”, planets in places other than in orbit around our Sun. One way to detect such planets is by looking for its gravitational tug on the star which they orbit, and the more massive the planet the larger the tug, thus many of the planets that have been found are massive. To keep the mass numbers in intuitive ranges rather than as “so many kilograms times 10 to the such-and-such power” it is common to express them as multiples of the mass of our own giant, Jupiter.

Conversion from the dimensionless unit M_J (how many “Jupiters”) to the conventional mass unit kilograms can be done with `differentDimensionalConvert()` by giving the knowledge base a simple linear dimension

conversion rule telling it to convert from attributes with dimensionless values but normalized units to attributes for domains with units by multiplying by the normalization factor (in this case the mass of Jupiter: 1.8986×10^{27} kg (Williams 2010)). Although this rule is analytically true and thus makes no assumptions, many assumptions probably went in to the normalization factor. Thus the routine for `expr.get(assumptionsA)` must be clever enough to gather the assumptions of normalization factor too. Additionally, because assumptions are cumulative any assumption that went into why, for example, we believe planet μ Ara b to be $1.68 M_J$ (Butler *et al* 2001), such as our estimate of the mass of its star μ Ara, this assumption would also be included in the resulting value.

This one rule covers a variety of normalizations. For example, consider the domain of powerful events, especially nuclear detonations, with its common unit “kilotons of trinitrotoluene (kT)”. By stating that 1 kiloton of TNT is 4.184×10^{12} joules, our system can convert such values.

6. Discussion

We have presented the first automated approach, to the best of our knowledge, that both tags values with metadata describing the theory and measurements upon which their computation depend, and that carries this metadata forward for the computation of subsequent values.

At least two outstanding issues remain including handling data that contradict in terms of accuracy, and how to handle conversions where the domain knowledge tells us to consider three or more attributes.

First, what should we do if one value assumes Jupiter’s mass is 1.8986×10^{27} kg while a later one assumes it is 1.89857×10^{27} kg? One could convert between the two by dividing out what one considers the “less” accurate value and multiplying with what considers the “more” accurate. This is arithmetically sound, but what of the assumptions that went into both kilogram figures for the mass of Jupiter? They may be contradictory in a deeper manner than “same formula with more precise numbers used” by, for example, considering secondary effects (*e.g.* relativity) or by being derived from a different formula altogether. In such cases one could dig deeper to look for potential contradictions, or take a precautionary stand and throw a `PotentiallyContradictoryAssumption` exception.

A second problem occurs when we handle the special relativity equation $E=mc^2$. No dimension conversion is necessary because both sides have the dimension “energy”. However, the set applicability of `differentDimensionalConvert()` is still needed because only in certain circumstances like matter-antimatter annihilation do we observe mass to energy interconversion.

Even if we add this knowledge and its necessary restrictions we still have handled only a special case of

matter-antimatter annihilation of particles *at rest*. If particles have significant speed then their kinetic energy also should be considered. This would necessitate revising `differentDimensionalConvert()` or writing a new function to handle multiple attributes (*e.g.* mass, rest energy, and relative speed by, for example, the Lorentz transformations).

7. Conclusion

We have presented algorithms and knowledge structures needed to safely handle interconversion among four types of units in common usage in the sciences. Further, we have discussed its limitations.

It would be a mistake to think the solution to the second issue as merely extending an algorithm over more attributes. Fundamentally we should give our system the ability to (re-)define its own dimensions as needed. (The *StructProc* knowledgebase is being built with an eye towards this. This is the reason why we had the word “fundamental” in quotes when describing dimensions.) Thus, it should be able to define the Lorentz transformations, not just apply them. Such searches for both accurate and at least somewhat intuitive definitions of time, space, *etc.* are at the heart of modern physics quest to unify quantum mechanics with relativity (Callender 2010, Musser 2011).

Artificial intelligence may play a role this search through the space of representations. If it does we must be clear about what our systems actually represent and symbolically manipulate. This paper attempts to do so for a limited domain.

References

Butler, R. Paul; Tinney, C. G.; Marcy, Geoffrey W.; Jones, Hugh R. A.; Penny, Alan J.; Apps, Kevin. 2001. “Two new planets from the Anglo-Australian planet search” *Astrophysical Journal*. 555 : 410-417, 2001 July 1.

Callender, Craig. 2010. “Is Time an Illusion?” *Scientific American*. 2010 June.

deCarvalho, Rob. 2006. “Simple Units and Dimensions for Matlab”<http://www.mathworks.com/matlabcentral/fileexchange/9873>. Originally appeared Feb 2, updated Mar 3.

Khanin, Raya. 2001. “Dimensional analysis in computer algebra.” *International Symposium on Symbolic and Algebraic Computation*. ACM. 2001.

Langley, Pat. Simon, Herbert. Bradshaw, Gary. Zytow, Jan. *Scientific Discovery: Computational Explorations of the Creative Processes*. MIT Press. Cambridge, MA. 1987.

Mariano, Adrian. “Manual page for GNU Units version 1.85” 2004.

Musser, George. 2011. “Forces to Reckon With: Does gravity muck up electromagnetism?” *Scientific American*. 2011 February.

Phillips, Joseph. 2010. “A Proposed Semantics for the Sampled Values and Metadata of Scientific Values.” *Midwest Artificial Intelligence and Cognitive Science Conference*.

SunOS “Manual page for SunOS’ `units` command” 1992 Sep 14.

Williams, David R. 2010. “Jupiter Fact Sheet.” <http://nssdc.nasa.gov/planetary/factsheet/jupiterfact.html>.

Last updated 2010 November 17.

New Features and Many Improvements to Analyze Morphology and Color of Digitalized Plant Organs Are Available in Tomato Analyzer 3.0

Gustavo Rodriguez, David Francis, and Esther van der Knaap

Department of Horticulture and Crop Science, The Ohio State University/ Ohio Agricultural Research and Development Center, Wooster, Ohio 44691

AND

Jaymie Strecker, Itai Njanji, Josh Thomas, and Atticus Jack

Department of Mathematics and Computer Science, The College of Wooster, Wooster, Ohio 44691

Abstract

Tomato Analyzer measures morphological and color attributes via image analysis in an objective, high-throughput, and semiautomatic manner. This software allows for reproducible quantification of phenotypic data that previously were done by hand or visual analysis. The new version has improved the accuracy of all measurements and reduced the need to make time-consuming manual adjustments. In this paper new morphological and color attributes available in Tomato Analyzer 3.0 as well as how the color test module was made more user-friendly are described.

Introduction

The species in the plant kingdom are characterized by a great diversity in color, shapes and size displayed in organs such as leaves, flowers, and fruits. Even within a particular species the individuals also can be distinguished by the morphology and color displayed in those organs. Biologists trying to understand the genetic and molecular basis for this variation need to measure morphological and color attributes in an objective and reproducible way. Most of this type of phenotypic analysis consists of time-consuming manual measurements or subjective visual scoring of characteristics that reduce the success of identifying genomic regions or physiological causes underlying this variation.

Tomato Analyzer (TA) is a software program designed to collect objective data from digital images obtained from plant organs (Brewer et al, 2006). Many of these data are nearly impossible to quantify manually, such as angles at the distal and proximal ends of the organs or the variation for color in their surfaces. Briefly, the software recognizes the objects (fruits, leaves or seed) in digitalized images and

from the detected boundaries in each object is able to obtain more than 35 morphological attributes. The pixels inside the boundaries recognized by the software are used to translate color data from the RGB system into the L*a*b* universal color space which is able to approximate human visual perception (Darrigues et al, 2008). Moreover, TA combines controlled vocabulary consistent with terms present in trait ontology databases and mathematical descriptors for each shape and color attribute. Even though the application was specifically developed to analyze tomato fruit, this software can be applied to analyze fruit of other species and other plant organs such as seeds, flowers, and leaves.

This paper describes how morphological and color analysis of plant organs can be precisely done using Tomato Analyzer. Lastly, some possible applications of Color Test in Tomato Analyzer 3.0 are discussed.

1. Morphological Analysis of Plant Organs Using Tomato Analyzer

Tomato Analyzer can do a high-throughput analysis of morphological traits in images obtained from plant organs. Moreover, the data obtained are unbiased compared to those manually measured by different researchers or using different instruments. This impartiality allows the reproducibility of the experiments as well as the compilation and analysis of data obtained from experiments conducted in several environments and years.

To date, most of the morphological classifications in plants were made based on eyeball observations. Instead, attributes of TA can be used to objectively classify plant organs into various morphological categories.

Tomato Analyzer 2.0 has been a valuable and effective tool to identify and confirm genomic regions that control tomato fruit shape as well as performing in-depth analyses of the effect of key fruit shape genes on plant morphology.

It was possible due to color of tomato fruits are contrasting enough with a black background used in the scanned images. However, the software was unable to detect other darker fruits or leaves.

1.1 Workflow for Morphological Analysis

- Scan plant organs against a black background to eliminate shadows.
- Open the image in TA, select the attributes to measure, tell TA to analyze the image.
- TA separates objects from background. It does this by looking at a histogram of luminance for the image and finding the separation point (area of low histogram values) between the foreground (lighter colors) and the background (darker colors). It then finds contiguous areas of foreground pixels (the objects) and calculates the boundaries around them.
- Resulting data appears in a spreadsheet panel in TA (screenshot) and can be exported as .csv file.
- Some measurements can be manually adjusted.

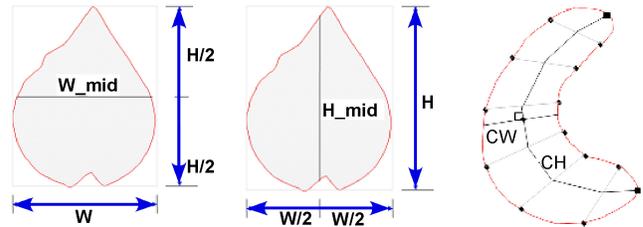
1.2 New Features and Attributes for Morphological Analysis in Tomato Analyzer 3.0

1.2.1 Reading TIFF images

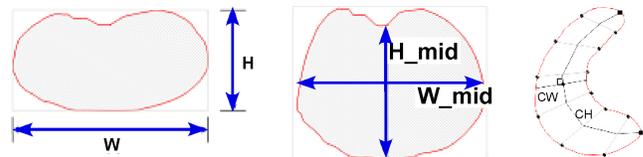
The previous version of Tomato Analyzer read only JPEG (.jpg) files but this new version can open both JPEG (.jpg) and TIFF (.tif) files. TIFF files are recommended because they preserve the image as it was originally scanned. JPEG images alter some of the colors in the image, reducing the accuracy of object boundary detection and color analysis. For example, this new feature has improved the boundary detection in dark green leaves and cucumber fruits. This improves the accuracy of all measurements and reduces the need to make time-consuming manual adjustments.

1.2.2 Length, width and fruit shape index attributes in curved fruits.

One of the most important features to analyze fruit shape in tomato is the fruit shape index (defined as the ratio between the width and the height of the fruits). When the tomato fruit or other type of fruits as cucumber is curved, the values for this index do not represent the actual value, they are underestimated. A new measurement, named curved height, was added in the new version of Tomato Analyzer (Figure 1). This attribute allows an accurate estimation of the length on curved fruits as well as is possible to be measured a new fruit shape index attribute.



- **Width Mid-height** (W_{mid}) – The width measured at $\frac{1}{2}$ of the fruit's height.
- **Maximum Width** (W) – The maximum horizontal distance of the fruit.
- **Height Mid-width** (H_{mid}) – The height measured at $\frac{1}{2}$ of the fruit's width.
- **Maximum Height** (H) – The maximum vertical distance of the fruit.
- **Curved Height** (CH) – The height measured along a curved line through the fruit (passing through the midpoints of opposing pairs of points on either side of the distal and proximal points).



- **Fruit Shape Index External I** (H / W) – The ratio of the Maximum Height to Maximum Width.
- **Fruit Shape Index External II** (H_{mid} / W_{mid}) – The ratio of Height Mid-width to Width Mid-height
- **Curved Fruit Shape Index** (CH / CW) – The ratio of Curved Height to the width of the fruit at mid-curved-height, as measured perpendicular to the curved height line.

Figure 1. Basic measurement attributes of Tomato Analyzer 3.0 and fruit shape index ratios based on this basic measurements.

1.2.3 Increased number of morphometric points

To measure shape without selecting individual attributes, TA offers a morphometric or geometric analysis of each object. This function finds points along the boundary of each tomato slice in the loaded image. Statistical tools such as Principal Component Analyses can be used to analyze the points in the exported data and it has been used to identifying tomato genome regions that control fruit morphology (Gonzalo et al, 2009). The distal and proximal ends are used as landmark points for every object (Figure 2). The number of points measured along the boundary is defined by the user and ranges from 4 to 200 in this new version of TA. The first morphometric point ($1x, 1y$) is always the proximal end point. The origin (0,0) of the coordinate system is located in the upper left corner of the rectangle defined by the Maximum Width and the Maximum Height.

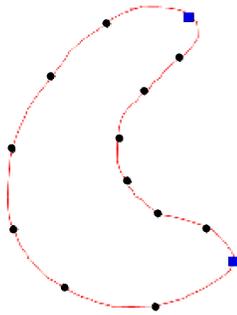


Figure 2. The morphometric points are a fixed number of points along the perimeter.

2. Color Analysis of Plant Organs Using Tomato Analyzer

Color is an important quality attribute in horticultural and floricultural crops defined by particular genes. However, color changes in plant organs also are indicators of biotic or abiotic stresses. The genetic bases for color attributes as well as factors affecting plant health would be better understood if computer-based analysis of digital images is applied instead of their subjective characterization. The Color Test module in Tomato Analyzer (TACT) is able of collecting and analyzing color parameters in an efficient, accurate and high-throughput manner from scanned images that contain plant organs. However, the scanner needs to be calibrated if the user intends to translate RGB values to L^* , a^* and b^* parameters. This is because scanners may change in accuracy and how they capture the color scheme over time. Moreover, scanners differ in how well they capture color information.

2.1 Workflow for Color Analysis

- Scan a color checker. Open the image in TA and perform a color calibration.
- Scan objects and open the image in TA, as you would for morphological analysis.
- Select the color attributes to measure, and tell TA to perform color analysis.
- Resulting data appears in the spreadsheet panel, as in morphological analysis.

2.2 New Features and Attributes for Color Analysis in Tomato Analyzer 3.0

2.2.1 Color Attributes Visualized in real time

The most important improvement related to the color test module in this new version of TA is that the results are

shown in the real time on the screen shot of the software (Figure 3). In the previous version, the results only could be visualized in .csv files after the analysis was done.



Figure 3. Screenshot of the Tomato Analyzer Test Color Module. This new version shows the color results in real time (right-down window).

2.2.2. A More User-friendly Test Color Module calibration

The user should chose a color checker with a black or very dark background based on the broad range of colors observed in the object of interest (Figure 4). Color checkers can be purchased custom made or standard.

	L^*	a^*	b^*
1	37.986	13.555	14.059
2	65.711	18.13	17.81
3	49.827	-4.88	-21.925
4	43.139	-13.095	21.905
5	55.112	8.844	-25.399
6	70.719	-33.397	-0.199
7	62.661	36.067	57.096
8	40.02	10.41	-45.964
9	51.124	48.239	16.248
10	30.325	22.976	-21.587
11	72.532	-23.709	57.255
12	71.941	19.363	67.857
13	28.778	14.179	-50.297
14	55.261	-38.342	31.37
15	42.101	53.378	28.19
16	81.733	4.039	79.819
17	51.935	49.986	-14.574
18	51.038	-28.631	-28.638
19	96.539	-0.425	1.186
20	81.257	-0.638	-0.335
21	66.766	-0.734	-0.504
22	50.867	-0.153	-0.27
23	35.656	-0.421	-1.231
24	20.461	-0.079	-0.973



Figure 4. Standard color checker from X-rite (Grand Rapids, MI) and actual L^* , a^* , b^* values for each tile of the color checker are shown in the table.

After the color checker was scanned should be opened and analyzed in TA. The software recognizes each tile as an

object. Then, the user needs to enter the actual L*,a*,b* values for each tile in the color checker, as provided by the manufacturer (Figure 4). TA uses the actual and observed L*, a*, and b* values to calculate the linear regression. After that, the color module is calibrated.

2.2.3 Three Methods to Analyze Color with Tomato Analyzer 3.0

- *Average color values.* The values displayed are: Average Red, Average Green, Average Blue, Average Luminosity, Average L* Value, Average a* Value, Average b* Value, Average Hue, Average Chroma. These average values are calculated taking account all pixel within the object.
- *L*, hue, chroma distributions.* These measurements provide histogram data for L*, hue, and chroma. The data appear in the tabs called L* Distributions, Hue Distributions, and Chroma Distributions, respectively. Each column shows the fraction of the object whose color falls within a certain range. For example, if the L[40..50) column in L* Distributions has the value 0.3, then 30% of the fruit has L* between 40 (inclusive) and 50 (exclusive).
- *Set custom color parameters.* Based on the L*, hue, chroma distributions, the user can define custom ranges of L*, hue, chroma, or a combination of the three. The User-Defined Color Ranges dialog appears as shown in Figure 5. The user can define up to 6 combinations of color ranges. In the example in Figure 5, Parameter 1 includes all colors where the hue is between 30 (inclusive) and 45 (exclusive); L* and chroma may be anything. Parameter 2 includes all colors where the L* is between 0 and 30 and the hue is between 0 and 90; the chroma may be anything. The data will appear in the data window tab called Custom Color Parameters.

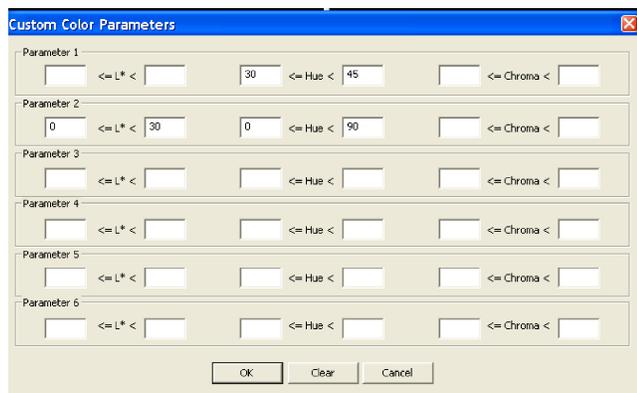


Figure 5. Set Custom Color parameters dialog box

2.3. Possible Applications Color Test Module of Tomato Analyzer 3.0

The Color Test module in Tomato Analyzer 3.0 is able to define the average for several color attributes inside the boundaries of a plant organ as well as the proportion of six different user-defined color parameters. This new feature can be useful to study pattern of color variation in some plant organ such as petals or leaves. In the plants, soil nutritional deficiencies, pesticides toxicities and even the severity of pathogen attacks affect the color pattern on some plant organ. For example, it has been demonstrated that the estimation of severity in a specific corn disease is highly affected by the rater experience when they directly estimate percentage of diseased leaf area and even more when they use a 0 to 9 ordinal rate scale (Poland and Nelson 2011). With TACT you can define a color range for the diseased portion and calculate the percentage of the leaf having that color. Thus, different researchers will get exactly the same results because they won't be interpreting the colors or the ratings differently. Therefore, Tomato Analyzer software would become in a powerful tool for this type of studies.

Acknowledgments

This project was funded by the National Science Foundation DBI 0227541 to Esther van der Knaap

References

- Brewer M.T.; Lang L.; Fujimura K.; Dujmovic N.; Gray, S.; and van der Knaap E. 2006. Development of a controlled vocabulary and software application to analyze fruit shape variation in tomato and other plant species. *Plant Physiology* 141 (1): 15-25.
- Darrigues A.; Hall J.; van der Knaap E.; and Francis D.M. 2008. Tomato analyzer-color test: A new tool for efficient digital phenotyping. *Journal of the American Society for Horticultural Science* 133 (4): 579-586.
- Gonzalo M.J.; Brewer M.T.; Anderson C.; Sullivan D.; Gray S.; and van der Knaap E. 2009. Tomato fruit shape analysis using morphometric and morphology attributes implemented in tomato analyzer software program. *Journal of the American Society for Horticultural Science* 134 (1): 77-87.
- Poland J.A. and Nelson R.J. 2011. In the Eye of the Beholder: The Effect of Rater Variability and Different Rating Scales on QTL Mapping. *Phytopathology* 101 (2): 290-298.

This page is intentionally left blank.

Information Retrieval

Chair: Jennifer Seitzer

Towards Agent-Oriented Knowledge Base Maintenance for Description Logic *ALCN*

Stanislav Ustymenko¹ and Daniel G. Schwartz²

¹ Meritus University, School of Information Technology,
30 Knowledge Park Drive (Suite 301), Fredericton, New Brunswick, Canada E3C 2R2 ,
sustymenko@meritusu.ca

² Department of Computer Science, Florida State University,
Tallahassee, Florida, USA
schwartz@cs.fsu.edu

Abstract

Artificial agents functioning in the Semantic Web are to be capable of getting knowledge from diverse sources. This implies the capability to continuously update their knowledge bases. New stream reasoning concepts make this need even more pressing. Semantic Web ontologies are commonly represented using description logic knowledge bases. We propose an agent architecture with such features, utilizing a Dynamic Reasoning System (DRS). This explicitly portrays reasoning as a process taking place in time and allows for manipulating inconsistent knowledge bases. We sketch a procedure for user-directed ontology debugging. This same mechanism can be used for automated belief revision. We identify important research directions that may benefit from this approach.

1 Introduction

The Semantic Web (SW) [4] is a common name of a family of technologies extending the Web with rich, machine-interpretable knowledge. The SW retains the massively decentralized nature of current World Wide Web, with an unlimited number of knowledge sources identifiable by unique URIs. It supports rich metadata annotation, including expressive ontology languages. Description Logics (DLs) [2] emerged as leading formalism for knowledge representation and reasoning on the Semantic Web.

Once widely implemented, the Semantic Web will support intelligent software agents that will work with massive, decentralized ontologies, while other agents modify them in possibly inconsistent ways. Agents will need a way to absorb new knowledge in a timely fashion, all the while protecting the consistency of their knowledge bases, or, alternatively, be able to draw useful inferences from inconsistent premises.

Several approaches have been proposed to model knowledge evolution over time. One of the most well-researched formalisms is *belief revision* [9, 10], specifically the classic AGM framework [1, 9]. Substantial efforts have been extended to apply this approach to description logics [13, 14, 19, 20], and the work is ongoing. However, the belief revision framework does not explicitly address knowledge evolution in time. Also, in its original

formulation, belief revision postulates are stated in terms of potentially infinite belief sets (although work has been done to address this issue). We believe that belief revision is a mature paradigm that can be valuable source for important insight. However, there is a need for formal approaches that address the practical challenges more directly.

New research direction under the tentative title “stream reasoning” [6] emerged within the Semantic Web community. It explicitly deals with reasoning over rapidly changing and time-dependent data in a way that can deliver answers to the user while they are still relevant. Stream reasoning is defined as “*the new multi-disciplinary approach which will provide the abstractions, foundations, methods, and tools required to integrate data streams and reasoning systems*” [7]. Della Valle et al. [5] write: “*Stream-reasoning research definitely needs new theoretical investigations that go beyond data-stream management systems, event-based systems, and complex event processing. Similarly, we must go beyond current existing theoretical frameworks such as belief revision and temporal logic*”. Currently, there is no consensus on a logic formalism appropriate for stream reasoning. There is an obvious practical need for such formalism to be able to integrate current, description logic-based Semantic Web standards.

Dynamic Reasoning Systems (DRS) [17] provide a formal framework for modeling the reasoning process of an artificial agent that “explicitly portrays reasoning as an activity that takes place in *time*”. It sidesteps the logical omniscience assumption of the classical AGM framework and has means of working with inconsistent knowledge bases by keeping track of a proposition's *derivation path*. The DRS framework has been shown to support non-monotonic reasoning in a natural way.

A DRS can be defined for any language. DLs present a challenge in that they do not have explicit derivation rules. Instead, DLs rely on *inference algorithms* to accomplish common reasoning tasks. One of the basic tasks is checking *subsumption* of concepts.

The goal of this paper is to present the DRS framework as a suitable formalism for Semantic Web reasoning. To this end, we give an instance of DRS capable of building a concept subsumption hierarchy for a well-known description logic.

We believe it to be an important foundation for research on belief dynamics for Semantic Web agents. Section 2 of this paper contains a brief formal introduction to Description Logics and the necessary definitions. Section 3 discusses Dynamic Reasoning Systems. Section 4 describes a DRS and agent reasoning process for deriving explicit subsumption hierarchies from description logic \mathcal{ALCN} terminology. Short abstract of this work appears in [20]. Finally, in Section 5, we draw some conclusions and discuss directions for future research.

2 Description Logics

Languages for any description logic contain *concept names*, *role names*, and *individual names*. Below, we will use uppercase A and B for concept names, uppercase letters R, P for role names, and lowercase x, y, z for individual names.

DL languages combine role and concept names into *concept definitions*. Concepts of a description logic \mathcal{AL} [16] are defined as follows:

$C, D \rightarrow$	A		(atomic concept)
	\top		(universal concept)
	\perp		(bottom concept)
	$\neg A$		(atomic negation)
	$C \cap D$		(intersection)
	$\forall R.C$		(value restriction)
	$\exists R.\top$		(limited existential quantification)

More expressive DLs extend \mathcal{AL} by the following constructs:

Indication	Syntax	Name
\cup	$C \cup D$	union
\exists	$\exists R.C$	full existential quantification
\mathcal{N}	$\leq n R, \geq n$	number restriction
\mathcal{C}	$\neg C$	full negation

The commonly used DL \mathcal{ALCN} extends \mathcal{AL} with full negation and number restriction. In the following sections, we will restrict ourselves to \mathcal{ALCN} .

An *Interpretation* of a DL is a structure $I = (\Delta^I, \cdot^I)$, where Δ^I is a nonempty set called *domain* and \cdot^I is an *interpretation function* that maps concept names to subsets of a domain, role names to subsets of $\Delta^I \times \Delta^I$,

and individual names to elements of Δ^I . The function \cdot^I extends to arbitrary concept definition in a rather intuitive way (for details, see [2], chapter 2). A concept is *unsatisfiable* if for any interpretation I , $C^I = \emptyset$.

Description Logic *knowledge bases* consist of two components: a TBox, a set of statements about concepts, and an ABox, a set of assertions about individuals. In general, a TBox T contains *general concept inclusion axioms* of the form $C \subseteq D$ (inclusion axiom). The pair of axioms $C \subseteq D, D \subseteq C$ is abbreviated $C \equiv D$ (equality axiom). An interpretation I satisfies an axiom $C \subseteq D$ if $C^I \subseteq D^I$. Interpretation I satisfies a TBox T if it satisfies every axiom in T .

A *definition* is an equality axiom with an atomic concept on the left hand side. A TBox is a *terminology* if it consists of definitions and no concept name is defined more than once. A concept name is a *defined name* if it appears on the left hand side of the axiom and a *base name* if it doesn't. A definition is in the *extended form* if only base concept names appear on the right hand side. A terminology is *definitorial* if every definition has exactly one extended form (not counting equivalent syntactic variants). In further discussion, we assume that our TBoxes are definitorial terminologies. Under this condition, we can assume, wlog, that definitions contain no cycles.

An ABox contains assertions regarding individual names. These include *concept assertions* $C(a)$ and *role assertions* $R(a, b)$. An interpretation I satisfies (or is a model of) $C(a)$ if $a^I \in C^I$ and it satisfies $R(a, b)$ if $(a^I, b^I) \in R^I$. Finally, I satisfies an assertion α (or an ABox A) with respect of a TBox T if it is a model of both an assertion (or an ABox) and the TBox.

An ontology of concepts can be expressed using a DL. The term *ontology* is often applied either to a TBox or to a full DL knowledge base. We will occasionally use *ontology* in the former sense.

3 Dynamic Reasoning Systems

The classical (propositional) notion of belief set [e.g., 9] models it as an (often infinite) set of formulas of the underlying logical language. In our view, a belief set should be finite and should represent the agent's knowledge and beliefs at a given point in time. Moreover, each formula in such a belief set should contain information indicating how it was obtained and whether it has been used in subsequent deductions, thereby enabling both backtracking and forward chaining through reasoning paths for so-called "reason maintenance".

To this end, in [17] there was defined the notion of a dynamic reasoning system (DRS), which explicitly portrays reasoning as an activity that takes place in time. This is

obtained from the conventional notion of formal logical system by lending special semantic status to the concept of a derivation path (i.e., a proof). Introduction of new knowledge or beliefs into the path occurs in two ways: either new propositions are added in the form of axioms, or some propositions are derived from earlier ones by means of an inference rule. In either case, the action is regarded as occurring in a discrete time step, and the new proposition is labeled with a time stamp (an integer) indicating the step at which this occurred. Moreover, for propositions entered into the path as a result of rule applications, the label additionally contains a record of which inference rule was used and which propositions were employed as premises.

At any given time, the contents of the path is regarded as being the sum total of the agent's knowledge and beliefs as of that time. Thus we here take this path as being the agent's belief set as of that time.

This is to be contrasted with other systems of belief revision, which assume that the agent additionally knows all the logical consequences of the basic belief set. Such systems are said to exhibit "logical omniscience." For an in-depth analysis of this issue, together with a manner of addressing it, see the paper by Fagin, Halpern, Moses, and Vardi [8].

For complete details of the notion of a DRS, please see [S97]. A brief outline is as follows. A labeled formula is defined as a pair (P, λ) where $P \in L$, where L is a logical language, and the label λ is an ordered

4-tuple (*index, from, to, status*), where:

1. *index* is a non-negative integer, the *index*, representing the formulas position in the belief set.
2. *from* is a *from list*, containing information about how the formula came to be entered into the belief set. Either it was received from an outside source (obtained from some other agent or through interaction with its environment), in which case the *from list* contains the token *rec*, or it was derived from some formulas occurring earlier in the belief set, in which case the *from list* contains the name of the derivation rule and the indexes of the formulas used as premises in the derivation. The types of formulas that can be received are understood to include both axioms of the propositional calculus and statements about the agents environment (sometimes distinguished as "logical" and "nonlogical" axioms).
3. *to* is a *to list*, containing the indexes of all formulas in the belief set for which the given formula served as a premise in the indexed formula's derivations.
4. *status* is a *status indicator*, taking values on or off, indicating whether the belief represented by the formula is currently held, i.e., whether the formula may or may not be used in any future derivations. Whenever a formula is initially entered into the belief set, its status is on.

For a given agent, let us denote the agent's belief set at time step i by Λ_i . Let $\Lambda_0 = \emptyset$. Thus the agent initially has no knowledge or beliefs. Then, given Λ_i , for $i \geq 0$, Λ_{i+1} can be obtained in any of the following ways:

1. A new formula is received from an outside source,
2. A formula is derived from some formulas in Λ_i by means of an inference rule,
3. A formula in Λ_i has its status changed from *on* to *off*.

Changing a formula's status from on to off occurs during a reason maintenance process that is invoked whenever an insatisfiability, i.e., a definition of the form $A \equiv \perp$ is entered into the agent's belief set. The objective of reason maintenance is to remove this insatisfiability.

This has two phases. First one starts back tracking from the insatisfiability, using the from lists in the formula labels, looking for the "culprit" formulas that occurred earlier and which led to the inconsistency. A decision then must be made to turn the status of at least one of these formulas to "off". Then one forward chains from this formula, using the to lists, to find all formulas whose derivations stemmed from the culprit formula, and likewise turns their status to "off". This will include the inconsistent formula that triggered the reason maintenance process.

Which culprit formula to deactivate can be determined by the various culprits' degrees of belief, to wit, remove the one that is least believed. In case the culprits are all believed equally, one can be chosen at random. Alternatively, an agent can remove the culprit formula that is the least important according to some reasonable criteria. One such criteria is a cumulative belief level of formulas derived from the culprit. This criteria provides a finite version of the AGM epistemic entrenchment relation.

This model of agent-oriented reasoning reflects that view that, at any given time, the agent's beliefs may harbor an inconsistency, but the agent does not become aware of this unless an inconsistent formula is explicitly entered into its belief set.

This, in our opinion, is a realistic model of natural human reasoning. Humans can comfortably maintain inconsistent beliefs for long periods of time without ever realizing this.

But once they become consciously aware of a contradiction, they typically rethink their position and modify their beliefs so that the contradiction is removed.

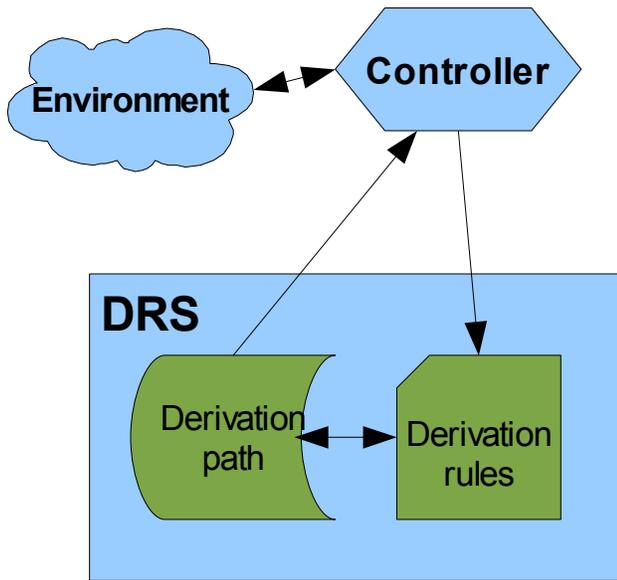


Fig.1 Reasoning agent employing a Dynamic Reasoning System

The reasoning agent (Fig. 1) uses a Dynamic Reasoning System to reach conclusions that help advance the agent's goals. A *controller* directs DRS behavior to steer it to such conclusions. The controller performs the following actions:

1. Receive information from the outside environment. The information can come from a human user, other agents, or be harvested by an agent through sensors. The latter can get information from any external data source.
2. Enter information, as a “nonlogical” axioms expressed in language L , into the DRS's inference path.
3. Apply an inference rule.
4. Act to remove insatiability, by invoking belief revision procedure described above.

The agent performs these actions in the order dictated by the agent's and environment's current state, presumably in a manner that would advance its goals. In the following, we are constructing an agent that would accept an ontology in the form of TBox definitions and construct a subsumption hierarchy of concept names implicit in this ontology.

4 Dynamic Reasoning for DL $ALCN$

A Dynamic Reasoning System is a model for knowledge base and reasoning process for artificial agent that assists a user. We describe an agent that extracts ontological knowledge from the Web and uses it to support a user's browsing and querying activities. To this end, an agent maintains two information stores:

1. Local copy of the ontology, expressed as an $ALCN$ TBox. This ontology consists of $ALCN$ definitions that occur in the derivation path.
2. A subsumption tree of concept names.

The latter can be used to support both browsing and user querying on both a TBox and an ABox. The user has a preference for satisfiable ontologies, so the agent has to detect and remove unsatisfiable concepts. Thus, our DRS needs to support 2 types of DL reasoning:

1. Check if a defined concept A is satisfiable
2. Deduce atomic subsumption, that is, a statement of the form $A \subseteq B$,

where A, B are concept names.

To construct the DRS, we first note that if A and B are defined by axioms $A \equiv C, B \equiv D$, where C, D are concept definition, then $A \subseteq B$ iff $C \subseteq D$. Second, note that $C \subseteq D$ iff $C \cup \neg D$ is unsatisfiable.

So both our reasoning tasks would require checking satisfiability of concepts. We are using a generic tableau-based satisfiability algorithm [2, 3].

Now we can build our dynamic reasoning system. First, We define the language, L . The symbols of L are the same as the symbols of logic $ALCN$. We use A, B for concept names occurring in the incoming statements and A', B' for the names introduced by the agents. The formulas of L are the following:

1. Equivalence statements of the form $A \equiv C$, where A is a concept name and C is concept definition. Without loss of generality, we assume all concept definitions are in negation normal form, i.e. negation only occurs in front of concept names.
2. Atomic subsumption statements of the form $A \subseteq B$, where A, B are concept names. These represent arcs of the subsumption tree the agent is building.
3. TBox assertions $C(a), R(a, b)$, where C is a concept, R is a role, and a, b are individual constants
4. Explicit inequality assertions $x \neq y$, where x, y are individual names.

Then we define inference rules. Implicitly, every rule that modifies a concept definition also puts the result into negation normal form. The inference rules will be:

1. *Substitution*: from $A \equiv C$ and $B \equiv D$ infer $A \equiv E$, where E is C with all occurrences of B replaced by D . For this treatment we assume that our TBox does not contain cycles in definitions. By repeatedly applying this rule, we obtain an *extension* of definition for A that only contain *ground* concept names on the right side.
2. *Subsumption test introduction*: from $A \equiv C, B \equiv D$ infer $A' \equiv C \cap \neg D$, where

A' is a previously-unused agent-generated concept name.

3. From $A \equiv C, B \equiv D$ and $A' \equiv \perp$, provided that name A' was introduced using rule 2 with $A \equiv C, B \equiv D$ as premises, derive $A \subseteq B$.

The following rules 4-10 are added to enable tableau-based consistency checks. These are derived from the transformation rules listed in [2], p. 81. Individual names x, y, z, \dots are unique names generated by the agent. All TBox statements are derived from the same ABox statement (that is undergoing satisfiability check) $A \equiv C_0$:

4. From $A \equiv C_0$, infer $C_o(x_0)$, if no ABox statements were inferred from $A \equiv C_0$.
5. From $A \equiv C_0$ and $(C_1 \cap C_2)(x)$, infer $C_1(x)$ and $C_2(x)$, if any one of them is not yet inferred.
6. From $A \equiv C_0$ and $(C_1 \cup C_2)(x)$, infer $C_1(x)$ or $C_2(x)$, if neither of them is inferred yet.
7. From $A \equiv C_0$ and $(\exists R.C)(x)$, infer $C(y)$ and $R(x, y)$, where y is a new generated name, if no z exists such that $C(z)$ and $R(x, z)$ are already derived.
8. From $A \equiv C_0$, $(\forall R.C)(x)$ and $R(x, y)$ infer $C(y)$, if not already derived.
9. From $A \equiv C_0$ and $(\geq n R)(x)$, infer $R(x, y_1), \dots, R(x, y_n)$ and $(y_i \neq y_j)$, and $R(x, y)$, unless $R(x, z_1), \dots, R(x, z_n)$ are already inferred.
10. From $A \equiv C_0$ and $(\leq n R)(x)$, if $R(x, y_1), \dots, R(x, y_{n+1})$ are in the derivation path and $y_i \neq y_j$ is not in the path for some $i \neq j$: replace all occurrences of y_i with y_j .

The following rules 11-13 detect inconsistency in TBoxes built using rules 4-10. As above, TBox statements are derived from $A \equiv \perp$:

11. From $A \equiv C_0$ and $\perp(x)$, derive $A \equiv \perp$, where x is any individual name.
12. From $A \equiv C_0$, $A_1(x)$ and $\neg A_1(x)$, derive $A \equiv \perp$, where x is any individual name and A_1 is any concept name.
13. From $A \equiv C_0$, $(\leq n R)(x)$, set $\{R(x, y_i) \vee 1 \leq i \leq n+1\}$ and set

$\{y_i \neq y_j \vee 1 \leq i \leq j \leq n+1\}$ derive $A \equiv \perp$, where x, y_1, \dots, y_{n+1} are individual names, R is a role name and $n > 0$.

Finally, rule 14 derives a subsumption axiom, using reduction to unsatisfiability:

14. From $A \equiv C$, $B \equiv D$, $A_1 \equiv C \cup \neg D$ and $A_1 \equiv \perp$, derive $A \subseteq B$

A Dynamic Reasoning System based on language L and rules 1-14 is capable of supporting an agent that builds an explicit subsumption hierarchy. We will now describe a controller that can achieve this goal.

An agent starts with an empty derivation path and empty subsumption hierarchy. It will receive TBox definitions from the user. To start the hierarchy, before receiving the first axiom, the controller will enter a root concept, $R \equiv \top$, as a first formula in the derivation path and R as a root node in the hierarchy.

Upon entering a new axiom of the form $A \equiv C$, the controller will perform the following actions:

1. Derive an expanded definition of A by repeatedly employing Rule 1 until the right side of the resulting definition contains no defined concept names.
2. Test satisfiability of A using Rules 4-13. If it is unsatisfiable, flag it for a belief revision procedure
3. Expand all (extended) definitions that depend on using Rule 1. Test the affected concepts' satisfiability, flagging for a belief revision process if unsatisfiable. Update the hierarchy of concepts affected by this step, testing subsumption by using Rules 2-14.
4. Place A into its appropriate place in the subsumption hierarchy, using Rules 2-14 to test subsumption with definitions of concept names already there.

To test satisfiability by employing Rules 3-13, an agent follows a tableau-based algorithm. Details of the appropriate algorithm, with discussion of termination and complexity, can be found in [2].

Rules 6 and 10 are *non-deterministic*: for a given ABox, they can be applied in finitely many different ways, leading to finitely many ABox'es. The concept is satisfiable if at least one such ABox is consistent. Each ABox is a branch in the satisfiability algorithm. The controller may handle branches by setting the belief status of statements on inactive branches to *off*. In practice, it may be useful to remove such statements from the path to save space.

We did not specify the details of modifying subsumption hierarchy on steps 3 and 4. In principle, the controller may simply search the existing hierarchy starting at the root, testing the concept in question's subsumption with each node. This is a natural and decidable procedure that will result in the correct hierarchy. Studying the complexity of such an

algorithm and exploring possible optimizations is a task left for future research.

In case an unsatisfiable concept is detected, an agent will generate and display to the user the list of definitions that lead to it. The user will have a choice to delete and modify one of them. Methods for assisting the user or for achieving this task without user interaction can be developed, based on research in ontology debugging and belief revision for description logics [12, 13, 14, 21]. Developing such algorithms is another task left for future research.

5 Conclusions and Further Research

In the present work, we argued for the suitability of Dynamic Reasoning Systems [17, 18] as formalism for agent reasoning on the Semantic Web. To this end, we presented a limited but realistic example of a DRS for performing a common reasoning task on a Description Logic ontology. We sketched a procedure for user-directed ontology debugging. This same mechanism can be used for automated belief revision.

Research in reasoning dynamics for the Semantic Web is a major part of the overall Semantic Web effort. The problem has been approached from belief revision [14], ontology debugging [12], and now stream reasoning [6] perspectives. We believe the present approach has the potential to contribute to all these efforts.

There are several directions for future research. First, the agent presented needs to be described in greater detail. Procedures need to be fleshed out, and potential performance problems need to be identified and addressed. Complexity issues need to be discussed. There is also a possibility to use data stored in the derivation path to speed up new reasoning. For example, incremental algorithms can be designed to utilize and extend existing derivation paths when a concept gets updated through incorporating new definitions.

The agent can also be extended to support more varied reasoning. It can be modified to accept more kinds of input, including, e. g. , general inclusion axioms and TBox assertions. A facility to deal with user queries on an ABox needs to be added. The agent can be used as a model to build DRSs capable of dealing with Semantic Web standards and more realistic scenarios (reasoning in the presence of loops and redefinitions of concepts). On the other hand, less expressive DLs can be investigated, in hope that they may guarantee moderate computational complexity.

Finally, the DRS formalism can be used to investigate belief revision techniques. Variants on the AGM framework's rationality postulates can be constructed for a finite DRS case, both in general and specifically for description logics. Feasible algorithms adhering to these principles need to be constructed. Finally, these postulates and algorithms can be applied to interesting practical cases, such as reasoning with multi sourced information that takes into account different degrees of the agent's belief and trust between agents.

References

1. Alchourrón, C.E., Gärdenfors, P., Makinson, D.: On the logic of theory change: partial meet contraction and revision functions, *Journal of Symbolic Logic*, **50**, 2 (1985)
2. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (eds.): *The Description Logic Handbook*, Cambridge University Press (2003)
3. Baader, F., Sattler, S.: Expressive number restrictions in Description Logics. *J. of Logic and Computation*, 9(3):319–350 (1999)
4. Bernes-Lee, T., Hendler, J., Lassila, O.: *The Semantic Web*. Scientific American (2001)
5. Della Valle, E., Ceri, S., van Hamelern, F., Fensel, D.: It's a streaming world! Reasoning upon rapidly changing information. *Intelligent Systems vol.24 no.6*, pp. 83-89 (2009a)
6. Della Valle, E., Ceri, S., Fensel, D., van Hamelern, F., Studer, R.: (Eds.): *Proc. 1st International Workshop on Stream Reasoning*, CEUR vol. 466 (2009)
7. Della Valle, E., Ceri, S., Braga, D., Celino, I., Fensel, D., van Hamelern, F., Unel, G., *Research chapters in stream reasoning*, In: *Proc. 1st International Workshop on Stream Reasoning*, CEUR vol. 466 (2009)
8. Fagin, R., Halpern, J., Moses, Y., Vardi, M. *Reasoning about knowledge*, MIT Press, Cambridge, MA, 1995
9. Gärdenfors, P.: *Knowledge in Flux: Modeling the Dynamics of Epistemic States*, MIT Press/Bradford Books, Cambridge, MA (1988).
10. Gärdenfors, P., ed.: *Belief Revision*, Cambridge University Press, NewYork (1992)
11. Hamilton, A.G., *Logic for Mathematicians*, Revised Edition, Cambridge University Press (1988).
12. Parsia, B., Sirin, E., Kalyanpur, A.: Debugging OWL ontologies, in *Proc. 14th International World Wide Web Conference (WWW'05)*, Chiba, Japan, May 10-14, 2005. ACM Press 2005
13. Ribeiro, M. M., Wassermann, R.: First Steps Towards Revising Ontologies, In: *Proc. 2nd Workshop on Ontologies and their Applications*, CEUR Workshop Proc. Vol. 166 (2006)
14. Ribeiro, M. M., Wassermann, R.: Base Revision in Description Logics - Preliminary Results, In: *International Workshop on Ontology Dynamics (IWOD)* (2007)
15. Smolka, G.: A feature logic with subsorts. Technical Report 33, IWBS, IBM Deutschland, P.O. Box 80 08 80 D-7000 Stuttgart 80, Germany (1988)
16. Schmidt-Schauß, M., Smolka, G.: Attributive concept descriptions with complements. *Artificial Intelligence*, 48(1):1–26, (1991)
17. Schwartz, D. G.: Dynamic reasoning with qualified syllogisms, *Artificial Intelligence*, 93(1-2) (1997) 103-167.
18. Schwartz, D.G.: Formal Specifications for a Document Management Assistant, In: *Proc. International Conference on Systems, Computing Sciences and Software Engineering* (2009)
19. Qi, G., Liu, W., Bell, D. A.: Knowledge Base Revision in Description Logics, In *Proc. European Conference on Logics in Artificial Intelligence (Jelia'06)*, Springer Verlag (2006)
20. Ustymenko, S., Schwartz, D.G., *Dynamic Reasoning for Description Logic Terminologies*. Canadian Conference on AI 2010: 340-343
21. Zhuang, Z. Q., and Pagnucco, M.: *Belief Contraction in the Description Logic EL*, In: *22nd Int. Workshop on Description Logics (DL 2009)*, CEUR Workshop Proc. Vol. 477 (2009)

Extracting Micro Ontologies from Interrogative Domains For Epistemic Agents

Tracey Hughes and Cameron Hughes

Ctest Laboratories
One University Plaza
Youngstown, Ohio, 44555
traceyhughes@ctestlabs.org
cameronhughes@computer.org

Abstract

A relevant and functional ontology continues to be one of the bottlenecks to the process of building epistemic agent oriented systems. While the construction of electronic ontologies is the focus of many ongoing efforts, ontology building in a timely manner remains an obstacle. Our current focus is directed toward the notion of automated identification of ontological artifacts from interrogative domains in real time. The artifacts that we are interested in form the basis for a micro ontology of the interrogative domain under consideration.

Introduction

Basic knowledge acquisition can proceed only after the fundamental ontology of the domain has been identified (Nirenburg and Raskin 2004). The problem of identifying an appropriate knowledge representation scheme is also constrained by the selection of a pertinent ontology (Brachman and Levesque 2004). Its for this reason an improvement of the ontology generation process will help to remove one of the bottlenecks to the process of building knowledge-based agent-oriented systems. While there are many efforts underway in the discipline to remedy this (Witherell, et. al. 2010), ontology building remains an obstacle (C-Y. Lu 1987). Our current focus is directed toward the notion of automated identification of ontological artifacts from interrogative domains in real-time. The artifacts that we are interested in form the basis for a micro-ontology of the interrogative domain under consideration. The ontological artifacts of interest are:

- vocabulary of the ontology
- base relationships in the ontology
- primary entities and objects of the ontology

Interrogative Domains as Sources of Knowledge

Interrogative domains consist primarily of questions and answers. A presenter presents an entity with one or more questions and the entity is challenged with providing an appropriate answer (Lehnert 1978). The interrogative domains we are concerned with are digital transcripts of trials, congressional hearings, interviews, surveys and law enforcement interrogations. What we find interesting about transcripts from these areas is the sheer scope of the subject matter. Legal transcripts cover argumentation on such wide ranging topics as when life begins in the womb to violations of civil rights by the patriot act. Transcripts of civil proceedings range from expert testimony on the effect of harmful contaminants in ground water to the effect of video games on adolescent weight gain. It is clear that interrogative domains present particularly fertile sources for knowledge and the value of extracting even micro ontologies from that knowledge should not be underestimated.

The knowledge acquisition process which is dependent on an underlying ontology can be expensive, error prone, and lengthy (C-Y. Lu 1987), and is in fact one of the bottlenecks to building epistemic-based agent systems. Our approach has been to find and experiment with new and different knowledge sources for the knowledge acquisition process (Hughes and Hughes, 2009). The scope and form of digital transcripts make them reasonable candidates for our investigation.

In this paper we describe epistemological and propositional knowledge analysis techniques that we are investigating at Ctest Laboratories. These analysis techniques are used to automatically discover ontological artifacts within the transcripts of an interrogative domain. These ontological artifacts are then used as the fundamental basis for building micro-ontologies of the subject matter found in each transcript. We are using ROGUE (Real-time Ontology Generation Using Epistemic Agents) to perform the transcript and text mining. ROGUE is an experimental multi-epistemic agent-based system under development at Ctest Laboratories. Ultimately, it is ROGUE agents that automatically generate the micro-ontology used as the basis

for knowledge acquisition and a knowledge space for an epistemic agent.

Interrogative Domains and Entailment

In a natural language processing or text mining context, it is not always clear when or how one grammatical item is related to another. It is not always clear when the scope or focus has changed or when new referents have been introduced or old ones dropped. This is part of what makes natural language processing challenging (Barton, et. al. 1987). What sets interrogative transcripts apart from other types of texts and documents is that they consist primarily of question and answer pairs. The relationship between a question and answer is predetermined and clear. The fact that the transcript consists primarily of questions and answers greatly simplifies the segmenting task (Blackburn and Bos 2005). Because questions and answers have a predetermined relationship, we can take advantage of interrogative entailment for transcript mining purposes. Most question and answer pairs entail one or more statements. For example, the question and answer pair in Table 1 is taken from one of the trial transcripts that we used in our data sets.

Question and Answer
<p>Q: <i>And you saw her take us all to a site that was in London is that correct?</i></p> <p>A: <i>Yes.</i></p>
Entailed Proposition
E1. <i>I saw her take us all to a site that was in London.</i>
Inferred Propositions
I1: <i>She took me to a site in London.</i>
I2: <i>We all went to a site in London.</i>

Table 1. The Q&A pair and there entailed and inferred propositions.

The simple meaning of the Q combined with A semantically entails Proposition E1 and from E1 we may infer Propositions I1 and I2. These propositions compose the propositional knowledge extracted from the digital transcripts.

Notions of Knowledge and Truth

Our discussion of knowledge is restricted to propositional knowledge. We use the Tripartite Analysis (Kant 1965) of Knowledge (TAK) as the basis for our epistemic agents. Although the Tripartite Analysis has short comings and has been thoroughly criticized, see (Gettier 1963) for the basic attack on the Tripartite deconstruction, it is well suited for our implementation of epistemic agents. In this analysis, propositional knowledge is understood as justified true belief. We use Kripke structures (Kripke 1963) as an

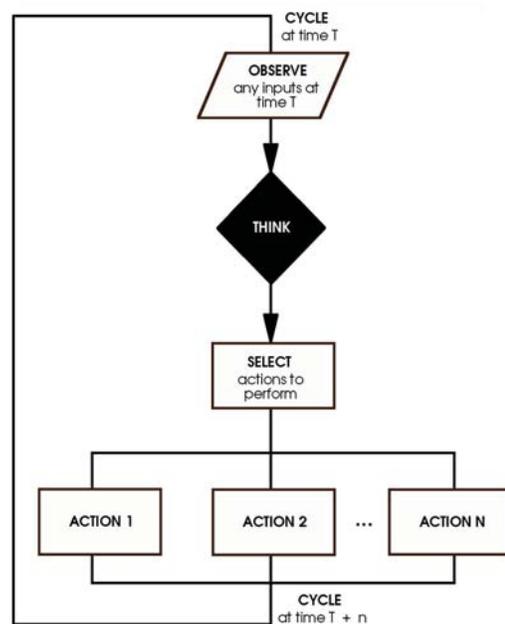


Figure 1. The classic observe think act agent cycle.

intersection between the Tripartite Analysis and our Epistemic Agent. If we let M be a Kripke structure:

$$M = (S, \pi, K_1, \dots, K_n) \tag{1}$$

Then K , what the agent knows, and S , the worlds the agent consider possible, are used to represent the notion of modal truth in the context of the Tripartite Analysis. Further, S is expanded and defined by our agent's Epistemic Structure (Hughes, et. al. 2008).

Contrast of Agents and Epistemic Agents

In the context of our work and this paper, an agent is recursively defined as a situated autonomous software component that has an agent cycle. A distinguishing feature of the agent is that it affects and is affected by the environment that it is situated in through the activity that takes place in the agent cycle as seen in Figure 1. While there are several classic notions of what constitutes an agent cycle, see (Shoham 1993), (Rao and Georgeff 1995) and (Wooldridge 2000), we take our cue from Kowalski's and Sadri's work in logic programming and multi-agent systems (Kowalski and Sadri 1999). We chose to build on (Kowalski and Sadri 1999) partially because it subsumes the notions described by (Shoham 1993) and (Rao and Georgeff 1995) but primarily because their unified agent cycle is compatible with our notion of an epistemic agent (Hughes and Hughes, 2009). In (Kowalski and Sadri 1999) unified agent cycle, the "think" process utilizes reasoning, logic programs, and integrity constraints. The goals of the agents include commands and queries, integrity constraints,

obligations and prohibitions, condition-action rules, and commitment rules.

Epistemic agents have an epistemic structure (Hughes, et. al. 2008), and their agent cycle is focused on acquiring, justifying, and maintaining propositional knowledge and classifying propositions which serve as the goals and the actions that can be performed. Further, epistemic agents have a DNA (Deductive Nuclear Architecture) (Hughes and Hughes, 2009) as opposed to a BDI architecture (Wooldridge 2000). The DNA is a structure designated by α :

$$\alpha = \langle E_s, \Theta, I, \delta \rangle \quad (2)$$

where :

- E_s is the Epistemic Structure,
- Θ is a deductive theory of inference over E_s
- I , is a set of interrogative types,
- δ is the deductive response, the set of conclusions reachable from E_s .

The epistemic structure is a knowledge representation scheme that models the TAK (justified true belief). It is defined as:

$$E_s = \langle G_1, G_2, J_s, V_c, F \rangle \quad (3)$$

where:

- G_1 is a graph of a *priori* propositions,
- G_2 is a graph of a *posteriori* propositions,
- J_s is a set of justification propositions,
- V_c is a vector of commitment,
- F is a non-monotonic truth maintenance function on E_s .

The a priori propositions stored in G_1 constitute the entailed propositions from the Q&A pairs of the digital transcript. The propositions in G_2 are inferred and acquired during the agent cycle and induction. Initially G_2 is $\{ \}$, and we have G_1 with a cardinality > 0 . Some of the a priori propositions will serve as justifications (populate J_s) and as a basis for the agent's level of commitment (populate V_c) (Hughes and Hughes 2009) to the other propositions in G_1 and to those that will later populate G_2 . The agents beliefs are based upon the V_c determined by J_s . *Initially*, the agent evaluates its level of commitment to the proposition contained in G_1 . As propositions are inferred and G_2 becomes populated, the level of commitment to propositions in both G_1 and G_2 are re-evaluated by F which in turns updates J_s and V_c . F functions as an integrity constraint assuring that the agents propositional knowledge is logical and consistent. Populating G_2 is determined by Θ , the

deductive theory of inference which defines the inference types. Depending on the proposition type (determined by the Q&A type from which it was entailed based on I_i) will dictate the inference type. Θ and I_i both works as a condition-action rules.

Defining the Knowledge Space

Let $d = \{ E_{s1}, E_{s2}, E_{s3}, \dots, E_{sn} \}$

where d is a set of epistemic structures representing a particular domain, or a collection of domains. Then we have:

$$K_s = \bigcup_{i=1}^N d \quad (4)$$

where K_s is a set union of the domains, or the total knowledge space of the agent, also called the *epistemic reality*. Building K is where one of the primary bottlenecks to deploying epistemic agent-oriented systems. Typically,

Techniques	Description	Pros/Cons
Interviews	Live Q & A sessions with actors used to produce protocols. They can be structured, semi and unstructured	Pros:
		Validations immediate
		Cons:
		Explicit knowledge not tacit knowledge
Questionnaires	Questions presented to a respondent that supplies written answers	Pros:
		Standard statistical treatment Simple to complete Not time consuming
		Cons:
		Answers are limited Little interaction
Observation	Observing the expert performing tasks and taking notes in order to document protocols	Pros:
		Low complexity and cost Gather process and tacit knowledge
		Cons:
		Time consuming process Total dependency on observer
Sorting	Used to capture the way respondents compare and order concepts, objects, attributes, and values associated with them	Pros:
		Explicit detailed knowledge about concepts
		Cons:
		Concepts must first be identified Experts may have varying views

Table 2. Description, pros and cons of KA techniques.

K is built using one or more of several standard knowledge acquisition techniques. These techniques are shown in Table 2.

It is in the knowledge acquisition phase where we look to speed up the process. Rather than acquiring the base ontology through the traditional processes of conducting interviews, disseminating questionnaires, repository evaluation local-remote observation and requirement analysis (Whitman, et. al. 2007), we are exploring the use of transcript mining (Hughes and Hughes 2009) to automatically identify the base ontology for the agent's a priori knowledge space. Once it's built, we can summarize the behavior of the epistemic agent.

If we let:

$$e_n / \alpha \quad (5)$$

be a set of agents with DNA then the Fundamental Epistemic Axiom can be stated as:

$$q \delta(q) \leftrightarrow K_s \quad \delta(q) \quad e_n / \alpha \Theta(I, (q)) \cap M \quad (6)$$

This axiom is used to guarantee the epistemological soundness of the work performed by the epistemic agents.

Ontological Artifacts

The ontological artifacts are taken directly from the propositional knowledge found in the transcripts. If we consider the transcript as the universe of discourse, then all the worlds that the agent considers possible will be contained within the transcript since it provides the complete context for the epistemic agent's knowledge. So when it comes to truth analysis, the epistemic reality (4) and the worlds the agent considers possible (Kripke 1963) are taken from the transcript. With this in mind we use transcript mining (Hughes and Hughes 2009) to automatically extract the entailed propositions of the transcript.

There are 5 basic steps in the extraction of the ontological artifacts from the transcript:

- Step 1:** Segment the transcript into blocks of Q&A pairs while maintaining the chronological appearance of witnesses, attorney's evidence, etc.
- Step 2:** Classify the Q&A pairs according to the 13 basic categories of questions and answers.
- Step 3:** Resolve anaphora, and mark substitutions between the question and answer pair blocks.
- Step 4:** Using interrogative entailment, convert the Q&A pairs to propositions.
- Step 5:** Using the predicates: `simple_subject()`, `simple_predicate()`, `simple_object()` extract the ontological artifacts from the propositions

The classification of the Q&A pairs has important ramifications because the Q&A pair classification determines the type of entailed proposition. If the Q&A pair is a location type, then the entailed proposition will also be a location type and will make an assertion about location. It is also important to do the Q&A classification at this point because it helps with the anaphora (Kamp and Reyle 1993) resolution rules.

So, if we let: $T = \{ \text{set of Q\&A in transcript} \}$ then:

$$T \quad T_i \quad (7)$$

where T_i is the set of propositions entailed from T .

Model Theoretic Semantic = Micro-Ontology

Once we have T_i we perform a Model Theoretic Analysis (MTA) on T_i to extract the ontological artifacts of our interest (Blackburn and Bos 2005), being:

- vocabulary of **T**
- base relationships in **T**
- primary entities and objects in **T**

Note that the MTA gives us the artifacts that form the basis of our micro-ontology. In fact, an MTA is sufficient to produce a micro-ontology for a transcript knowledge source. This analysis is done using three predicates from the ROGUE system:

```
simple_subject(Pi)
simple_predicate(Pi)
simple_object(Pi)
```

where P_i is an interrogatively entailed proposition. And the three simple predicates take P_i as arguments and produce the corresponding ontological artifact. For example, we can extract ontological artifacts for the entailed proposition in Table 1:

E1: *I saw her take us all to a site that was in London.*

In Step 3, the anaphora are resolved so the propositions will contain these substitutions:

E1 w/ substitutions:
Mr. Garcia saw Ms Runga take Mr. Garcia, Mr. Cannon, and Mr. Homes to a site that was in London.

The resulting extracted ontological artifacts would be:

```
was_in(Mr. Garcia,London_site)
was_in(Mr. Cannon,London_site)
was_in(Mr. Homes,London_site)
```

The ontological artifacts are captured by the model theoretic semantics of the transcript. Here the model is described as $m = (D, f)$ where m is a model theoretic semantic representation of all the language that is contained in the trial corpus. m consists of a pair (D, f) where D is the *Domain* which is the set of people and things referenced in the corpus (e.g. defendants, jurors, attorneys, witnesses) plus the relations (e.g. $lawyer(X, Y)$, $trial_day(N)$, etc.) between those people and things. f is an *Interpretation Function* which maps everything in the language onto something in the domain (Blackburn and Bos 2005). f is implemented using the Lambda Calculus operator λ and β -conversion. m captures completely our ontological artifacts. It is important to note here that the process that extracts m from T_i requires robust natural language processing (NLP). At Ctest Labs, we use ISIS, an agent-oriented system dedicated to NLP. The micro-ontology is a component of the Cognopaedia which serves as the universe of discourse.

ROGUE Agent Architecture

As mentioned earlier at Ctest labs, we are using ROGUE system to extract these ontological artifacts. ROGUE (Real-time Ontology Generation Using Epistemic Agents) is an experimental multi-epistemic agent-based system that automatically generates the micro-ontology used as the basis for knowledge acquisition and a knowledge space for an epistemic agent. The ROGUE agent exists in an environment from which it senses percepts (inputs), perform actions, and then outputs to the environment. There can be 1 to n ROGUE agents in a given environment working in parallel. Figure 2 shows the architecture of the

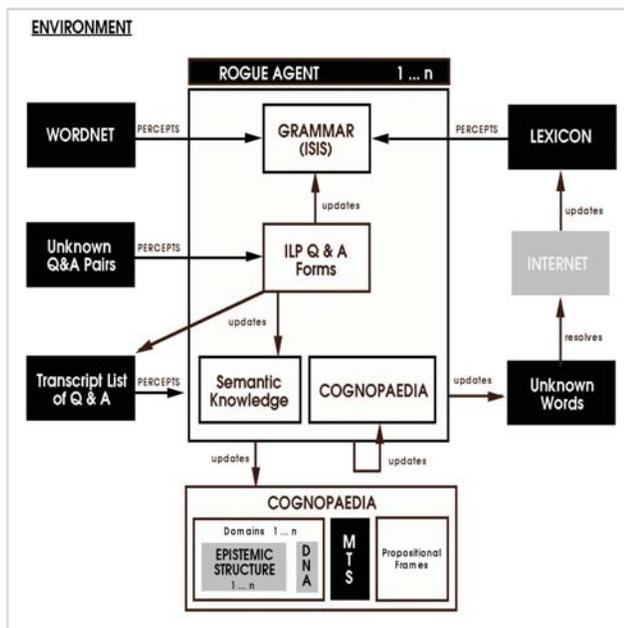


Figure 2. The architecture of the ROGUE Agent.

ROGUE agent.

The ROGUE agent has GRAMMAR (ISIS) and ILP Q&A Forms. ISIS (interrogative Sentence Interface Subsystem) provides a NLI (Natural Language Interface) to the ROGUE system. It is involved in processing the Q&A pairs and mapping them to propositions and OAV tuples. ISIS receives inputs from the WORDNET and Lexicon components. ILP Q&A Forms is an inductive logic program that processes unknown Q&A pairs. When ILP becomes aware of an unknown Q&A form, it attempts to identify it using ILP. If it is able to identify the unknown Q&A form, it updates ISIS so it can now process that form when it is again presented. The now known Q&A pairs of that form is then re-introduced to the agent from the Transcript List Q&A component. The ILP Q&A Forms also updates the agent's Semantic (Procedural) Knowledge.

The COGNOPAEDIA is an ontology and knowledge construct that replaces the application domain. It contains 1 to nE_s , an ontology (and micro-ontology), and propositional frames that models the propositions. When propositions are entailed or inferred, and tuples are extracted, the E_s and ontology contained in the COGNOPAEDIA are updated. The "on board" knowledge construct is required by the ROGUE agent in order to update the COGNOPAEDIA that is apart of the environment. The agent's "on board" knowledge construct is updated by the agent itself.

Discussion And Future Work

In this paper, we have presented some of our work at Ctest Laboratories where we are using ROGUE to extract the micro-ontology or model theoretic semantic from digital transcripts. We are interested in the transcripts because of the sheer scope and subject matter that transcripts cover. The notion that we could improve traditional knowledge acquisition techniques by automating the process of building the micro-ontology is very enticing. However, we may be too ambitious in our objectives because to date, we've only built micro-ontologies for less than two dozen transcripts and the transcript subject matter was not sufficiently diverse. Also we lacked diversity in the types of transcripts that we used. In particular, we have only looked at trial transcripts as opposed to interviews, congressional hearings, surveys, interrogations, etc. However, because building the micro-ontology is usually a prerequisite for the full knowledge acquisition phase, we are persuaded that the ROGUE approach will prove useful and we are already in the process of setting up more formal and exhaustive experiments. We are in the process of attempting multi-domain knowledge spaces.

Finally, the question and answers themselves require robust natural language processing in the interrogative entailment phase. From the transcripts we have processed, about 20% of the Q&A pairs are still not intelligible to our grammar and semantic parsers. So while the digital transcript as a knowledge source on the surface appears to be very fertile ground, we still have much work to do. We

believe that access to a large number of transcript will help the transcript mining process produce multi-domain models. Currently we are using data sets of less than two dozen transcripts due to the difficulty of access. We believe that the more transcripts we consider on a topic the higher the knowledge credential and payoff.

We are considering ways we can enhance our ISIS NLP system as a result of its failure to process 100% of Q&A pairs presented to it. We still see a great deal of promise in using digital transcripts to automate parts or all of the knowledge acquisition process.

References

Witherell, P., Krishnamurty, S., Grosse, I., and Wileden, J. 2010. Improved Knowledge Management Through First-Order Logic in Engineering Design Ontologies. *Journal of Artificial Intelligence for Engineering Design, Analysis and Manufacturing* Vol. 24: 245-257.

C-Y. Lu, S. 1987. Knowledge Map: An Approach to Knowledge Acquisition in Developing Engineering Expert Systems. *Journal of Engineering with Computers* Vol 3: 59-68.

Breitman, K., Casanova, M.a., Truszkowski, W. 2007. *Semantic Web Concepts, Technologies and Applications*. NASA Monographs in Systems and Software Engineering: Springer.

Hughes, C., and Hughes, T. 2009. Transcript Mining Using Epistemic Agents and Interrogative Entailment. In *Proceedings of IEEE International Conference on Intelligent Computing and Intelligent Systems*, Vol. 1, 857-861. Beijing, China: IEEE Press.

Kant, I. 1965. *The Critique of Pure Reason*. trans Norman Kemp Smith. New York: St. Martin.

Gettier, E. 1963. Is Justified True Belief Knowledge? *Analysis*, Vol.23: 121-123.

Kripke, S. 1963. Semantical Considerations on Modal Logic. *Acta Philosophica Fennica*, vol. 16: 83-94.

Hughes, T., Hughes, C., and Lazar, A. 2008. Epistemic Structured Representation for Legal Transcript Analysis. *Advances in Computer and Information Sciences and Engineering*. Springer: 101-107.

Shoham, Y. 1993. Agent Oriented Programming. *AI Journal* 60(1): 51-92.

Rao, A.S., and Georgeff, M.P. 1995. An Abstract Architecture for Rational Agents. In *Proceedings of the International Conference on Logic Programming*, 67-81. Kanagawa, Japan: MIT Press.

Wooldridge, M. 2000. *Reasoning About Rational Agents*. Cambridge, Massachusetts: MIT Press.

Kowalski, R., and Sadri, F. 1999. From Logic Programming towards Multi-Agent Systems. *Annals of Mathematics and Artificial Intelligence*, Vol 25, issue 3/4: 391-419.

Kamp, H., and Reyle, U. 1993. *From Discourse to Logic, Introduction to Model Theoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*. Netherlands: Kluwer Academic Publishers.

Brachman, R. J., and Levesque, H. J. 2004. *Knowledge Representation and Reasoning*. San Francisco, CA.: Morgan Kaufman.

Barton, G. E., Berwick, R.C, and Ristad, E.S. 1987. *Computational Complexity and Natural Language*. Cambridge, Massachusetts: MIT Press.

Blackburn, P., and Bos, J. 2005. *Representation and Inference for Natural Language*. Stanford, CA.: CSLI Publications.

Nirenburg, S., and Raskin, V. 2004. *Ontological Semantics*. Cambridge, MA.: MIT Press.

Lehnert, W.G. 1978. *The Process of Question Answering*. Hillsdale, NJ.: Lawrence Erlbaum Associates, Inc.

An Ontological Semantic Account of Relative Quantification in English

Whitney R. Vandiver

Purdue University
Linguistics Program
West Lafayette, Indiana
wvandive@purdue.edu

Abstract

This paper proposes a linguistic analysis of the semantic behavior of relative quantifiers in English, those for which an absolute value cannot be determined, with attention to the differences in properties and meaning between individual quantifiers and the semantic subclasses created by these quantifiers. Represented formally within Ontological Semantic Technology (OST), the semantic nature of such relative quantifiers is also described for computational purposes, with consideration of the related mathematical qualities of quantification that must be captured for adequate description. Among the English quantifiers considered here are *few*, *a few*, *little*, *a little*, *a bit*, *some*, *several*, *many*, *much*, *most*, *a lot*, the comparative forms of *more*, *less*, and *fewer*, compositions of combined relative quantifiers, such as *much more*, and the intensification of quantification with *very* and *too*.

Quantification in English can occur in two forms, as absolute with numerical equivalents or as relative with variable, inconsistent values that appear to be contextually dependent. Relative quantification, the focus of this paper, has been commonly treated with syntactic analyses, with quantifiers seeing little in the way of meaningful semantic descriptions. A large portion of these syntactic accounts aim at describing quantification through solely formal mathematical and logical representations (Keenan 1973; Partee 1978; Barwise and Cooper 1981; Keenan and Stavi, 1986), despite both realms failing to produce any definable subclasses or conclusive semantic properties for further application due to their inability to represent the syntactic relationships of natural language and the minor distinctions of quantifier meaning (Nirenburg and Raskin 2004). However, existing linguistic descriptions of quantifier

behavior has provided some insight into their semantics. Jespersen's earlier work (1933) is absent of the term *quantifier*, discussing instead only indefinite numerals and totality, the latter in keeping with Sapir's (1930) discussion of the meanings of *all*, with some consideration of *each* and *every*. While Jespersen's (1969) later description contributes adjectival modification (*few women*, *three students*), he shows the beginning of a semantic characterization with the note that quantifiers vary from syntactically similar adjectives in that they do not mark anything about kinds, only numbers. Quirk et al. (1985) set up a more detailed syntactic description regarding quantifiers' nominal co-occurrence but provide little toward semantic descriptions. Keenan and Stavi (1986) touch on the semantics of quantifiers in their description of natural language determiners of *some*, *several*, *few* and *most*, but view them in light of determiner behavior for a limited analysis which gives more attention to the behavior of the syntactic class rather than the semantic phenomenon of quantification. In their work on the psychology of quantifiers, Sanford, Moxey, and Paterson (1994) consider two classes of relative quantification—though, oversimplifying the task greatly—as denoting only small or large amounts, leaving middle-range quantifiers in their overlapping boundaries and awkwardly classified as a result. However, there lies in English quantification more than simple proportional comparisons and syntactic descriptions.

The simplest case of English quantification is that provided by numerals, which give exact amounts¹, and their lexical and morphological equivalents, such as singular nouns. However, some English quantifiers offer a relative quantification for which a numerical equivalent cannot be consistently established (Bradburn and Miles 1979; Routh 1994; Wright et al. 1994). Consequently, some researchers argue that the conceptual definition of a linguistic quantifier should be no more than a variable reference point (Sanford, Moxey, and Paterson 1994), eliminating the general notion of a lexical class (Nouwen 2010). Contrary to this belief, the behavior of relative quantifiers can be shown to create a cohesive netting of lexical items with similar semantic meaning. When

¹ For a similarly computational account of numerals, see Taylor et al. (2010).

analyzed purely semantically, relative quantification reveals two classes of behavior, both of which can be determined by how a relative quantificational range may vary its boundaries with respect to a domain. Each domain may be represented as a scale of possible quantification, and each possible range of a quantifier remains fixed to a scale. Such ranges create the consistent interpretations of each relative quantifier regardless of context, eliminating what Barwise and Cooper (1981) term the “fixed context assumption” (p. 163) and with which they dismiss the need to account for the variable meaning of “non-logical” (p. 163) quantifiers.

Formal Semantic Representation with OST

Ontological Semantic Technology² (OST; Nirenburg and Raskin 2004, superseded by Raskin, Hempelmann, and Taylor 2010) is the implementation of the formal representation of lexical meaning and respective word classes for computational purposes. OST results in the production of text-meaning representations (TMRs) that become the basis for reasoning and inferencing processes to imitate the meaning-driven competency of the human mind³, providing the tools and format necessary to describe natural language meaning⁴. The primary resources of OST are a language-independent ontology and a language-dependent lexicon. The structure of the former captures the relationships between events, objects, and properties of concepts to represent the world knowledge necessary in natural language understanding. As a lattice of concepts, each defined as a set of properties, the ontology uses hundreds of properties that may be combined to describe any semantic structure (sem-struct) of a lexical sense and thereby represent differences in meaning.

All lexical entries of a language are collected in a language’s lexicon, which is a representation of each word sense’s individual sem-struct, along with additional pertinent information such as a syntactic structure (syn-struct). In formal representation, each concept and its relative properties can be combined to describe different meanings so that each sense in a lexicon will have a different meaning or be represented as a synonym of another entry with identical sem-structs. These combinations of concepts and properties are given to the OST analyzer in the form of sem-structs, along with all the other information in each lexical entry, for the production of TMRs to show the possible interpretations of a text,

² See Raskin et al. (2010) for theoretical revisions and implementational deviations from initial notions.

³ The goal of OST is not to describe what the human mind accomplishes in the way of representations and processes but to represent meaning and process it with results equivalent of human abilities (Nirenburg and Raskin 2004).

⁴ For description of OST tools, see Nirenburg and Raskin (2004) for the preliminary development or Taylor, Hempelmann, and Raskin (2010) for updated methods and uses in automatic acquisition.

providing it with the information necessary to discern in meaning of the words in the input text.

The semantic properties of relative quantifiers in English are easily represented in OST by providing each quantifier with a quantificational range for describing a particular property. The scale is determined by the domain being quantified with each quantifier’s range being unique to an individual quantifier’s use so that no two quantifiers necessarily provide the same quantification. Semantically, relative quantifiers select a range of possible quantification while being imprecise about which value in that range is actually quantifying the objects, i.e. *some books* may mean two, three, or four books. These inexact values produce one of the primary obstacles for a computational analysis of scalar quantification, but this property of flexible quantification is adequately captured in OST.

Semantic Analysis of Relative Quantification

Relative quantifiers are unique in that they create vague quantifications yet are capable of being used in a variety of contexts. The possible values that a relative quantifier may represent are in part reliant on what values other relative quantifiers may represent on the same scale (in the same domain) while the relationship between each relative quantifier will always remain the same⁵, i.e. *little* involves less than *some*, which involves fewer objects than *many*. Regardless of the domain, the semantic behavior of relative quantifiers is always consistent in interpretation. Two subclasses of relative quantification may be delineated based on semantic behavior: stationary quantifiers have a definite range with unmovable boundaries, while drifting comparative quantifiers have ranges that are anchored with one endpoint and move along different values on a scale in comparison to another known value.

Stationary Quantification

Stationary quantifiers have a definite range of quantification that cannot be adjusted along a given scale of quantification. Relative quantifiers that represent the smallest amounts of stationary quantification are *(a) few*, *(a) little*, and *a bit*. It is significant to note that, regardless of their relative nature or the domain in which they are used, these lexical items will always be used at the lower end of the spectrum to communicate relatively smaller amounts in English. Semantically, *(a) few* and *(a) little* are synonyms and represent the same values of quantification, with the only difference being that the former modifies count nouns (*cars, books, students*) while the latter modifies only non-count or mass nouns (*coffee, tea, excitement*):

- (i) Mary brought a few books with her
- (ii) Mary brought a little coffee with her

⁵ This is in disagreement with Nouwen’s (2010) argument that relativity among quantifiers suggests their independency.

Additionally, both quantifiers, unlike *a bit*, may operate with or without the determiner *a*. Moxey and Sanford (1986) touch on the difference between *few* and *a few*, offering empirical evidence that the two quantifiers experience a divergence in communicating a speaker's expectations. *A few*, they argue, is a quantification without expectation, while *few* implies that a speaker expected a larger amount. Compare (i) with (iii):

- (iii) Mary brought few books with her

While (i) is a simple quantification, (iii) carries the possibility that the speaker is expressing that she expected Mary to bring more books than she did. The same difference exists between *little* and *a little*. This difference in expectation does not remove the relative nature of the quantifiers and their meaning of a relatively small amount. It is interesting to also note that *little* and *a bit* may be combined for a restriction (or perhaps, intensification) of a measurement: *a little bit*. However the reverse modification is ungrammatical: **a bit little*.

OST easily captures this behavior when it is noted that members of the stationary class have a fixed range of quantification that can be represented as a crisp (as opposed to 'fuzzy') set along a given scale. To account for the consistent interpretation of their quantification, a definite range can be determined for each quantifier, with its range remaining the same regardless of the domain. In OST, the crisp set is expressed formally by the numerical fillers of the facets EQUAL-TO, GREATER-EQUAL, GREATER-THAN, LESS-THAN, and LESS-EQUAL, in the description of the given property, RELATIVE-NUMBER in the case of (*a*) *few* and RELATIVE-AMOUNT in the case of (*a*) *little*, which represent the maximum and minimum values of a quantifier's [0, 1] range. Therefore, taking *a few* as an example:

(rel-number (greater-than (0))(less-equal (0.2)))

As this sem-struct illustrates, the semantic properties of *a few* give a relative quantification that is greater than 0 (which are the representative values for the quantifiers *no/none*) but is either less than or equal to 0.2 on the scale of quantification. It is significant to note here that the values in the sem-struct are not representative of the numerical equivalent of a quantifier but the values on a scale with which other quantifiers are compared—therefore, the ranges represent the consistent relationship that each quantifier has with other members of its class. This definite range will remain the same regardless of the domain because the fixed range is used in respect to other quantifiers on the same scale. Because *a little* and *a bit* may be used interchangeably to quantify non-count/mass nouns just as *a few* quantifies count nouns, the sem-structs of the former two will look identical because they have the

same semantic properties, differing from *a few* only in which property is quantified⁶:

(rel-amount (greater-than (0))(less-equal (0.2)))

To capture the difference in expectation created by the loss of the article, a precondition of an event is added formally, represented here as *A* and to which a comparison is made. Consequently, the sem-structs of *few* and *little* differ only in the quantified property:

(rel-number (greater (0))(less-equal (0.2))
(precondition (value (A))))
(rel-amount (greater (0))(less-equal (0.2))
(precondition (value (A))))

Similar to quantifiers of smaller quantities and amounts, the relative stationary quantifier *some* may also represent a small amount bordering on the values of *a few* or *a little*, but this quantifier may also touch the rims of larger quantities, offering a broad range of quantification. The difference in its range may be seen in comparing (vi), (vii), and (viii):

- (vi) Mary bought a few books, and John bought some
- (vii) Mary bought some books, but John bought a lot
- (viii) Mary bought some books, and John bought several

The interpretation of *some* in (vi) is that John brought more books than Mary, while its interpretation in (vii) and (viii) is that Mary bought fewer books than John. In both examples, the quantifier communicates a vague amount whose comparative range is determined by the co-occurring quantifier in the other clause. Consequently, the sem-struct for *some* is constructed using the same facets as with its smaller relative quantifiers to capture the definite range of quantification of both its use with count and non-count nouns, respectively:

(rel-number (greater-equal (0.3))(less-equal (0.6)))
(rel-amount (greater-equal (0.3))(less-equal (0.6)))

The sem-struct above, when compared with those of *a little*, *a few*, and *a bit*, reveals that the quantificational range of *some* is in a consistent relationship with other quantifiers when applied to the same scale. Additionally, the use of the conjunction in (vi) and (viii) demonstrates how *some* may share in the meaning of a small quantification closing in on the range of *few* while also sharing the boundaries of a larger quantification such as *several* in (viii). However, the disjunction in (vii) illustrates that the range of

⁶ Some distinctions in the use of *fewer/less* are disappearing in current American usage, especially in the comparative degree, as in *there are less students in my class*.

quantification must fall somewhere between the two ends of relative quantification, perhaps with overlap of what constitutes *some* and what constitutes *a lot*. Interestingly, the grammatical combinations of conjunction and disjunction do not exhibit the same patterns between quantifier clauses, with conjunction always being grammatical while disjunction has the restriction of requiring that the comparative quantifier ranges not be overlapping:

- (ix) Mary bought a few books and/*but John bought some books
- (x) Mary bought several books and/*but John bought some books
- (xi) Mary bought some books and/*but John bought a few books⁷
- (xii) Mary bought some books and/but John bought many/several/a lot of books

As (ix)-(xi) shows, *some* may not be used in disjunction with other quantifiers if their ranges have overlapping values, except in constructions like (xii) where the overlap occurs with a higher range.

These examples show that *some* quantifies a relatively broad range of possible amounts—something greater than *few* but commonly less than *many* or *a lot*. This scalar property allows for overlap to occur between the ranges of different quantifiers. In other words, there is no definitive answer for where the range of *a little* ends and the quantification of *some* begins, and this semantic feature must be accounted for computationally.

This is accomplished with the RELAXABLE-DOWN-TO (REL-DOWN-TO) and the RELAXABLE-UP-TO (REL-UP-TO) facets⁸, which create an overlap of quantifier ranges and allow an extension of a definite range⁹. Therefore, a *few* becomes

(rel-amount (greater-than (0))(less-equal (0.2))
(rel-up-to (0.3)))

Its range is therefore allowed to extend to a larger value to compensate for the quantifier's use beyond its definite range to account for the inability to distinguish, even within a single domain, the exact endpoint of a relative quantifier. Likewise, the same facets may be used with *some*:

(rel-amount(rel-down-to(0.2))(greater-equal(0.3))
(less-equal (0.6))(rel-up-to (0.7)))

⁷ The reading of (xi) with disjunction may be made grammatical with the addition of “only” before the quantifier, which restricts the quantifier range so that it does not overlap with *some*.

⁸ These facets are based on the RELAXABLE-TO facet (Nirenburg and Raskin 2004), which allows the domain of a property to include concepts that are uncommon fillers or contradict the ontology but which may occur in natural language.

⁹ For a discussion of how OST computes imprecise and/or fuzzy semantic information, review Taylor and Raskin (2010), and Raskin and Taylor (2009).

With the relaxable facets, the ranges of *a few* and *some* overlap with the range of [0.2, 0.3], meaning that they both may quantify within this range.

Some might be thought of as having the greatest relativity in quantification because of its two possible interpretations, such as those with (xiii):

- (xiii) I know John bought some books, but I don't know how many

In the case of (ix), rather than quantifying how many books John bought, the speaker is meaning that she knows John purchased books with no idea of how many. *Some* is instead being used to assure the listener of John's purchase. Here, a new sense of *some* is used with only a minimum limit, at least one, which is also easily represented as a simplification of its sem-struct above:

(number (greater-equal (1)))¹⁰

The ability to distinguish which interpretation is correct with the use of *some*, the remainder of the discourse must be taken into consideration, such as the second clause in (xiii). Such information identifies the attitude of the speaker toward the information and may provide clues as to which sense of the quantifier is meant.

Still within the class of stationary relative quantifiers, we find larger quantifiers with *several*, *many*, *much*, and *a lot*. These quantifiers represent greater values that border on the higher end of *some* but stop just shy of *all*. As with (*a few* and (*a little*), the differences between *several* and *many* with *much* is the matter of quantifying count and non-count/mass nouns:

- (xiv) Many/several students attended the workshop
- (xv) Much attention was paid to the issue

However, *a lot* may be used to quantify both count and non-count nouns and, therefore, may replace the quantifiers in both (xiv) and (xv). Again, keeping with the same facets, the sem-structs for *several*, *many*, *much*, and *a lot*, respectively, are represented below:

(rel-number(rel-down-to (0.5))(greater-equal(0.6))
(less-equal (0.7))(rel-up-to (0.8)))
(rel-number(rel-down-to(0.6))(greater-equal(0.7))
(less-equal (0.8))(rel-up-to (0.9)))
(rel-amount(rel-down-to(0.6))(greater-equal(0.7))
(less-equal (0.8))(rel-up-to (0.9)))
(rel-number(rel-down-to(0.7))(greater-equal(0.8))
(less-equal (0.9))(rel-up-to (0.95)))

¹⁰ NUMBER as opposed to RELATIVE-NUMBER is the property for absolute quantification, used here to show that exactly one object is being quantified.

The quantifier *most* is sometimes included as a relative quantifier but behaves slightly differently than other relative quantifiers, in that *most* is a proportional quantifier. Compare (xiv) with (xvi):

(xvi) Most students attended the workshop

This quantifier is synonymous with the meaning of *the majority*. While it is still relative, *most* acts differently because it creates a proportion of another set. In the comparison of (xiv) with (xvi), both give a larger quantification, but *most* gives a meaning that a relative quantification of a set is being given for the creation of a new, smaller set. Other relative quantifiers, such as *many*, do not have this proportional property. Regardless of this difference, *most* is still capable of being represented similarly in OST with the variable range boundaries by making them relative to the larger known set, the domain:

$(\text{rel-number}(\text{rel-down-to}(0.51))(\text{greater-equal-to}(0.75))(\text{less-equal}(0.95))(\text{rel-up-to}(0.99)))$

As the sem-struct shows, *most* creates a range of at least 75% up to a maximum of 99% of a larger known set—which allows it to apply to both absolute cardinal values, such as *most of the forty*, and relative quantities, such as *most students attended*—relaxable down to 51% of the set. Relating the sem-struct to (xvi), the domain would be the number of possible students that could have attended the workshop, with *most* representing a large proportion of this set for the creation of a new set—those students who did attend the workshop. Depending on the domain, its cardinality may be stated explicitly in the text and added in during processing for an exact value of the domain, or it may be implicit to result in an implied value of the domain.

Additionally, English quantifiers may be intensified semantically with the modifier *very* to strengthen the quantification and further restrict the range of possible values, i.e. *very little*. Two phenomena are worthy of note regarding the use of intensifiers with quantifiers. First, *very* does not result in an increase of quantification when combined with all relative quantifiers and may be used with some quantifiers and negation to lessen quantification:

- (xvii) We had very few students attend the workshop
- (xviii) We didn't have very many students attend the workshop
- (xix) ?We had very many students attend the workshop
- (xx) Did very many students attend the workshop?

As (xvii) illustrates, the intensification of *very* on a negative quantifier¹¹ results in a lesser quantification (fewer objects than simply *few*), while *very* creates a greater quantification with a positive quantifier (more objects than simply *many*) as in (xix). The occurrence in (xix), however, is awkward without an emphatic stress and is more acceptable with the replacement of *a lot*, which may be interpreted as a higher quantificational range equivalent to an intensified *many*. The intensification within a question, as in (xx), is grammatical, while the intensification within the scope of negation reduces the value.

This intensifier behaves differently depending on whether it is acting on a positive or negative quantifier, but its semantic behavior is consistent for both situations¹² and is, therefore, captured in the sem-structs below representing the intensification of negative and positive intensification, respectively:

$(\text{rel-number}(\text{value}(\text{\$var1}^2)))$
 $(\text{rel-number}(\text{value}(\text{\$var1}^{(1/2)})))$

The description of quantifier intensification introduces mathematical operations into the sem-structs, primarily square and square root. The intensification of quantifiers is dependent on the polarity of the intensified quantifier (Vandiver, forthcoming) which is what determines whether a quantifier is termed positive or negative. The use of the square and square root functions intensify the polarity of a quantifier on its given scale of quantification.

As shown in sem-structs, when a negative quantifier is intensified, the original value of the greater values in the range are restricted by squaring their original values. The intensification of positive quantifiers, however, restricts the lesser range values by taking the square root of such values. Examples of the resulting ranges in TMRs are shown in the last section of this paper.

Secondly, the modifier *too* may also be used with relative quantifiers but, while making use of the same mathematical operations, creates a different meaning than the intensification of *very*. Rather than simply restricting the range of quantification, *too* means that quantification occurs by surpassing a limit or expectation, either a minimum or maximum, as with (xxi) and (xxii):

- (xxi) Too many students registered for the workshop

¹¹ The terms *negative and positive quantifier* are used here to refer to the polarity of a quantifier's range in respect to a given scale, which has a direct relationship with a quantifier's intensified range. A negative quantifier is one whose range represents values lesser than those of *some*, while a positive quantifier is one whose range represents values greater than the same boundary. This is reinforced by the inability for *some* to be intensified due to its overlap with lesser and greater ranges, as is also exemplified in relative frequency quantifiers, such as *sometimes*.

¹² This treatment acknowledges Zadeh's (1976) analysis of *very* as the square of the initial linguistic value; however, it also adds the square root calculation for positive quantifiers, which Zadeh says is specifically for the behavior of *more* and *less*.

- (xxii) We have too little coffee left to give everyone a refill

However, this pre-determined limit is contextual and does not affect the semantic structure of the intensifier; therefore, the sem-struct for *too* will be similar to that of *very* and make use of the same mathematical operations, only in reverse relationships with negative and positive quantifiers, respectively, as shown below:

(rel-number(value(var1^{1/2})))
(rel-number(value(var1²)))

The sem-structs above for *too* show only the calculation of the amount, while the comparison with the known limit will be reflected in the TMR. In this way, *too* and *very* have similar ways of restricting quantification, but *too* does so in relation to a known limit or expectation.

Finally, debunking the classification of some quantifiers as adjectives (Jespersen 1933, 1969; Quirk et al. 1985), not all quantifiers that are used with adjectives may be used with quantifiers, such as *really*, *entirely*, *totally*, *absolutely*, *utterly*, or *completely*.

Drifting Comparative Quantification

English also has relative quantification that is not only contextually variable but is created in comparison to another known amount rather than in respect to other vague quantifiers, creating the class of drifting comparatives. This is accomplished with *more (than)*, *less (than)*, and *fewer (than)*, which create a relative range by comparing the unknown value to an already established value. As above, there is a distinction in use between quantification of count and non-count/mass nouns:

- (xxiii) John drank less tea than Mary
(xxiv) John ate fewer cookies than Mary
(xxv) John drank more coffee/cups of coffee than Mary

Additionally, *more (than)* may quantify both kinds of nouns, as in (xxv).

Drifting quantification continues to modify the property of relative quantification as with stationary quantifiers and is represented with the same range-defining facets. The difference with this class is that the range of quantification may have one endpoint that is moved along the scale because the values are established in comparison to another value. This predetermined value, represented here as B, designates the only anchored value of a comparative quantifier's range and will be readjusted with each new domain. Therefore, the sem-struct of the comparative *more* is formulated in reference to the value of B:

(rel-number (greater-than (B)))

This sem-struct describes the value of *more* as being anchored at the known value of B and having a range of

any value greater than this variable. Depending on the value of B, the minimum endpoint of *more* will vary with each domain. In this way, *more* is a drifting quantifier because its range of possible values drifts along the scale depending on where the endpoint is anchored. For example, the anchored values of B in (xxvi) and (xxvii) represent different values on which *more* is anchored:

- (xxvi) Mary bought more books than John
(xxvii) Greta drank more tea than Leo

In (xxvi), B represents the number of books bought by John, and it becomes the minimum value after which the range of *more* will begin. Likewise, in (xxvii), B represents the amount of tea drank by Leo, which becomes the anchoring minimum value for *more*. Because the range of *more* anchors at its lowest end and quantifies upward, it is a positive quantifier.

Similarly, the sem-structs of *fewer* and *less* are also anchored by a known value but are in a reverse relationship with B. Instead of being anchored to a minimum amount, they are anchored to a maximum amount with B. The sem-structs for *fewer* and *less*, respectively, are shown below:

(rel-number (less-than (B)))
(rel-number (less-than (B)))

In this way, the difference in quantification in (xxviii) and (xxix) is accomplished by anchoring the range of *less* and *fewer* to different amounts, in addition to their difference in quantification of count and non-count/mass nouns.

- (xxviii) Mary bought fewer books than John
(xxix) Greta drank less tea than Leo

Anchoring them on their highest amounts, these two quantifiers act as negative quantifiers. In this way, the behavior of drifting quantifiers may be seen as a recalculating of one end point of their ranges while maintaining the mass inclusion of the remaining values of one side of this anchored value.

Interestingly, as (xxx) and (xxxi) exemplify, drifting quantifiers have the ability to maintain their comparisons within the phenomenon of simple syntactic ellipsis, a common occurrence in English:

- (xxx) Mary brought more books
(xxxi) Mary brought more books than John

Ellipsis is the process by which information is omitted from a sentence and left to be filled back in by contextual information. With (xxx) and (xxxi), *more* is building a larger value on top of the known value, the amount of books brought by John. The accompaniment of *than* signals that the known value of B is explicitly stated in the sentence; consequently, the *than* phrase is optional in English in such constructions, and its absence has no direct

effect on the meaning of the quantifier or its formal representation.

Composites

Relative quantifiers may also combine with each other for the creation of composites:

- (xxxii) Chicago received much more snow than Boston
- (xxxiii) A few of the many students who enrolled in the course submitted their papers early

These combinations allow for the quantification of another quantifier's range. Such composites maintain the relative quality of providing an imprecise amount or value, though they produce the most intense form of relative quantification in English. This semantic phenomenon occurs when one quantifier is established and then is either restricted or intensified by another relative quantifier, similar to intensification with *very*. In other words, the same quantification as above occurs, but one quantificational range is modifying another for a double-quantification rather than a restricted single range. Individual sem-structs for composites are not needed because each quantifier will have its own sem-struct, and the composite aspect will be captured in the TMR.

Text Meaning Representation of Quantification

The quantification of objects in English is translated into a TMR for sentential meaning. Given *The woman bought a few books* and *The woman bought several books*, the analyzer will produce the following TMRs to distinguish between the differing relative quantification:

```
(buy(agent(sem(human(gender(value(female))))))
(theme(sem(books(rel-number (greater-than(0))(less-
equal(0.2))(rel-up-to(0.3)))))))))
```

```
(buy(agent(sem(human(gender(value(female))))))
(theme(sem(books(rel-number(rel-down-to(0.5))(greater-
equal(0.6))(less-equal(0.7))(rel-up-to(0.8)))))))))
```

Conclusion

Despite arguments that quantification cannot be described as a cohesive class, a purely semantic account of quantification in English is captured here by the Ontological Semantics Technology. Their behavior can be described consistently as the determination of overlapping ranges of quantification within a given domain with each relative quantifier maintaining a consistent relationship with the next or a single anchored point. Range restrictions can be imposed by intensifiers, and the strongest quantification may occur as compositions of quantifiers. While this analysis focuses on English, relative

quantification in other languages are likely to be found to exhibit similar behavior and may be described similarly in Ontological Semantics Technology formal representation for computational purposes.

References

- Barwise, J. and Cooper, R. 1981. Generalized quantifiers and natural language. *Linguistics and Philosophy* 4: 159-219.
- Bradburn, N. M. and Miles, C. 1979. Vague quantifiers. *Public Opinion Quarterly* 43(1): 92-101.
- Brems, L., and Davidse, K. 2003. Absolute and relative quantification: Beyond mutually exclusive word classes. *Belgian Journal of English Language and Literatures (BELL)*: 49-60.
- Jespersen, O. 1933. *Essentials of English Grammar*. New York: Henry Holt and Company.
- Jespersen, O. 1969. *Analytic Syntax*, The Transatlantic Series in Linguistics, Ed. Samuel R. Levin. New York: Holt, Rinehart and Winston, Inc.
- Keenan, E. 1973. *Formal semantics of natural language: Papers from a colloquium*. New York: Cambridge University Press.
- Keenan, E., and Stavi, J. 1986. A semantic characterization of natural language determiners. *Linguistics and Philosophy* 9(3): 253-326.
- Moxey, L. A., and Sanford, A. J. 1986. Quantifiers and focus. *Journal of Semantics* 5: 189-206.
- Nirenburg, S. and Raskin, V. 2004. *Ontological Semantics*. Cambridge: MIT Press.
- Nouwen, R. 2010. What's in a quantifier?, in *Theoretical Validity and Psychological Reality*. Eds. M. Everaet, T. Lentz, H. De Mulder, O. Nilsen, and A. Zondervan. Benjamins. Forthcoming.
- Partee, B. H. 1978. *Fundamentals of mathematics for linguistics*. Stanford: Greylock.
- Quirk, R., Greenbaum, S. Leech, G., and Svartvik, J. 1985. *A Comprehensive Grammar of the English Language*. England: Pearson Education Limited.
- Raskin, V., Hempelmann, C. F., and Taylor, J. M. 2010. Guessing vs. knowing: The two approaches to semantics in natural language processing. In *Proceedings of Dialog 2010*. Moscow, Russia.

Raskin, V., and Taylor, J. M. 2009. The (not so) unbearable fuzziness of natural language: The ontological semantic way of computing with words. In 28th International Conference of the North American Fuzzy Information Processing Society. Cincinnati, Ohio.

Routh, D. 1994. On representation of quantifiers. *Journal of Semantics* 11: 194-215.

Sanford, A. J., Moxey, L. M., and Paterson, K. 1994. Psychological studies of quantifiers. *Journal of Semantics* 10: 153-170.

Sapir, E. 1930. Totality. *Language* 6(3): 7-28.

Taylor, J. M., Hempelmann, C. F., and Raskin, V.. 2010. On an automatic acquisition toolbox for ontologies and lexicons. In Proceedings of International Conference on Artificial Intelligence. Las Vegas, Nevada.

Taylor, J. M., Raskin, V., Hempelmann, C. F., and Vandiver, W. R. 2010. Computing the meaning of number expressions in English: The common case. In Proceedings of International Conference on Artificial Intelligence. Las Vegas, Nevada.

Taylor, J. M. and Raskin, V. 2010. Fuzzy ontology for natural language. In Proceedings of the 29th International Conference of the North American Fuzzy Information Processing Society. Toronto, Canada.

Vandiver, W. R. Forthcoming. The Ontological Semantics of Quantifiers in English. PhD Dissertation. Purdue University.

Wright, D. B., Gaskell, G. D., O'Muircheartaigh, C. A. 1994. How much is "quite a bit"? Mapping between numerical values and vague quantifiers. *Applied Cognitive Psychology* 8: 479-496.

Zadeh, L. A. 1976. A fuzzy-algorithmic approach to the definition of complex or imprecise concepts. *International Journal of Man-Machine Studies* 8: 249-291.

This page is intentionally left blank.

Intelligent Systems

Chair: Dana Vrajitoru

Spatiotemporal knowledge representation and reasoning under uncertainty for action recognition in smart homes

Farzad Amirjavid, Kevin Bouchard, Abdenour Bouzouane, Bruno Bouchard

Department of mathematics and computer science
555, university boulevard, Chicoutimi, Quebec, Canada G7H2B1, University of Quebec At Chicoutimi (UQAC)
farzad.amirjavid@uqac.ca
Kevin.bouchard@uqac.ca
Abdenour_bouzouane@uqac.ca
Bruno_bouchard@uqac.ca

Abstract

We apply artificial intelligence techniques to perform data analysis and activity recognition in smart homes. Sensors embedded in smart home provide primary data to reason about observations and provide appropriate assistance for residents to complete their Activities Daily Livings (ADLs). These residents may suffer from different levels of Alzheimer disease. In this paper, we introduce a qualitative approach that considers spatiotemporal specifications of activities in the Activity Recognition Agent (ARA) to do knowledge representation and reasoning about the observations. In this paper, we consider different existing uncertainties within sensors observations and Observed Agent's activities. In the introduced approach if the more details about environment context be provided, the less activity recognition process complexity and more precise functionality is expected.

1 Introduction

Smart home mostly addresses the health-care problem of performing automated assessment of functional health for elder adults and provision of automated assistance that will allow people suffering from Alzheimer to remain independent [16]. In order to live independently at home, adults need to be able to complete key activities of Daily Living, or ADLs, however tracking of ADL accomplishment is a time consuming task for caregivers. To provide automated assistance we apply Activity Recognition Agent (ARA) to reason about observations provided by the embedded sensors in Smart Home. In this paper, we deal with the activity recognition process performing in Activity Recognition Agent (ARA). Event Recognition Agent (ERA) detects realized events and report them to the ARA. ARA provides a report for the Plan Recognition Agent (PRA) about observed and inferred activities and finally the Assistance Provision Agent (APA) would provide appropriate assistance for the Observed Agent (OA). The schema1 shows the general process in the smart home.

Although uncertainty and imprecision is included always with the action recognition field, in most of the performed researches up to now [1,6,11,13,14,15,16,23,24] the existing uncertainty and imprecision in OA's behavior and home state is not considered and they are not robust if activity realization models change. Furthermore, any small change in sensors network, sensors locations and sensors number could lead to restricting all their applied models and all the previous training tests would not be useful any more. Moreover, objects movement, which provides important information in activity recognition, has not been considered.

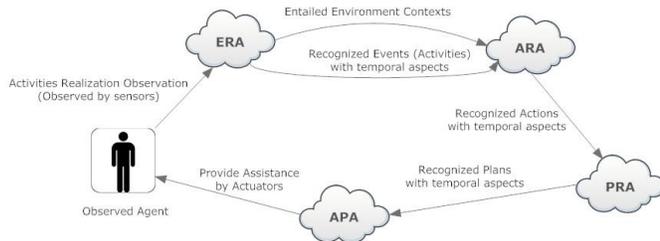
Most of the surveyed activity recognition approaches do not tolerate relatively detailed information about the real world and even they may avoid more sensors for not to receive complementary information about the activities. The reason is that increase in number of applied sensors could lead to process complexity and they would need a huge dataset for training. I contrast, the introduced approach in this paper welcomes the increase in input information and in the case of change in sensors network structure, and the old knowledge would be still valid. Furthermore, the increase in provided information would even cause to decrease in process complexity.

In this paper, we are explaining an intelligent agent that tries to explain the observations and detects anomalies in the case that there is no explanation. Applied knowledge representation and reasoning techniques that benefit from activities temporal and spatial specifications is discussed and we introduce fuzzy contexts that can briefly indicate the home state and possible events that could occur in contexts.

The art of ranking and classification between generated hypotheses inferred from available knowledge and present observations can lead to better adjustment between system's inference and the real world. In this way, reasoning can be less complicated and so it causes less

error to choose the right decision in decision-making process.

A brief explanation about general activity recognition process is that after that ERA provides ARA the current home state and happened events in fuzzy context and fuzzy events frame (knowledge representation), the possible hypotheses through time line are generated and ranked dynamically. Then in the reasoning process, the explanations about observations would be provided.



schema1- the general smart home process model

2 Knowledge Representation

A knowledge representation system is applied to interpret sentences in the logic in order to derive inferences from them. When we design a knowledge representation system, we have to make choices across a number of design spaces. The single most important decision to be made, is the expressivity of the KR. Our desire is to include more effective parameters in action recognition process who may make the knowledge representation enough expressive and may make the reasoning process not so relatively difficult. Brahman and Levesque [1984] introduced the mentioned desire as contradictory goals; however, we believe that applying fuzzy context can lead to more expressiveness and simpler reasoning in an intelligent agent. That is because fuzzy context holds more details at one hand and at the other hand the defuzzified context prevents to generate many relatively similar contexts that can make the reasoning process complicated. Here we introduce two key knowledge types and their representation methods.

2.1 Environmental parameters or context items

Embedded sensors in the smart home provide primary data for the Activity Recognition Agent (ARA). The received data by sensors that is raw and unprocessed introduce the environmental components (such as temperature, doors state, heater state, Observed Agent’s position, etc) that may be effective on action recognition process. In fact, the mentioned components form the body of contexts and are named as context items. Unfuzzy context is a context that is constituted from a set of items and we define fuzzy context as context constituted from fuzzified items.

2.2 World state and fuzzy context

“Fuzzy Context” is the term used to express the home state with it. In this way, environmental parameters (called *items* and indicated by i_x) are measured and then fuzzified by fuzzy membership functions. To express a general form of fuzzy context, we apply the following form:

$$\tilde{C}(\tilde{i}_1, \tilde{i}_2, \dots, \tilde{i}_n)$$

Temperature is an example for item. For instance, when the thermometer indicates 37 degree it can be inferred that it belongs to “Warm” class (applying fuzzy roles and defuzzification functions) and finally warm is reported instead of 37 degree. Home state is finally formed by such this information. As a simple example for home state, consider a home that includes some embedded sensors to indicate the home state. These sensors indicate “OA location”, “door state”, “heater state”, “oven state” and “temperature”. Mentioned sensors generate continuously values along time axis. The following indicates the final defuzzified home state:

$$C_{obs} (OA : at_oven, door : closed, heater : off, Oven : off, temperature : warm)$$

2.3 Events

We define *events* as each meaningful change in sensors generated values. ERA simply receives generated values from the sensors and checks whether the value belongs still to a new class. A change in received values class means an event has happened and the event is reported to the ARA.

2.4 Discussion

Allen temporal logic is a famous temporal logic that introduced thirteen temporal relations between actions. Morchen argued that Allen’s temporal patterns are not robust and small differences in boundaries lead to different patterns for similar situations [2]. Furthermore, the complexity increases if the OA performs multiple actions simultaneously. Moreover, it does not also indicate the actions beginning and terminating moments.

From the mentioned problems, we have inspired the idea that we can consider the beginning event (temporal point) instead of interval consideration and so in this way, it would be necessary just to compare beginning points of actions and their durations would be justified as their components that contain fuzzy, relative and estimative measures as value. So, in brief it can be said that only the before relation would be considered and the possible moments that other actions can begin on them.

To implement the mentioned idea we have applied the possibility theory that was first introduced by Zadeh[7,8,9,10]. In summary, it is assumed that after observation of an event, all the possible actions can begin simultaneously and the most possible moments for events occurrence is indicated. The farther from *most possible*

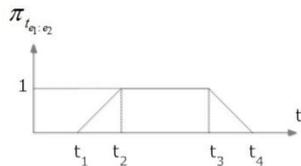
occurrence moments the less ranking value in hypotheses ranking introduced in “3.3” section.

The result is that multiple simultaneous running actions can be considered and it is enough flexible to consider different possible temporal relations between actions and gives an estimation (by defuzzifying the fuzzy time up to next Action’s beginning moment) to predict the action termination moment.

For example, for the action entering to the kitchen, the table1 indicates the possible events (actions beginning points) that are possible to occur after previously assumed occurred events and their possible occurrence moments.

$t_0 \backslash t_1$	kitchen_door_open	kitchen_door_close	temperature_increase	temperature_decrease	heater_on	heater_off
kitchen_door_open	0	1	0	0.9	0	0
kitchen_door_close	0	0	0	1	1	0
temperature_increase	0	0	0	0	0	1
temperature_decrease	0	0	0	0	1	0
heater_on	0	0	1	0	0	0.2
heater_off	0	0	0.3	0.7	0	0

Table1. Possibility distributions for relations between events for action “entering to the kitchen”



Schema2. Possibility distributions for occurrence moments

In the table1 the possibility distribution for the “before” relation is indicated by the normalized numbers (from 0 to 1) and in schema2 the possibility distribution for possible occurrence moments of the next event is indicated by

$\pi_{t_{e1}, e2}$ which is a trapezoid fuzzy number. In this digit t_1 is the soonest moment that event2 can occur after another event1, moments between t_2 and t_3 are the most possible moments that event2 can occur and t_4 is the latest moment that event2 can occur. (We have forborne to include the necessity distributions in our calculations, which is already dependent to the possibility distributions.)

The table1 is implemented as a data table in database and it indicates the effective environmental parameters to recognize the action “entering to the kitchen”.

2.5 Temporal Knowledge Representation

We define the term temporal knowledge as a kind of knowledge that is dependant to the time and may lead to different inferences in different temporal contexts;

however, this knowledge can include some temporal information about next possible contexts that can possibly happen in future. We refer to the first introduced type as absolute time and the second one as relative time.

To represent temporal dependency (absolute time), we insert a new item to the fuzzy context ontology that is called fuzzy time item. In this way, contexts for similar conditions but different temporal conditions are made. A function is implemented to check whether the current time is adjustable to the defuzzified *time item* existing in the fuzzy context.

Time elapse as a possible *fuzzy event* is also applicable. An example for defuzzified item of fuzzy time can be like “morning”.

To represent temporal information (relative time) using a fuzzy trapezoidal digit, we indicate the possible transition moments to different possible contexts and it is implemented by a simple table containing the concerning data. This relative data is converted to the real time at the running time.

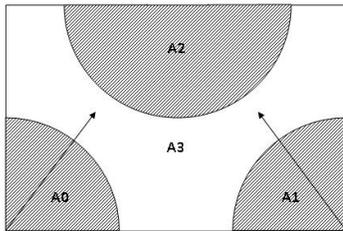
2.6 Spatial Knowledge Representation

Another key knowledge that is helpful to do better reasoning is spatial knowledge that indicates the context dependency to the objects locations. As it was mentioned earlier, movement of objects in the real world provides noticeable information for the activity recognition process. There can be considered two general spatial knowledge forms. The first one, which would be referred to as absolute position, indicates the objects positions in the real world and the second one that would be referred to as relative positions indicates the position of objects to each other. In the fuzzy context, a section is dedicated for the objects positions in the home (first spatial knowledge type) and the second spatial knowledge type is indicated in the Event Recognition Agent (ERA). One example for absolute position application in activity recognition is that, to infer the *cooking activity* it is necessary to observe the pan on the oven. An example for the relative position inference is that if approach of pot to glass be observed it can inferred that OA has fulfilled the glass with the pot’s containing liquid such as coffee. ERA provides this information as recognized event for the ARA.

2.6.1 Discussion

To recognize objects movements we have applied RFID tags and antennas. This process is done in ERA and we would have a short introduction of it in here.

In a brief description, we have attached RFID tags on the objects and used RFID antennas to recognize the OA's activities. We have made a program in Java to recognize the performed activities by the OA. Every six microseconds applied RFID antennas check the environment to detect the RFID tags. By having just one RFID antenna and attaching RFID tags on the objects, we are able to recognize if the object is close or far from the antenna. By adding the second antenna, we would be able to make four regions. The first region is the region around the first antenna, the second region would be around the second antenna, and the third region is the region in front of both antennas and the region that both antennas show equal signal strength to detect the objects and the fourth region is the region that no antenna can easily detect the object (see schema3).



Schema3. Regions defined by RFID antennas

In ERA, entering and exiting a region is recognizable by the available equipments and the concerning events are reported to the ARA. In the absolute position recognition, it's enough to find the object's location in one of the mentioned regions, however in relative position recognition we should find two target objects in one region.

2.7 Spatiotemporal Knowledge Representation

Spatiotemporal knowledge is key environmental information to do activity recognition; however, there is other effective environmental information such as temperature, door's position and other items that are also useful for controlling affairs in smart home. to represent such this knowledge we have divided fuzzy context into three major sections. One section for temporal knowledge, another section for spatial knowledge and third section for controlling items is provided.

Introduced fuzzy context let us consider different knowledge types in action recognition and the controlling affairs (using checking functions) are done at the transition moments. Transition between contexts is also indicated by the observed fuzzy events reported by ERA.

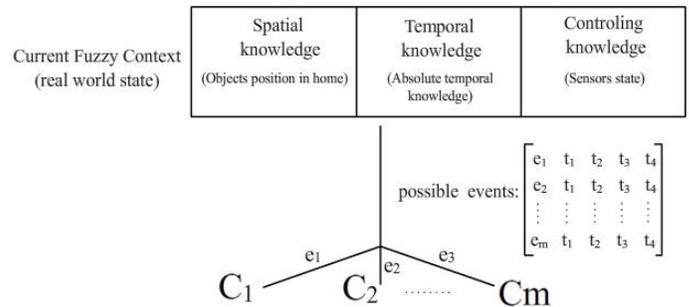
3 Reasoning

The reasoning process in activity recognition follows the observation, hypothesis generation and hypothesis pruning steps.

3.1 Hypothesis Generation

Hypotheses are generated only in the case of event recognition reported by the ERA. Movement of objects, elapse of time and a switch in controlling sensors states are possible observable events. In fact, the generated hypotheses indicate the possible future contexts could possibly be observed in the future.

The hypothesis generation process in summary is that at first hypotheses are generated based on a table named as possible fuzzy events (see table1) that could have been generated in future in the current context. At the second step, they are assigned the possible observation moments by the use of trapezoidal fuzzy digit (see schema2) and finally they are ranked or weighted (see part 3.3). the mentioned process is illustrated in schema4.



Schema4 hypothesis generation

3.2 Hypothesis Generation through the time line

Considering uncertainties for unrecognized but in reality happened events (there are several reasons for it), it is possible that it defects the reasoning process and so ARA wrongly detects normal actions or activities as anomaly. To improve the activity recognition efficiency we consider that possible events may have happened but not observed and they are generated and pruned through the time line. A question that may arise in here is that what could be the occurrence time of undetected event? The answer is that the defuzzified value of the fuzzy trapezoid number can indicate the possible moment that the event has happened. In the case of anomaly detection, it would be checked whether there have been no undetected event and there is no previously generated hypotheses that can explain the occurred events.

3.3 Hypothesis Ranking

When new hypotheses are generated, they are inserted as tree leaves (we can call it also decision tree) and then they are ordered by defuzzified occurrence moment from left to right. To describe briefly the ranking process, we assign each observed and proved a higher point and in contrast unobserved or not yet proved hypotheses are assigned lower points.

The rank and weight of generated hypotheses ($w_i(t)$) can change dynamically by elapse of time. The primary assigned weight is derived from the possibility distribution for occurrence of event ($\pi_{e_1:e_2}$ existing in table1) and as the fuzzy trapezoid number affects it, so by elapse of time it can differ to the past weights ($\pi_{t_1:e_2}$, schema2). The third parameter to affect the hypotheses rankings is the possibility distribution of the upper node occurrence (W_u). Finally, γ affects the ranking value. γ is a value that is resulted from a trade-off between smart home precision in event detection and uncertainties about behaviours of Observed Agent (OA) or in other words Alzheimer severity degree. At one side, the more severity in Alzheimer illness the less confidence on the OA and at the other side the more precision in event recognition, the more confidence on the reports and so it would be less necessary to trace the tree down to a lot of levels. The ranking formula is indicated as:

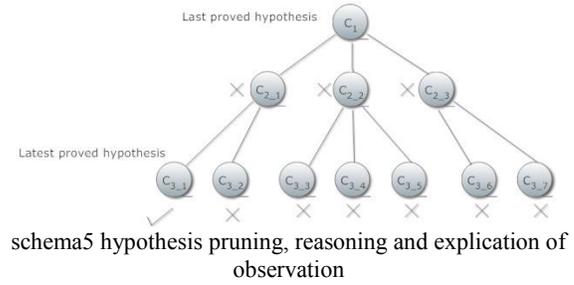
$$w_i(t) = \gamma \cdot W_u \cdot \sqrt{\pi_{e_1:e_2} \cdot \pi_{t_1:e_2}}$$

3.4 Hypothesis Pruning

To prevent the increase in number of less possible hypotheses, pruning is necessary. Pruning is applied in the case of low possibility distribution of event occurrence. In addition, observation of a possible event that could have happened calls the pruning function¹. Another way is to

¹ To estimate the closeness of new observation to the previously generated and assumptive hypotheses, we applied the $\pm \frac{1}{\gamma}$ formula to check the difference between values of the observed and assumptive context items. If all the differences between all the items be more than $\pm \frac{1}{\gamma}$ then no explain is found.

limit the pruning to a fix number of levels. Whenever a hypothesis be proved, the concerning weight for that node is assigned one.



In the schema5 the sequence of C1 and C2_1 and C3_1 indicate an explication about the latest observations.

3.5 Reasoning Process

Our goal in reasoning process is to find an explication that can explain the observations. Observation of a fuzzy event is a good reason to decide whether there are anomalies or not. However, the more OA be conscious the more rely on unproved hypotheses. The sequence of observed events can explain the current activities and actions. Furthermore, the contexts can explain the precedence of home states. So, recognition of current context from the previously generated hypotheses can well explain the observations and current activity(ies). Whenever no explanation for the observation is found or the explanation does not include minimum acceptance weight (dependent to γ), so the observed action would be recognized as abnormal action.

4 Implementation and Conclusion

The ARA was implemented in VB net environment and it was simulated in SIMACT [27]. The activity “entering to the kitchen” was simulated in different scenarios (but the same old embedded sensors) and some uncertainties in event recognition (see picture1). Anomaly detection would not be better than 50% done if the unproved hypotheses grow deeper than three levels in decision tree. In spatial reasoning it can be said that the more antennas be applied, the more precise hypotheses would be generated. It can be inferred that in the introduced approach, in the case of increasing the sensors number, more precise hypotheses would be generated and proved. Fuzzy context at one hand can express well the real world state and it can decrease reasoning complexity if it be well defuzzified.



Picture1- the activity “entering to the kitchen” simulation in SIMACT

5 Future Works

We recommend the interested researches to survey the Activity Recognition in the case of multiple residents in smart homes and also to introduce an optimization model for fuzzy roles to decrease the activity recognition mistakes.

6 References

- 1- Bruno B., Bouzouane A., Giroux S.: A Keyhole Plan Recognition Model for Alzheimer's Patients: First Results, *Journal of Applied Artificial Intelligence (AAI)*, Taylor & Francis publisher, Vol. 22 (7), pp. 623-658, July 2007.
- 2- Didier Dubois and Eyke Hullermeier, comparing Probability Measures Using Possibility Theory: A Notion of Relative Peakedness, *International Journal of Approximate Reasoning*, 2007.
- 3- D. Dubois, H. Prade and R. Sabbadin (2001) Decision-theoretic foundations of possibility theory. *Eur. J. Operational Research*, 128: 459-478.
- 4- D. Dubois and H. Prado. *Possibility Theory*, Plenum Press, New York, 1988.
- 5- D. Dubois, H.T. Nguyen and H. Prade, Fuzzy sets and probability: misunderstandings, bridges and gaps. In: D. Dubois and H. Prade, eds, *Fundamentals of Fuzzy Sets*. Boston, Mass: Kluwer, 343-438, 2000.
- 6- Cook, D.J. Youngblood, M. Heierman, E.O., III Gopalratnam, K. Rao, S. Litvin, A. Khawaja, F. MavHome an agent-based smart home, *IEEE Computer Society* Washington DC, USA, 2003.
- 7- Probability measures of fuzzy events, L.A.Zadeh, *Journal Math. Anal. Appl.*, vol 23, pp. 421-427, 1968.
- 8- Fuzzy Sets as a basis for a theory of possibility, L.A. Zadeh, *Fuzzy Sets and Systems*, vol. 1, pp. 3-28, 1978.
- 9- *Possibility Theory*, D.Dubios, H.Prade, Plenum Press, 1988.
- 10- Fuzzy sets and probability : Misunderstandings, bridges and gaps, D.Dubois, H. Prade, *Proc. of the Second IEEE Inter. Conf. on Fuzzy Systems*, volume 2, pp. 1059-1068, 1993.
- 11- Vikramaditya R. Jakkula, and Diane J. Cook, "Learning Temporal Relations in Smart Home Data", *Proceedings of the Second International Conference on Technology and Aging*, Canada, June 2007.
- 12- James F. Allen: *Maintaining knowledge about temporal intervals*. In: *Communications of the ACM*. 26/11/1983. ACM Press. S. 832-843, ISSN 0001-0782.
- 13- Roy P., Bouchard B., Bouzouane A., Giroux S.: A possibilistic approach for activity recognition in smart homes for cognitive assistance to Alzheimer's patient. In *Activity Recognition in Pervasive Intelligent Environment (Atlantis Ambient and Pervasive Intelligence)*, L. Chen, C. Nugent, J. Biswas, J. Hoey Editors, World Scientific Publishing Company, ISBN: 978-9078677352, pp. 1-20, September 2010.
- 14- Roy P., Bouchard B., Bouzouane A., Giroux S.: Challenging issues of ambient activity recognition for cognitive assistance. *Handbook of research on Ambient Intelligence and Smart Environments: Trends and Perspectives*, IGI global, F. Mastrogiovanni and N. Chong Editors, Information Science Publishing, ISBN: 1616928573, pp. 1-25, august 2010.
- 15- Roy P., Bouchard B., Bouzouane A., Giroux S: Combining pervasive computing with activity recognition and learning, *Web Intelligence and Intelligent Agents*, Zeeshan-hassan Usmani (Ed.), ISBN: 978-953-7619-85-5, INTECH, pp. 447-462, 2010.
- 16- G.Singla, D. Cook, and M. Schmitter-Edgecombe. [Incorporating temporal reasoning into activity recognition for smart home residents](#). *Proceedings of the AAAI Workshop on Spatial and Temporal Reasoning*, pages 53-61, 2008.
- 17- Diamond J. A report on Alzheimer disease and current research. Technical report, Alzheimer Society of Canada, (2005), 1-19.
- 18- Baum C., Edwards D.,: cognitive performance in senile dementia of the alzheimers type: The kitchen task assessment. *The American Journal of Occupational Therapy*. 1993, Vol. 47 (5), 431436.
- 19- Jensen, F. V., *Bayesian Networks and Decision Graphs* (Springer 2001)
- 20- M. Iosifescu, "Finite Markov processes and their applications", Wiley (1980)
- 21- N.A. Abdul-Manaf, M.R. Beikzadeh, representation and Reasoning of Fuzzy Temporal Knowledge (2006), *IEEE Int. Conferences on Cybernetics & Intelligent Systems and Robotics, Automation & Mechanics (CIS-RAM 2006)*.

- 22- Dubois, D., Prade, H. and Sandri, S. (1991). "On possibility/Probability transformations", proc. Of the 4th International Fuzzy Systems Association (IFSA'91) Congress, Brussels, Mathematics, PP. 50-53.
- 23- V. Jakkula, J. Cook. "Temporal pattern discovery for anomaly detection in a smart home", 3rd IET International Conference on Intelligent Environments (IE 07), 2007.
- 24- S. Luhr, G. West, S. Venkatesh. „Recognition of emergent human behaviour in a smart home: A data mining approach“, Pervasive and Mobile Computing Volume 3 , Issue 2, 2007.
- 25- G. Nagypal, "a fuzzy model for representing uncertain, subjective and vague temporal knowledge in ontologies", <http://www.springerlink.com/content/atnluqe2gn8y7h27>, 2003.
- 26- D. Dubois and H. Prade (1998) Possibility theory: Qualitative and quantitative aspects. In D. M. Gabbay and P. Smets P., editors Handbook of Defeasible Reasoning and Uncertainty Management Systems, Vol. 1., Dordrecht: Kluwer Academic, 169-226.
- 27- <http://www.springerlink.com/content/j4g35l38913w2j0t>

Characteristics of Computational Intelligence (Quantitative Approach)

Shiva Vafadar, Ahmad Abdollahzadeh Barfouroush

Intelligent Systems Lab
Computer Engineering and Information Faculty
Amirkabir University of Technology
Tehran, Iran
vafadar@aut.ac.ir; ahmad@ce.aut.ac.ir

Abstract

These days, intelligence is one of the features of software agents. However, developing this feature via a systematic software engineering approach suffers from some shortages. One of the open issues is related to specifying intelligence features that are expected from intelligent agents. The source of the problem is different definitions of intelligence that are presented from different perspectives. Consequently, there is not a predefined set of characteristics for intelligence that can be used as a baseline for specifying intelligence requirements of the system. As a result, intelligence is defined (or interpreted) differently between various stakeholders of the system. This will lead to the ambiguity of the requirements, which is the source of serious problems in developing software systems.

In this paper, we look at intelligence of agents from a software engineering point of view. In this way, we analyze more than 70 different definitions of intelligence (in different areas such as public notions, psychology and AI) to extract different characteristics that are considered as features of intelligence. By eliminating non-computational capabilities of intelligence, we investigate a set of characteristics of computational intelligence.

In this way, we use a quantitative approach. We rank identified characteristics according to the frequency of their appearance in various definitions. The result is that learning, adaptation to new situations and environment, goal-orientation, problem solving, acquiring and using knowledge and reasoning are the top ranked issues of intelligence. Because the extracted features belong to different levels of abstraction, we classify them into four groups that are non-functional, general capabilities, basic AI techniques and Infrastructural. In addition, we investigate the relationship between intelligence characteristics (e.g. learning) and the other quality attributes of software systems.

Introduction

Usually, intelligence is an expected capability of software agents. One of the promises of agent oriented software engineering is to bring artificial intelligence findings to everyday practices of software development [1]. We believe that as intelligence is one of the features of software agents, similar to the other features of software systems, it should be developed via applying a complete process, which covers all the activities of software development such as requirement engineering, analysis, design, implementation and test.

However, some research have been performed on analysis and design of intelligence features of the agents (such as autonomy [2], reasoning [3] and learning [4]), but currently this process is more focused on implementation of artificial intelligence software systems. Consequently, by ignoring requirements specification, analysis and (somehow) design of intelligence features, implementation suffers from a complete and comprehensive input from earlier phases of software development. Taking into account the cost and probability of the failure in such an incomplete process, importance of developing intelligence via an engineering approach is made obvious.

In a complete software engineering process, the first activity is defining and specifying requirements and expected features of software system. By considering intelligence as a software requirement, a set of characteristics is needed to be used as a reference for defining intelligence requirements. But, unfortunately, intelligence itself is a vague term and there are different definitions for it [6]. Consequently, there is not an agreement on intelligence, not only between customers and developers but also between experts of each group. This will result ambiguity in intelligence as a requirement, because it is interpreted differently by different stakeholders of the system.

To solve this problem, in this paper we present a software engineering view on intelligence as a requirement of

software agents. Our approach is decomposing intelligence - as an ambitious term- to a set of more concrete features or characteristics that are considered as elements of intelligence. In this way, we perform a quantitative analysis on various perspectives of intelligence in different fields (psychology and AI). According to this approach, we can distinguish the main features of intelligence based on the experts' point of view. The frequency of each feature in different definitions can be interpreted as an evidence of implicit agreement on it as a feature of intelligence. To achieve this goal, we take into account more than 70 different definitions of intelligence which have been presented by experts in psychology and AI, in addition to the popular notions about intelligence (which are presented in dictionaries and encyclopedias) and we extract different issues that are mentioned in them as characteristics of intelligence.

Our survey shows that there are 28 distinct characteristics in these definitions. By omitting non-computational characteristics and features that are trivial capabilities for machines, we consider 16 characteristic as computational intelligence features which are ranked based on their frequency and importance in various fields.

The main contribution of this work is that it breaks down intelligence into a set of more concrete characteristics that can be defined and specified as requirements of software agents. The result of this research is a set of characteristics (or features) which can be considered as computational intelligence requirements. This set can be used as a basis for eliciting and specifying intelligence requirements of the system. To this end, requirements engineer uses this set of features as a common language between different stakeholders of the system to interpret intelligence from their point of view. This activity is the first step for moving towards a complete software engineering process for intelligence requirements of agents. Consequently, it can be used as a basis for analysis, design, implementation and test of these features.

The remainder of this paper is organized as follows: In Section 2, we analyze different definitions of intelligence in public notion, psychology and AI. In Section 3, we present the set of characteristics for computational intelligence by analyzing the results of Section 2 and we also classify these characteristics into four main groups. In Section 4, we investigate the relationship between learning (as an intelligence characteristic) and non-functional requirements of software systems. Finally, in Section 5, we conclude and introduce further works of this research.

Analyzing Different Definitions of Intelligence

During last century, various researches have been performed on human and artificial intelligence. Despite such a long history, still there is not a standard definition

of intelligence neither in psychology for human intelligence nor in AI for artificial intelligence.

One of the methods that can help us to overcome ambiguity of a term is to describe it via its characteristics. For example, in software engineering quality -as a vague term, is defined via its characteristics such as reliability, usability, etc. [5]. Therefore, to solve the same problem for intelligence, it can be defined via its characteristics as well. In this way, we break down different definitions of intelligence to investigate attributes that are identified as characteristics of intelligence.

Our observation leads us to believe that intelligence is not a single unitary ability, but rather a composition of several functions. This result confirms our first hypothesis for defining intelligence via its characteristics that helps us to specify intelligence via its features in software systems. This interprets intelligence (as a vague requirement) to a set of definable features of the system.

In the following, we analyze definitions of intelligence in three categories. Our analysis is performed on a set of definitions that is considered as the largest and most well-references collection on intelligence [6]. It contains 71 definitions of intelligence in psychology, AI and popular notions about intelligence.

The goal of this analysis is to distinguish characteristics that are considered as elements of intelligence according to the experts' point of view in each field.

Public Notions of Intelligence

There are 18 definitions in this group. This group represents definitions that have been proposed by groups or organizations and definitions of intelligence given in dictionaries and encyclopedias [6]. We consider these definitions as popular notions about intelligence because they construct or represent general ideas about intelligence in public. Since customers are a main group of stakeholders for defining requirements of the system, this category of definitions is important in our survey because it represents customers' point of view about intelligence.

By reviewing these definitions, we identify the following characteristics as attributes of intelligence: learning and understanding (e.g., facts, truth, meanings) (12 times each), reasoning (9 times), ability to adapt to the environment or new situations (6 times), capability to solve problems (5 times), capability to acquire and apply knowledge (5 times), profit from experience (4 times), capability of planning, thinking abstractly (or generalization) (each one 3 times), having judgment, perceiving relationships, using memory, comprehending language (two times) and finally being able to classify, calculation and imagination (each one once). Figure 1 shows the results of analysis of this group definition. In this figure, red bars demonstrate the characteristics that are common between all the groups that we have surveyed.

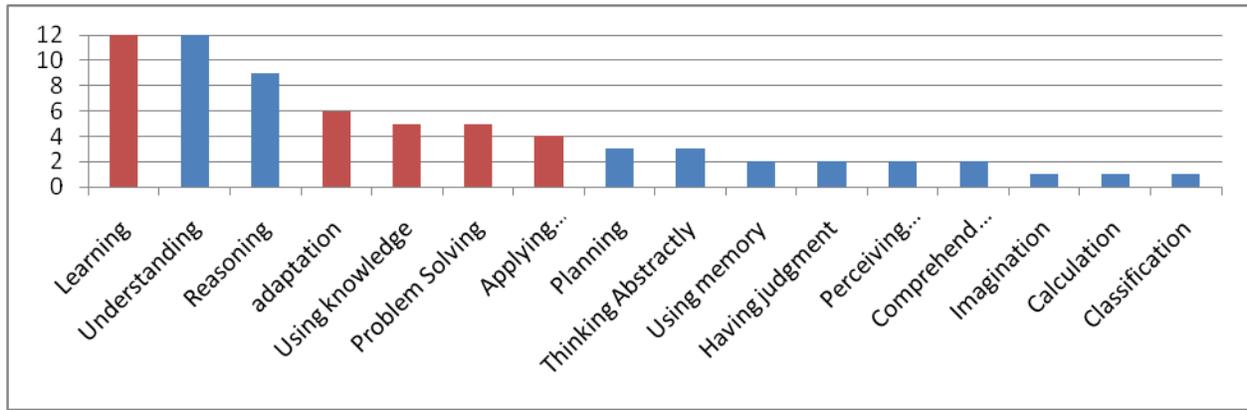


Figure1: Characteristics of Intelligence Based on Public Notion

Psychologists Definitions

This category contains 35 definitions from psychologists [6]. Taking into account these definitions in our survey helps us to understand elements of human intelligence according to psychologists and consider related attributes for computational intelligence.

We distinguish 23 issues that have been mentioned as features of humane intelligence in psychologists' definitions. They are ranked as the following according to their frequency in surveyed definitions: ability to adapt to the environment or new situations (8 times), learning, ability to solve problems and capability to acquire and

apply knowledge (5 times each), thinking abstractly, having judgment, applying experience, imagination and perceiving relationship and generalization capability (each one 3 times), reasoning, perceptual recognition, capability to produce product, using memory (each one twice), and finally planning, quickness, flexibility, attention, pattern recognition, being educable, discrimination, sensation, cognitive ability (each one once). Figure 2 shows the results of analysis of this group's definitions. In this figure, red bars highlight the characteristics that are common between all the groups that we have surveyed.

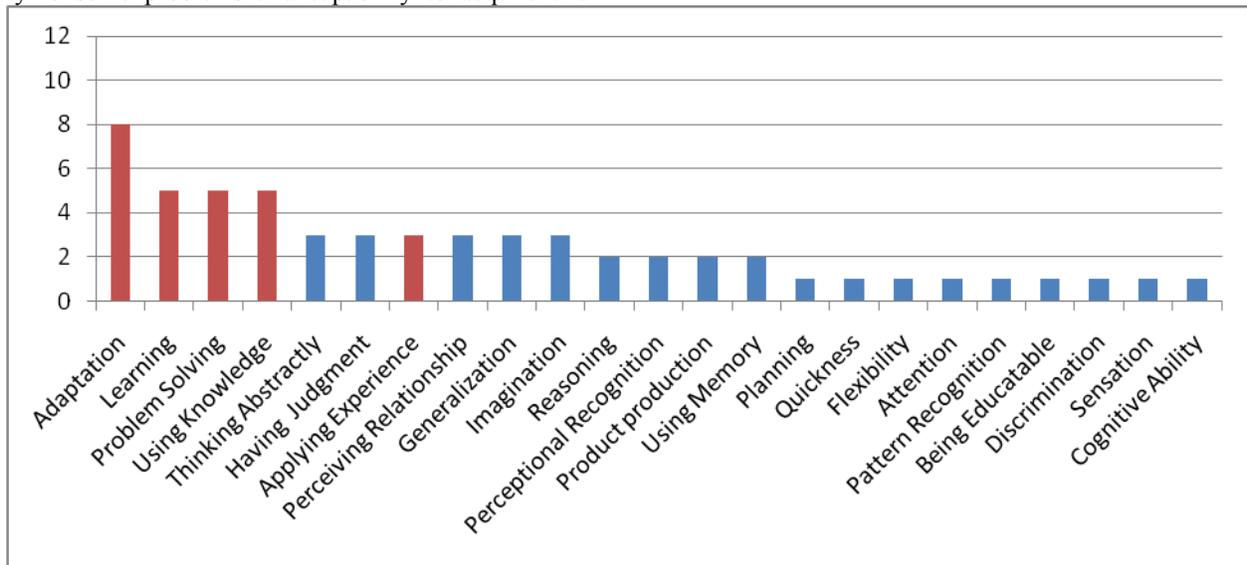


Figure2: Characteristics of Intelligence Based on Psychologists' Definitions

AI researchers Definitions

In this section, we analyze 18 definitions of intelligence from researches in artificial intelligence. The complete list of these definitions can be found in [6]. By reviewing these definitions, we identify that there are eight different topics as characteristics of intelligence which are: Goal-

orientation (9 times), ability to adapt to the environment or new situations (4 times), learning (3 times), capability to solve problems (2 times), capability to acquire or apply knowledge (2 times) and applying experience and autonomy (each one once). Figure 3 shows the results of this analysis.

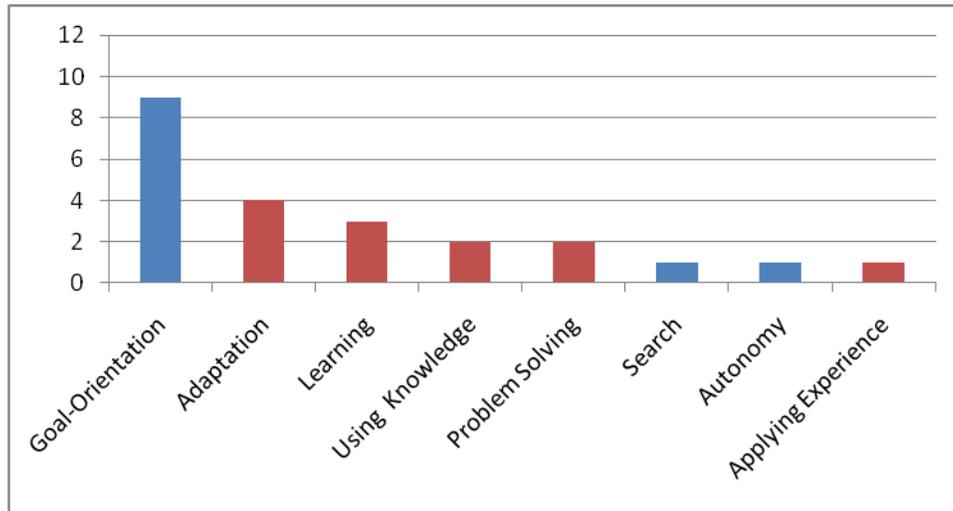


Figure3: Characteristics of Intelligence Based on AI Researchers' Definitions

Intelligence Characteristics in Computational Systems

By reviewing the results of analyzing definitions in different categories, strong similarities between many of these definitions quickly becomes obvious. This shows that there is an implicit agreement on some characteristics of intelligence. In addition, by taking into account their frequency of appearance, we conclude that some of them are more accepted as intelligence characteristic among experts than the others.

At the other hand, some of these features (especially features of intelligence in psychology and public definitions) are not suitable options for computational systems because they are not computable. In order to choose features of computational intelligence among distinguished set of characteristics, we omit these types of characteristics such as thinking, judgment, imagination, understanding, attention, product production, being educable, discrimination and understanding language. Because we are interested in intelligence as a behavior of software systems, we also ignore cognitive ability from our list. We also omit characteristics such as using memory or computational capabilities because they are the base of all computational systems. Otherwise, all computational systems would be intelligent and we are not interested in such a definition of intelligence.

After removing mentioned features, 16 characteristics remain in our list. To rank these features, we weight characteristics of different groups. We believe that characteristics that are mentioned by AI researchers are

more important in computational intelligence than those that are considered in human intelligence. Therefore, AI features have more weight than features in public notions and psychology features get the least weight. The result of this approach is shown in Figure 4. In this figure, red bars demonstrate shared characteristics in all the groups of our survey.

According to this approach, features of computational intelligence are ranked as follow:

- Learning
- Adaption
- Goal-Orientation
- Using knowledge
- Problem Solving
- Reasoning
- Applying Experience
- Generalization
- Perceiving Relationships
- Planning
- Autonomy
- Perceptual Recognition
- Classification
- Quickness
- Flexibility
- Pattern Recognition

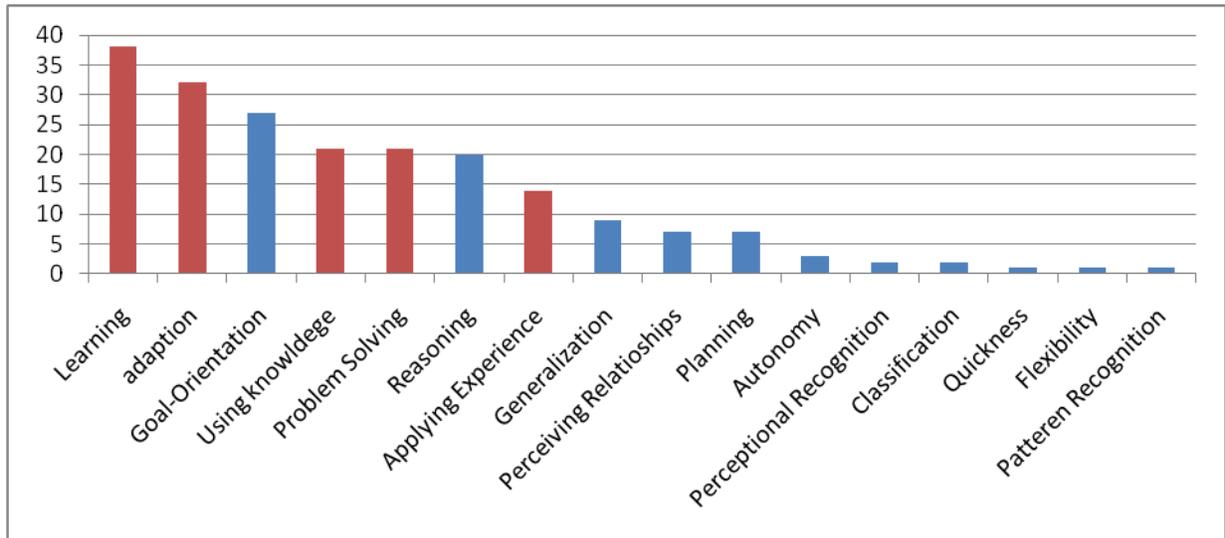


Figure4: Characteristics of Computational Intelligence

By examining these characteristic, we understand that these features are at different levels of abstractions. For example, some features are more general than the others, some of them are a subset of the others, and there are also some distinct features that refer to different capabilities of the system. To organize them, we divide these features into four categories. In this classification, intelligence is a feature of the system that improves non-functional requirements of the software system such as quickness and flexibility. To achieve this improvement, an intelligent system should be able to solve the problems in a goal-oriented manner. It also should be autonomous. In order to provide these general capabilities, intelligent systems need artificial intelligence capabilities such as reasoning, planning and learning that can be achieved through adaptation, pattern recognition, classification and applying experience. Intelligence of the system is founded on an infrastructure that contains knowledge and sensations or perceptions. Table 1 summarizes this classification.

Table1: Classification of Computational Intelligence Characteristics

Type	Characteristics
Non-Functional Requirements	Quickness, Flexibility
General Capabilities	Problem Solving, Goal Orientation, Autonomy
AI Techniques	Reasoning (Generalization), Planning, Learning (Adaptation, Pattern Recognition, Classification, Applying Experience,)
Infrastructures	Knowledge (Facts, Relationships), Perception (or Sensation)

Intelligence Characteristics and Non-Functional Requirements

When a capability is added to a software system, it improves software functionality. But there are some non-functional or quality requirements in the systems that should be taken into account during requirements engineering as well. In some cases, there are contradictions between these requirements. This means that by having some requirements in the system, we may lose or degrade the others. For example, by adding security features to the system, in general, more computations should be performed in the system. This can affect performance and efficiency of the system. In these cases, software engineer should choose a subset of requirements by considering the trade-off between requirements.

For looking at intelligence as a requirement of software systems, we need to analyze the side effects of intelligence and its characteristics on the other quality attributes of the system. For example, requirements engineer should pay attention to the side effects of considering learning requirement –as a feature of intelligence- on other quality attributes of the software system. Table 2 shows the relationship between learning and non-functional requirements. Characteristics of quality requirements in table 2 have been selected according to the classification of ISO 9126 [5]. In this table, “+” means that adding learning improves the non-functional characteristic. “-” means that learning potentially may decrease the sub-characteristic and “*” means that the relationship between learning and specified characteristic of non-functional requirement is neutral.

As this table shows, learning has a positive effect on characteristics of the system such as suitability, accuracy, interoperability, security, fault tolerance and adaptability.

But it has a negative relationship with efficiency and maintainability characteristics of system in general. This means that if quick response is a critical quality requirement and there is shortage of resources in the system, then adding learning to the intelligent requirements of the system should be done cautiously. The main reason is that learning utilizes extra time and resources of the system that may decrease its efficiency. Adding learning to the system also makes the code more complex, therefore changing the code or analyzing it when there is an error in the system becomes more complex and time consuming. Therefore, maintainability of the software decreases in general. But if adaptability is a required quality characteristic of the software system, adding learning as a requirement helps to attain an adaptable system.

Table2: Relationship between learning and quality requirements

CHARACTERISTIC	SUB-CHARACTERISTIC	Learning
Functionality	Suitability	+
	Accuracy	+
	Interoperability	+
	Security	+
	Compliance	*
Reliability	Maturity (hardware/software/data)	+
	Fault tolerance	+
	Recoverability (data, process, technology)	*
	Compliance	*
Usability	Understandability	*
	Learnability	*
	Operability	*
	Attractiveness	*
	Compliance	*
Efficiency	Time behavior	-
	Resource utilization	-
	Compliance	-
Maintainability	Analyzability	-
	Changeability	-
	Stability	-
	Testability	-
	Compliance	-
Portability	Adaptability	+
	Installability	*
	Co-existence	*
	Replaceability	*
	Compliance	*

Conclusion and Further Work

The aim of this research was identifying the main characteristics (or capabilities) of computational intelligence based on various definitions of intelligence. To achieve this goal, we analyzed more than 70 definitions of intelligence in various fields such as popular notion of intelligence, psychology and AI. According to the results of our survey, we distinguished 16 characteristics for computational intelligence.

This set can be used as a guideline (or reference) for eliciting and specifying expected capabilities (features) of intelligence system during requirement engineering. To develop an intelligent software (agent) system, first we should define intelligence requirements of the software according to the system or stakeholder's needs. As extracted characteristics are based on public notions of intelligence in addition to the AI experts' point of view, it can be considered as a common language between different stakeholders of the software such as developers and customers.

By specifying expected features during requirements engineering, later activities of software development such as analysis, architectural and detailed design and test of intelligence are based on a predefined set of capabilities. Furthermore, this set of requirements can be used as a basis for comparing intelligent agents in COTS (Component Of The Shelf) software development. To achieve this goal, intelligence requirements of the system (or agent) should be specified based on the proposed set of characteristics. At the other hand components that are developed should be defined according to this set as well. Having these preconditions, according to the intelligence requirements of the system (or agent), we can choose the most appropriate available component (or agent) for the system. For example, available components or agents are tagged according to their capabilities. In this case, if system needs an intelligent agent (component) that should be autonomous and being able to learn, we can choose the agent with these capabilities, according to the tags of available ones.

Our further works to extend our research are:

- Defining relationship between these features, in addition to the relationship with other non-functional requirements of software systems.
- Developing analysis patterns as the next activity of software development for these features such as learning analysis patterns[4]
- Defining validation and verification approaches for these features based on the existing methods for testing computational intelligence [7,8]

References

- [1] Zambonelli, F., Omicini, A.. *Challenges and Research Directions in Agent-Oriented Software Engineering*, Autonomous Agents and Multi-Agent Systems, Volume.9 No.3, p.253-283 (2004)
- [2] Weiss G., Fischer F., Nickles M., Rovatsos M. (2006). Operational modelling of agent autonomy: theoretical aspects and a formal language. Proceedings of the 6th International Workshop on Agent-Oriented Software Engineering (AOSE, pp. 1-15). Lecture Notes in Computer Science, Vol. 3950. Springer-Verlag.
- [3] Bosse T., Jonker C. M., Treur J. (2005). Requirements Analysis of an Agent's Reasoning Capability, Proceeding of the 7th International Workshop on Agent-Oriented Information Systems, AOIS'05.
- [4] Vafadar, S., Abdollahzadeh Barfouroush, A.: *Towards Requirement Analysis Pattern for Learning Agents*, Agent Oriented Software Engineering (AOSE) Workshop, short paper, Toronto, Canada (2010)
- [5] ISO/IEC 9126-1: Information technology - Software quality characteristics and metrics - Part 1: Quality characteristics and subcharacteristics
- [6] Legg, S., Hutter, M. *A Collection of Definitions of Intelligence*, Advances in Artificial General Intelligence: Concepts, Architectures and Algorithms, volume 157 of Frontiers in Artificial Intelligence and Applications (2007)
- [7] Hernández-Orallo, J., Dowe, D.L. *Measuring universal intelligence: Towards an anytime intelligence test*, Artificial Intelligence Journal, Volume (2010)
- [8] Sanghi, P., Dowe, D.L., A computer program capable of passing IQ tests, in: Proceedings of the 4th ICCS International Conference on Cognitive Science (ICCS'03), Sydney, Australia, July 2003, pp. 570–575.

Toward Robust Features for Remote Audio-Visual Classroom

Isaac Schlittenhart Jason Winters Kyle Springer Atsushi Inoue *

Eastern Washington University
Cheney, WA 99004 USA

Abstract

We present two studies on robustness of feature extractions for an remote classroom intelligent autopilot: (1) robust feature extractions and (2) a simple automated calibration of webcams. For the robust feature extractions, use of quantified vectors is studied as feature extractions of fuzzy classifiers in Perceptual State Machine, i.e. our core Computational Intelligence model for this intelligent autopilot. The simple automated calibration of devices is studied mainly for the sake of maximizing device utility. Those studies have shown promising results for actual use of this intelligent autopilot in ordinary classrooms that are not necessarily ideal for teleconference lectures.

Keywords: Remote Classroom, Autopilot, Perceptual State Machine, Fuzzy Classifiers.

Introduction

In this paper we present an improvement on robustness of an intelligent autopilot for remote audio-visual classrooms. This intelligent autopilot, that is currently under development, intelligently recognizes students in a remote classroom who need their instructor's attention. It controls various audio and visual devices, such as microphones and CCD cameras, autonomously as deemed appropriate. Currently, we incorporate a simple Computational Intelligence model, so-called Perceptual State Machine, i.e. a finite state machine with the use of fuzzy classifiers as its transition functions, and study mainly on its feature extractions in order to achieve our satisfactory performance. This paper describes our most recent progress on their robustness against lighting conditions of the remote classrooms that are often considered problematic for image processing.

Background and Motivation

Operating standard distance learning remote classrooms requires skilled operators. This often causes the costs of remote classrooms to be prohibitive for smaller institutions and may result in additional costs to student tuition and the institution. Further, high quality audio and visual devices often demand frequent calibration. This can require specialized, skilled technicians and generate yet another cost. As a

consequence, real-time remote lectures are frequently considered infeasible and expensive, despite their potentials and needs.

Due to the recent growth in Internet communication, there is more availability of cost effective webcams, condensed microphones, projectors or large screens, and ordinary PCs (desktop and laptop). In addition to hardware, there has been a large increase of useful online services and methods of delivering remote content, such as remote desktop controls, video chat, conference calls, etc. Considering such availability of off-the-shelf products, we anticipate to utilize those in order to suppress the implementation costs.

Since all the products we are employing are ready to use out of the box, the main issue is device control and integration. Our goal is a robust and intelligent autopilot that utilizes fuzzy sets, so that off-the-shelf products are to be maximumly utilized while requiring very little or no calibration, i.e. virtually free maintenance. In addition, it is ideal to make the entire remote classroom system (consisting of off-the-shelf devices, PCs, software, and this intelligent autopilot) compact and portable, e.g. a few components on a cart. When such a system is available, remote lectures can be set in any standard classrooms within a matter of minutes by anyone (e.g. student assistants) without requiring extensive technical training.

Perceptual State Machine, our Computational Intelligence model, is essentially a finite state machine that makes use of fuzzy classifiers as its transition functions (Beaver and Inoue 2006). Those fuzzy classifiers map between perceptual states naturally recognized by human beings (e.g. 'no session', 'need attention' and 'in session') and inputs (i.e. features) extracted from video and audio streams, that are captured through physical sensors such as webcams and condensed microphones. All the previous studies on feature extractions have focused only on various pixel histograms in certain color ranges mostly for the sake of simplicity and real-time responses (Beaver and Inoue 2005; Moore et al. 2008). Those studies include palm pixel count and extension of palm pixel count, frame pixel differentiation, gesture recognition using hue saturation value, pixel count/position histograms, counting moving colored pixels, and locating a student via audio amplitude. These feature extractions performed at a satisfactory level thus held promise under ideal lighting conditions, e.g. no sunshine coming into

*E-mail: inoueatsushij@gmail.com

the room as a result of shutting off the window shades.

Anticipated Improvements

The following two anticipations are presented in this paper for improving the robustness against lighting conditions that are not necessarily ideal but are rather common in many ordinary classrooms. If those anticipations are successful, our technology advancement is very significant.

Toward robust question detection using quantified vectors. The well known pitfall in those pixel counting feature extractions concerns scalability and position shifts in the frame. Assuming the video frame was cropped into sections of interest what would happen if someone's foot or hand protruded into the cropped area of interest or the subject shifted rapidly in the frame? What if the individual in the frame had a large amount of skin exposed or their skin was a different hue than expected? The result was a potentially false question state when no question existed or the lack of detecting a question at all. Use of quantified vectors presented here potentially solves this problem and seems well suited to larger scale.

Toward robust images using color and lighting correction. Using off-the-shelf components has its benefits but also adds issues concerning quality control and calibration. We found that many of the inexpensive webcams do have automatic white balance and exposure controls but that these controls can be inadequate in various classroom lighting conditions. Two simple correction methods are discussed in this paper that likely improve the quality of video images that are to be fed to the intelligent autopilot.

Question Detection Using Quantified Vectors

The detection of any object through computer vision has been an evolutionary process. The real-time requirement aspect of this intelligent autopilot system puts some constraints on processing power. Because of this, more simplistic methodologies have been employed in detecting question states. Initially in Beaver's work (Beaver and Inoue 2005), a method was proposed by which pixels matching the grayscale color of the palm were counted. Through the use of a fuzzy classifier three states were identified: in session, no session, and question. Beaver's method worked well under ideally set-up conditions, but such pixel count methods can have difficulty with scale or computing distance from the camera. A following work (Moore et al. 2008), while still focusing on pixel counting, includes pixel location extracted from vertical and horizontal histograms of pixels in certain color ranges. Additionally, instead of using only grayscale palm color range, a skin detection using hue saturation value has been employed. This method has held promise and showed selected skin segments well enough when taken under a white balanced condition.

Although the pixel count methods and skin detection data plots have been distinct and held promise, they are plagued by stability and reliability issues when considering lighting conditions of ordinary classrooms. The premise of the intelligent autopilot system is for a simple cart, requiring little

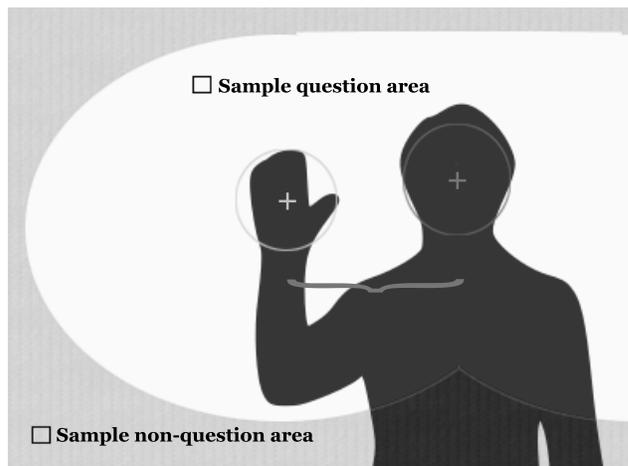


Figure 1: Sample raised hand range

to no calibration, to be wheeled into a classroom and function. Through experimentation, we have found that white balance affects color and, more specifically, white balance varies from both internal and external light sources and from camera hardware. The variation in lighting causes the skin color detection algorithms to mistake objects in the room as skin. This had a very large impact on the ability to detect a question. Color change in image pixels also suffers from other problems that are difficult to address: variation in skin color, skin-colored objects in the room, individuals with large amounts of skin exposed, and shifts in scale and framing of individuals. Clearly, Color change in image pixels alone is not robust for recognizing perceptual states of students in remote classrooms.

With further increasing computational power and recent advancements of devices, issues of computational cost has become less significant. As a result of this, more informative methods for detecting objects in video frames are to be feasible. In this study, we propose a feature extraction method of combining hand and face position data as quantified vectors for better robustness in perceptual state recognition.

Approach

Even in live classrooms, evaluating if a student has a question is subjective. Since the likelihood of a question is not an absolute yes or an absolute no, fuzzy sets are the best suited for determining such perceptual states of a classroom.

From a human perspective, we generally and naturally analyze the location of the hand in relation to the face. Since our minds are assumed to process the position of the hand in relation to the face, we do not process this as exactly computed; but we rather simply know (i.e. perceive) if an individual has a question by observing them. Taking this into consideration, we utilize the centers of the face and the hand in such a way that a line can be drawn between them. We simply consider this as a vector. As far as its coordinate is considered, given the inherent presentation of such a vector in an image composed of rows of pixels, we use polar coordinate rather than Cartesian. In doing so, vector coordinates

of the data points can be represented as x-axis distance and y-axis distance from center to center and their angles and magnitudes are inherently contained within themselves.

Since the position of the face can be described relative to the hand straightforwardly using this polar vector, the only remaining aspect that has to be addressed is how to actually recognize the hands and faces in the target image. Jones (Viola and Jones 2001) has suggested a method that has proven to be highly effective in recognizing objects given a set of training images. Other studies have shown that Haar-like classifiers proposed in their studies are superior in recognition rates per CPU cycles than many other conventional methods (Santana et al. 2008). Since Haar-like classifiers are based on trained boosted classifiers with image integrals, our current concerns such as color, white balance and skin-colored objects in the room no longer impact the feature extractions (Viola and Jones 2001).

Model

A single video frame is assumed to contain an image of the classroom. The three perceptual states of the classroom can then be outlined from a logical viewpoint as follows:

No session No faces present in the video frame.

In session Faces present in video frame but not hands (if any) in question positions.

Question Faces and raised hands present in the video frame.

First, proximities of faces and hands are presented as vectors in Cartesian coordinate such that Haar-like classifiers find the following coordinates:

- The centroid of the i -th face $F_i(x_1, y_1)$.
- The centroid of the i -th hand $H_i(x_2, y_2)$.
- The width of the face s , i.e. a scaler.

Then the quantified vector V is constructed in Cartesian coordinate such that

$$V = \langle x_2 - x_1, y_2 - y_1 \rangle \cdot s$$

Finally this is converted into the polar coordinate such that

$$V = \langle r, \theta \rangle$$

where $r = |V| = s \cdot \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$ and $\theta = \arctan \frac{y_2 - y_1}{x_2 - x_1}$ if $x_2 - x_1 > 0$ and $\theta = \frac{\pi}{2}$ if $x_2 - x_1 = 0$. Clusters of such vectors are summarized (i.e. their histograms are generated) in order to generate fuzzy sets for the fuzzy classifiers.

Experiment and Evaluation

An open source software called OpenCV provides the necessary tools in order to train Haar-like classifiers. The experimental procedure follows:

1. Train Haar-like classifiers for hands and faces using OpenCV software. Alternatively, there are pre-configured Haar-like classifiers in OpenCV.
2. Use those trained hand Haar-like classifiers in order to detect faces and hands in a video frame.

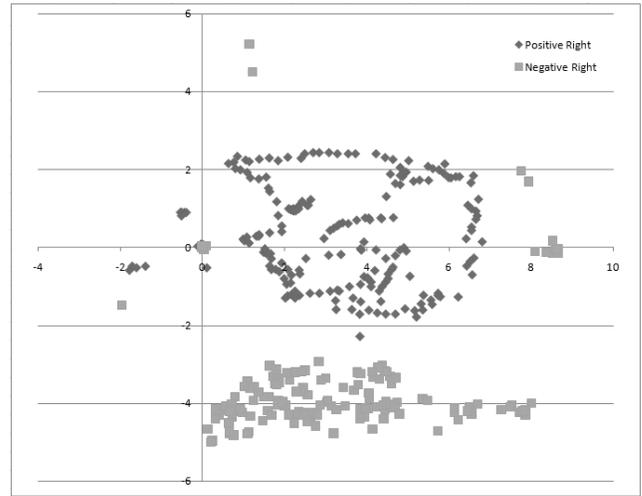


Figure 2: Sample data points

3. Collect the quantified vector in order to identify fuzzy classifiers for the perceptual state recognition.

In our experiment, we have used printed images of faces and hands then captured video frames of those through a webcam so that vector data points can be simulated as if actual images of actual students are captured. The pre-configured face classifiers have worked quite well in our experiment while the pre-configured hand classifiers do not. As a result, we have encountered some training overheads for those classifiers (a future work). Some results can be seen in figure 2. The y-axis displays the range that the hand can be located from the head vertically. This information combined with the x-axis showing the hand face distance horizontally creates a visual map that can be used to define fuzzy partitions. As shown, the data groupings are very distinct and the fuzzy classifiers should well be identified.

Findings

The robustness of using quantified vectors as features holds promise and seems to bypass many of the issues encountered with pixel/color methods. The distinct data sets in figure 2 show that this method is very distinct and will be able to recognize classroom states to a high degree of accuracy. However, a new set of issues is introduced. Haar-like classifiers may have trouble with detecting objects if the object is rotated slightly. This could pose a problem for both faces and hands. A solution has been suggested by (Barczak, Johnson, and Messom 2005) and further investigation must follow. Another issue is the dependencies of Haar-like classifiers upon a number of parameters including sample size, training parameters, and optimal training image size. For the time being, default parameter setting appears to be sufficient. It also appears that a larger sample of images are highly demanded for the satisfactory classifier training. If this is indeed the case, Haar-like classifiers may not be suitable for this intelligent autopilot. A further investigation is currently underway on this critical issue.



Figure 3: Image of excessive light coming into the camera

Robust Images Using Color and Lighting Correction

Despite some extreme lighting conditions such as a direct sun light coming into a classroom, sensory devices may still be able to capture images. In theory, our Computational Intelligence model is capable of recognizing states as long as the images are captured distinctively enough from noises, even if they are not in good quality for human eyes. The goal this work is to maximize the recognition performance as a result of color and light correction. The following two steps are considered to maximize the image quality coming into the system under extreme conditions:

1. Detect whether the appropriate magnitude of light is there.
2. If the magnitude of light is not appropriate (i.e. excessive or insufficient), perform a color correction.

Step 1: Lighting Conditions

Lighting conditions in a room can greatly affect the quality of an image taken through a digital camera such as a web camera. If the amount of light in a room is insufficient, there is a chance that the camera will not pick up on details of objects in the room. If there is excessive lighting, chances that the picture is bleached out are very high. Two examples can be seen below of how the amount of lighting can affect the image taken through a web cam: figure 3 and figure 4. By looking at those images it is possible to see how badly the details of the image can be lost depending on the lighting conditions coming into the camera.

It is impossible to correct images that have either too much or too little light due to the fact that details in the image will be missing. This means in the system for the virtual classroom there needs to be an automated warning system to alert the user that the amount of lighting coming into the camera is not appropriate for the system. The right amount of lighting depends on the camera being used. Each



Figure 4: Image of insufficient light coming into the camera



Figure 5: Histogram of excessive light coming into the camera

camera will have a different aperture size. The smaller the aperture, the less light will enter the lens and the bigger the aperture, the more light will be brought in (Busch 2008). Since most web cameras have a fixed aperture size the lighting of the room will have to be adjusted to meet the needs of the camera selected for the virtual classroom. Traditional methods to detect the correct amount of lighting for an image taken are the use of histograms (Busch 2008). If most of the weight of the histogram falls on the left side this means that there is not enough light in the room to ensure details. If the histogram has most of the weight on the right side, this indicates there is too much lighting for the room. See 5 and 6 for examples of the histograms for the images above.

Once a histogram of an image is made it is possible to automate the process of having the system alert the user of improper lighting conditions for the lens being used inside the classroom. OpenCV has several tools to build histograms of images (Gary Bradski 2008). These histograms can be built and analyzed in real time. The data in the histograms can be placed into buckets. Each bucket will contain the count of a certain color. If there is too much black in the image, this means there is insufficient light for the camera. If there is too much white, it means there is too much light for the camera.

At this point it is unknown how much black and how much white indicates a problem with the lighting in the



Figure 6: Histogram of insufficient light coming into the camera

system. Forty images were tested for this purpose; twenty of them were examples of insufficient light while the other twenty contained too much light. It was found that if 30 percent of them fell in the white range or black range the lighting conditions were not right for the room. If the room contains a lot of black or a lot of white to begin with the 30 percent may not be a good threshold for that room. This means that the threshold will have to be adjusted by the user if needed depending on the objects in the room.

Result 1: Lighting Conditions

To test the system 30 images were used; ten each of excessive, insufficient, and reasonable lighting. Each image was fed into the system to see if it detected the type of the input image. At this stage, "reasonable" lighting does not mean perfect lighting. The lighting in the room just needs to be the right amount for the system to work reasonably well. Of the images tested, all the poor images were detected as "reasonable". However, out of the good images, two were detected as poor lighting conditions. Given that the purpose of this part of the system is just to alert the user that there could be a problem, this rate of false positives is acceptable.

Step 2: Color Correction

The next step in improving the image is white balancing the image – color correction. In any image taken the light in the scene affects the color of the image. The only type of light that is not affected is white light (light that comes from the sun). All artificial light sources contain color known as temperature (Busch 2008). The temperature changes the color of the objects in the scene. The goal of white balance is to take the color out of the objects in the scene and restore the color of the objects as if they appeared in natural white light.

While most decent image processing software packages can white balance an image, the system for the virtual classroom needs to do it in real time since the lighting conditions of the room may change while the system is in session by various lights being turned on or off. Another factor to consider is that two different cameras (even the same model) might capture images of different color values (Gary Bradski 2008). The goal of the experiment is to take an image of a known color value, see how the image is changed by the

lighting conditions of the room, and correct the values of the image of the room by the values changed on the target.

The target for the test was a piece of white foam board picked up at an arts and craft store. Given that white reflects all colors, and we are looking for the color change of the target and not the color of the target itself this material seemed to be a reasonable choice (Serway 1996). Next, two cameras were tested to see if they picked up different color values. This was done by putting three colored papers (green, red and blue) in front of each camera and recording the values they saw. In each instance, camera number 2 recorded two values more red than camera number 1. It is possible to calibrate the system by adding two values to the red in camera number 1 or by subtracting two values from camera number 2.

The next step is to set up the cameras to get input into the system. One camera is always going to point at the target; the other camera is always going to point into the room. The captured image will then be color-corrected in accordance with the color change caused by the temperature of the lighting used in the scene.

To summarize:

- There are two cameras.
- The color difference of the two cameras is tested and corrected.
- One camera points at a known color target.
- One camera points into the room (audience-facing).
- The color difference between the known color of the target and the color detected in the room is used to provide color correction for the audience-facing camera.

Result 2: Color Correction

Image color correction is subjective and based on human perspective. We feel that the results of our color correction method show promise. The corrected images' color was greatly improved. The brown tint of the images caused by the temperature of the lighting disappeared and the images looked more like they were taken in natural sunlight.

Integrated Test Result

We propose that using color-corrected images likely produces better results than previously tested methods for the system. In the original study it has been shown it is possible to count the number of pixels of skin tone in the image to see if there is a hand raised or not (Moore et al. 2008). One problem is that if the temperature of the lighting of the room is not properly white balanced, objects in the room such as tables and clothing possibly looks like flesh tones to the system. During the original test it has been found that the best accuracy of the system is 86 percent. Our test images have showed an accuracy of 85 percent with good lighting to the room. With poor lighting that accuracy dropped down to about 64 percent, which indicates the impact proper lighting has on the system. After having the system correct the images, the good images have not changed but the poor images have had 73 percent accuracy. This means by having the system improve the images it is possible to increase the

accuracy of the system by correcting the lighting conditions of the room.

Conclusion

In this paper, two studies on robustness of feature extractions from images are presented. They are necessary in order to achieve the goal of our intelligent autopilot to be placed in ordinary classrooms instead of ideally set-up experimental environments for many image processing works.

Use of quantified vectors that are generated by Haar-like classifiers provides a more robust feature extractions that is virtually immune to many encountered issues about scaling and positioning of objects in classroom images. Future works should easily allow for multiple individuals in each frame since a hand could be mapped to the nearest face while generating the quantified vectors, whereas multiple individuals residing in one frame posed a large problem for methods based on change in colors of pixels. Quantified vectors as robust features would potentially increase the accuracy when multiple individuals are present.

Furthermore, it has been shown that it is possible to improve the video quality through color and lighting correction. This simple method provides an automated solution to a problem commonly encountered in the ordinary classrooms that are not necessarily opted for teleconference lectures. This should bring us a feasibility that the system may be used in any ordinary classrooms (as long as a net connection is available) and that the state recognition may achieve a satisfactory level.

The next step of this intelligent autopilot is mainly about implementation and system integration for the first complete prototype.

Acknowledgment

This research was conducted as a 10-week course work of CSCD581 Computational Intelligence at EWU in Winter 2009. The authors would like to thank James Lamphere for an additional system administration support, Brian Kamp and his students for looking at bright lights while waving at

the camera, and Emily Schlittenhart for her help editing and proofing this paper.

References

- Barczak, A. L. C.; Johnson, M. J.; and Messom, C. H. 2005. Realtime computation of haar-like features at generic angles for detection algorithms. In *Research Letters in the Information and Mathematical Sciences - ISSN 1175-2777*.
- Beaver, I., and Inoue, A. 2005. Perceptual Recognition of States in Remote Classrooms. In *Proceedings of International Conference of North America Fuzzy Information Processing Society (NAFIPS05)*.
- Beaver, I., and Inoue, A. 2006. Using Fuzzy Classifiers for Perceptual State Recognition. In *International Conference on Information Processing and Management of Uncertainty in Knowledge-based Systems (IPMU2006)*.
- Busch, D. 2008. *Mastering digital SLR Photography*. Thomson Course Technology.
- Gary Bradski, A. K. 2008. *Learning OpenCV*. O'Reilly, first edition.
- Moore, Z. I.; Schlittenhart, I. W.; Simpson, D. M.; Sorna, C. T.; Springer, K. A.; and Inoue, A. 2008. Intelligent Autopilot for Remote Classroom: Feature Extraction. In *Proceedings of Midwest Artificial Intelligence and Cognitive Science Conference (MAICS 2008)*.
- Santana, M. C.; Déniz-Suárez, O.; Antón-Canalís, L.; and Lorenzo-Navarro, J. 2008. Face and Facial Feature Detection Evaluation - Performance Evaluation of Public Domain Haar Detectors for Face and Facial Feature Detection. In Ranchordas, A., and Araújo, H., eds., *VISAPP (2)*, 167–172. INSTICC - Institute for Systems and Technologies of Information, Control and Communication.
- Serway, R. A. 1996. *Physics for scientists and engineers with modern physics*. Saunders college publishing, fourth edition.
- Viola, P., and Jones, M. 2001. Robust Real-time Object Detection. In *International Journal of Computer Vision*.

Hybrid Direct Neural Network Controller With Linear Feedback Compensator

Dr.Sadhana K. Chidrawar¹, Dr. Balasaheb M. Patre²

¹Dean , Matoshree Engineering, Nanded (MS) 431 602

E-mail: sadhana_kc@rediff.com

²Professor S.G.G.S. Institute of Engineering and Technology,
Nanded (MS) 431 606

E-mail: bmpatre@yahoo.com

Abstract

In this paper Hybrid Direct Neural Controller (HDNC) with Linear Feedback Compensator (LFBC) has been developed. Proper initialization of neural network weights is a critical problem. This paper presents two different neural network configurations with unity and random weight initialization while using it as a direct controller and linear feedback compensator. The performances of these controller configurations are demonstrated on the two different applications i.e. Continues Stirred Tank Reactor as nonlinear and DC Motor as linear. In this work a direct neural control strategy with linear feedback compensator is used to control the process. Error back propagation algorithm based on gradient algorithm is used to minimize the error between the plant output and desired output signal. The Direct Neural Controller (DNC) and Hybrid Direct Neural Controller (HDNC) are compared in terms of the Integral Square Error (ISE) and Integral Absolute Error (IAE). Addition of a linear feedback compensator helps to improve both the transient as well as steady state response of the system

Introduction

There are many industrial applications where the direct and coordination control strategies are required. Different types of controller are in use to provide appropriate control inputs to process plants to obtain desired outputs by changing its parameters. Neural network has been applied successfully in the identification and control of dynamical systems (Wang et al.2005). (Yuan et al. 2006) give the methodology of design of a conventional model reference adaptive control system extended to design a direct neural control for a class of nonlinear system. (Peng and Huang 2006) has given a novel hybrid forward algorithm (FA) for the construction of radial basis neural network with tunable nodes. (Huang and Lee 2002) develop a decentralize neural network controller for a class of large scale nonlinear high order interconnections. He also proves that this NN controller can achieve for large scale systems. (Castilo and Melin 2002) has describe a new method for estimation of the fractal dimension of a geometry fuzzy logic technique.

They also develop a hybrid intelligent system combining neuro fuzzy logic and fractal dimension for the problem of time series prediction. (Xianzhong and Shin 1993) presented a novel method using direct adaptive controller and a coordinator using neural network. The developments in neural network based control systems for real time control applications are still in early stage. There is still necessity of carrying out lot of work to reach a stage of perfection, the stage after which, the ANN based networks may be freely used for all types of process control applications in the industry. This paper presents a work carried out to develop a hybrid direct neural controller that may find wider applications in different types of industrial control environments.

The specific contribution in this paper is respect to (i) The development of a direct Neural Network Controller for studying the effect of initialization of unity and random weights in neural network control structure. (ii) The development of a Hybrid Direct Neural Controller. The HDNC has been developed by modifying a Direct Neural Controller (DNC) by adding a Linear Feedback Compensator (LFBC) in parallel with the neural network controllers. The comparison of both the controllers i.e. DNC and HDNC in terms of the Integral Square Error (ISE) and Integral Absolute Error (IAE). The test results are highly encouraging and establish the superiority of HDNC over the other controller being used in the process industry for linear as well as nonlinear systems.

ANN Techniques

Fully connected neural network used in this work, consists of an input layer with six neurons, one hidden layer with seven neurons and a single neuron in output layer as shown in Fig. 1. To reflect the status of the controlled system, the inputs of the neural network controller are chosen as the desired system outputs, actual output and the output errors: $Y_D(k)$, $Y_D(k-1)$, $Y(k)$, $Y(k-1)$, $e(k)$, $e(k-1)$.

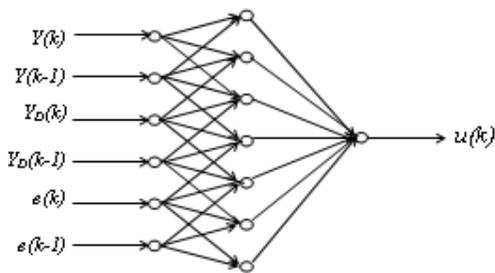


Fig. 1. Neural Network Architecture

ANN Method For Direct Control

A control system with DNC is shown in Fig. 2 Error Back Propagation Algorithm (Nahas, Henson and Seborg 1992) based on gradient algorithm is used to minimize the error between plant output and the desired output signal. Without a specific pre-training stage the weights of the neural network are adjusted online to minimize the error.

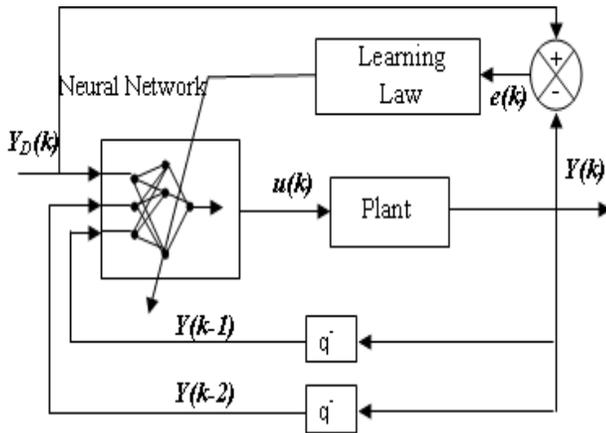


Fig. 2. Direct Neural Control System

$Y_D(k)$ is the desired process output, $Y(k)$ is the actual process output, $u(k)$ is the output of the neural network and $e(k)$ is the network error output.

DNC With Linear Feedback Compensator

In order to overcome problems associated with direct neural controller architecture a linear feedback compensator (LFBC) is placed in parallel with the neural controller. The application arrangement of the proposed hybrid scheme is shown in fig 3.

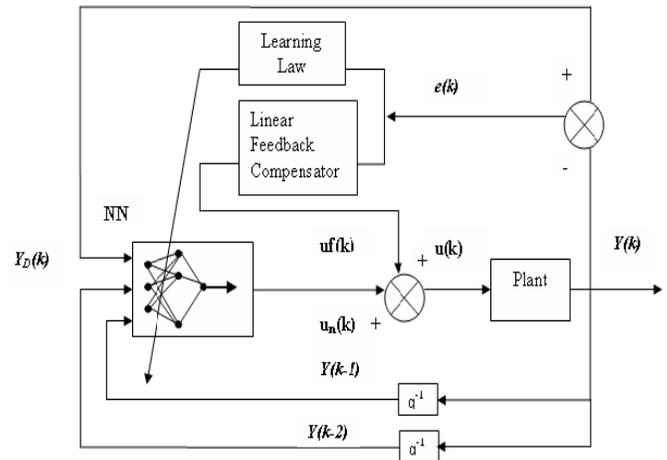


Fig.3. Direct neural controller with linear feedback compensator

Addition of a LFBC helps to improve both the transient as well as steady state response of the system. The hybrid combination of neural network and LFBC helps to eliminate the need of auto tuning of constants K_1 , K_2 and K_3 as required in conventional PID and Adaptive controllers. Once the values of constants are selected properly at one operating point, then these help to produce good results throughout the operating region of the systems. The hybrid combination of the neural network and the linear feedback compensator helps to compensate the limitation of individual controllers. The actual controlling signals $u(k)$ is the sum of output of neural controller and LFBC and is expressed as follows:

$$u(k) = u_n(k) + u_f(k) \quad (1)$$

Where $u_n(k)$ is the output of the neural network controller and $u_f(k)$ is the output of the linear feedback compensator (LFBC). Linear feedback compensator is a three term controller and expressed as

$$u_f(k) = K_1 e(k) + K_2 \Delta e(k) + K_3 \sum_{i=0}^k e(k) \quad (2)$$

Where, $e(k) = Y_D(k) - Y(k)$ and

$$\Delta e(k) = e(k) - e(k-1)$$

And K_1 , K_2 and K_3 are the constants. The limitation of using LFBC with ANN configuration is in the initial selection of values of the fixed constant K_1 , K_2 and K_3 to get the best performance. The constants K_1 , K_2 and K_3 are the basic design parameters of LFBC. The values of these constants can be obtained by trial and error procedure by observing the effect of these constants on the performance of the system.

Result

To evaluate the applicability of the controller, the performance of the controller has been studied on a simulated system.

Effect of Neural Network Weights Initialization for Non linear Application

Example 1

In this section neural controller is applied to a highly nonlinear CSTR system given in (Mitra and Pal 1996). A schematic of the CSTR system is shown in Fig. 4. A single irreversible, exothermic reaction $A \rightarrow B$ is assumed to occur in the reactor.

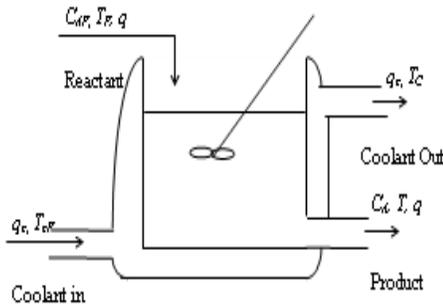


Fig. 4. Continuous Stirred Tank Reactor

Here objective is to control the effluent concentration by manipulating coolant flow rate in the jacket. Following differential equations given in Equation (3) describes the behavior of this CSTR:

$$\frac{dC_A}{dt} = \frac{q}{V} (C_{Af} - C_A) - k_0 C_A e^{\left(\frac{-E}{RT}\right)}$$

$$\frac{dT}{dt} = \frac{q}{V} (T_f - T) + \frac{\Delta H}{\rho C_p} \frac{k_0 C_A e^{\left(\frac{-E}{RT}\right)}}{q_c (1 + e^{\frac{hA}{\rho C_p T_c}}) T_{cf}} T$$

(3)

Where, C_{Af} is feed concentration, C_A is product concentration. T_f , T and T_c are feed, product and coolant temperature respectively. q and q_c are feed and coolant flow rate. Here temperature T is controlled by manipulating coolant flow rate q_c . Initially operating conditions are set to: $q=100$ lit/min, $C_{Af}=1$ mol/lit, $T_f=350$ K, $T_{CF}=350$ K, $V=100$ lit, $hA=7 \times 10^5$ cal/min K, $k_0=7.2 \times 10^{10}$ /min, $T=440.2$ K, $E/R=9.95 \times 10^3$ K, $-\Delta H=2 \times 10^5$ cal/mol, $\rho, \rho_c=1000$ gm/lit, $C_p, C_{pC}=1$ cal/gmK, $q_c=103.41$ lit/min, $C_A=8.36 \times 10^{-2}$ mol/lit

In Fig. 5, the set point tracking behavior of neural controller with unity weights initialization is shown.

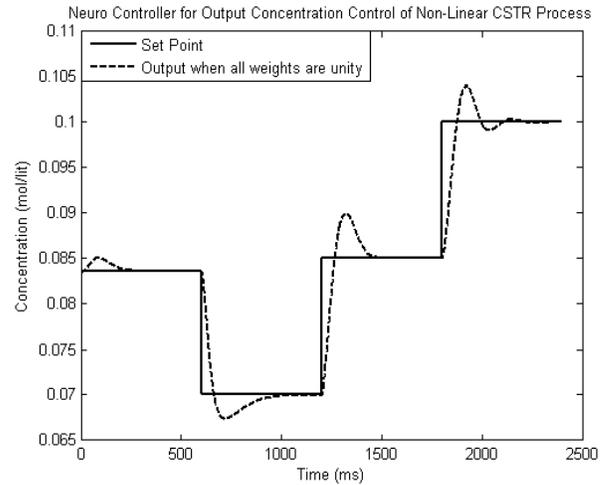


Fig. 5. Set point Tracking Performance of CSTR using DNC when Initial Weights of Network are 1

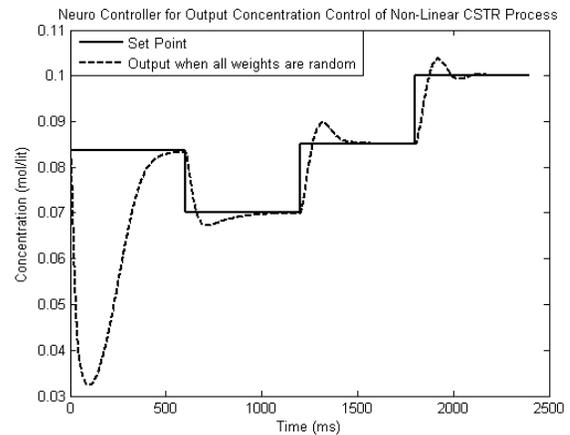


Fig. 6. Set point tracking performance of CSTR using DNC when initial weights of network are Random

In Fig. 6, the set point tracking behavior of neural controller with random weights in the range of 0 to 1 is shown. In order to complement the visual indications of performance for the simulation runs was made using ISE (integral of square errors) and IAE (integral of absolute error) criteria, which demonstrate the tracking ability of the system. Table I gives the ISE and IAE values for both the neural configurations.

Table I
Comparison Of Performance Of CSTR Process using DNC When Initial Neural Weights Are 1 And Random

Set point	All Initial Network Weights are 1		All Initial Network Weights are Random	
	ISE	IAE	ISE	IAE
0.0700	0.0050	0.8538	0.0046	0.8800
0.0836	0.0002	0.1850	0.2695	8.8873
0.0850	0.0075	1.0442	0.0071	1.0562
0.1000	0.0077	0.9988	0.0073	0.9363

In Fig. 7, the set point tracking behavior of neural controller with LFBC for unity weights initialization is shown and in Fig. 8, the set point tracking behavior of neural controller with LFBC for random weights in the range of 0 to 1 is shown.

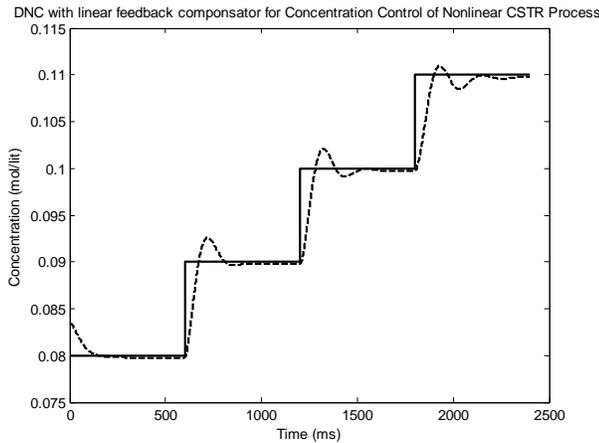


Fig. 7. Set point tracking performance of CSTR using Direct Neural Controller with LFBC when initial weights of network are 1

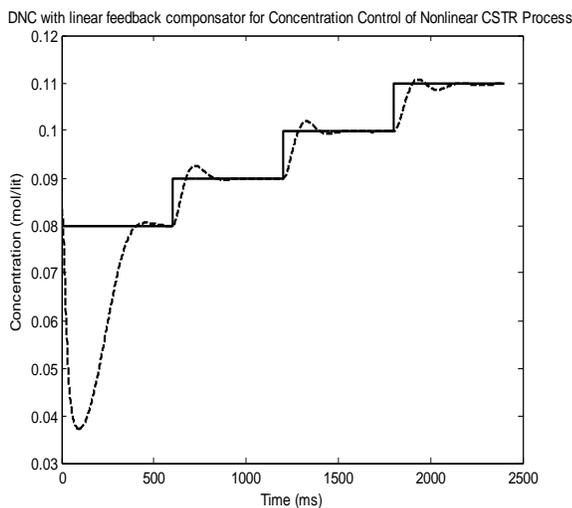


Fig. 8. Set point tracking performance of CSTR using Direct Neural Controller with LFBC when initial weights of network are Random

Table II gives the ISE and IAE values for both the neural configurations.

Table II

Comparison Of Performance Of CSTR Process, Using DNC With LFBC When Initial Neural Weights Are 1 And Random

Set point	All Initial Network Weights are 1		All Initial Network Weights are Random	
	ISE	IAE	ISE	IAE
0.08	1.9740	0.0817	1.1895	0.0398
0.09	1.9730	0.0816	1.1894	0.0397
0.10	1.9720	0.0815	1.1893	0.0396
0.11	1.4694	0.953	2.3870	0.0465

Effect of Neural Network Weights Initialization for linear Application

Example 2

In this section the neural controller is applied to a linear system. Here a DC motor is considered as a linear system from (Dorf and Bishop, 1998). A simple model of a DC motor driving an inertial load shows the angular rate of the load, $\omega(t)$, as the output and applied voltage, V_{app} , as the input. The ultimate goal of this example is to control the angular rate by varying the applied voltage. Fig. 9 shows a simple model of the DC motor driving an inertial load J .

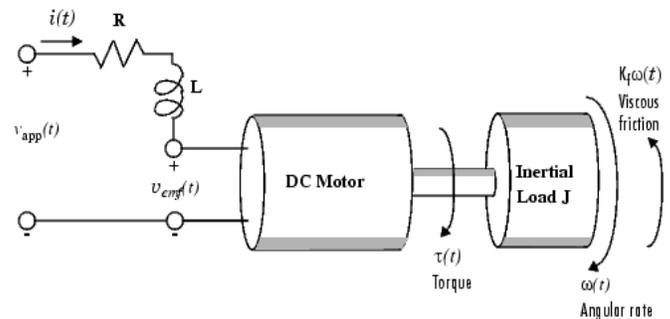


Fig. 9. DC motor driving inertial load

In this model, the dynamics of the motor itself are idealized for instance, the magnetic field is assumed to be constant. The resistance of the circuit is denoted by R and the self-inductance of the armature by L . The important thing here is that with this simple model and basic laws of physics, it is possible to develop differential equations that describe the behavior of this electromechanical system. In this example, the relationships between electric potential and mechanical force are Faraday's law of induction and Ampere's law for the force on a conductor moving through a magnetic field.

A set of two differential equations given in Equation (4) describes the behavior of the motor. The first for the induced current, and the second for the angular rate,

$$\begin{aligned} \frac{di}{dt} &= -\frac{R}{L} \cdot i(t) - \frac{K_b}{L} \cdot \omega(t) + \frac{1}{L} \cdot V_{app} \\ \frac{d\omega}{dt} &= -\frac{K_F}{J} \omega(t) + \frac{K_m}{J} \cdot i(t) \end{aligned} \quad (4)$$

Here objective is to control angular velocity ω by manipulating applied voltage, V_{app} . Initially operating conditions are set to: $R=2\Omega$, $L=0.5H$, $K_m=0.015$ (Torque Constant), $K_b=0.015$ (emf Constant), $K_F=0.2Nms$, $J=0.02 Kg.m^2/sec^2$.

In Fig. 10, the set point tracking behavior of neural controller with unity weights initialization is shown and in Fig. 11, the set point tracking behavior of neural controller with random weights in the range of 0 to 1 is shown.

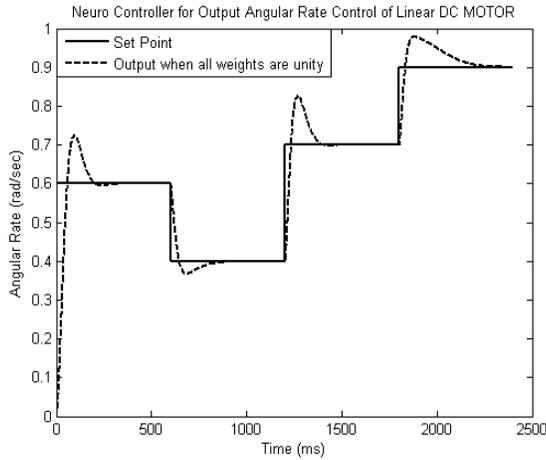


Fig. 10. Set point tracking performance of DC Motor using DNC when initial weights of network are 1

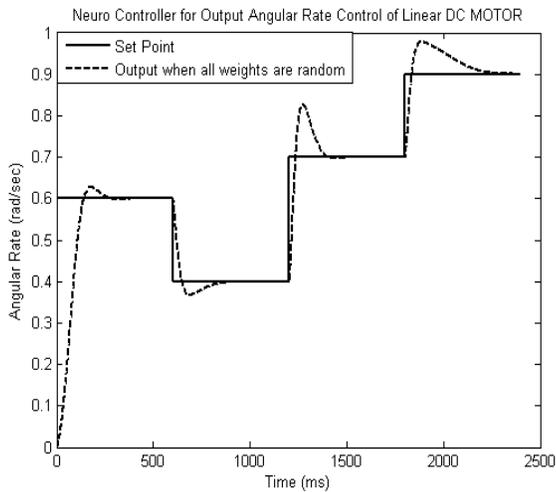


Fig. 11. Set point tracking performance of DC Motor using DNC when initial weights of network are Random

Table III gives the ISE and IAE values for both the neural configurations in DC Motor application.

Table III
Comparison Of Performance Of DC Motor Using DNC When Initial Neural Weights Are 1 And Random

Set point	All Initial Network Weights are 1		All Initial Network Weights are Random	
	ISE	IAE	ISE	IAE
0.4	0.648	7.926	0.676	8.208
0.6	8.437	27.356	18.490	44.684
0.7	2.049	15.303	2.159	16.108
0.9	1.365	19.294	1.426	20.049

In Fig. 12, the set point tracking behavior of neural controller with LFBC for unity weights initialization is shown. In Fig. 13, the set point tracking behavior of neural

controller with random weights in the range of 0 to 1 is shown.

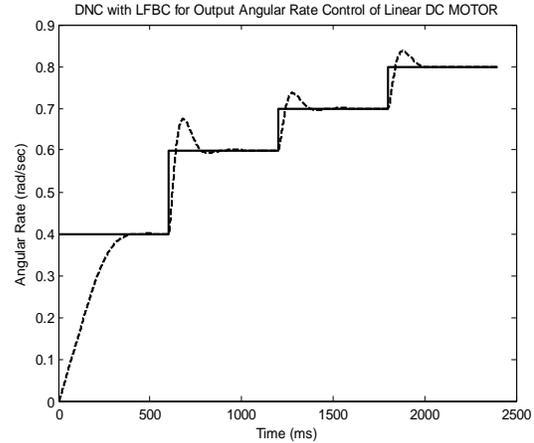


Fig. 12. Set point tracking performance of DC Motor using Direct Neural Controller with LFBC when initial weights of network are 1

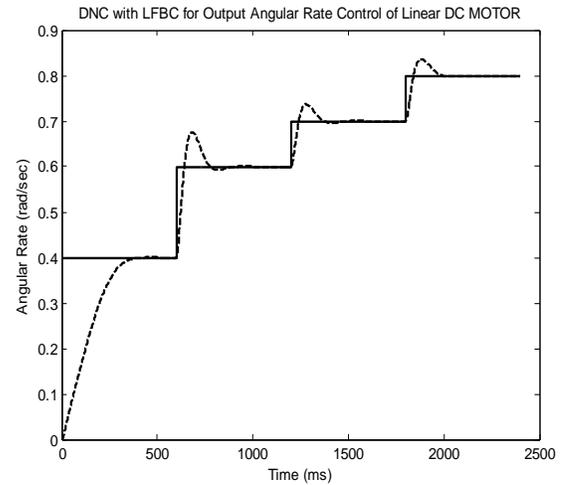


Fig. 13. Set point tracking performance of DC Motor using Direct Neural Controller with LFBC when initial weights of network are Random

Table IV gives the ISE and IAE values for both the neural configurations in DC Motor application.

Table IV
Comparison Of Performance Of DC Motor Using DNC With LFBC When Initial Neural Weights Are 1 And Random

Set point	All Initial Network Weights are 1		All Initial Network Weights are Random	
	ISE	IAE	ISE	IAE
0.4	0.0016	0.9941	0.0017	0.9958
0.6	0.0019	1.1598	0.0019	0.1618
0.7	0.0033	1.9883	0.0033	1.9917
0.9	0.0019	1.1598	0.0019	1.1618

Conclusion

In this paper, a Hybrid Direct Neural Control configuration has been proposed. A Linear Feedback Compensator is used to improve the performance of the Direct Neural Controller. The DNC and proposed HDNC have been tested on a nonlinear application of CSTR and a linear application of DC Motor. The performance of these two controllers was tested when neural networks are initialized with all unity parameters and random parameters. It is found that neural network with unity weight initialization is always better choice for any linear or nonlinear applications in DNC configuration while random weight initialization is better choice for nonlinear application using HDNC configuration. The unity or random weight initialization for linear application in HDNC configuration gives similar results. It is found that for all set point changes, neural controller with LFBC yields a fast response with little overshoots. In contrast with the direct neural controller has sluggish behavior for every set point. The test results of hybrid direct neural controller with linear feedback compensator are highly encouraging and establish the superiority of HDNC over the other controller being used in the process industry for linear as well as nonlinear systems.

References

- Nahas, E.P., Henson M.A. and Seborg D. E. 1992. Nonlinear internal model control strategy for neural network models, *Computers Chemical Engineering*, vol.16, pp. 1039-1057.
- Jacek M. Zurada, 2006. *Introduction to Artificial Neural Systems*, Jaico Publishing House.
- Dorf, R.C. and Bishop R.H. 1998. *Modern control systems*, Addison-Wesley, Menlo Park, CA.
- Mitra Sushmita and sankar K. Pal, 1996. Neuro-fuzzy expert systems: Relevance features and methodologies, *Journal of the IETE*, vol.42, no.4 and 5, pp.335-347.
- EI-Wang, He Feng, Xiao Zong, 2005. Smith predictive control based on NN, *IEEE proceeding of Machine, learning and cybernetic, Guangzhou*, p.p.4179-4183.
- Orlando De, Arjopsan P, Martin Hagan, 2001. A comparison of Neural Network algorithm, *IEEE proceeding*, p.p. 521-526.
- Yuan M., Poo G.S., 1995. Direct neural control system: nonlinear extension of adaptive control, *IEE Proceedings of Control Theory Applications*, vol. 142, No. 6, pp. 661-667.
- Jian-xun Peng, De-Shaung Huang, 2006. A hybrid forward algorithm for RBF neural network control, *IEEE transaction on neural network*, vol.17, issue 6, pp.1439-1451.
- Huang S .N, Lee T.H., 2002. A decentralize control of interconnected systems using neural network, *IEEE transaction on neural network*, vol.13, issue 6, pp.1554-1557.
- Castilo O., Melin P, 2002. Hybrid intelligent systems for time series prediction using neural networks, fuzzy logic and fractal theory, *IEEE transaction on neural network*, vol.13, issue 6, pp.1395-1408.
- Xianzhong cui, Kang G. Shin, 1993. Direct control and coordination using neural network, *IEEE transaction on systems, man and cybernetics*, Vol.23, No.3, pp 686-697.

This page is intentionally left blank.

***Special Session: Artificial Intelligence in
Biometrics and Identity Sciences II***

Chair: Gerry Dozier

Comparison of Genetic-based Feature Extraction Methods for Facial Recognition

#Joseph Shelton¹, #Gerry Dozier², #Kelvin Bryant³, #Lasanio Smalls⁴, #Joshua Adams⁵,

#Khary Popplewell⁶, #Tamirat Abegaz⁷, ^Damon L. Woodard⁸, ~Karl Ricanek⁹

*#Computer Science Department, North Carolina Agricultural and Technical State University
1601 East Market St. Greensboro, NC, 27411, United States of America*

^314 McAdams Hall, Clemson University, Clemson, S.C. 29634-0974, United States of America

~CIS Building, Room 2010, 601 South College Road, Wilmington NC, 28403

jashelt1@ncat.edu¹, gvdozier@ncat.edu², ksbryant@ncat.edu³, lrsmall@ncat.edu⁴, jcadams2@ncat.edu⁵,
tkpopple@ncat.edu⁶, tamirat@programmer.net⁷, woodard@clemson.edu⁸, ricanek@uncw.edu⁹

Abstract

In previous research, Shelton et al. presented a genetic-based method for evolving feature extractors for facial recognition. The technique presented evolved feature extractors that consisted of non-uniform, overlapping patches and did not cover the entire image. In this paper, we compare the use of non-uniform, overlapping patches with uniform, overlapping patches. Our results show a statistically significant performance improvement over the technique presented in Shelton's previous paper.

Introduction

Biometric recognition is the science of identifying an individual or group of individuals based on physical/behavioral characteristics or traits (Ross, 2007). One of the most popular biometric modalities is the face (Li and Jain, 2005; Ahonen, Hadid and Pietikinen 2006; Matas et al., 2002) and perhaps one of the more widely used techniques for extracting features from facial images for the purpose of biometric recognition is the Local Binary Pattern (LBP) method (Ojala and Pietikinen, 2002).

Shelton et al. introduces a genetic-based method, GEFE (Genetic & Evolutionary Feature Extraction), for evolving LBP feature extractors that consisted of non-uniformed, unevenly distributed patches that do not cover the entire image. The proposed method proved superior to the traditional LBP which uses uniform, evenly distributed, non-overlapping patches, that cover the entire image. In this paper, we introduce an alternative GEFE approach which is similar to the original GEFE approach with the exception that the unevenly distributed, overlapping patches are of uniform size.

The original GEFE method was theorized to have a bias due to the selection process of the coordinates for a patch. The coordinates represented the left corner of a patch, which increased the possibility of the patch dimensions exceeding the boundaries of a facial image. Any patch that exceeded the bounds was shifted till the whole patch was within the image space.

Because the possible coordinates could be anywhere on an image, the probability of selecting a patch that would just end up in the lower right hand corner was greater than any other location on the image. The method used in this research seeks to eliminate any potential bias by representing the coordinates of a patch as the center of it. This creates a greater probability of a patch being placed on all corners of an image.

The remainder of this paper is as follows: we will introduce the concept of GEFE for evolving LBP Feature Extractors composed non-uniform and uniform patches, as well as describing the genetic algorithm used in this research. We will discuss our experimental setup, we will show our results and finally we will present our conclusions and future work.

GEFE using Non-Uniform and Uniform Patch Sizes

LBP is a texture operator that can be used to extract texture information in the form of image features. The images used in this work are gray-scale, and are all facial images. For the standard LBP technique (Ojala and Pietikinen, 2002), a number of uniform, non-overlapping, and evenly distributed patches are used to cover an image. Texture features are then extracted from each patch area of the image.

Shelton et al. developed a genetic-based method for evolving LBP feature extractors that were composed of patches that were non-uniform, overlapping, unevenly distributed, and did not cover the entire image. Figure 1 provides an example of the patch layout of the standard LBP method (Figure 1a) and the original GEFE method (Figure 1b).



Figure 1a: Standard LBP

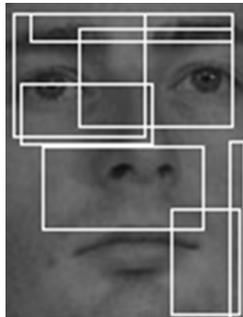


Figure 1b: GEFE

Figure 1: Gray Scale Images fitted with patches

Given a layout of patches for an LBP feature extractor, the LBP method is applied to each interior pixel within the patch. Each pixel has a value between 0 and 255 that represents the intensity of its gray level. When LBP is applied to the pixels of a patch, a histogram is created that represents the unique texture pattern for that particular patch. The histograms of every patch on the image are then concatenated to form a unique set of features that represents the image.

Figure 2 provides an example of the LBP method being applied to a particular pixel value for the center pixel with an intensity value of 120. The center pixel is surrounded by 8 neighboring pixels, shown in the 1st sample pattern in Figure 2. The differences are calculated and shown in the 2nd pattern.

Upon inspection of the 3rd pattern, one can see a series of zeros and ones. This pattern is created by taking the difference between each neighbor pixel and the center pixel. If the difference is negative, then the conversion value will be zero. If the difference is zero or greater, then the conversion value for that neighbor will be one. The third pattern is then ‘unwrapped’ to form a binary string and the string is converted to an integer number, which is the LBP value for that center pixel. For the center value in Figure 2, the LBP is 14 due to the sequence: 00001110. Where the binary string starts its unwrapping depends on the user, but this research starts the unwrapping process at the leftmost corner.

The number of possible binary patterns using 8 neighbors is 256. Each binary pattern is classified as either uniform or non-uniform. A uniform pattern is a bit string that has two or less bit changes (including the wrap-around from the last bit to the first bit). A non-uniform pattern is a bit string that has more than two bit changes (once again, including the wrap-around).

bit string that has more than two bit changes (once again, including the wrap-around).

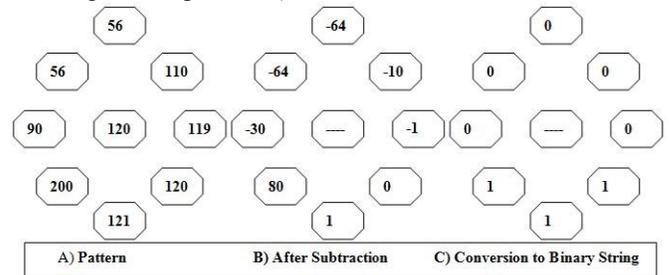


Figure 2: The LBP Method

The number of possible binary patterns using 8 neighbors is 256. Each binary pattern is classified as either uniform or non-uniform. A uniform pattern is a bit string that has two or less bit changes (including the wrap-around from the last bit to the first bit). A non-uniform pattern is a bit string that has more than two bit changes (once again, including the wrap-around). As shown in Figure 3, the uniform pattern has one change between the fourth and fifth bits, and one change between the eighth and first bits. Since the string wraps around, the last and first bits in the string must be compared. The non-uniform pattern has changes between the second and third bits, the third and fourth bits, the fifth and sixth bits, and the seventh and eighth bits. Out of the total 256 possible patterns, 58 of those patterns are uniform. Two of the 58 patterns are 00000000 and 11111111.

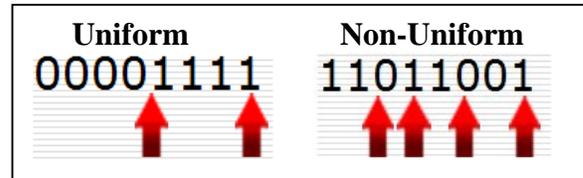


Figure 3: Uniform VS Non-Uniform

A subhistogram is created for every patch of an image, and is composed of 59 bins. Bins 1 to 58 correspond to the 58 possible uniform patterns using 8 neighbors. The 59th is a bin that holds the count of all non-uniform patterns found in the patch. The work of Ojala and Matti Pietikainen suggests that the most discriminating features of a facial image contain predominantly uniform patterns. The subhistograms associated with each patch are then concatenated to form a histogram representing the features extracted by the LBP method.

As in Shelton et al.’s previous research, this paper uses a steady-state GA (SSGA) to evolve a population of feature extractors (FE) (Davis, 1991; Fogel, 2000). In previous research, Shelton et al. evolved FEs consisting of non-uniform patches (not to be confused with the uniformity/non-uniformity of LBP patterns presented earlier).

Non Uniform GEFE

A candidate FE, fe_i , is a 6-tuple, $\langle X_i, Y_i, W_i, H_i, M_i, f_i \rangle$, where $X_i = \{x_{i,0}, x_{i,1}, \dots, x_{i,n-1}\}$ represents the x-coordinates of the center pixel of the n possible patches, $Y_i = \{y_{i,0}, y_{i,1}, \dots, y_{i,n-1}\}$ represents the y-coordinates of the center pixel of the possible patches, $W_i = \{w_{i,0}, w_{i,1}, \dots, w_{i,n-1}\}$ represents the widths of the n possible patches, $H_i = \{h_{i,0}, h_{i,1}, \dots, h_{i,n-1}\}$ represents the heights, $M_i = \{m_{i,0}, m_{i,1}, \dots, m_{i,n-1}\}$ represents the mask for each patch, and fit_i represents the fitness of fe_i . The mask is a vector that determines which patches are used when building the feature vector for an image. The purpose of masking out patches is to reduce the number of features that need to be compared when measuring similarity between images. The FE can create patches with non-uniform sizes, meaning that the widths and heights for each n patch can be unique. Given a probe set and a gallery set the fitness is the number of errors made when comparing each probe to the gallery multiplied by 10 plus the fraction of the n patches from which features were extracted (1).

$$fit_i = NumErrors * 10 + \left(\frac{\sum_{k=0}^{n-1} m_{i,k}}{n+1} \right) \quad (1)$$

Uniform GEFE

Candidate FEs consisting of patches with uniform patch size are similar with the exception that for any FE, fe_k , $W_k = \{w_{k,0}, w_{k,1}, \dots, w_{k,n-1}\}$ is of the form, $w_{k,0} = w_{k,1}, \dots, w_{k,n-2} = w_{k,n-1}$, meaning that the widths of every patch is the same. Similarly, $H_k = \{h_{k,0}, h_{k,1}, \dots, h_{k,n-1}\}$ is of the form, $h_{k,0} = h_{k,1}, \dots, h_{k,n-2} = h_{k,n-1}$, meaning that the height of every patch is the same.

Steady State Genetic Algorithm

The SSGA used to evolve candidate FEs works as follows. First a population of candidate FEs is randomly generated. Each candidate FE is then evaluated and assigned a fitness. After the initial population has been created, two parents are selected via binary tournament selection (Fogel, 2000, Abraham, Nedjah and Mourelle, 2006) and are used to create one offspring via uniform crossover and Gaussian mutation (Davis, 1991; Fogel, 2000; Kennedy and Eberhart 2001; Abraham, Nedjah and Mourelle, 2006). The offspring is then evaluated, assigned a fitness, and replaces the worst fit candidate FE in the population. The evolutionary process of selecting parents, creating a offspring, and replacing the worst fit FE in the population is repeated a user specified number of times. Figure 4 provides a pseudocode version of an SSGA.

```

compute SSGA{
t = 0;
initialize pop(t)
evaluate pop(t)
while(Not done){
    Parent1 = Select_From_Pop(t)
    Parent2 = Select_From_Pop(t)
    Child = Procreate(Parent1, Parent2)
    Evaluate(Child)
    Replace(Worst(Pop(t+1)), Child)
    t = t+1;
}
}

```

Figure 4: Pseudo-code for the GEFE SSGA

Experiment

We performed our experiment on a subset of 105 subjects taken from the Facial Recognition Grand Challenge (FRGC) dataset (Phillips et al., 2005). Each subject in the FRGC dataset has three slightly different images associated with it, as seen in Figure 5. Our dataset of 105 subjects consisted of a probe set (one image per subject), and a gallery set (two images per subject).

The probe set contains one of the images of each subject, and the gallery set contains the other two images for each of the subjects. Since our dataset contained 105 subjects, a total of 105 images were in the probe set and 210 images were in the gallery set. The dimensions of our images were 100 by 127 pixels.

For this experiment, we compared the Standard LBP method (SLBP), GEFE with non-uniform sized patches (GEFE_n), and GEFE with uniform sized patches (GEFE_u).



Figure 5: Subject 27's Snapshots

Results

For our results, an SSGA was used to evolve a population of 20 candidate feature extractors. The SSGA used uniform crossover and Gaussian mutation, (where the Gaussian $\mu = 0.1$). The SSGA was run 30 times for GEFE_u and GEFE_n. For each run, a total of 1000 function evaluations were allowed.

In Table I, the average performance of the three methods is shown. The SLBPM needed to be run only once since the patch characteristics were deterministic. GEFE_n used an average of 36.90% of patches, with an average accuracy of 99.84% while GEFE_u used an average of 35.82% of patches, with an average accuracy of 100%. Both GEFE_u

and $GEFE_n$ outperformed SLBPM in terms of accuracy while using a fewer number of features.

A t-test was used to confirm the observation that $GEFE_u$ had a statistically significant better performance (in terms of accuracy) than $GEFE_n$.

Research has been done that notes certain areas of a face to be discriminating enough to effectively distinguish between different persons (Matas et al., 2002). Figure 6 shows an approximate positioning of patches for the best feature extractors created using the $GEFE_n$ and the $GEFE_u$. For Figure 6b, the patches are meant to be the same size.

To avoid confusion, one of the patches was drawn in green.

It is interesting to see that the majority of patches are around the ocular region. Because the $GEFE_n$ and the $GEFE_u$ choose this region to focus on suggests that this area holds textures that are unique enough to differentiate individuals from one another. This result is consistent with conclusions presented in other research (Woodard et al., 2010; Miller et al., 2010).

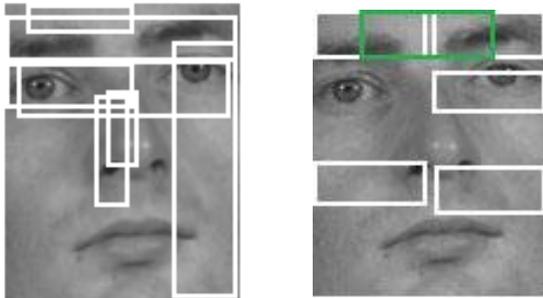


Figure 6a: SSGA Non_Uniform Figure 6b: SSGA Uniform

Figure 6: Best Individuals

Conclusion and future Work

In this paper, two forms of GEFE were compared (along with SLBPM). Both $GEFE_u$ and $GEFE_n$ had better performance than SLBPM. $GEFE_u$ had a better performance than $GEFE_n$. Our future work will be devoted toward the investigation and comparison of GEFE using a variety of other forms of Genetic and Evolutionary Computing. A second endeavor will be to use the smaller feature sets evolved by GEFE in an effort to develop hierarchical biometrics systems similar to the one proposed in Gentile's paper (Gentile, 2009).

Acknowledgments

This research was funded by the office of the Director of National Intelligence (ODNI), Center for Academic Excellence (CAE) for the multi-university, Center for Advanced Studies in Identity Sciences (CASIS) and by the National Science Foundation (NSF), Science & Technology Center: Bio/Computational Evolution in

Action Consortium (BEACON). The authors would like to thank the ODNI and the NSF for their support of this research.

TABLE I

Experimental results for LBP (even distribution) and SSGA Experiments

Methods	Patches Used	Avg. Accuracy	Best Accuracy
SLBPM	100.0%	99.04%	99.04%
$GEFE_n$	38.65%	99.68%	100.0%
$GEFE_u$	35.82%	100.0%	100.0%

References

- [1] Timo Ojala, Matti Pietikainen, *Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns*, IEEE Trans. Pattern Analysis and Machine Intelligence, 971-987; 2002
- [2] Arun Ross, *An Introduction To Multibiometrics*, Appeared in Proc. of the 15th European Signal Processing Conference (EUSIPCO), (Poznanm Poland), September 2007
- [3] S. Z. Li and A. K. Jain, Eds., *Handbook of Face Recognition*. Springer, 2005.
- [4] Damon L. Woodard Shrinivas J. Pundlik Jamie R. Lyle Philip E. Miller, *Periocular Region Appearance Cues for Biometric Identification In CVPR Workshop on Biometrics, 2010*
- [5] James E. Gentile, Nalini Ratha, Jonathan Connell *SLIC Short-Length Iris Codes*, in *Submission to Biometrics Theory, Applications and Systems (BTAS '09), 2010*
- [6] P. Miller, A. Rawls, S. Pundlik, and D. Woodard, *Personal identification using periocular skin texture*, in *SAC '10 Proceedings of the 2010 ACM symposium on Applied Computing*. New York, NY, USA: ACM, 2010.
- [7] L. Davis, *Handbook of Genetic Algorithms*. New York: Van Nostrand Reinhold, 1991.
- [8] D. Fogel, *Evolutionary Computation Toward a New Philosophy of Machine Intelligence*. IEEE Press, 2000.
- [9] J. Kennedy and R. Eberhart, *Swarm Intelligence*. Morgan Kaufmann, 2001.
- [10] Timo Ahonen, Abdenour Hadid, Matti Pietikinen, "Face Description with Local Binary Patterns Application to Face Recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, no. 12, pp. 2037-2041, Dec. 2006, doi:10.1109/TPAMI.2006.244.

- [11] Jiming Liu, Y. Y. Tang, Y. C. Cao, *An Evolutionary Autonomous Agents Approach to Image Feature Extraction* *IEEE Trans. Evolutionary Comput.* **1** 2 (1997), pp. 141–158.
- [12] Ajith Abraham, Nadia Nedjah, Luiza de Macedo Mourelle *Evolutionary Computation from Genetic Algorithms to Genetic Programming* “Studies in Computational Intelligence” Springer (2006), 1-20.
- [13] P. J. Phillips, P. J. Flynn, T. Scruggs, K. W. Bowyer, J. Chang, K. Hoff, J. Marques, J. Min, and W. Worek, “Overview of face recognition grand challenge,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2005.
- [14] Joseph Shelton, Kelvin Bryant, Joshua Adams, Khary Popplewell, Tamirat Abegaz, Kamilah Purrington, Damon L. Woodard, Karl Ricanek, Gerry Dozier, *Genetic Based LBP Feature Extraction and Selection for Facial Recognition* in *ACM Southeast Conference (ACMSE)*, 2011.
- [15] J. Matas, P. Břilek, M. Hamouz, and J. Kittler, *Discriminative Regions for Human Face Detection* in *The 5th Asian Conference on Computer Vision*, 23–25 January 2002, Melbourne, Australia
- [16] J. Gentile, N. Ratha, and J. Connell. *An efficient, two-stage iris recognition system* in *International Conference on Biometrics Theory, Applications, and Systems (BTAS '09)*, pages 1–5, 2009

Genetic-Based Selection and Weighting for LBP, oLBP, and Eigenface Feature Extraction

Tamirat Abegaz[#], Gerry Dozier[#], Kelvin Bryant[#], Joshua Adams[#], Brandon Baker[#], Joseph Shelton[#], Karl Ricanek[^], Damon L. Woodard^{*}

[#]North Carolina A&T State University

[^]The University of North Carolina at Wilmington

^{*}Clemson University

Abstract In this paper, we have investigated the use of genetic-based feature selection (GEFeS), genetic-based feature weighting (GEFeW) on feature sets obtained by Eigenface and LBP. Our results indicate that GEFeS and GEFeW enhances the overall performance of both the Eigenface and LBP-based techniques. Compared to Eigenface hybrid, our result shows that both LBP and oLBP hybrids perform better in terms of accuracy. In addition, the results show that GEFeS reduces the number of features needed by approximately 50% while obtaining a significant improvement in accuracy.

Keywords— Local Binary Pattern (LBP), Eigenface, Steady State Genetic Algorithm (SSGA), Overlapping Patches, Feature Selection.

I. INTRODUCTION

Feature Selection is a computational technique that attempts to identify a subset of features that are most relevant to a particular task (such as biometric identification) [1]. The ideal feature selection technique removes those features that are less discriminative and keeps those features that have high discriminatory power. A number of feature selection techniques have been developed and can be classified as: Enumeration Algorithms (EAs), Sequential Search Algorithms (SSAs), and Genetic Algorithms (GAs). EAs guarantee the optimal subset of features by evaluating all possible subsets of the features. This works well for a very small sized feature sets, however, it is computationally infeasible when the size of the feature set is large [2].

SSAs attempts to divide a feature set, U , into two subsets of features, X , and Y , where X denotes the selected features and Y denotes the remaining ones. Based on user specified criteria, SSAs select the least significant features from the subset X and moves those features into Y while selecting the most significant features from Y and moving them into X . While SSAs are suitable for small and medium size problems, they are too computationally expensive to use on large problems [2].

GAs attempt to find an optimal (or near optimal) subset of features for a specific problem [3, 4, 5, 6, 7, 8, 9, 10]. First, a number of individuals or candidate Feature Subsets (FSs) are generated to form an initial population. Each FS is then evaluated and assigned a fitness obtained from the evaluation

function specific to the problem at hand. Parents are then selected based on fitness. New FSs are produced from the selected parents by the processes of reproduction. Survivors are selected from the previous generation and combined with the offspring to form the next generation. This process continues for user specified number of cycles.

This work is an extension of the research performed by Abegaz et. al [10]. In their work, Abegaz et al. used Genetic and Evolutionary Feature Selection (GEFeS), GEFeS+ (which is a co evolutionary version of GEFeS) , and Genetic and Evolutionary Feature Weighting (GEFeW), Eigenface algorithm. In their work, Abegaz et. al. reported that Eigen GEFeS, Eigen GEFeS+, and Egen GEFeW enhanced the overall performance of the Eigenface method while reducing the number of features needed. Comparing Eigen GEFeS, Eigen GEFeS+, and Eigen GEFeW, they reported that Eigen GEFeW performed best in terms of accuracy even though it used a significantly larger number of features as compared to either Eigen GEFeS or Eigen GEFeS+. In this paper, we extend the work of Abegaz et. al compare GEFeS, GEFeS+, and GEFeW hybrids using Eigenface, LBP, and overlapped LBP (oLBP).

Our work is partly motivated by the research of Gentile et. al [11, 12]. Gentile et. al proposed a hierarchical two stage process to reduce the number of feature checks required for an iris based biometric recognition system. The claimed that a shorter representation of the iris template by pre aligning the probe to each gallery sample and generate a shortlist of match candidates. Our target is a similar system for Face Recognition (FR) based on short length biometric templates that are able to achieve higher recognition accuracies.

The remainder of this paper is as follows. Section II explains the feature extraction techniques used as input for the GEFeS, GEFeS+, and GEFeW. Section III provides an overview of GEFeS, GEFeS+, and GEFeW. Section IV presents our experiment, and in Section V we present our results. Finally, our conclusions and future work are presented in Section VI.

II. FEATURE EXTRACTION USING EIGENFACE, LBP, AND oLBP

In a typical biometric system, the task of sample acquisition and feature extraction are always performed [13]. Sample acquisition is the gathering of biometric traits such as fingerprints, iris scan, periocular images, or facial images.

From the acquired sample, feature extraction is performed to create a feature vector to be used for comparison. In the case of a facial biometric sample, Eigenface and LBP are commonly used feature extractors. For a typical feature extractor, the pre enrolled images (and their associated feature vectors) are stored in a database commonly referred to as gallery [13], while newly acquired images (and their feature vectors) are called probes [13].

For Eigenface based feature extraction [14], each image in the training dataset was converted into a single vector. This conversion is necessary because one needs a square matrix (transformation matrix or covariance matrix) to compute the Eigenvectors (Eigenfaces) and the Eigenvalues. The gallery images have been used to construct a face space spanned by the Eigenfaces. Each image is then projected into the face space spanned by the Eigenfaces. 560 discriminatory feature weights were extracted for each image and stored for the feature selection experiments.

For LBP based feature extraction [15, 16], an image is first divided into several patches (blocks) from which local binary patterns are extracted to produce histograms from every non border pixels. The histogram obtained from each patch is concatenated to construct the global feature histogram that represents both the micro patterns and their spatial location. In other words, the histograms contain description of the images on three different levels of localities. The first one indicates that the labels for histograms contain information about the pattern on a pixel level. Second, the summation of the labels obtained in the patch level to produce the information on a regional level. Finally, the histograms at the regional level are concatenated to produce the global descriptor of the image.

The standard LBP uses those labels which have at most one 0 1 and one 1 0 transitions when viewed as a circular bit string. Such labels are known as uniform patterns [17] For uniform pattern LBP, every patch (block) consists of $P(P - 1) + 3$ bins where $P(P - 1)$ represents the bins for the patterns with two transitions [18]. The remaining three bins represents the bins for the patterns with 0 transitions (all zeros (00000000) and all ones (11111111), and for all non uniform patterns (bin that represents more than two transitions) [18]. The total number of histogram is computed using the formula, $B(P(P - 1) + 3)$, where B represents the number of blocks and P represents the of sampling points. For our research, we use $P = 8$, and $B = 36$ to obtain a feature vector of 2124.

oLBP based feature extraction [18] is a variant of LBP that attempts to include the internal border pixels that are left out during the process of logical portioning on the standard LBP feature extraction method. This is done by logically overlapping the patches horizontally, vertically, and both horizontally and vertically with a one pixel overlap. This provides information to determine whether including the middle border pixels have impact on the recognition rate of the LBP based face recognition algorithm.

III. GEFES, GEFES+, AND GEFEW

GEFeS, GEFES+, and GEFEW were designed for selecting and/or weighting the most discriminatory features for recognition. GEFES, GEFES+, and GEFEW are instances of a Steady State GA(SSGA) with in eXplanatory Toolset for the Optimization Of Launch and Space Systems (X TOOLSS) [19]. In order to describe GEFES, consider the following feature vector.

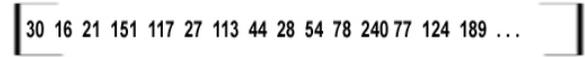


Figure 1: Sample feature vector

Furthermore, consider the vector shown in Figure 2 as a candidate real coded feature mask.



Figure 2: Real Coded Feature Mask

For GEFES a masking threshold value of 0.5 is used to create a binary coded candidate feature mask which will be used as condition for masking features. If the random real number generated is less than the threshold (0.5 in this case), then the value corresponding to the real generated number is set to 0 in the candidate feature mask vector or 1 otherwise. The candidate feature mask is used to mask out a feature set extracted for a given biometric modality. Figure 3 shows the candidate binary coded feature mask matrix obtained from the random real numbers generated in Figure 2. The masking threshold value is applied on the real numbers to obtain the binary representation



Figure 3: Binary coded candidate feature mask

When Comparing the candidate feature mask with the feature matrix, if a position corresponding to the feature matrix value in the candidate feature mask is 0 then that feature value will be masked out from being considered in the distance computation. Figure 4 shows the result of the features in Figure 1 when feature masking (Figure 3) is applied to a feature vector.



Figure 4: The Resulting feature vector after feature masking

GEFeS + is a co evolutionary version of GEFES where that instead of using the static threshold value of 0.5, we evolve a threshold value between 0 and 1. So each random number generated using a uniform distribution has a masking

threshold value that determines whether the feature corresponding to features is masked out or not.

For GEFeW, the real coded candidate feature mask is used to weight features within the feature matrix. The real coded candidate feature mask value is multiplied by each feature value to provide a weighted feature. If the number generated is 0 (or approximately equal to 0) the feature value is 0, which basically means that the feature is masked.

As given in Equation 1, the fitness returned by the evaluation function is the number of recognition errors encountered after applying the feature mask multiplied by 10 plus the percentage of features used. The selection of the parent is based on smaller fitness values because the optimization goal is to reduce the number of recognition errors (i.e. increasing the accuracy) while reducing the number of features.

$$fitness = (number\ of\ errors) * 10 + \%Features\ Used \quad (1)$$

IV. EXPERIMENTS

The dataset used in this research is a subset of the Face Recognition Grand Challenge (FRGC) dataset [20]. In our dataset, 280 subjects were used, with each subject having a total of 3 associated images with it. Out of 840 images, 280 were used as probe and 560 images were selected for training images. The images had passed the pre processing stages such as eye rotation alignment, histogram equalization, masking resizing (each with 225 by 195), and conversion of the images into greyscale.

For the GEFeS, GEFeS+, and GEFeW, the inputs used were the features extracted using Eigenface, LBP, and oLBP feature extraction methods. These methods were used on a subset of the FRGC dataset. This subset was selected because it contains a variety of imaging conditions such as different ethnic origins, frontal images that were neutral, and frontal images that had facial expressions.

The objective of this experiment is to compare the impact of applying GEFeS, GEFeS+, GEFeW on the Eigenface, LBP, and oLBP based feature extraction methods.

V. RESULTS

For our experiment, nine GEFeS, GEFeS+, GEFeW instances were used. These instances all have a population size of 20, Gaussian mutation rate of 1 and mutation range of 0.2. The Mutation rate value of 1 implies that all children (100%) must undergo mutation. The mutation range provides a window from the current value (obtained value after recombination) that the new value will be mutated. Furthermore, they were each run a total of 30 times with a maximum of 1000 function evaluations. GEFeS, GEFeS+, and GEFeW were designed for selecting and/or weighting the most discriminatory features for recognition. Our results are shown in Tables I.

In Table I, the columns represent the method used, the percentage of the average features, the average accuracy, and the best accuracy obtained. The percentage of the average

accuracy is computed using the results obtained from the 30 runs. The best accuracy is selected from the run that resulted in the smallest number of errors.

ANOVA and t Tests were used to divide the GEFeS, GEFeS+, GEFeW instances and the baseline algorithms into equivalence classes. As shown in Table 1, comparing the baseline algorithms, the Eigenface method performs best. The results show that when using 100 percent of the features, the maximum accuracy obtained for the baseline LBP was 70.36%. While the Baseline_{LBPbest} performs slightly better than the baseline Baseline_{LBP}, it still uses the entire feature set. As can be seen in Table 1, applying GEFeS on the feature set extracted by the standard LBP significantly improves accuracy from a 70.36% to 96.62%. This result shows that GEFeS is actually masking out those features which are less relevant for recognition. This improvement in accuracy comes also with a reduction in the number of features used for recognition.

TABLE I
EXPERIMENTAL RESULTS OF THE LBP BASELINE, oLBP AND THE EIGENFACE METHODS

Methods	Number of Features Used	% Accuracy	Best Accuracy
Baseline _{LBP}	2124	70.36	70.36
Baseline _{oLBPbest}	2124	70.71	70.71
Baseline _{Eigenface}	560	87.14	87.14
Eigen GEFeS	291.2	86.67	87.85
LBP GEFeS	1022.1	96.62	97.14
oLBP GEFeS	1018.46	96.43	96.79
Eigen GEFeS+	476	88.48	88.92
LBP GEFeS+	463.24	96.52	97.14
oLBP GEFeS+	446.89	96.50	97.14
Eigen GEFeW	492.8	91.42	92.5
LBP GEFeW	1865.29	95.33	95.71
oLBP GEFeW	1865.08	95.33	96.07

Compared to GEFeS and GEFeS+, all of the results show that GEFeW used a larger number of features. Using a larger number of features brings a better result in the case Eigen GEFeW as compared to Eigen GEFeS, and Eigen GEFeS+. Surprisingly, in the case of LBP GEFeW and oLBP GEFeW the result is the opposite. Utilizing a significantly larger number of features actually decreases the accuracy for both LBP GEFeW and oLBP GEFeW as compared to their corresponding methods.

LBP GEFeS, LBP GEFeS+, oLBP GEFeS, and oLBP GEFeS+ fall in the best equivalence class with respect to accuracy. This means that there is no statistical difference among them. All performed well in terms of reducing the number of features needed and in producing a significant improvement in accuracy from their corresponding baseline methods.

Figure 1 shows the Cumulative Match Characteristic (CMC) curve for the Baseline_{LBP}, Baseline_{oLBPbest}, Baseline_{Eigenface} and for the methods that fall in the first equivalent class. As can be seen from the Figure 1, LBP GEFeS, LBP GEFeS+, oLBP GEFeS, and oLBP GEFeS+ obtain approximately 97.5%

accuracy at rank 10. However, both $\text{Baseline}_{\text{Eigenface}}$ and Eigen GEFES performed well (approximately 96%) at rank 10. $\text{Baseline}_{\text{LBP}}$ performed relatively poorly in terms of accuracy.

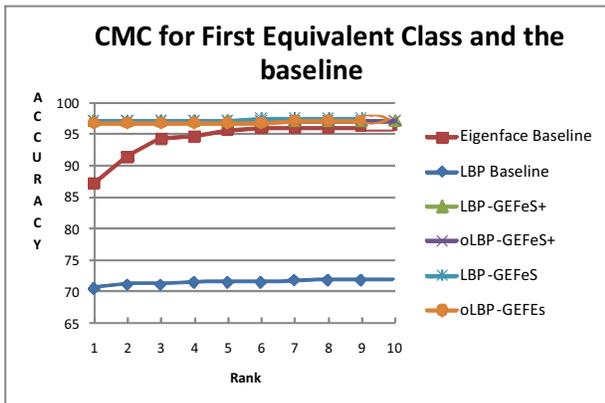


Figure 5: Comparisons of CMC results for baseline and the best performing algorithms

VI. CONCLUSION AND FUTURE WORK

Our results using GEFES, GEFES+, and GEFESW suggests that hybrid GAs for feature selection/weighting enhances the overall performance of the Eigenface, LBP, and oLBP methods while reducing the number of features needed. When comparing the baseline accuracy, the Eigenface method performed far better than both LBP and oLBP. However, the hybrid GAs result show that both LBP and oLBP hybrids performed much better than the Eigenface hybrid method.

Our future work will be devoted towards the investigation of GEFES, GEFES+, and GEFESW based on other forms of Genetic and Evolutionary Computation[21, 22, 23, 24]

ACKNOWLEDGMENT

This research was funded by the Office of the Director of National Intelligence (ODNI), Center for Academic Excellence (CAE) for the multi university Center for Advanced Studies in Identity Sciences (CASIS) and by the National Science Foundation (NSF) Science & Technology Center: Bio/computational Evolution in Action CONSORTIUM (BEACON). The authors would like to thank the ODNI and the NSF for their support of this research

REFERENCES

- [1] Ajay Kumar "ENCYCLOPEDIA OF BIOMETRICS" 2009, Part 6, 597-602, DOI: 10.1007/978-0-387-73003-5_157
- [2] M. Dash and H. Liu, Bartlett, Javier R. Movellan, and Terrence J. Sejnowski., "Feature Selection for Classification" *Genetic Algorithms for Feature Selection*, Intelligent Data Analysis, vol. 1, no. 3, pp. 131-156, 1997.
- [3] J. Adams, D. L. Woodard, G. Dozier, P. Miller, K. Bryant, and G. Glenn. *Genetic-based type II feature extraction for periocular biometric recognition: Less is more*. In Proc. Int. Conf. on Pattern Recognition, 2010. to appear.
- [4] Huang C. L. and Wang C. J. "GA-based feature selection and parameters optimization for support vector machines", *C.-L. Huang, C.-J. Wang / Expert Systems with Applications*. Vol. 31(2), 2006, pp231-240.

- [5] Adams, J., Woodard, D. L., Dozier, G., Miller, P., Glenn, G., Bryant, K. "GEFE: Genetic & Evolutionary Feature Extraction for Periocular-Based Biometric Recognition," Proceedings 2010 ACM Southeast Conference, April 15-17, 2010, Oxford, MS.
- [6] Dozier, G., Adams, J., Woodard, D. L., Miller, P., Bryant, K. "A Comparison of Two Genetic and Evolutionary Feature Selection Strategies for Periocular-Based Biometric Recognition via X-TOOLSS," Proceedings of the 2010 International Conference on Genetic and Evolutionary Methods (GEM'10: July 12-15, 2010, Las Vegas, USA).
- [7] Simpson, L., Dozier, G., Adams, J., Woodard, D. L., Dozier, G., Miller, P., Glenn, G., Bryant, K.. "GEC-Based Type-II Feature Extraction for Periocular Recognition via X-TOOLSS," Proceedings 2010 Congress on Evolutionary Computation, July 18-23, Barcelona, Spain.
- [8] Dozier, G., Bell, D., Barnes, L., and Bryant, K. (2009). "Refining Iris Templates via Weighted Bit Consistency", Proceedings of the 2009 Midwest Artificial Intelligence & Cognitive Science (MAICS) Conference, Fort Wayne, April 18-19, 2009.
- [9] Dozier, G., Adams, J., Woodard, D. L., Miller, P., Bryant, K. "A Comparison of Two Genetic and Evolutionary Feature Selection Strategies for Periocular-Based Biometric Recognition via X-TOOLSS", (to appear in) The Proceedings of the 2010 International Conference on Genetic and Evolutionary Methods (GEM'10: July 12-15, 2010, Las Vegas, USA).
- [10] Tamirat Abegaz, Gerry Dozier, Kelvin Bryant Joshua Adams, Khary Popplewell, Joseph Shelton, Karl Ricanek, Damon L. Woodard "Hybrid GAs for Eigen-Based Facial Recognition", accepted for IEEE Symposium Series in Computational Intelligence 2011 (SSCI 2011)
- [11] J.E Gentile, N. Ratha, and J. Connell, "SLIC: Short-length iris codes," In Proc. IEEE 3rd International Conference on Biometrics: Theory, Applications, and Systems, 2009. BTAS '09, 28-30 Sept. 2009, pp.1-5.
- [12] J.E. Gentile, N. Ratha, and J. Connell, "An efficient, two-stage iris recognition system", In Proc. 3rd International Conference on Biometrics: Theory, Applications, and Systems (BTAS), 2009.
- [13] Peter T. Higgins, "Introduction to Biometrics", The Proceeding of Biometrics consortium conference 2006, Baltimore", MD, USA, Sept. 2006.
- [14] M. Turk and A. Pentland, "Eigenfaces for recognition", *Journal of Cognitive Neuroscience*, Vol. 13, No. 1, pp. 71-86, 1991.
- [15] Caifeng Shan and Tommaso Gritti, " Learning Discriminative LBP-Histogram Bins for Facial Expression Recognition", Proc. of 15th EUSIPCO, Poznan, Poland, September 2007.
- [16] Goldberg, Toimo Ahonen, Abdenour Hadid, and Matti Pietik'ainen " Learning Face Expression Recognition", <http://www.ee.oulu.fi/mvg/>, visited on sept 10, 2120.
- [17] J. Zhao, H. Wang, H. Ren, and S. C. Kee," LBP discriminant analysis for face verification," in Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05), vol 3, pp. 167-172, June 2005.
- [18] Tamirat Abegaz, "GENETIC AND EVOLUTIONARY FEATURE SELECTION AND WEIGHTING FOR FACE RECOGNITION", thesis submitted to North Carolina A&T State University
- [19] M. L. Tinker, G. Dozier, and A. Garrett, "The exploratory toolset for the optimization of launch and space systems (x-toolss)", <http://xtoolss.msfc.nasa.gov/>, 2010.
- [20] P. Jonathon Phillips¹, Patrick J. Flynn², Todd Scruggs³, Kevin W. Bowyer² Jin Chang², Kevin Hoffman³, Joe Marques⁴, Jaesik Min², William Worek³, " Overview of the Face Recognition Grand Challenge", IEEE Conference on Computer Vision and Pattern Recognition, 2005.
- [21] Danial Ashlock. "Evolutionary Computation for Modeling and Optimization.", Springer, 2005.
- [22] Dozier, G., Homaifar, A., Tunstel, E., and Battle, D. (2001). "An Introduction to Evolutionary Computation" (Chapter 17), Intelligent Control Systems Using Soft Computing Methodologies, A. Zilouchian & M. Jamshidi (Eds.), pp. 365-380, CRC press.
- [23] D. Guillaumet, & J. Vitri'a, "Evaluation of distance metrics for recognition based on non-negative matrix factorization", *Pattern Recognition Letters*, 24(9-10), 2003, 1599-1605.
- [24] Fogel, D. *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*. IEEE Press, 2nd Edition., 2000.

Ethnicity Prediction Based on Iris Texture Features

Stephen Lagree and Kevin W. Bowyer

Department of Computer Science and Engineering
University of Notre Dame
Notre Dame, Indiana 46556 USA
slagree@nd.edu, kwb@cse.nd.edu

Abstract

This paper examines the possibility of predicting ethnicity based on iris texture. This is possible if there are similarities of the iris texture of a certain ethnicity, and these similarities differ from ethnicity to ethnicity. This sort of “soft biometric” prediction could be used, for example, to narrow the search of an enrollment database for a match to probe sample. Using an iris image dataset representing 120 persons and 10-fold person-disjoint cross validation, we obtain 91% correct Asian / Caucasian ethnicity classification.

Introduction

Iris texture has been shown to be useful for biometric identification and verification (Bowyer, Hollingsworth, and Flynn 2008; Phillips et al. 2005; Phillips et al. 2010; Daugman 2006). Studies have been done to determine if iris texture contains information that can determine “soft biometric” attributes of a person, such as ethnicity (Qiu, Sun, and Tan 2006; Qiu, Sun, and Tan 2007a) or gender (Thomas et al. 2007). This paper analyzes the possibility of ethnicity prediction based on iris texture. The ability of biometric systems to recognize the ethnicity of a subject could allow automatic classification without human input. Also, in an iris recognition system, an identification request includes a “probe” iris, which is checked against a “gallery” of enrolled images, to find the correct identity of the requested iris. One application of this feature is to narrow down the gallery of subjects to compare an iris to for identification purposes. In a system with millions of enrolled subjects, comparing an iris to all subjects could take an extremely long time. Narrowing down the gallery to only irises with the same ethnicity as the probe iris for comparison could give a great speed improvement.

Related Work

The CASIA biometrics research group has performed research on iris texture elements, including studies (Qiu, Sun, and Tan 2006; Qiu, Sun, and Tan 2007a; Qiu, Sun, and Tan 2007b) on determining ethnicity based on iris texture. To our knowledge, this is the only other work on predicting ethnicity from iris texture. In (Qiu, Sun, and Tan 2006), they report 86% accuracy in Asian / Caucasian classification. Thomas et al. (2007) suggests that the work in (Qiu, Sun, and Tan 2006) may be biased due to illumination differences in the two datasets the images were taken from, the Asian subject images coming from one dataset and the Caucasian subject images from another dataset. If one dataset was generally brighter or darker than the other, this factor could have entered into the learned algorithm for separating the subjects based on lighting, not iris texture. In the results presented in this paper, we eliminate this issue by using images taken from a single database to build our classifier, so that any acquisition setup differences are just as likely to appear in either ethnicity class. In (Qiu, Sun, and Tan 2007a), the CASIA group reports 91% accuracy in Asian / non-Asian ethnicity classification, using support vector machines and texton features. The dataset in this work is composed of 2,400 images representing 60 different persons, so that there are 20 images per iris. They divide the dataset into a 1,200-image training set and a 1,200-image test set, with training and test set not specified to be person-disjoint. In general, if iris images from the same person appear in both the training and the test set, then the performance estimate obtained is optimistically biased. In the results presented in this paper, we eliminate this issue by using a person-disjoint ten-fold cross-validation.

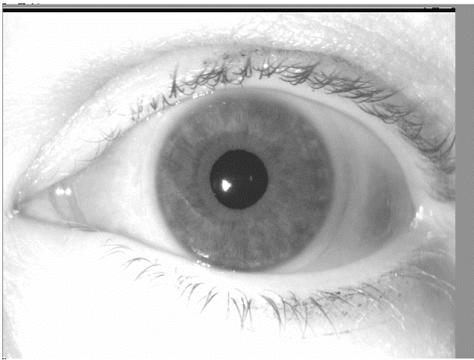
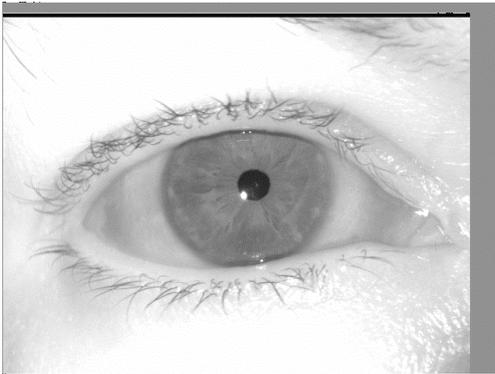
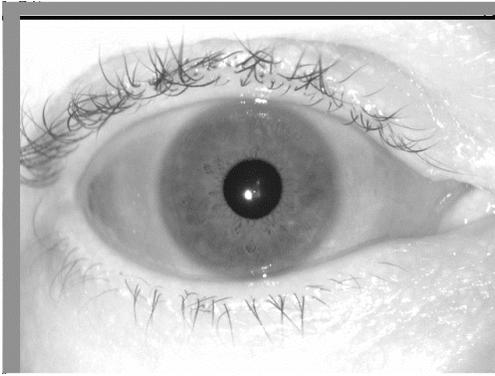


Figure 1 – Example LG 4000 Iris Images From Subjects with Caucasian Ethnicity (top: image 02463d1892; middle: image 04327d1264; bottom: image 04397d1461).

In a study of how human observers categorize images, Stark, Bowyer, and Siena (2010) found that humans perceive general differences in iris texture that can be used to classify iris textures into categories of similar texture pattern. Observers grouped a set of 100 iris images into categories of similar texture. The 100 images represented 100 different persons, and the 100 persons were balanced on gender and on Asian / Caucasian ethnicity. The

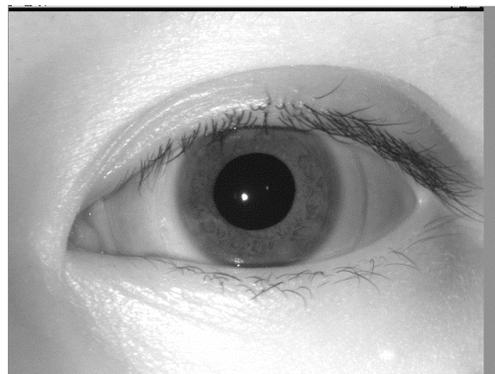
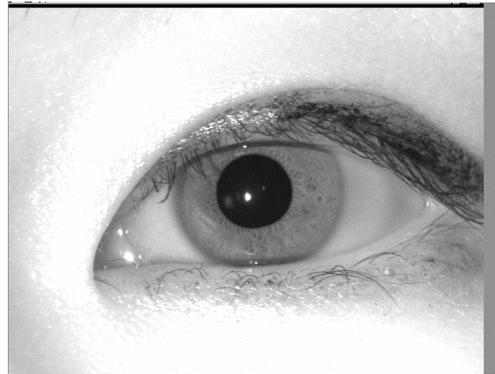
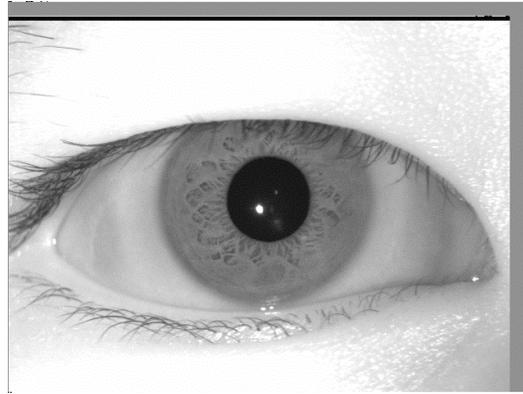


Figure 2 – Example LG 4000 Iris Images From Subjects with Asian Ethnicity (top: image 04815d908; middle: image 04629d1385; bottom: image 05404d80).

observers did not know the gender or ethnicity of the persons in the iris images. However, the grouping of images into categories of similar iris texture resulted in categories that were, on average, split 80% / 20% on ethnicity. The same categories were on average divided much more closely to 50% / 50% on gender. Thus, one result of Stark's work (2010) is that human observers perceive consistent ethnicity-related differences in iris

texture. In this paper, we want to train a classifier to explicitly perform the sort of ethnicity classification that was found as a side effect of the texture similarity grouping done by humans in (Stark, Bowyer, and Siena 2010) and that was previously explored in (Qiu, Sun, and Tan 2006; Qiu, Sun, and Tan 2007a).

Dataset

We want to see how accurately we can identify ethnicity based on iris texture. For this study we will use two ethnicity classes, Caucasian and Asian. This study used 1200 iris images selected from the University of Notre Dame’s iris image database. (This is a newer database than was released to the iris biometrics research community for the government’s Iris Challenge Evaluation (ICE) program (Phillips et al. 2005; Phillips et al. 2010).) All images were obtained using an LG 4000 sensor at Notre Dame. As with all commercial iris biometrics systems that we are aware of, the images are obtained using near-infrared illumination, and are 480x640 in size. One half of the images, 600, were of persons whose ethnicity is classified as Asian and the other half were from persons classified as Caucasian. For each ethnicity, the 600 images represented 60 different persons, with 5 left iris images and 5 right iris images per person. This 1,200-image dataset was randomly divided into 10 folds of 120 images each, with 6 persons of each ethnicity in each fold. Thus the images in the folds are person-disjoint; that is, each person’s images appear in just one fold.

Segmentation

For this iris texture prediction study, we want to base our findings solely on iris texture. Therefore we exclude periocular clues that might be used as an indicator of ethnicity. We segment the images to obtain the region of interest, and mask out the eyelid-occluded portions of the iris. We use Notre Dame’s IrisBee software to perform the segmentations (Phillips et al. 2005). The output from IrisBee that we use for texture examination is a 240x40 pixel normalized iris image along with the corresponding bitmask of eyelid and eyelash occlusion locations. The image segmentation and masking are exactly those that would be used by IrisBEE in processing the images for biometric recognition of a person’s identity. However, the normalized images are not processed by the log-Gabor filters that are used by IrisBEE to create the “iris code” for biometric recognition. We create a different texture feature vector for ethnicity prediction.

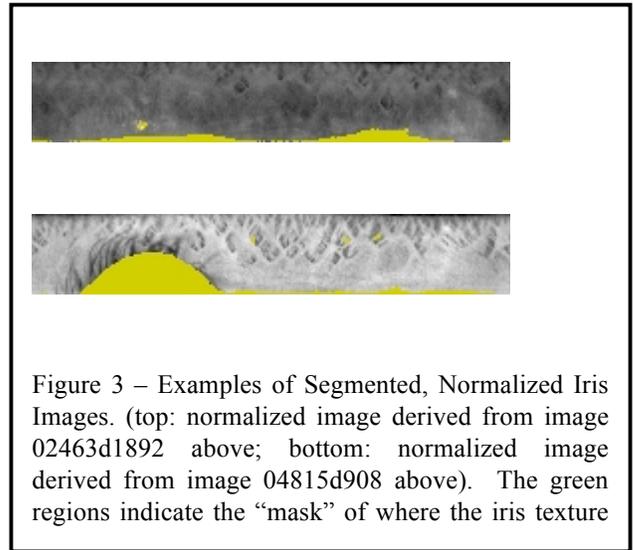


Figure 3 – Examples of Segmented, Normalized Iris Images. (top: normalized image derived from image 02463d1892 above; bottom: normalized image derived from image 04815d908 above). The green regions indicate the “mask” of where the iris texture

Feature Generation

After an image is segmented and normalized, we compute texture features that can be used in training a classifier to categorize images according to ethnicity. To do this we apply different filters to the image at every non-masked pixel location, and use the results of the filter to build a feature vector. Six of the filters we have used are “spot detectors” and “line detectors” of various sizes, as depicted in Tables I to VI. For a given point in the image, if applying a given filter would result in using any pixel that is masked, then that filter application is skipped for that point. The rest of the filters, depicted in Tables VI-VIII, were created using Laws’ Texture Measures (Laws 1980). These are designed to give responses for various types of textures when convolved with images.

A feature vector that describes the texture is computed for each iris image. We divided the normalized image array into a number of smaller sections in order to compute statistics for sub-regions of the normalized image. This is so that classification could be based on, for example, relative differences between the band of the iris nearer the pupil versus the band of the iris furthest from the pupil. These regions were ten four-pixel horizontal bands and four 60-pixel vertical bands of neighboring pixels in the normalized iris image. The ten horizontal bands correspond to concentric circular bands of the iris, running from the pupil out to the sclera (white) of the eye. The four vertical bands correspond roughly to the top, right, bottom and left parts of the iris. Since the filters are looking for different phenomena in the image, we find statistics for the filter response of each image. Each image contains 630 features, with 5 statistics calculated for each of the 9 filters on all of the 14 regions. The five statistics are: (1) average value of filter response, (2) standard

deviation of filter response, (3) 90th percentile value of filter response, (4) 10th percentile value of filter response, and (5) range between 90th and 10th percentile value. The motivation for using the average value is to represent the strength of a given spot size or line width in the texture. The motivation for using the standard deviation is to represent the degree of variation in the response. The motivation for using the percentiles and range is to have an alternate representation of the variation that is not affected by small amounts of image segmentation error.

TABLE I: Small Spot Detector Filter

-1/8	-1/8	-1/8
-1/8	+1	-1/8
-1/8	-1/8	-1/8

TABLE II: Large Spot Detector Filter

-1/16	-1/16	-1/16	-1/16	-1/16
-1/16	+1/9	+1/9	+1/9	-1/16
-1/16	+1/9	+1/9	+1/9	-1/16
-1/16	+1/9	+1/9	+1/9	-1/16
-1/16	-1/16	-1/16	-1/16	-1/16

TABLE III: Vertical Line Detector Filter

-1/20	-1/20	+1/5	-1/20	-1/20
-1/20	-1/20	+1/5	-1/20	-1/20
-1/20	-1/20	+1/5	-1/20	-1/20
-1/20	-1/20	+1/5	-1/20	-1/20
-1/20	-1/20	+1/5	-1/20	-1/20

TABLE IV: Wide Vertical Line Detector Filter

-1/10	+1/15	+1/15	+1/15	-1/10
-1/10	+1/15	+1/15	+1/15	-1/10
-1/10	+1/15	+1/15	+1/15	-1/10
-1/10	+1/15	+1/15	+1/15	-1/10
-1/10	+1/15	+1/15	+1/15	-1/10

TABLE V: Horizontal Line Detector Filter

-1/20	-1/20	-1/20	-1/20	-1/20
-1/20	-1/20	-1/20	-1/20	-1/20
+1/5	+1/5	+1/5	+1/5	+1/5
-1/20	-1/20	-1/20	-1/20	-1/20
-1/20	-1/20	-1/20	-1/20	-1/20

TABLE VI: Wide Horizontal Line Detector

-1/10	-1/10	-1/10	-1/10	-1/10
+1/15	+1/15	+1/15	+1/15	+1/15
+1/15	+1/15	+1/15	+1/15	+1/15
+1/15	+1/15	+1/15	+1/15	+1/15
-1/10	-1/10	-1/10	-1/10	-1/10

TABLE VII: S5S5

+1	0	-2	0	1
0	0	0	0	0
-2	0	+4	0	-2
0	0	0	0	0
+1	0	-1	0	+1

TABLE VIII: R5R5

-1	-4	6	-4	+1
-4	+16	-24	+16	-4
6	-24	+36	-24	+6
-4	+16	-24	+16	-4
+1	-4	+6	-4	+1

Results

We tried a variety of different classification algorithms included in the WEKA package (Weka). This included using meta-algorithms like Bagging with other classifiers. By changing parameters, we achieved performance gains on some of the algorithms. However, we found our best results using the SMO algorithm with the default parameters in WEKA for classification. The SMO algorithm implements “Sequential Minimal Optimization”, John Platt’s algorithm for building a support vector machine classifier (Weka). The input to the SMO algorithm is the feature vectors of all 1200 iris images that we have computed. To assess the results of our classifier we use cross-fold validation with ten folds using stratification based on ethnicity. These folds are also subject-disjoint to ensure the persons whose images are in the test data have not been seen by the classification algorithm in the training data.

The SMO classifier results in higher accuracy compared to a broad range of other classifiers, including decision tree based algorithms and bagging. Using Bagging on the top two classifiers, SMO and Random Forest, did not improve performance. Running the experiment with the SMO classifier and the feature vector as described above gives us an accuracy of 90.58%. This is good accuracy, representing an improvement on the 86% reported in (Qiu, Sun, and Tan 2006) and close to the 91% reported in (Qiu, Sun, and Tan 2007a) for a train-test split that was not

person-disjoint. When we do not use person disjoint results, we see an accuracy of 96.17%, which is significantly higher than Qiu, Sun, and Tan (2006; 2007a) reported.

We computed the classification accuracy for each feature separately to see the impact of individual features. Table X shows that some of the single features have almost have the performance of all of the features together. However none of them do as well as the combination of all of the features. Some filters may be redundant; a combination of a few might reproduce the performance of all nine filters.

To ensure that the size of our training dataset was not limiting our accuracy levels, we ran the classifier with different numbers of folds. Table XI shows the results we achieved using 5, 10, and 20 fold cross validation. The accuracy levels are all within one percent, indicating that our performance should not be limited by our dataset size.

TABLE IX: Results for Different Classifiers

Algorithm	Accuracy (%)
SMO	90.58
RandomForest (100 Trees/Features)	89.50
Bagged FT	89.33
FT	87.67
ADTree	85.25
J48Graft	83.67
J48	83.08
Naïve Bayes	68.42

TABLE X: Feature Performance with SMO

Feature	Accuracy (%)
Small Spot Detector	85.58
Large Spot Detector	85.67
Vertical Line Detector	87.42
Wide Vertical Line	85.50
Horizontal Line Detector	78.92
Wide Horizontal Line Detector	78.33
S5S5	78.17
R5R5	73.33
E5E5	88.0
All Features	90.58

TABLE XI: SMO By Number of Folds Used in Cross Validation

Folds	Accuracy (%)
5	90.00
10	90.583
20	90.1667

TABLE XII: SMO Accuracy By Fold Using 10 Fold Cross Validation

Fold	Accuracy (%)
1	91.667
2	100.000
3	88.333
4	90.833
5	97.500
6	82.500
7	98.333
8	90.000
9	87.500
10	79.167
Average	90.583

Future Work

To achieve even greater accuracy, we intend to implement additional and more sophisticated features, and to look at the effects of the size of the training set. We envision that the number of different persons represented in the training data is likely to be more important than the number of images in the training set; that is, doubling the training set by using twice as many images per person is likely not as powerful as doubling the number of persons.

For this experiment, we only looked at very broad ethnicity classifications. More work could be done to examine finer categories, such as Indian and Southeast Asian. The performance of a classifier such as this has not been tested on subjects of multiple ethnic backgrounds either.

Acknowledgments

This work is supported by the Technical Support Working Group under US Army contract W91CRB-08-C-0093, and by the Central Intelligence Agency. The opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of our sponsors.

References

- Bowyer, K.W.; Hollingsworth, K.; and Flynn, P. J. Image Understanding for Iris Biometrics: A Survey, *Computer Vision and Image Understanding*, 110(2), 281-307, May 2008.
- Daugman, J., Probing the Uniqueness and Randomness of Iris Codes: Results From 200 Billion Iris Pair Comparisons, *Proceedings of the IEEE*, Nov. 2006, 94 (11), 1927 – 1935.
- Laws, K. Textured Image Segmentation, Ph.D. Dissertation, University of Southern California, January 1980.

Phillips, P. J.; Bowyer, K. W.; Flynn, P.J; Liu, X; and Scruggs, T. W. The Iris Challenge Evaluation 2005, *Biometrics: Theory, Applications and Systems (BTAS 08)*, September 2008, Washington, DC.

Phillips, P. J.; Scruggs, W. T.; O'Toole, A.; Flynn, P.J.; Bowyer, K.W.; Schott, C. L.; and Sharpe, M. FRVT 2006 and ICE 2006 Large-Scale Experimental Results, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32 (5), May 2010, 831-846..

Stark, L; Bowyer, K.W.; and Siena,S. Human perceptual categorization of iris texture patterns, *Biometrics Theory, Applications and Systems (BTAS)*, September 2010.

Thomas, V; Chawla, N; Bowyer, K. W.; and Flynn, P. J. Learning to predict gender from iris images. In *Proc. IEEE Int. Conf. on Biometrics: Theory, Applications, and Systems*, Sept 2007.

Qiu, X. C.; Sun, Z. A.; and Tan, T. N. Global texture analysis of iris images for ethnic classification. In *Springer LNCS 3832: Int. Conf. on Biometrics*, pages 411-418, June 2006.

Qiu, X. C.; Sun, Z. A.; and Tan, T. N. Learning appearance primitives of iris images for ethnic classification. In *Int. Conf. on Image Processing*, pages II: 405-408, 2007a.

Qiu, X. C.; Sun, Z. A.; and Tan, T. N. Coarse iris classification by learned visual dictionary. In *Springer LNCS 4642: Int. Conf. on Biometrics*, pages 770-779, Aug 2007b.

Weka 3. <http://www.cs.waikato.ac.nz/ml/weka/>.

Author Index

- Abegaz, Tamirat86, 216, 221
Adams, Joshua86, 216, 221
Ahmad, Mohammad133
Alford, Aniesha86
Amirjavid, Farzad188
Andonie, Răzvan146
Baker, Brandon221
Barfouroush, Ahmad Abdollahzadeh195
Baryamureeba, Venansius73
Benze, James62
Bian, Haiyun128
Bockhorst, Joseph16
Bouchard, Bruno188
Bouchard, Kevin188
Bouzouane, Abdenour188
Bowyer, Kevin W.99, 225
Bryant, Kelvin86, 216, 221
Carroll, Thomas E.114
Chidrawar, Sadhana K.208
Connaughton, Ryan99
Doyle, James S., Jr.91
Dozier, Gerry86, 216, 221
Flynn, Patrick J.91, 99
Francis, David160
Giese, Andrew67
Graesser, Arthur C.23
Hammell, Robert J., II47
Haning, Jacob133
Hayashi, Isao2
Hossain, Shamina114
Hu, Yanlong133
Hughes, Cameron172
Hughes, Tracey172
Inoue, Atsushi8, 40, 202
Jack, Atticus160
Janning, Michael133
Kelly, John86
Knaap, Esther van der120, 160
Korukonda, Harika79
Kudoh, Suguru N.2
Lagree, Stephen225
Lambert, Matthew133
Laughlin, Andrew40
Lazar, Alina108
Lőrentz, István146
Malița, Mihaela146
Mazlack, Lawrence J.54
McQuighan, Joseph M.47
Moldovan, Cristian23
Munimadugu, Hareendra138
Neorr, Peter114
Njanji, Itai160
Nsang, Augustine S.30
Oldfather, Chad16
Olson, Joshua40
Patre, Balasaheb M.208
Paulson, Patrick114
Phillips, Joseph154
Poplewell, Khary86, 216
Purdy, Carla79

Ralescu, Anca	30, 120, 138, 142
Ramsay, Brian	120
Ricanek, Karl	216, 221
Rodriguez, Gustavo	160
Rus, Vasile	23
Schlittenhart, Isaac	202
Schwartz, Daniel G.	166
Seethakkagari, Swathi	142
Seitzer, Jennifer	62, 67
Shellito, Bradley A.	108
Shelton, Joseph	86, 216, 221
Simpson, Donny	40
Sivaraman, Chitra	114
Smalls, Lasanio	216
Snyder, Bennjamin	133
Springer, Kyle	202
Strecker, Jaymie	160
Thomas, Josh	133, 160
Unwin, Stephen D.	114
Ustymenko, Stanislav	166
Vafadar, Shiva	195
Vanderbeck, Scott	16
Vandiver, Whitney R.	178
Visa, Sofia	120, 133
Wakabi-Waiswa, Peter P.	73
White, Ethan	54
Winters, Jason	202
Woodard, Damon L.	216, 221
Zier, Brian	8