Proceedings of the 2nd Workshop on Semantic Personalized Information Management: Retrieval and Recommendation

SPIM 2011

Workshop Organizers and Program Chairs

Marco de Gemmis
Department of Computer Science,
University of Bari "Aldo Moro", Italy.

Ernesto William De Luca
School IV – Electrical Engineering and Computer Science,
Berlin Institute of Technology, Germany.

Tommaso Di Noia
Electrical & Electronics Engineering Department,
Technical University of Bari, Italy.

Aldo Gangemi
Italian National Research Council (ISTC-CNR),
Institute for Cognitive Sciences and Technology, Italy.

Michael Hausenblas
National University of Ireland (NUIG), Galway.
DERI – Digital Enterprise Research Institute, Ireland.

Pasquale Lops
Department of Computer Science,
University of Bari "Aldo Moro", Bari, Italy.

Thomas Lukasiewicz
Department of Computer Science,
University of Oxford, United Kingdom

Till Plumbaum
School IV – Electrical Engineering and Computer Science,
Berlin Institute of Technology, Germany.

Giovanni Semeraro
Department of Computer Science,
University of Bari "Aldo Moro", Italy.

These are the Proceedings of the *2nd Workshop on Semantic Personalized Information Management: Retrieval and Recommendation (SPIM 2011)*, held in conjunction with the *10th International Semantic Web Conference (ISWC 2011*). The workshop aims at improving the exchange of ideas between the different communities involved in the research on semantic personalized information management and covers a wide range of interdisciplinary topics: semantic social web, machine learning hybridized with semantics for personalization, techniques for (semantic) user modeling, recommender systems, personalized information retrieval, semantic interaction, use of semantic technologies in UI/HCI, linked data consumption for PIM, semantic search and exploratory browsing.

The workshop received an enthusiastic feedback from the SPIM community with a total of 20 submitted papers. 13 papers have been accepted and this highlights an increasing interest in the workshop topics. Indeed, during the first workshop edition in 2010, 7 papers were presented. This is a clear indication that "semantic personalized information management" is a very interesting and timely topic.

The set of accepted papers substantially covers the proposed topics, with some additional specific subjects: folksonomies, interaction and knowledge patterns for automatic explanation, CMS, business intelligence, etc. We can coarsely group the 13 accepted papers as follows:

Recommendation and classification:

- *Improving Tag-based Resource Recommendation with Association Rules on Folksonomies*
- *Finding similar research papers using language models*
- *Towards Ranking in Folksonomies for Personalized Recommender Systems in E-Learning*
- *User's food preference extraction for cooking recipe recommendation*
- *Performance Measures for Multi-Graded Relevance*
- *A Dimensionality Reduction Approach for Semantic Document Classification*
- *Personalized Filtering of Twitter Stream*

User modelling

- *Classifying Users and Identifying their Interests in Folksonomies*
- *User Modeling for the Social Semantic Web*

Various PIM support

- *Personalization in Skipforward, an Ontology-Based Distributed Annotation System*
- *A Model for Assisting Business Users along Analytical Processes*
- *A Privacy Preference Manager for the Social Semantic Web*
- *User-sensitive Explanations under a Knowledge Pattern Lens*

In the following, we summarize the background motivation for the scientific and practical relevance of the workshop.

Motivation

Finding and managing information is a crucial task in our everyday life, and especially on the Web, the user is confronted with a huge amount of information. Therefore, search engines have become an essential tool for the majority of users for finding information on the Web.

While search engines implementing the canonical search paradigm are adequate for most ad-hoc keyword-based retrieval tasks, they reach limits when user needs have to be satisfied in a personalized way. Today’s search engines have a very limited consideration of individual user’s preferences or context given by previous searches for distinguishing the relevance of a document with respect to the meaning of a user query (experiences so far seem restricted to massive log analyses and experimental things like Google Squared, which however does not address

personalization). With the advent of the Semantic Web, new opportunities emerge for semantic information retrieval systems to better match user needs. Next-generation search engines should implement a novel search paradigm, where the user perspective is completely reversed: *from finding to being found*. *Recommender Systems* may help to support this new perspective, because they have the effect of pushing relevant objects to potentially interested users. An emerging approach is to use Web 2.0 and Semantic Web technologies to model information about users, their needs and preferences, their context and relations, and to incorporate data from other resources like Linked Open Data (http://linkeddata.org). This data might be useful to interlink diverse information about users, items, and their relations and implement reasoning mechanisms that can support and improve the search and recommendation process, better satisfying the users' information need.

A new generation of systems is emerging, *which fully understand the items they deal with*, and new methods for modelling user information, combining user content and Semantic Web resources, as well as new algorithms for processing that data, are thus needed.

Why the topic is of particular interest at this time

More and more real-world applications in different areas are going to integrate recommender systems to personalize retrieval issues, results, and in general the user interaction.

Successful workshops and international conferences in the last few years (ACM Recommender Systems, User Modelling, AAAI, ECAI, IJCAI, SIGIR) show the growing interest and research potential of these systems. Recent developments of the Semantic Web community offer novel strategies to represent data about users, items and their relations that might improve the current state of the art of search and recommendation systems.

The challenge is to investigate *whether* and *how* this large amount of wide-coverage and linked semantic knowledge can significantly improve the search/recommendation process in those tasks that cannot be solved merely through a straightforward matching of queries and documents.

We wish to thank all authors who submitted papers and all workshop participants for fruitful discussions. We would like to thank the program committee members and external referees for their timely expertise in carefully reviewing the submissions.

October2011
The workshop chairs

Marco de Gemmis
Ernesto William De Luca
Tommaso Di Noia
Aldo Gangemi
Michael Hausenblas
Pasquale Lops
Thomas Lukasiewicz
Till Plumbaum
Giovanni Semeraro

Program Committee

Fabian Abel	L3S Research Center
Sahin Albayrak	DAI-Labor, Technische Universität Berlin, Germany
Claudio Bartolini	
Marco Brambilla	Politecnico di Milano
Andrea Cali	University of London, Birkbeck College
Charles Callaway	University of Haifa
Ivan Cantador	Universidad Autonoma de Madrid
Pablo Castells	Universidad Autónoma de Madrid
Federica Cena	Department of Computer Science, University of Torino
Philipp Cimiano	
Mathieu D'Aquin	Knowledge Media Institute, the Open University
Marco De Gemmis	Dipartimento di Informatica - University of Bari
Ernesto William De Luca	Technische Universität Berlin
Tommaso Di Noia	Politecnico di Bari
Nicola Fanizzi	Dipartimento di Informatica, Università di Bari
Bettina Fazzinga	DEIS - University of Calabria
Miriam Fernandez	Knowledge Media Institute
Tim Furche	University of Munich
Aldo Gangemi	CNR-ISTC
Michael Hausenblas	Digital Enterprise Research Institute (DERI), NUI Galway
Tom Heath	Talis Systems Ltd
Dominikus Heckmann	
Eelco Herder	
Dietmar Jannach	TU Dortmund
Pasquale Lops	University of Bari
Thomas Lukasiewicz	Oxford University
Till Plumbaum	DAI-Labor, Technische Universität Berlin, Germany
Georg Ruß	Otto-von-Guericke-University of Magdeburg
Alan Said	TU Berlin
Giovanni Semeraro	Dipartimento di Informatica - University of Bari
Wolf Siberski	L3S Research Center
Armando Stellato	University of Rome, Tor Vergata
Tania Tudorache	Stanford University

Table of Contents

Personalized Filtering of the Twitter Stream

Pavan Kapanipathi[1,2], Fabrizio Orlandi[1], Amit Sheth[2], and Alexandre Passant[1]

[1] Digital Enterprise Research Institute, Galway, Ireland
{fabrizio.orlandi, alexandre.passant}@deri.org
[2] Kno.e.sis Center, Dayton, OH - USA
{pavan, amit}@knoesis.org

Abstract. With the rapid growth in users on social networks, there is a corresponding increase in user-generated content, in turn resulting in information overload. On Twitter, for example, users tend to receive uninterested information due to their non-overlapping interests from the people whom they follow. In this paper we present a Semantic Web approach to filter public tweets matching interests from personalized user profiles. Our approach includes automatic generation of multi-domain and personalized user profiles, filtering Twitter stream based on the generated profiles and delivering them in real-time. Given that users interests and personalization needs change with time, we also discuss how our application can adapt with these changes.

Keywords: Semantic Web, Social Network, Twitter, PubSubHubbub, User Profiling, Personalization

1 Introduction

Online Social Networks have become a popular way to communicate and network in the recent times, well known ones include Facebook, MySpace, Twitter, Google+, etc. Twitter, in specific, has rapidly grown in the recent years, reaching 460,000 average number of new users per day in the month of March 2011. These numbers have in turn played a crucial role to increase the number of tweets from 65 million to 200 million[3] in the past year. This proves that the interested users are therefore facing the problem of information overload. Filtering uninteresting posts for users is a necessity and plays a crucial role [8] to handle the information overload problem on Twitter.

On Twitter it is necessary to *follow* another user in order to receive his/her tweets. The user who receives the tweets is called a *follower* and the user who generates the tweet is called a *followee*. However, they receive all the tweets from the users that are also not of their interests. Twitter by itself provides features such as keyword/hashtag search as a naïve solution for the information overload problem, but these filters are not sufficient to provide complete personalized information for a user. Although Twarql [6] improved the filtering mechanism

[3] `http://blog.twitter.com/2011/08/your-world-more-connected.html`

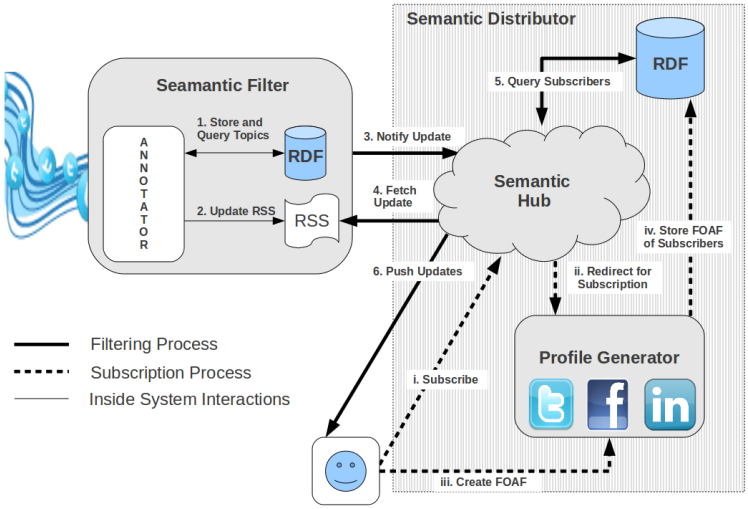

Fig. 1. System Architecture

for Twitter by leveraging Semantic Web technologies, the user still needs to track information by manual selection or formulation of SPARQL Query using Twarql's interface. So far applications such as TweetTopic [1] and "Post Post"[4] focus on filtering the stream of tweets generated from the people who are followed by the user. Instead of limiting the user experience only to his/her personal stream we propose a Semantic Web approach to deliver interesting tweets to the user from the entire public Twitter stream. This helps filtering tweets that the user is not interested in, which in turn reduces the information overload.

Our contributions include (1) automatic generation of user profiles (primarily interests) based on the user's activities on multiple social networks (Twitter, Facebook, Linkedin). This is achieved by retrieving users' interests, some implicit (analyzing user generated content) and some explicit (interests mentioned by the user in his/her SN profile). (2) Collecting tweets from the Twitter stream and mapping (annotating) each tweet to its corresponding topics from Linked Open Data. (3) Delivering the annotated tweets to users with appropriate interests in (near) real-time.

2 Architecture

Our architecture can be separated into three modules (1) Semantic Filter (**SF**) (2) Profile Generator (**PG**) (3) Semantic Hub (**SemHub**) as illustrated in Fig-

[4] `http://postpo.st/`

ure 1. In this section we first explain the interaction between the three modules, later each one is explained in detail.

In the above architecture two processes run in parallel (a) Filtering of tweets (b) Subscription to the System. The sequence for each process is represented by different types of arrows in Figure 1. The *Subscription to the system* is included in the Semantic Distributor. The Semantic Distributor (**SD**) comprises of both SH and PG. Once the user requests for the subscription (Seq. *i* in Figure 1) he/she is redirected to the PG (Seq. *ii*). PG generates the profiles based on the the user's activities on multiple social networks (Seq. *iii*). These profiles are stored in the SemHubs' RDF store (Seq. *iv*) using PuSH vocabulary [5]. On the other hand, *Filtering of tweets* is performed by annotating tweets from Twitter stream in SF. The annotations are further transformed to a representation of groups (SPARQL queries) of users who have interests corresponding to the tweet (Seq. *1*). These SPARQL Queries are termed as Semantic Groups (**SG**) in this paper. The tweet with its SG is updated as an RSS feed (Seq. *2*) and notified to SemHub (Seq. *3*). SemHub then fetches the updates (Seq. *4*) and retrieves the list of subscribers whose interests match the group representation of the tweet (Seq. *5*). Further the tweet is pushed to the filtered subscribers (Seq. *6*).

2.1 Semantic Filter

Semantic Filter (Figure 1), primarily performs two functions: (1) Representing tweets as RDF (2) Forming interested groups of users for the tweet.

First, information about the tweet is collected to represent the tweet in RDF. Twitter provides information of the tweet such as author, location, time, "reply-to", etc. via its streaming API. Including this, extraction of entities from the tweet content (content-dependent metadata) is performed using the same technique used in Twarql. The extraction technique is dictionary-based, which provides flexibility to use any dictionary for extraction. In our system the dictionary used to annotate the tweet is a set of concepts[6] from the Linked Open Data [2] (LOD)[7]. The same set is also used to create profiles, as described in the next Section 2.2. After the extraction of entities, the tweets are represented in RDF using lightweight vocabularies such as FOAF, SIOC, OPO and MOAT. This transforms the unstructured tweet to a structured representation using popular ontologies. The triples (RDF) of the tweet are temporarily stored in an RDF store.

The annotated entities represent the topic of the tweet. These topics act as the key in filtering the subset of users who receive the tweet. Topics are queried from the RDF store to be included in SGs that are created to act as the filter. The SG once executed at the Semantic Hub fetches all the users whose interests match to the topic of the tweet. If there are multiple topics for the tweet then the SG is created to fetch the union of users who are interested in at least one topic of the tweet.

[5] http://vocab.deri.ie/push
[6] Topic and concept are used interchangeably.
[7] http://richard.cyganiak.de/2007/10/lod/

2.2 User Profile Generator

The extraction and generation of user profiles from social networking websites is composed of two basic parts: (1) data extraction and (2) generation of application-dependent user profiles. After this phase other important steps for our work involve the representation of the user models using popular ontologies, and then, finally, the aggregation of the distributed profiles.

```
<foaf:topic_interest rdf:resource="http://dbpedia.org/resource/Semantic_Web" />
<wi:preference>
  <wi:WeightedInterest>
    <wi:topic rdf:resource="http://dbpedia.org/resource/Semantic_Web" />
    <rdfs:label>Semantic Web</rdfs:label>
    <wo:weight>
      <wo:Weight>
  <wo:weight_value rdf:datatype="http://www.w3.org/2001/XMLSchema#double">0.5</wo:
       weight_value>
  <wo:scale rdf:resource="http://example.org/01Scale" />
      </wo:Weight>
    </wo:weight>
    <opm:wasDerivedFrom rdf:resource="http://www.twitter.com/BadmotorF" />
    <opm:wasDerivedFrom rdf:resource="http://www.linkedin.com/in/fabriziorlandi" />
  </wi:WeightedInterest>
</wi:preference>
[...]
<wo:Scale rdf:about="http://example.org/01Scale">
  <wo:max_weight rdf:datatype="http://www.w3.org/2001/XMLSchema#decimal">1.0</wo:
       max_weight>
  <wo:min_weight rdf:datatype="http://www.w3.org/2001/XMLSchema#decimal">0.0</wo:
       min_weight>
</wo:Scale>
```

Fig. 2. Representing an interest (*Semantic Web*) and its weight (*0.5*) found in two sources (Twitter and LinkedIn)

First, in order to collect private data about users on social websites it is necessary to have access granted to the data by the users. Then, once the authentication step is accomplished, the two most common ways to fetch the profile data is by using an API provided by the system or by parsing the Web pages. Once the data is retrieved the next step is the data modeling using standard ontologies. In this case, a possible way to model profile data is to generate RDF-based profiles described using the FOAF vocabulary [4]. We then extend FOAF with the SIOC ontology [3] to represent more precisely online accounts of the person on the Social Web. Additional personal information about users' affiliation, education, and job experiences can be modeled using the DOAC vocabulary[8]. This allows us to represent the past working experiences of the users and their cultural background. Another important part of a user profile is represented by the user's interests. In Figure 2 we display an example of an interest about "Semantic Web" with a weight of 0.5 on a specific scale (from 0 to 1) using the Weighted IntListingerests Vocabulary (WI)[9] and the Weighting Ontology (WO)[10]. In order to compute the weights for the interests common approaches are based on the number of occurrences of the entities, their frequency, etc.

[8] DOAC Specification: `http://ramonantonio.net/doac/0.1/`
[9] WI Specification: `http://purl.org/ontology/wi/core#`
[10] WO Specification: `http://purl.org/ontology/wo/core#`

Finally, the phase that follows the modeling of the FOAF-based user profiles and the computation of the weights for the interests is the aggregation of the distributed user profiles. When merging user profiles it is necessary to avoid duplicate statements (and this is done automatically by a triplestore during the insertion of the statements). Furthermore, as in the case of the interests, if the same interest is present on two different profiles it is necessary to: represent the interest only once, recalculate its weight, and update the provenance of the interest keeping track of the source where the interest was derived from. As regards the provenance of the interest, as showed in Figure 2, we use the property `wasDerivedFrom` from the Open Provenance Model[11] (OPM) to state that the interest was originated by a specific website.

As regards the computation of the aggregated global weight for the interest generated by multiple sources, we propose a simple generic formula that can be adopted for merging the interest values of many different sources. The formula is as follows:

$$G_i = \sum_s w_s * w_i \tag{1}$$

Where: G_i is the global weight for interest i; w_s is the weight associated to the source s; w_i is the weight for the interest i in source s.

2.3 Semantic Hub

The Semantic Distributor module comprises of Semantic Hub [5] and Profile Generator. Semantic Hub (SemHub) is an extension of Google's PubSubHubbub (PuSH) using Semantic Web technologies to provide publisher-controlled real-time notifications. PuSH is a decentralized publish-subscribe protocol which extends Atom and RSS to enable real-time streams. It allows parties understanding it to get near-instant notifications of the content they are subscribed to, as PuSH immediately *pushes* new data from publisher to subscriber(s) where traditional RSS readers periodically *pull* new data. The PuSH ecosystem consists of a few hubs, many publishers, and a large number of subscribers. Hubs enable (1) publishers to offload the task of broadcasting new data to subscribers; and (2) subscribers to avoid constantly polling for new data, as the hub pushes the data updates to the subscribers. In addition, the PuSH protocol is designed to handle all the complexity in the communication easing the tasks of publishers and subscribers.

The extension from PuSH protocol to Semantic Hub is described in [5]. In our work, SemHub performs the functionality of distributing the tweets to its interested users corresponding to the Semantic Groups generated by SF. The SemHub will have only one publisher as shown in Figure 1 which is the SF, and there can be multiple subscribers. SemHub, as in our previous work, does not focus on creating a social graph of the publisher, the PG is responsible to store the subscribers's FOAF profile in the RDF store accesssed by the SemHub.

[11] OPM Specification: `http://openprovenance.org/`

3 Implementation

In this section we provide the implementation details for each module in the architecture. Firstly to collect tweets we use the *twitter4j Streaming API* [12]. Starting with SF, the entity extraction of tweets is dictionary-based similar to the extraction technique used in *Twarql* [7]. This technique is opted due to performance requirements for real-time notifications. A set of 3.5 million entities[13] from DBpedia is built as an in-memory representation for time-efficient and longest sub-string matching. The in-memory representation is known as ternary interval search tree (Trie) and the longest sub-string match using trie is performed at time complexity of *O(LT)* where L is the number of characters and T is the number of tokens in the tweet.

```
<http://twitter.com/rob/statuses/123456789>
  rdf:type    sioct:MicroblogPost ;
  sioc:content    What is the over/under on the Kim Kardashian / Kris Humphries
       Hollywood wedding lasting more than 5 years? #fb
  sioc:has_creator    <http://twitter.com/rob> ;
  foaf:maker    <http://example.org/rob> ;
  moat:taggedWith    dbpedia:Kim_Kardashian ;
  moat:taggedWith    dbpedia:Kris_Humphries ;
  moat:taggedWith    dbpedia:Hollywood .

<http://twitter.com/rob/statuses/123456789#presence>
  rdf:type    opo:OnlinePresence ;
  opo:startTime    2010-03-20T17:55:42+00:00    ;
  opo:customMessage <http://twitter.com/rob/statuses/123456789> .

<http://twitter.com/rob> geonames:locatedIn Dbpedia:Ohio .
[...]
```

Fig. 3. Representing Tweet in RDF

As mentioned in section 2.1, tweets are transformed into RDF using some lightweight vocabularies, see Figure 3 for an example. The RDF is then stored in an RDF store using SPARQL Update via HTTP. For performance issues it is preferable to have the RDF Store on the same server. However, architecturally it can be located anywhere on the Web and accessed via HTTP and the SPARQL Protocol for RDF. Presently, this RDF generated for each tweet is stored in a temporary graph and topics/concepts of the tweet are queried. These concepts are then used to formulate the SPARQL representation of the group (SG) of users who are interested in the tweet. The RSS is updated as per the format specified in [5] with the SG and the Semantic Hub is notified. The SG for the tweet in Figure 3 will retrieve all the users who are interested in at least one of the extracted interests (*dbpedia:Kim_Kardashian, dbpedia:Kris_Humphries, dbpedia:Hollywood*).

The Semantic Hub used for our implementation is hosted at `http://semantichub.appspot.com`. The SemHub executes the SG on the graph

[12] `http://stream.twitter.com`

[13] `http://wiki.dbpedia.org/About` (July 2011)

that contains the FOAF profiles of subscribers generated by PG. The corresponding tweets are *pushed* to the resulting users.

Profile Generator considers three different social networking sites: Twitter, LinkedIn and Facebook for generating user profiles. In order to collect user data from each of those platforms, we developed three different types of applications. For Twitter and Facebook we implemented similar PHP scripts that makes use of the respective query API publicly accessible on the Web. For LinkedIn we use a XSLT script that parses the LinkedIn user profile page and generates an XML file containing all the attributes found on the page. The user information collected from Twitter is the publicly available data posted by the user, *i.e.* his/her latest 500 microblog posts. The technique used for entity recognition in the tweets of the user is the same one used in SF for annotating the tweets. The extracted concepts are then ranked and weighted using their frequency of occurrences. A similar approach is described in [9].

While on Twitter we create profiles with implicitly inferred interests, on LinkedIn and Facebook we collect not only interests that have been explicitly stated by the users, but also their personal details such as contacts, workplace and education. The user personal data is fetched through the Facebook Graph API as well as the interests (*likes*) that are then mapped to the related Facebook pages representing the entities. We represent the entities/concepts on which the user is interested in using both DBpedia and Facebook resources.

The weights for the interests are calculated in two different ways depending on whether or not the interest has been *implicitly* inferred by the entity extraction algorithm (the Twitter case) or *explicitly* recorded by the user (the LinkedIn and Facebook cases). In the first case, the weight of the interest is calculated dividing the number of occurrences of the entity in the latest 500 tweets by the total number of entities identified in the same 500 tweets. In the second case, since the interest has been manually set by the user, we assume that the weight for that source (or social networking site) is 1 (on a scale from 0 to 1). So we give the maximum possible value to the interest if it has been explicitly set by the user.

Our approach as regards the computation of the new weights as a result of the aggregation of the profiles is straightforward. We consider every social website equally in terms of relevance, hence we multiply each of the three weights by a constant of 1/3 (approximately 0.33) and then we sum the results. According to the previously described formula (1) in this case we use the following values: $w_s = 1/3. \forall s$.

4 Conclusion and Future Work

In this paper we described an architecture for filtering the public Twitter stream and delivering the interesting tweets directly to the users according to their multi-domain user profile of interests. We explained how we generate comprehensive user profiles of interests by fetching and aggregating user information from different sources (*i.e.* Twitter, Facebook and LinkedIn). Then, we detailed

how we extract entities and interests from tweets, how we model them using Semantic Web technologies, and how is possible to automatically create dynamic groups of users related to the extracted interests. According to the user groups the tweets are then "pushed" to the users using the Semantic Hub architecture.

In future, we want to extend our work to handle social streams in general (not only Twitter). Also, leveraging inferencing (category - subcategory relationships) on LOD, rather than just filtering based on concepts. Our extention would also include users not only subscribe to concepts from LOD as interests but also subscribe to a SPARQL Query as in Twarql. We are also working on providing interesting information and ranking based on the user's social graph.

5 Acknowledgements

This work is funded by (1) Science Foundation Ireland under grant number SFI/08/CE/I1380 (Líon 2) and by an IRCSET scholarship supported by Cisco Systems (2) Social Media Enhanced Organizational Sensemaking in Emergency Response, National Science Foundation under award IIS-1111182, 09/01/2011 - 08/31/2014.

References

1. M.S. Bernstein, Bongwon Suh, Lichan Hong, Jilin Chen, Sanjay Kairam, and E.H. Chi. Eddi: interactive topic-based browsing of social status streams. In *The 23rd annual ACM symposium on User interface software and technology*, 2010.
2. Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data - the story so far. *Int. J. Semantic Web Inf. Syst.*, 5(3):1–22, 2009.
3. John Breslin, Uldis Bojars, Alexandre Passant, Sergio Fernàndez, and Stefan Decker. SIOC: Content Exchange and Semantic Interoperability Between Social Networks. In *W3C Workshop on the Future of Social Networking*, January 2009.
4. Dan Brickley and Libby Miller. FOAF Vocabulary Specification 0.98. Namespace Document 9 August 2010 - Marco Polo Edition. `http://xmlns.com/foaf/spec/`, 2010.
5. Pavan Kapanipathi, Julia Anaya, Amit Sheth, Brett Slatkin, and Alexandre Passant. Privacy-Aware and Scalable Content Dissemination in Distributed Social Networks. In *ISWC 2011 - Semantic Web In Use*, 2011.
6. Pablo N. Mendes, Alexandre Passant, and Pavan Kapanipathi. Twarql: tapping into the wisdom of the crowd. I-SEMANTICS '10, 2010.
7. Pablo N. Mendes, Alexandre Passant, Pavan Kapanipathi, and Amit P. Sheth. Linked Open Social Signals. In *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, 2010.
8. D. Ramage, S. Dumais, and D. Liebling. Characterizing microblogs with topic models. In *ICWSM*, 2010.
9. Ke Tao, Fabian Abel, Qi Gao, and G.J. Houben. TUMS: Twitter-based User Modeling Service. In *Workshop on User Profile Data on the Social Semantic Web (UWeb), ESWC 2011*, 2011.

User-sensitive Explanations under a Knowledge Pattern Lens

Alessandro Adamou[1,2], Paolo Ciancarini[1], Aldo Gangemi[2], and Valentina Presutti[1,2]

[1] Alma Mater Studiorum Università di Bologna, Italy
[2] ISTC, National Research Council, Italy

Abstract. This paper introduces our ongoing research on a general-purpose methodology for generating explanations for occurrences of interaction patterns. Explanations are tailored around a user's profile or interaction history in interactive systems. The approach relies on the recognition of interlinks between interaction patterns and knowledge patterns, both formally modeled as networked ontologies. In addition, the method constructs its statements using the same knowledge shared by the hosting system, including distributed knowledge such as Linked Data. We plan to implement and evaluate this approach in the context of Content Management Systems and related interaction patterns such as query disambiguation, content recommendation, faceted search and browsing.

1 Introduction

In interactive systems, the need for interpreting system behaviour has seen a steady growth as the supported recurring schemes, or *patterns* in human-computer interaction (recommendation, tagging, query disambiguation, faceted search and browsing being some) increase in amount and complexity. The more a system replaces natural language with an iconic or mutimodal one, the less it tends to be self-explanatory. We collectively dub the functionalities that address this issue (a simple example being *tooltips*) as "explanations", in that they justify either the content of system feedback, or its form of presentation to the user. Examples include providing information on an entity portrayed in a picture, the meaning of a chart portion being shown in a certain color, or a justification as to why a certain multimedia item is being recommended to a customer.

Software systems tend to short-circuit this issue by hardwiring ad-hoc functionalities, each dealing with a specific use case and whose output is tightly bound with the functionality they serve. For instance, explanation-featuring recommender systems support the entire cycle autonomously, by generating the exact sentence that will be delivered to the user along with the recommendation. Despite its basis on general principles common to other interaction-oriented functionalities, an approach like this is hardly reusable. In addition, when these functionalities work in a "boxed" fashion, not interoperable with the rest of the system, they may fail to deliver "user-sensitive" explanations. Here, user-sensitivity denotes the ability of a system to deliver explanations that are (i)

tailored around a user's interaction context and/or profile; (ii) comprehensible by an agent whose only required knowledge is that of the domain at hand.

Consider for example a member of a research project X browsing the annual report on its expenditures. A chart shows the Q1-2011 portion marked yellow as opposed to others marked green. If the *project manager* hovers on that chart portion, a tooltip or text console will explain the color as "Project X has spent $2,000 above its planned quota in Q1, 2011". For another member, the message could instead be "Project X has slightly overspent" or "has spent 5% more than its planned quota", depending on what information that user has access to.

As it emerges from the examples above, an explanation generator does not guess why the system provided a certain type of feedback. This information should come from the system itself: in another example, it is the recommender system that knows why it came to select a particular item for recommendation. The challenge for explanation functionalities is to deliver this information, or part thereof, in a user-sensitive way, by selecting relevant pieces of knowledge in a context, and providing directives as to how they should be *assembled* together.

Ontologies, knowledge patterns, inference rules and reasoning are a solid set of tools for sewing together these mutually agnostic systems and functionalities into a general-purpose semantic framework that can serve multiple interaction patterns. This paper describes the first insights into our ongoing research on a method that can accomplish these goals. After a brief overview on existing work in Section 2, we sketch in Section 3 the general workflow of our method under research and the types of resource it is based upon. In Section 4 we show with an example how this method applies to possible occurrences of the interaction patterns we can support. We conclude with Section 5, by describing the ongoing work and our plan for a proof-of-concept implementation of the method.

2 Related work

Historically, explanation generation has been one of the fields of study of computational linguistics for over two decades [6]. An algorithm based on abductive reasoning, which targets the explanation of queried events, has been around since 1987 [1]. Another explanation system grounded in the biology domain was Knight [5], which combined early knowledge pattern identification and usage in order to provide definitions of entities in the given domain.

We acknowledge the above as groundbreaking work that inspired our research. Nowadays however, with the rise of Linked Data and heterogeneous knowledge sources on one hand, and the increasing interaction patterns support on the other hand, demands of cross-domain and cross-application flexibility are being pushed beyond those systems. As to modern interaction patterns, *Tagsplanations* address the relevance and sentiment of users towards tags for generating recommendations [10], where tags on content items are sorted and selected to justify a recommendation. We argue that an approach such as this should not limit to processing user-generated tags and supporting the recommendation pattern, but should instead be a cornerstone for selecting relevant statements in general-

purpose explanation approaches for entities, facts and interactions. Tintarev et al. analyzed the goals and metrics involved with explaining recommendations, e.g. effectiveness, efficiency, persuasion and transparency; and investigated on a method to elicit effectiveness [9]. Experiments for evaluating the impact of explanations in automated collaborative filtering systems were also conducted [4]. They used a white-box model of explanations, which was designed ad-hoc for this single interaction pattern, as it arguably does not involve reasoning.

Finally, the exploitation of interaction patterns and their representation using formal semantics has also begun to occur in studies on multimodality [8]. Multimodal interaction patterns in this study encompass the spatial and temporal relations between modalities, such a sequentiality and simultaneity. While we are not currently targeting multimodality, we gain inspiration from this study with respect to the formal treatment of sequential input in interaction models.

3 Approach

Our proposed general-purpose approach relies on the following knowledge, either made available by the host system, e.g. a Content Management System (CMS), or authored as part of the explanation strategy itself.

Interaction Pattern catalog. A collection of solutions to common usability problems, which become recurring interaction schemes in a user-system dialog. Numerous libraries of interaction pattern specifications are available, one being by Martijn van Welie[3]. To reason upon them, we will require a model for representing them as ontologies. This is work-in-progress for this research[4], which concentrates on interaction patterns for manipulating content and knowledge. We will focus on interaction patterns expected to occur in a CMS, such as recommendation, faceted search and browsing, annotation and query disambiguation.

Knowledge Pattern catalog. A collection of formal minimal models used to describe a concept, state or event in the real world. Knowledge patterns (KPs) are *invariances across observed data or objects* that allow a *formal or cognitive interpretation* [3]. These will contribute to the selection of statements for explanations: in this respect, the ties with linguistic frames and FrameNet [7] as a repository of such models are evident. The Content Pattern section of the ODP portal[5], which we also plan to enrich, will be our experimental basis.

Real-world knowledge. With this term, we denote the content of the knowledge base managed by the host system. Other than its rendering as RDF, no assumption is made as to where this knowledge is stored and which vocabularies are used. It may, for example, be a simple hub that crawls and indexes the Linked Data cloud, or a centralized repository of in-house knowledge.

[3] Interaction Pattern Library, `http://www.welie.com/patterns/`

[4] Interaction pattern model WIP, version control at `https://iks-project.googlecode.com/svn/sandbox/explanation/trunk/src/main/resources/model/`

[5] Ontology Design Patterns, `http://www.ontologydesignpatterns.org`

User interaction trace. For explanations to be tailored around the flow of interaction of a given user with a given system, the system itself has to provide the interaction history, or *discourse context*. It is essentially a semantic log of user interaction and the interface elements, widgets or multimodal channels involved. It is an extension of the interaction pattern metamodel that we will provide.

Mappings of two different kinds:

- The *interpretation* of user interaction elements as the real-world entities they denote; for example, the avatar icon of a person can be interpreted as that very person; a drop area can be interpreted as a physical location, or a task; selecting a point on a map can be interpreted as requesting the points of interest nearby, or setting it as the next destination of a trip. Interpretations should be provided by the host system. The mapping vocabulary will be an extension of our interaction pattern metamodel.
- *Alignments* between controlled vocabularies and KPs, which are not built using these vocabularies. They can be e.g. `owl:equivalentClass` axioms between `kp:Person`[6] and the classes `Person` in FOAF and DBPedia. Alignments can be assumed to be provided by the knowledge pattern authors.

Annotations of knowledge patterns, with the dual purpose of marking their key elements in order to (i) determine if a pattern can be matched in the knowledge base; (ii) select entities and facts that should participate into statements that will form the explanation. Groups of such elements can also be marked as key.

Our explanation strategy proceeds along the lines of the following workflow. We assume it to be triggered every time a user action or system feedback is issued and logged by the host system, i.e. stored in the *interaction trace*.

1. **Interaction pattern detection.** If this information is registered by the host system, this step is unnecessary; otherwise, we will need to establish whether the most recent sequence of utterances in a user-system dialog matches an occurrence of one of the interaction patterns in our catalog. This is done by matching the registered actions and their involved UI elements against the classes of actions and UI elements defined in each interaction pattern (which are represented using the same ontology). If more than one such interaction pattern is satisfied, heuristics may be applied to select the most likely (e.g. the one that traces furthest back in the interaction history).
2. **Interpretation grounding.** For every individual in the detected interaction pattern occurrence, check which individuals in the knowledge base it maps to, according to the *interpretation mapping*. Extract the types - both asserted and inferred - of every such individual.
3. **Candidate knowledge pattern set construction.** If the interpretation explicitly states which knowledge pattern(s) the detected interaction pattern maps to, let $\mathcal{KP}$ be the set of these knowledge pattern(s) *and* their specializations, if any; otherwise, let $\mathcal{KP}$ be the entire set of knowledge patterns.

[6] The `kp` prefix is a placeholder for the namespace to be used for knowledge patterns.

4. **Knowledge pattern satisfiability check.** For each knowledge pattern in $\mathcal{KP}$, determine if it is satisfied. A knowledge pattern is satisfied iff all its elements marked as *key elements* are filled, i.e. for each key class or property there is a corresponding individual or predicate in the knowledge base. Let $\mathcal{KP}'$ be the set of *satisfied* knowledge patterns.
5. **Explanation assembly.** For each satisfied knowledge pattern in $\mathcal{KP}'$, select the statements (assertions or inferences) in the knowledge base that (i) map to the knowledge pattern and (ii) have the greatest weights according to their annotations in the knowledge pattern model. Weight corrections can be applied using a coefficient calculated by arbitrary heuristics, e.g. the number of mentions of that entity in the interaction context. The list of selected statements is the *explanation* for this occurrence of the detected KP.

4 Example

An explanation is, in general, an *interpretation* of an *interaction pattern occurrence* in a running system. One of our goals is to *classify* interaction patterns in terms of the types of explanations that typically accompany them. Note that explanations themselves can be part of some interaction patterns, such as those classifiable as *explanation requests*, e.g. tooltips that appear upon hovering on a widget, or clickable "*more...*" or "*why?*" links. When this is not the case, explanation can be seen as an *ancillary* interaction pattern, which conceptually accompanies another one such as recommendation, search or annotation.

In the conceptually simplest cases, an explanation consists of providing a summary description, or *synopsis*, of an entity in the 'real' (as opposed to 'virtual') world. Typically, this is conveyed through the provision of attributes and/or facts involving the object to be explained. We shall now exemplify this for an interaction pattern with a peculiar requirement in this matter.

4.1 Query disambiguation

An as-you-type entity search is issued for annotating the text item "George Bush" in the content being written. When the query returns both former Presidents of the United States, the system prompts the user to select the appropriate one. To aid the user in the selection, a description must be provided for both results, but these descriptions must differ enough for the user to be able to disambiguate. An explanation whose statements are "former President of the U.S.; war with Iraq under his administration", if perfectly sound for either George Bush individual, would be totally ineffective for the interaction pattern at hand. In other words, the synopses of the two entities must minimize their overlap.

Figure 1 exemplifies our rationale for this interaction pattern. A simple Query Disambiguation pattern can be formalized as including the following individuals:

- a *system action*, which in this case consists of responding to a previous query;
- a *selector widget* populated by the system action with a list of ambiguous entries, each of them mapping to (i.e. `interpretedAs`) a real-world entity.

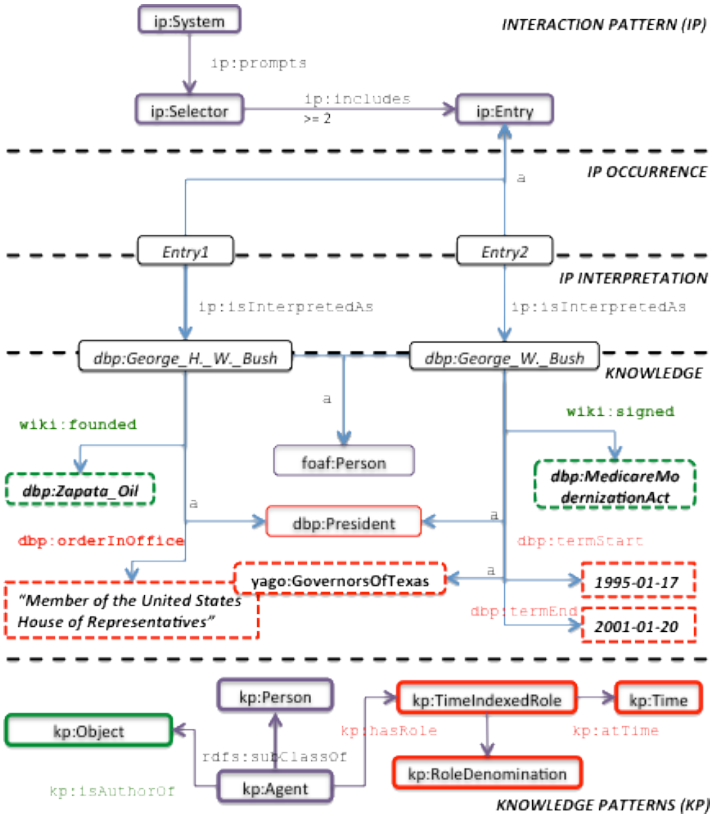

Fig. 1. Selection of statements for explaining disambiguation on the query "George Bush", from the interaction pattern (top) to the knowledge pattern (bottom) level. Entities marked in red indicate an association to a *time-indexed role* knowledge pattern; those in green indicate an association to an *authorship* knowledge pattern.

Suppose the explanation generator has access to all the named entities recognized for the same content, as well as the same knowledge base used by the semantic search engine that handled the query (see "knowledge" portion of the figure). Also, suppose the knowledge base types each "Bush" individual as a `foaf:Person`. The Query Disambiguation interaction pattern does not map to any specific knowledge pattern, since we have to discover which knowledge patterns apply to each entity. However, we do know we can restrict to those knowledge patterns that involve a `kp:Person` (for which we have mappings for commonly used types like `foaf:Person` in FOAF, `schemaorg:Person` in schema.org or `dbp:Person` in DBPedia). From all the facts - both asserted and inferred - about these two entities in the knowledge base, we detect several occurrences that satisfy a *time-indexed role* pattern, e.g. George W. Bush's office as Governor of Texas between 1995 and 2001, and George H.W. Bush's office as U.S. congressman 1967-1971. For a general *authorship* pattern, an instantiation is

detected for George H.W. Bush as a founder of the Zapata Oil company, and for George W. Bush as signer of the Medicare Act of 2003.

These facts satisfying some knowledge pattern are also weighed according to an arbitrary set of heuristics, which may include the number of occurrences in the knowledge base (suppose it to be, for example, a hub of large Linked Data sets where the same statements can occur multiple times using different identifiers) and the amount of mentions of related entities (such as Texas or Medicare) in the content item edited by the user. The greatest-weighed facts (denoted by the colored entities with a dashed outline in Figure 1) are selected and included as statements in the synopsis for each entity responding to a "George Bush" query.

If there are semantic relations involving the current user, then profile information can be included in the context and the KPs satisfied by these relations can be prioritized. For instance, if the knowledge base states that a `foaf:Person` matching this user (via alignments to a *self* pattern, e.g. by matching user names in the system or `foaf:name` property values) has worked as a White House administrative between 2002 and 2007, the combined *membership*[7] and *hierarchy* KPs can be satisfied. Then, a statement such as "your former boss" for George W. Bush will be viable for generation and have a greater weight for inclusion.

5 Ongoing and future work

Our current focus is on defining, reusing and reengineering formal schemas, annotation vocabularies and content for the knowledge required per the first part of Section 3. This includes knowledge and interaction pattern models, extensions for interaction traces, interpretations and mappings with popular and emerging controlled vocabularies such as DBpedia[8] and `schema.org`. For the interaction metamodel, we are taking inspiration from the interaction and interface modules of the C-ODO Light ontology network for managing ontology lifecycles [2].

As a basis for evaluating the designed methodology, the implementation of a proof of concept is in progress. We will focus on Content Management Systems (CMS) as versatile interactive systems in collaborative contexts where content and knowledge are managed. We intend to prioritize the association of explanation support with the interaction patterns that most frequently occur in such systems[9], e.g. (1) annotation of content with knowledge; (2) autocompletion; (3) faceted search and browsing; (4) query disambiguation; (5) content recommendation; (6) drag-&-drop (in latest-generation WebCMS).

The implementation under construction is a set of plugins for the **Apache Stanbol** service platform for semantic CMS[10], which provides functionalities for improving knowledge management and interaction. In particular, an inference rule language, compatible with the SWRL rule and SPARQL query languages, will be used for defining interpretations and alignments whenever assertions more

[7] `http://ontologydesignpatterns.org/cp/owl/timeindexedmembership.owl`
[8] DBpedia TBox, `http://downloads.dbpedia.org/3.6/dbpedia_3.6.owl.bz2`
[9] CMS interaction, `http://wiki.iks-project.eu/index.php/InteractionPatterns`
[10] Apache Stanbol incubation home, `http://incubator.apache.org/stanbol`

complex than equivalence or subsumption statements are required. Its ontology registry support will be used to manage multiple catalogs, such as knowledge patterns, interaction patterns and mappings. Its controlled environment for ontology networks allows us to scale reasoning and pattern detection tasks only on viable candidates or satisfiable knowledge and interaction patterns. A Web Service API will allow other modules in the host system to store the semantics of their user interaction traces, in accordance with the prescribed terminology.

Our explanation method will be user-evaluated against experimental and control groups extracted from Semantic CMS community members, which will include their users as well as their adopters and providers. The effectiveness of our approach will be evaluated by applying our proof of concept to specific use cases and domains which typically employ such interactive systems, e.g. project management and news publishing. Efficiency will be assessed through quantitative measurements such as the lag over on the main functionality output and the response time of users in the accomplishment of tasks with and without explanation support, as well as with traditional hardcoded explanations.

Acknowledgements This work has been part-funded by the European Commission under grant agreement FP7-ICT-2007-3/ No. 231527 (IKS - Interactive Knowledge Stack)

References

1. Coombs, M.J., Hartley, R.T.: The MGR algorithm and its application to the generation of explanations for novel events. International Journal of Man-Machine Studies 27(5-6), 679–708 (1987)
2. Gangemi, A., Lehmann, J., Presutti, V., Nissim, M., Catenacci, C.: C-ODO: an OWL meta-model for collaborative ontology design. In: Noy, N.F., Alani, H., Stumme, G., Mika, P., Sure, Y., Vrandecic, D. (eds.) CKC. CEUR Workshop Proceedings, vol. 273. CEUR-WS.org (2007)
3. Gangemi, A., Presutti, V.: Towards a pattern science for the Semantic Web. Semantic Web 1(1-2), 61–68 (2010)
4. Herlocker, J.L., Konstan, J.A., Riedl, J.: Explaining collaborative filtering recommendations. In: CSCW. pp. 241–250 (2000)
5. Lester, J.C., Porter, B.W.: Developing and empirically evaluating robust explanation generators: The KNIGHT experiments. Computational Linguistics 23(1), 65–101 (1997)
6. Maybury, M.T.: Communicative acts for explanation generation. International Journal of Man-Machine Studies 37(2), 135–172 (1992)
7. Nuzzolese, A.G., Gangemi, A., Presutti, V.: Gathering Lexical Linked Data and Knowledge Patterns from FrameNet. In: Proc. of the 6th International Conference on Knowledge Capture (K-CAP). pp. 41–48. Banff, Alberta, Canada (2011)
8. Taib, R., Ruiz, N.: Integrating semantics into multimodal interaction patterns. In: Popescu-Belis, A., Renals, S., Bourlard, H. (eds.) MLMI. Lecture Notes in Computer Science, vol. 4892, pp. 96–107. Springer (2007)
9. Tintarev, N., Masthoff, J.: Effective explanations of recommendations: user-centered design. In: Konstan, J.A., Riedl, J., Smyth, B. (eds.) RecSys. pp. 153–156. ACM (2007)
10. Vig, J., Sen, S., Riedl, J.: Tagsplanations: explaining recommendations using tags. In: Conati, C., Bauer, M., Oliver, N., Weld, D.S. (eds.) IUI. pp. 47–56. ACM (2009)

Towards Ranking in Folksonomies for Personalized Recommender Systems in E-Learning

Mojisola Anjorin, Christoph Rensing, and Ralf Steinmetz

Technische Universität Darmstadt,
Multimedia Communications Lab,
64283 Darmstadt, Germany
{Mojisola.Anjorin, Christoph.Rensing, Ralf.Steinmetz}@kom.tu-darmstadt.de

Abstract. Recommender systems offer the opportunity for users to no longer have to search for resources but rather have these resources offered to them considering their personal needs and contexts. Additional semantics found in a folksonomy can be exploited to enhance the ranking of resources. These semantics have been analyzed in an e-learning scenario: CROKODIL. CROKODIL is a platform which supports the collaborative acquisition and management of learning resources. This paper proposes a conceptual architecture describing how these semantics can be integrated in a personalized recommender system for learning purposes.

1 Introduction

Recent research on personalized recommender systems has shown that the exploitation of semantic information found in folksonomy systems have led to improved recommendations [1]. Recommender systems have been applied to e-learning scenarios to help provide personalized support to learners by suggesting relevant items for learning purposes [6]. This raises the challenge of identifying relevant resources which match the current personal context and needs of the learners. Recommender systems in learning scenarios pose new requirements such as exploring and identifying which attributes represent relevance in a learning context [6]. It is therefore an ongoing challenge to meet the requirements of recommender systems in an e-learning scenario such as CROKODIL[1]. CROKODIL is based on a pedagogical concept which focuses on activities as the central structure to organize learning resources. The platform offers collaborative semantic tagging (thereby creating a folksonomy) as well as social network functionality to support the learning community [3].

Section 3 gives a brief analysis of the semantic information which could be exploited for the context-specific ranking of learning resources in the CROKODIL e-learning scenario. Section 4 describes a preliminary conceptual architecture of a personalized recommender system considering context-specific ranking. This concept will be implemented and evaluated in future work.

[1] http://www.crokodil.de, http://demo.crokodil.de (online as of 12.09.2011)

2 Related Work

A survey of the state-of-the-art on social tagging systems and how they extend the capabilities of recommender systems is given in [7]. Abel [1] shows that it is worth exploiting additional context information which are found in folksonomies to improve ranking strategies. Approaches are introduced which extend FolkRank to a context-sensitive ranking algorithm exploiting the additional semantics relating to groups of resources in GroupMe! [1]. In e-learning, recommender systems exist using context variables such as user attributes and domain specific information to provide personalized recommendations [6]. The concept in this paper proposes to use contextual information in folksonomies to rank learning resources in a personalized recommender system for e-learning.

3 Context Feature Analysis in an E-Learning Scenario

Contextual information in folksonomies can be categorized into four dimensions [1]: the *user context*, the *tag context*, the *resource context* and the *tag assignment context* (when a user attaches a tag to a resource). Considering the e-learning scenario CROKODIL, the available contextual information can be categorized as shown in Table 1. The user context comprises of: *learner groups* working together on a common task or activity, *user roles* such as the tutor role and *friendships* existing between individual learners. In future work, additional social information could be inferred from the learner's social network. When tagging resources in CROKODIL, *tag types* such as genre, topic, location, person and event can be assigned to the tags (a tag without a type is also possible). For example, the tag "Beer" of type "Location" refers to the town in Devon, England and not to the alcoholic beverage "beer", thus providing contextual information to the individual tags as well as to the tag assignments. *Activities* provide contextual information to a resource. In CROKODIL, activities structure which tasks need to be performed to achieve a defined goal. For example, in order to get ready to hold a presentation on German Culture, a learner creates an activity called "Prepare a talk about the Oktoberfest in Germany" having a sub-activity "Describe popular brands of beer in Bavaria". The required knowledge for this presentation

Table 1. Context Dimensions Applied to the CROKODIL Scenario

User Context	**Tag Context**	**Resource Context**	**Tag Assignment Context**
learner groups, user roles, friendships	tag types	activities	tag types, activities

is sought mostly via resources on the Web, such as on blogs or in Wikipedia. The appropriate resources are then attached to the activity. This activity thus provides contextual information to the resources. The tag assignment context comprises of *tag types* and *activities* as these both give additional contextual meaning when tagging.

4 Conceptual Architecture of a Personalized Recommender System for E-Learning

Fig.1 shows the design of a conceptual architecture of a personalized recommender system considering the CROKODIL e-learning scenario. This concept incorporates the ranking of learning resources considering the context features discussed in Section 3. Resources from friends, group members or the tutor could be given a higher weight than other resources. The tag types will have different weights according to the popularity of the tag type [2]. For example, a topic tag "Oktoberfest" is weighted higher than a genre tag "blog". Considering the activity the learner is presently working on, for example, "Prepare a talk about the Oktoberfest in Germany", resources belonging to activities nearer in the hierarchy will be weighted higher than resources belonging to an activity further away [2]. Therefore, resources belonging to its direct sub-activity "Describe popular brands of beer in Bavaria" are weighted higher than resources belonging to other activities further down the hierarchy. The results from the

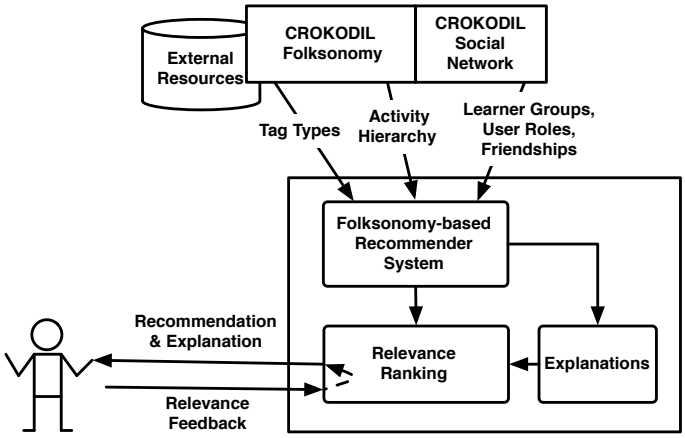

Fig. 1. Conceptual Architecture of a Personalized Recommender System

folksonomy-based recommnender system will be offered to the learner in the form of a ranked list. The learner is given the opportunity to give explicit feedback regarding these recommendations via a simple like/ dislike binary rating. This feedback is integrated into the ranking algorithm by applying Rochio's relevance feedback approach [5]. The feedback is thus used to further adapt and fit the recommendations to the learner's present learning context. In addition, explanations will be made about the recommended item, giving reasons why this item was recommended [4]. This will help to give a better understanding and stimulate the learner to reflect about the recommended learning resources. The

learner is then able to give qualified feedback about whether this recommendation is appropriate to the current learning needs or not. Finally, in order to enrich the variety of resources suggested by the recommender system, external learning resources from existing learning repositories such as ARIADNE [8] or the Open University's OpenLearn [2] will be considered.

5 Conclusion

In this paper, a conceptual design of a personalized recommender system for e-learning is described applying context-specific ranking of resources. An approach for a graph-based recommender system using semantic tag types has already been proposed in [2]. Next steps will be to implement these concepts and integrate them in CROKODIL. The impact of the various semantic information sources will be evaluated by considering several variants of the ranking algorithm, thereby showing which context features or combinations thereof are suitable to the CROKODIL learning scenario. Furthermore, investigations will need to be made on how explanations can be generated. In addition, the acceptance of these explanations, relevance feedback and recommendations of external learning resources will be evaluated with learners in a usability study.

Acknowledgements This work is supported by funds from the German Federal Ministry of Education and Research and the mark 01 PF 08015 A and from the European Social Fund of the European Union (ESF). The responsibility for the contents of this publication lies with the authors.

References

1. Abel, F.: Contextualization, User Modeling and Personalization in the Social Web. Phd thesis, Gottfried Wilhelm Leibniz University Hannover (2011)
2. Anjorin, M., Böhnstedt, D., Rensing, C.: Towards graph-based recommendations for resource-based learning using semantic tag types. DeLFI (Sept 2011)
3. Anjorin, M., Rensing, C., Bischoff, K., Bogner, C., Lehmann, L., Reger, A.L., Faltin, N., Steinacker, A., Lüdemann, A., Garcia, R.D.: Crokodil - a platform for collaborative resource-based learning. To be published: EC-TEL (Sept 2011)
4. Jannach, D., Zanker, M., Felfernig, A., Friedrich, G.: Recommender Systems An Introduction. CUP (2011)
5. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval: Relevance-Feedback und Query-Expansion. CUP (2009)
6. Manouselis, N., Drachsler, H., Vuorikari, R., Hummel, H.G.K., Koper, R.: Recommender systems in technology enhanced learning. In: Recommender Systems Handbook. Springer (2011)
7. Milicevic, A., Nanopoulos, A., Ivanovic, M.: Social tagging in recommender systems: a survey of the state-of-the-art and possible extensions. Artificial Intelligence Review 33, 187–209 (2010)
8. Neven, F., Duval, E.: Reusable learning objects: a survey of lom-based repositories. Proceedings of the tenth ACM int. conf. on Multimedia (2002)

[2] http://openlearn.open.ac.uk (online as of 12.09.2011)

Improving Tag-based Resource Recommendation with Association Rules on Folksonomies

Samia Beldjoudi[1], Hassina Seridi[1], and Catherine Faron Zucker[2]

[1] Laboratory of Electronic Document Management LabGED
Badji Mokhtar University, Annaba, Algeria
{seridi,beldjoudi}@labged.net
[2] I3S, Université Nice - Sophia Antipolis, CNRS
930 route des Colles, BP 145, 06930 Sophia Antipolis cedex, France
catherine.faron-zucker@unice.fr

Abstract. In this paper, we propose a method to analyze user profiles according to their tags in order to personalize the recommendation of resources. Our objective is to enrich the profiles of folksonomy users with pertinent resources. We argue that the automatic sharing of resources strengthens social links among actors and we exploit this idea to enrich user profiles by increasing the weights associated to web resources according to social relations. We base upon association rules which are a powerful method for discovering interesting relationships among a large set of data on the web. We extract association rules from folksonomies and use them to recommend supplementary resources associated to the tags involved in these rules. In this recommendation process, we reduce tag ambiguity by taking into account social similarities calculated on folksonomies.

Keywords: Folksonomies, Social Tagging, Association Rules, Tag-based Resource Recommendation, Tag Ambiguity.

1 Introduction

Web 2.0 technologies have created the conditions for new usages on the web which has become a social web. Users create, annotate, share and make public what they find interesting on the web and therefore are greatly involved in the evolution of the web. Folksonomies are one of the keystones of these new social practices: they are systems of classification derived from the practice and method of collaboratively creating and managing tags to annotate and categorize content. This practice is known as collaborative tagging or social tagging. Among the most popular social websites based on folksonomies let us cite Delicious which offers an effective way to conduct the collaborative management of social bookmarking, Flickr which is a photo management and sharing web application, Youtube and Dailymotion designed for sharing videos, Myspace and Odeo for sharing music files.

The basic principle of social tagging relies on three main elements: the user, the resource and the tag. The combination of these three elements enables the

development of semantic tools exploiting both folksonomies and annotations of web resources by users with tags. In this paper, we propose a method to analyze user profiles according to their tags in order to predict interesting personalized resources and recommend them. In other words, our objective is to enrich the profiles of folksonomy users with pertinent resources. We argue that the automatic sharing of resources strengthens social links among actors and we exploit this idea to reduce tag ambiguity in the recommendation process by increasing the weights associated to web resources according to social similarities. We base upon association rules which are a powerfull method for discovering interesting relationships among a large set of data on the web.

We insist on the fact that our final aim is not to suggest tags to users: each time a resource is presented to a user, the tags already used to annotate this resource are indicated but the user is free to tag the resource by choosing a tag among them or using a new one. Our aim is to recommend resources which are annotated with tags suggested by association rules, in order to enrich user profiles with these resources (if they validate them). In other words, our aim is to enrich user profiles based on similarities between users and association rules and by doing so to increase the community effect when suggesting resources to a given user. Our approach comes from a new view on the community effect in folksonomies since it aims at automatically strengthening existing correlations between different members of online communities, without involving the user in this process. The fact of suggesting to each user some resources considered useful or interesting for him without him specifying explicit tags, this can significantly improve folksonomy-based recommendation systems, because the man-machine interaction and therefore the user effort are considerably reduced.

This paper is organized as follows: Section 2 is an overview of the main contributions related to our work. Section 3 is dedicated to the presentation of our approach. In section 4 we present and discuss the results of some experiences we conducted to measure the performance of our approach. Conclusion and future works are described in Section 6.

2 Related Works

2.1 Tag Recommendation

The general aim of tag recommendation systems is to help users choose the appropriate tags when annotating resources in order to increase the weights associated to each tag and so cross a step up to building a common vocabulary in these systems. Among the many works adressing this problem, let us cite that of Schmitz et al. [13] who showed how association rules can be adopted to analyze and structure folksonomies and how these folksonomies can be used for learning ontologies and supporting emergent semantics. Their approach consists in reducing the ternary dimension of a folksonomy by projecting it on a triadic context, and then in extracting association rules from this two-dimensional projection. An association rule $A \Rightarrow B$ is interpreted in two ways: users assigning tags in

A to some resources often also assign tags in B to them or users labeling the resources in A with some tags often also assign these tags to the resources in B.

Another noticeable contribution is that of Jäschke et al. [7] who present a formal model and a new search algorithm called FolkRank, especially designed for folksonomies. It is also applied to find communities within a folksonomy and is used to structure search results. The authors have exploited the idea of the PageRank algorithm, which consists in considering a web page as important when there are several other pages connected to it. In FolkRank, a resource tagged by an important number of users with an important number of tags becomes important. The same type of relationships becomes true for tags and users. The idea is to create graphs, and to associate to each node of these graphs a weight indicating its importance.

Gemmell et al. [5] propose a tag-based recommendation method based on the adaptation of the K-nearest neighbor algorithm so that it accepts as input both a user U and a resource R and gives out a set of tags T. The interest of this approach is to orient users to use the same tags, and thus increase the chance of building a common vocabulary used by all members.

2.2 Resource Recommendation

The general aim of resource recommendation systems is to enrich the quantity and relevance of the recommended resources. Among the works adressing this problem, let us cite De Meo et al. [4] who propose an approach based on the principle of query expansion. The aim is to recommend resources to users searching by tags by enhancing their profiles represented by their tag-based queries. The principle of the approach is to enrich user profiles by additional tags discovered through the exploration of the two graphs TRG and TUG representing the relations respectively between tags and resources and between tags and users.

Let us note that, when compared to the works on tag recommendation, the principle is the same: extract the most appropriate tags. Most of the techniques performed in this process demonstrate their contributions for building a language more or less common between users of folksonomies. However the methods that are used to achieve this goal are different from one approach to another. Regarding the work of De Meo et al., we can say that the results obtained with their approach show that the idea of proposing a system of resource recommendation is pertinent: the rates of precision and recall are optimistic. However the fact of forcing the user to specify a list of tags in order to get resources can generate a cognitive overload and it obliges the system to focus on the participation of the user to perform its recommendation procedure. Moreover the technique that has been designed in this work does not take into account the semantics between tags, in particular it cannot distinguish between ambiguous tags and therefore it may recommend resources that will be rejected by the user because they are not close to his preferences.

2.3 Resolving Tag Ambiguity

According to Mathes [9], "the terms in a folksonomy have inherent ambiguity as different users apply terms to documents in different ways. There are no explicit systematic guidelines and no scope notes". For this reason we are concerned by the problem of tag ambiguity in our approach of tag-based resource recommendation.

Among the most important contributions on resolving tag ambiguity or extracting the semantic links between tags in a folksonomy, we start with [10] where Mika has proposed to extend the traditional bipartite model of ontologies to a tripartite one where the instances are keywords used by the actors of the system in order to annotate web resources. In his paper, Mika focuses on social network analysis in order to extract lightweight ontologies, and therefore semantics between the terms used by the actors. In [6], Gruber stated that there is no contrast between ontologies and folksonomies, and so recommended to build an ontology of folksonomy. According to Gruber, the problem of the lack of semantic links between terms in folksonomies can be easily resolved by representing folksonomies with ontologies. Specia and Motta [14] in their turn have preferred the use of ontologies to extract the semantics of tags. The proposed method consists in building clusters of tags, and then trying to identify possible relationships between tags in the same cluster. The authors have chosen to reuse available ontologies on the semantic web in order to express correlations which can hold between tags. An attempt to automate this method has been done by Angeletou et al. [2].

Buffa et al. [3] present a semantic wiki reconciling two trends of the future web: a semantically augmented web and a web of social applications where every user is an active provider as well as a consumer of information. The goal here is to exploit the force of ontologies and semantic web standard languages in order to improve social tagging. According to the authors, with this approach, tagging remains easy and becomes both motivating and unambiguous. The niceTag project of Limpens et al. [8] is focused on the same principle: the use of ontologies in order to extract the semantics between tags in a system. In addition, the interactions among users and the system are used to validate or invalidate automatic treatments carried out on tags. The authors have proposed methods to build lightweight ontologies which can be used to suggest terms semantically close during a tag-based search of documents. Pan et al. [11] address the problem of tag ambiguity by expanding folksonomy search with ontologies. They proposed to expand folksonomies in order to avoid bothering users with the rigidity of ontologies. During a keyword-based search of resources, the set of ambiguous used terms is concatenated with other tags so as to increase the precision of the search results.

To sum up, most of the works aspire to bring together ontologies and folksonomies as a solution to resolve tag ambiguity and overcome the lack of semantic links between tags. Sure enough the approaches described in this section show that the social nature of resource sharing is not in contradiction with the possibilities offered by ontology-based systems. But the rigidity that characterizes

ontologies and the need for an expert who must control and organize the links between terms as in [6] seem a little cumbersome and too much expensive. Even the structures automatically extracted as in [10] still suffer from the ambiguity of concepts. Regarding the work of [14], we can say that the use of semantic web ontologies for extracting relationships between terms is not sufficient, because as the semantic web includes some specific domain ontology, that will push back the problem. Also the expertise of users which was introduced in [8] is characterized by the complexity of its exploitation. As a result we propose an approach of tag-based resource recommendation where we aim to resolve tag ambiguity without explicitly using ontologies.

3 Resource Recommendations based on Association Rules

3.1 Association Rules: Basic Definitions

In data mining, learning association rules is a widely used method for discovering interesting relations among variables in large databases. [12] describes analyzing and presenting *strong rules* discovered in databases using different measures of interestingness. Based on this concept of *strong rules*, [1] introduces association rules for discovering regularities between products in large scale transaction data recorded by point-of-sale (POS) systems in supermarkets. For example, the rule $\{onions, potatoes\} \Rightarrow burgers$ found in the sales data of a supermarket indicates that if a customer buys onions and potatoes together, he is likely to also buy burgers[3]. According to the original definition by [1], the problem of association rule mining is defined as follows:

Definition 1. *Let* $I = \{i_1, \ldots, i_n\}$ *be a set of* n *binary attributes called* items. *A* rule *is an implication* $X \Rightarrow Y$ *where* $X, Y \subseteq I$ *and* $X \cap Y = \emptyset$. *The sets of items (itemsets)* X *and* Y *are called* antecedent *and* consequent *of the rule respectively.*

To select interesting rules from the set of all possible rules in a database $D = \{d_1, \ldots, d_m\}$, with each transaction in D containing a subset of items in I, two measures are commonly used: support and confidence.

Definition 2. *The* support $supp(X)$ *of an itemset* X *is the proportion of transactions in* D *which contain* X.

Definition 3. *The* confidence $conf(X \rightarrow Y)$ *of a rule* $X \rightarrow Y$ *measures the proportion of transactions in* D *that contain* Y *among those that contain* X. $conf(X \rightarrow Y) = \frac{supp(X \cup Y)}{supp(X)}$.

Let us illustrate these notions on the following dataset.

[3] http://en.wikipedia.org/wiki/Association rules [Retrieved 13 May 2011]

Transaction ID	ItemSet
1	Bread, Cream, Water
2	Cream
3	Bread, Cream, Milk
4	Water
5	Cream, Water

From this dataset, we can extract the rule $Bread \Rightarrow Cream$ with a confidence $conf(Bread \Rightarrow Cream) = \frac{supp(\{Cream, Bread\})}{supp(Bread} \frac{2/5}{4/5} = 1/2$.

To be selected as significant and interesting, association rules are usually required to satisfy a user-specified minimum support and a user-specified minimum confidence. The process of generating association rules is usually split up into two separate steps: First, the minimum support constraint is applied to find all frequent itemsets in a database. Second, the minimum confidence constraint is applied among the rules involving these frequent itemsets. The quality of the extraction algorithm thus strongly depends on the values chosen by the user for the minimum support and minimum confidence, which adequacy is relative to the application.

3.2 Association Rules and Folksonomies

A folksonomy is a tuple $F =< U, T, R, A >$ where U, T and R represent respectively a set of users, a set of tags and a set of resources, and A represents the relationships between the three preceding elements, i.e. $A \subseteq U \times T \times R$ [10]. In our approach we consider a folksonomy as being a tripartite model where web resources are associated with a user to a list of tags. Therefore we have extracted three social networks from our folksonomy, which represent three different viewpoints on social interactions: one network relating tags and users, a second one relating tags and resources and a third one relating users and resources. We represent these social networks by three matrices TU, TR, UR:

- $TU = [X_{ij}]$ where $X_{ij} = \begin{cases} 1 \text{ if } \exists r \in R, < u_j, t_i, r > \in A \\ 0 \text{ otherwise} \end{cases}$
- $TR = [Y_{ij}]$ where $Y_{ij} = \begin{cases} 1 \text{ if } \exists u \in U, < u, t_i, r_j > \in A \\ 0 \text{ otherwise} \end{cases}$
- $UR = [Z_{ij}]$ where $Z_{ij} = \begin{cases} 1 \text{ if } \exists t \in T, < u_i, t, r_j > \in A \\ 0 \text{ otherwise} \end{cases}$
- RU, RT and UT are the transposed matrices of UR, TR and TU.

This enables us to analyze the correlations captured from the different social interactions. We used Pajek, a tool which has already been used by Mika to analyze large networks [10].

To apply an association rule method to folksonomies, we represent each user in a folksonomy by a transaction ID and the tags he uses by the set of items which are in this transaction. The following table provides an illustrative example of a dataset of user tags.

Transaction ID	Itemset
U_1	Computer, Programming
U_2	Computer, Apple
U_3	Kitchen, Apple
U_4	Programming
U_5	Kitchen

Our goal is to find correlations between tags, i.e. to find tags which frequently appear together, in order to extract those which are not used by one particular user but which are often used by other users close to him in the social network. For example, let us consider a dataset in which it occurs that many users who use the tag *Software* also employ the tag *Java*. We aim at extracting a rule $Software \Rightarrow Java$ so that we can enrich the profiles of users who employ the tag *Software* but not the tag *Java*, by the resources tagged with *Java*. Among the wide range of algorithms proposed to extract interesting association rules, we use the one known as *Apriori* [1].

Once the rules are extracted, our recommendation system proceeds as follows. For each extracted rule, we test whether the tags which are in the antecedent of the rule are used by the current user. If it is the case then the resources tagged with each tag found in the consequent of the rule are candidate to be recommended by the system. The effectiveness of the recommendation depends on the resolution of tag ambiguity, as explained in the next section.

3.3 Resolving Tag Ambiguity in Recommendation

A tag can have several meanings, i.e. refer to several concepts. Therefore, a basic tag-based recommendation system would equally recommend resources relative to fruits or to computers for a user searching with the tag *apple*. The resolution of tag ambiguity is specially crucial in our approach where some tags which are used to recommend resources are not directly used by the user but deduced with association rules. To resolve the problem of tag ambiguity in recommendation, we propose to measure the similarity between users to identify those who have similar preferences and therefore adapt the recommendation to user profiles.

Similarity between Users. For each extracted association rule whose antecedent applies to the user searching for resources, we measure the similarities between this user and the users of his social network who use the tags occuring in the consequent of the rule. The resources associated to these tags are recommended to the user depending on these similarities.

To measure the similarity between two users u_1 and u_2, we represent each of them by the vector of binary numbers representing all his tags (extracted from matrix UR) and we calculate the cosines of the angle between the two vectors:

$$sim(u_1, u_2) = cos(v_1, v_2) = \frac{v_1.v_2}{\|v_1\|^2.\|v_2\|^2}$$

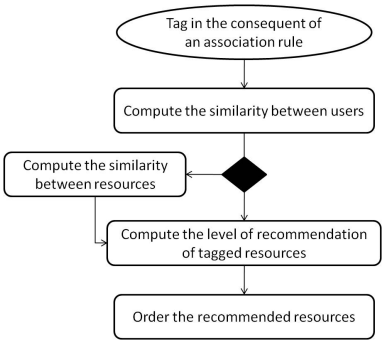

Fig. 1. Overview of the recommendation process

Similarity between Resources. To avoid the *cold start* problem which generally results from a lack of data required by the system in order to make a good recommendation, when the user of the recommendation system is not yet similar to other users, we also measure the similarity between the resources which would be recommended by the system (as related to a tag occuring in the consequent of an association rule) and those which are already recommended to the user.

To measure the similarity between two resources r_1 and r_2, we represent each of them by the vector of binary numbers representing all its tags (extracted from matrix TR) and we calculate the cosines of the angle between the two vectors.

Levels of Recommendation. Each resource recommended by the system is first associated an initial weight based on the similarities between users. Above a threshold fixed in $[0, \dots 1]$, we qualify the resource as *highly recommended*. Under this threshold, we consider the similarity between resources and we similarly highly recommend the resources which weights calculated on the product matrix $RR = RT \times TR$ are above a given threshold. Otherwise, we compute the average ratio between the number of resources shared by the user of the recommendation system with his social network and the number of resources used by him. These numbers are given by the product matrix $RR = RU \times UR$. Above a threshold fixed in [0..1], we qualify the resource as *highly recommended*; under this threshold, it is simply *recommended* or *weakly recommended*if the similarity is close to zero.

Whole Process of Recommendation. The activity diagram in Figure 1 gives an overview of the whole process of recommendation including the key steps described above to analyze existing interactions between the different elements of a folksonomy, especially those between users.

Let us note that our recommendation system is flexible, since the user can interact to accept or reject the recommended resources.

Let us consider the example of a folksonomy represented through the following three matrices TU, TR and UR:

	U_1	U_2	U_3	U_4	U_5
computer	1	1	0	0	0
kitchen	0	0	1	0	1
programming	1	0	0	1	0
apple	0	1	1	0	0

	R_1	R_2	R_3	R_4	R_5
computer	1	1	0	0	0
kitchen	0	0	0	1	1
programming	1	1	0	0	0
apple	0	0	1	0	1

	R_1	R_2	R_3	R_4	R_5
U_1	1	1	0	0	0
U_2	0	1	1	0	0
U_3	0	0	0	1	1
U_4	1	0	0	0	0
U_5	1	0	0	1	1

Let us now suppose that we have extracted the interesting association rule *computer* $\Rightarrow$ *apple*. Matrix TU shows that tag *computer* is used by user U_1. Since *apple* is in the consequent of the rule, matrix TR shows that resources R_3 and R_5 are candidates for a recommendation to U_1. Matrix UT shows that *apple* is used by users U_2 and U_3. Then we calculate the similarity between U_1 and U_2 and the similarity between U_1 and U_5, based on matrix $UU = UT \times TU$:

	U_1	U_2	U_3	U_4	U_5
U_1	2	1	0	1	0
U_2	1	2	1	0	0
U_3	0	1	2	0	1
U_4	1	0	0	1	0
U_5	0	0	1	0	1

$$sim(U_1, U_2) = cos(UU_1, UU_2) = \frac{(2\ 1\ 0\ 1\ 0)\times(1\ 2\ 1\ 0\ 0)}{\sqrt{4+1+0+1+0}\times\sqrt{1+4+1+0+0}} = \frac{4}{2\times\sqrt{6}} = 0.66$$

$$sim(U_1, U_3) = cos(UU_1, UU_3) = \frac{(2\ 1\ 0\ 1\ 0)\times(0\ 1\ 2\ 0\ 1)}{\sqrt{4+1+0+1+0}\times\sqrt{0+1+4+0+1}} = \frac{1}{2\times\sqrt{6}} = 0.16$$

U_1 and U_2 show higher cosine similarity than U_1 and U_3 . Then, among the resources tagged with *apple*, namely R_3 and R_5 (see matrix TR), those tagged by U_2 are highly recommended to U_1: it is only the case of R_3 (see matrix UR).

U_1 and U_3 are not similar. Then, among the resources tagged with *apple*, we compute the similarity of those tagged by U_3, namely R_5, with those already recommended by the system, namely R_3. It is based on matrix $RR = RT \times TR$:

$$sim(R_3, R_5) = cos(RR_3, RR_5)$$

R_5 and R_3 are not similar. Then R_5 is weakly recommended to U_1.

4 Experiments

In this section, we describe some experiments over two datasets and we analyze and discuss our results. We have developped a simple application with a convivial interface enabling the user to log in and get a personalized ordered list of recommended resources — depending on his tagging activity and social network.

4.1 Experiment over a subset of the del.icio.us database

In order to validate our approach, we have conducted a first experiment with the del.icio.us database. Our test base comprises 207 tag assignments involving 21 users, 97 tags — some of which are ambiguous —, 92 resources — each having possibly several tags and several users. Our system has extracted a set of 17 association rules from the analysis of the dataset with a support equal to 0.5 and a confidence equal to 0.6. We have for example the rule $news \Rightarrow software$: 60% of the users using the tag *news* also use the tag $software$.

To demonstrate the validity of our approach, we have distinguished two classes of users: the first one contains the users who have employed ambiguous tags and the other one those who did not use those tags. This ambiguity of tags has been subjectively decided: for instance *apple* is ambiguous and *software* is not.

Not surprisingly, our experiment has showed that, by applying the extracted association rules, the resources associated to non ambiguous tags are highly recommended. It has also showed that, in the case of rules involving ambiguous tags, our system recommends to the user the resources which are close to his interests with a high level of recommendation and, on the contrary, those which are far from his interests with a low level of recommendation.

4.2 Evaluation of our Recommendation System over an Experimental Dataset

To evaluate the quality of our recommendation system, we used the following three metrics: recall, precision and F1 metric. Precision measures the ability of the system to reject all the resources which are not relevant. It is given by the ratio between the number of the relevant resources recommended and that of the recommended resources. Recall measures the ability of the system to retrieve all the relevant resources. It is given by the ratio between the number of relevant resources recommended and that of all the relevant resources in the database. F1 is a combination of the two previous metrics; it is defined by the following formula:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Because the calculation of these metrics requires the knowledge of all relevant resources for each user in order to compare the results provided by our recommendation system and those which are actually preferred by each user, we have built a database by inviting 6 users to participate to an experiment.

We first made a prototype of a folksonomy in the form of a website. Then we asked the users to specify their preferred resources. Finally, we asked each user to tag a set of resources among 18 ones available on our website, by using free keywords. Based on this dataset, we extracted 10 association rules with a support equal to 0.5 and a confidence equal to 0.6. Afterwards we calculated the three metrics for each participant in our test, for each tag. The following table presents the average values of the metrics we obtained for our 6 users:

$User$	$Precision$	$Recall$	$F1$
U_1	0.66	0.57	0.61
U_2	1.00	0.85	0.91
U_3	0.60	1.00	0.75
U_4	0.44	0.80	0.56
U_5	0.66	0.66	0.66
U_6	0.37	1.00	0.54
$Average$	0.62	0.81	0.67

These are quite encouraging results, showing that our approach of recommendation adapted to user profiles is truly able to help users when searching for resources.

4.3 Discussion

We have shown through our experiments that the use of data mining methods and tools has proved its effectiveness for folksonomy-based recommendation. The results of our data sample are optimistic and so we can say that the community effect which characterizes folksonomies has showed its power in users profiles enrichment. This enhancement can significantly help to improve recommendation systems. At the same time that our approach contributes to increase the weights associated to the relevant resources, it reduces tag ambiguity: every time when there are shared resources between two users, the system can avoid the trap of tag ambiguity in the research phase and it will test the similarity between resources when the users are not similar. The extraction of association rules is based on tags rather than on resources because we believe that tag popularity in folksonomies is greater than resource popularity and the meaning of tags in these systems is more significant than that of resources: the same resource can be used for many different purposes.

5 Conclusion and Future Works

In this paper, we have proposed a method to automatically enrich user profiles with a set of relevant resources, based on social networks and folksonomies. We exploit association rules extracted from the social relations in a folksonomy to recommend resources tagged with terms occuring in these rules by other users close in the social network. Our objective is to create a consensus among users of a same network in order to teach them how they can organize their web resources in a correct and optimal manner.

We have tested our approach on a small amount of data where we have obtained good results, but the validation of our approach still requires a larger sample set. In order to continue and improve our work, we aim to address the problem of scalability of our approach on larger databases. The measure of similarity we use is based on several products of matrices whose dimensions are the

numbers of resources and tags of a folksonomy. In real scenariis, these dimensions are usually too large. We are intending to explore matrix factorization and latent semantic analysis.

References

1. R. Agraval, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proc. of the ACM SIGMOD Int. Conference on Management of Data, Washington, USA*, 1993.
2. S. Angeletou, M. Sabou, L. Specia, and E. Motta. Bridging the Gap Between Folksonomies and the Semantic Web: An Experience Report. In *Proc. of ESWC workshop on Bridging the Gap between Semantic Web and Web*, 2007.
3. M. Buffa, F. Gandon, G. Ereteo, P. Sander, and C. Faron. SweetWiki: A semantic Wiki. *Journal of Web Semantics*.
4. P. De Meo, G. Quattrone, and D. Ursino. A query expansion and user profile enrichment approach to improve the performance of recommender systems operating on a folksonomy. *User Modeling and User-Adapted Interaction*, 20(1), 2010.
5. J. Gemmell, T. Schimoler, M. Ramezani, and B. Mobasher. Adapting k-nearest neighbor for tag recommendation in folksonomies. In *Proc. of 7th Workshop on Intelligent Techniques for Web Personalization & Recommender Systems (ITWP'09), Pasadena, California, USA, in conjunction with IJCAI 2009*, 2009.
6. T. Gruber. *TagOntology - a way to agree on the semantics of tagging data*. http://tomgruber.org/writing/tagontology.htm, 2005.
7. R. Jäschke, L.B. Marinho, Hotho A., L. Schmidt-Thieme, and G. Stumme. Tag recommendations in folksonomies. In *Proc. of 11th Eur. Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD), Warsaw, Poland*, volume 4702 of *LNCS*. Springer, 2007.
8. F. Limpens, F. Gandon, and M. Buffa. Collaborative semantic structuring of folksonomies. In *Proc. of IEEE/WIC/ACM Int. Conference on Web Intelligence (WI 2009), Milan, Italy*, 2009.
9. A. Mathes. *Folksonomies - Cooperative Classification and Communication Through Shared Metadata*. http://www.adammathes.com/academic/computer-mediated-communication/folksonomies.html, 2004.
10. P. Mika. Ontologies are us: A unified model of social networks and semantics. In *Proc. of 4th Int. Semantic Web Conference (ISWC 2005), Galway, Ireland*, volume 3729 of *LNCS*. Springer, 2005.
11. J.Z. Pan, S. Taylor, and E. Thomas. Reducing ambiguity in tagging systems with folksonomy search expansion. In *Proc. of 6th Eur. Semantic Web Conference (ESWC 2009), Heraklion, Greece*, volume 5554 of *LNCS*. Springer, 2009.
12. G. Piatetsky-Shapiro. Discovery, analysis, and presentation of strong rules. In *Knowledge Discovery in Databases*. AAAI/MIT Press, 1991.
13. C. Schmitz, A. Hotho, R. Jäschkee, and G. Stumme. Mining association rules in folksonomies. In *Proc. of IFCS 2006 Conference: Data Science and Classification, Ljubljana, Slovenia*. Springer, 2006.
14. L. Specia and E. Motta. Integrating folksonomies with the semantic web. In *Proc. of 4th Eur. Semantic Web Conference (ESWC 2007), Innsbruck, Austria*, volume 4519 of *Lecture Notes in Computer Science*. Springer, 2007.

A Model for Assisting Business Users along Analytical Processes

Corentin Follenfant[1,2], David Trastour[1], and Olivier Corby[2]

[1] SAP Research, SAP Labs France SAS
805 avenue du Dr. Maurice Donat, BP 1216, 06254 Mougins Cedex, France
firstname.lastname@sap.com
[2] INRIA Sophia Antipolis Méditerranée,
2004 route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex, France
firstname.lastname@inria.fr

Abstract. User-centric business intelligence aims at empowering analysts who interact with complex tools, by allowing them to perform accurate data manipulations and analysis without necessarily requiring IT expertise and knowledge of underlying data specifications. Recommender systems contribute to easing their tasks but most of them operate inside walled gardens and cannot assist properly the user throughout his BI workflow. In this paper we introduce a lightweight vocabulary intended to capture fragments of analytical workflows as multidimensional data transformations, within a Semantic Web framework. We utilize this model for calculating content-based recommendations.

Keywords: business intelligence, content-based recommendation, analytical layers, usage semantics

1 Introduction

Traditional Business Intelligence (BI) platforms provide tools that are designed to cover a wide range of operations in a data-driven decision making workflow. The prerequisites steps concern data extraction, cleansing and integration. On top of them come what we call analytical processes: it includes querying, analysis and visual data consumption. These operations often require various technical competencies, for instance SQL expertise and a good understanding of underlying relational models. Since the current businesses landscapes rarely allow users to maintain both technical and analytical profiles, this hinders the decision makers' capacity to leverage the tools at their full potential without requiring extensive assistance from IT departments.

A common approach to tackle this problem is by providing contextual assistance through recommender systems in order to suggest items such as datasets, business entities, queries or visualizations, depending on the current step of the user into his analytical process. Although those systems start to work beyond the legacy $User \times Item$ space and integrate broader contextual information [1],

they can hardly be applied on the whole analytical process as items become heterogeneous and implicit rating functions complex.

In this paper we propose an information model based on Semantic Web technologies, designed to capture semantics of sequential transformations applied on multidimensional data structures. We describe a content-based recommendation use case of this model, where items' granularity vary from business entities to analytical processes aspects, and their utility is computed by arbitrary functions over their usage statistics.

2 Context

BI systems architecture can be split in three layers: first, raw data mainly comes from operational systems where it is stored into heterogeneous databases. Secondly, Extract, Transform, Load (ETL) and integration processes federate those sources into data warehouses. Combined with metadata management components, they expose data through a (hyper)cube model of business entities named after users' familiar terminology: *measures* (factual data, e.g. `Revenue`) that can be driven by *dimensions* (dimensional data, e.g. `Country`, `Year`, `Store`). Thirdly, analytical processes of end user applications such as reporting tools begin by querying the data warehouse layer to retrieve multidimensional data, before applying transformations and visualizations as the user authors his report.

Number of efforts are devoted to making these tools more usable and accessible by involving recommender systems for specific steps of analytical processes. This goes from querying the data warehouse [2, 4], to higher-level workflows such as exploration [6, 3]. Assisting users throughout the analytical process requires a common metamodel to capture multidimensional operations.

3 An Information Model to Capture Usage Semantics

The RDF Data Cube vocabulary introduced by Cyganiak et al.[3] is mainly intended to enable the publication of statistics, and thus provides a metamodel for multidimensional datasets. In order to enable high level description of analytical processes that can be performed within reporting tools, we extend the vocabulary[4] as presented in figure 1. Processes are split into sequences of multidimensional data transformations that

are derived from users' interactions with the report design tool. Each transformation corresponds to a `mda:AnalysisLayer` subclass instance. Like a web service in the OWL-S[5] fashion, it consumes and exposes interfaces of multidimensional data structures described by `qb:DataStructureDefinition` individuals

[3] http://publishing-statistical-data.googlecode.com/svn/trunk/specs/src/main/html/cube.html

[4] The RDFS classes and properties of our extension use the `mda:` prefix, for MultiDimensional Analysis. The `qb:` prefix refers to Data Cube vocabulary.

[5] http://www.ai.sri.com/daml/services/owl-s/1.2/overview/

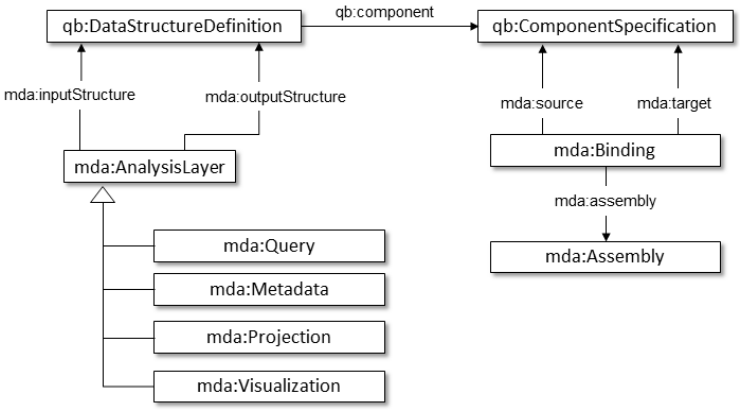

Fig. 1: Multidimensional Analysis extension outline

through `mda:inputStructure` and `mda:outputStructure` properties. Transformations can thus be interchanged and reused for describing different snippets (reports elements) that share layers of the analytical process. These layers are connected together through sets of bindings, assemblies, that plug the multidimensional structures atoms, business entities represented by `qb:ComponentSpecification` individuals.

4 Experiments

Aiming at providing assistance and reuse capabilities to business users who consume reports and have authoring expectations, we leverage our model to compute content-based recommendations. We ran a snippet crawler against a repository of BI reports in order to harvest snippets' underlying analysis sequences and populate an RDF graph with generated triples. The source is an internal repository storing 645 reports used to perform analysis on 101 data warehouse models. A total of 8121 snippets were extracted, all of them being split into up to five layers of transformations over business entities.

Usage statistics measures are extracted from this graph and then used to feed utility functions of a recommender system, for which we identified two granularities of recommendations. First, basic top-k SPARQL queries can suggest business entities that are likely to complete a `ProjectionLayer`, that is adding dimensions or measures to a snippet's axis. As opposed to this horizontal recommendation, the vertical one aims at recommending entire layers in the analytical process in order to assist a user into reusing relevant transformations or visualizations that can be applied on top of a query. To do so, we compute item similiarity measures for `AnalyticalLayer` individuals that are not already connected together through assemblies. The similarity measure strategy is adapted from the Levenshtein distance implemented in the iSPARQL extension [5].

5 Conclusion & Future Work

We introduced an approach to reuse-oriented analytical processes modelling with Semantic Web technologies, which captures the different steps of analysis as multidimensional structures transformations. The first use case concerns BI reporting applications, for which we exemplified our model by triplifying a repository of reports snippets. The graph data resulting from this initial experiment can be queried for basic usage statistics or content similarity measures with simple SPARQL aggregates or iSPARQL statements.

Areas of research that we expect will require further investigation include the formal definition of matching criteria between layers of analytical processes, and its implementation as a specific similarity strategy for analytical layers' RDF resources in iSPARQL; and mechanisms to capture or infer the provenance of the data surfacing into end user visualizations [7]. Finally, we will check the model's genericity by using crawlers for BI applications besides reporting tools, such as dashboarding or exploration ones. This will enable representing analytical processes composed of transformations and data derived from different environments, for instance statistical data published with respect to RDF Data Cube vocabulary and external to the data warehouses.

References

1. Adomavicius, G., Tuzhilin, A.: Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. IEEE Transactions on Knowledge and Data Engineering 17, 734–749 (2005)
2. Chatzopoulou, G., Eirinaki, M., Polyzotis, N.: Query Recommendations for Interactive Database Exploration. In: Proceedings of the 21st International Conference on Scientific and Statistical Database Management. pp. 3–18. SSDBM 2009, Springer-Verlag, Berlin, Heidelberg (2009)
3. Jerbi, H., Ravat, F., Teste, O., Zurfluh, G.: Applying Recommendation Technology in OLAP Systems. In: Aalst, W., Mylopoulos, J., Rosemann, M., Shaw, M.J., Szyperski, C., Filipe, J., Cordeiro, J. (eds.) Enterprise Information Systems. Lecture Notes in Business Information Processing, Springer Berlin Heidelberg (2009)
4. Khoussainova, N., Kwon, Y., Balazinska, M., Suciu, D.: SnipSuggest: Context-aware Autocompletion for SQL. Proc. VLDB Endow. 4, 22–33 (October 2010)
5. Kiefer, C., Bernstein, A., Stocker, M.: The Fundamentals of iSPARQL: A Virtual Triple Approach for Similarity-Based Semantic Web Tasks. In: The Semantic Web. Lecture Notes in Computer Science, Springer Berlin / Heidelberg (2007)
6. Marcel, P., Negre, E.: A Survey of Query Recommendation Techniques for Datawarehouse Exploration. In: Proceedings of the 7th Conference on Data Warehousing and On-Line Analysis. EDA '11 (June 2011)
7. Reisser, A., Priebe, T.: Utilizing Semantic Web Technologies for Efficient Data Lineage and Impact Analyses in Data Warehouse Environments. In: Proceedings of the 2009 20th International Workshop on Database and Expert Systems Application. DEXA '09, IEEE Computer Society, Washington, DC, USA (2009)

A Privacy Preference Manager for the Social Semantic Web*

Owen Sacco and Alexandre Passant

Digital Enterprise Research Institute,
National University of Ireland, Galway, Ireland
{firstname.lastname}@deri.org
http://www.deri.ie/

Abstract. Current Social Web applications provide users with means to easily publish their personal information on the Web. However, once published, users cannot control how their data can be accessed apart from applying generic preferences (such as "friends" or "family"). In this paper, we describe how we enable finer-grained privacy preferences using the Privacy Preference Ontology (PPO); a light-weight vocabulary for defining privacy settings on the Social Web. In particular, we describe the formal semantic model of PPO and also present *MyPrivacyManager*, a privacy preference manager that let users (1) create privacy preferences using the aforementioned ontology and (2) restrict access to their data to third-party users based on profile features such as interests, relationships and common attributes.

1 Introduction

In the past few years, the growing number of personal information shared on the Web (through Web 2.0 applications) increased awareness regarding privacy and personal data. A recent study [2] showed that privacy in Social Networks is a major concern when private news are publicly shared, revealing that most users are aware of privacy settings and have set them at least once since 2009.

Most Social Networks provide privacy settings restricting access to private data to those who are in the user's friends lists (i.e. their "social graph") such as Facebook's privacy preferences and Google+ circles. Yet, the study shows that users require more complex privacy settings as current systems do not meet their requirements.

We aim to solve these privacy shortfalls with the Privacy Preference Ontology (PPO)[1]. This model can be applied to any social data as long as it is modelled in RDF (for instance using FOAF[2], SIOC[3] or OGP[4] can be used to

* This work is funded by the Science Foundation Ireland under grant number SFI/08/CE/I1380 (Líon 2) and by an IRCSET scholarship co-funded by Cisco systems.

[1] http://vocab.deri.ie/ppo#

[2] Friend-of-a-Friend — http://www.foaf-project.org

[3] Semantically-Interlinked Online Communities - http://sioc-project.org/

[4] Open Graph Protocol - http://ogp.me/

define such fine-grained settings. While data from major websites is generally not modelled directly in RDF, wrappers can easily be implemented through their API. In addition, PPO can be natively used in Social Semantic Web applications, i.e. Social Web applications directly using RDF to model their data, such as Semantic MediaWiki or Drupal 7.

In this paper, we detail the formal model of PPO, and also present a privacy preference manager (*MyPrivacyManager*), letting users: (1) create privacy preferences described using PPO for their FOAF profiles; and (2) view other user's profiles, filtered according to their privacy preferences.

The remainder of the paper is organised as follows: Section 2 provides an overview of the Privacy Preference Ontology (PPO) and presents use cases for PPO. In Section 3, we present our formal model. In section 4 we present the implementation of *MyPrivacyManager*. Section 5 discusses related work and Section 6 presents future work and concludes the paper.

2 The Privacy Preference Ontology (PPO)

2.1 Overview

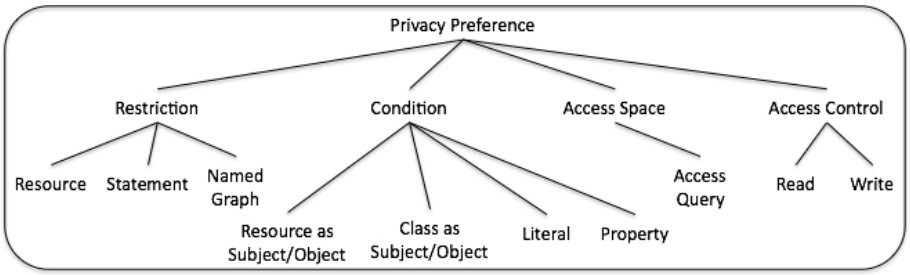

Fig. 1. A high level graphical representation of the properties that make up a privacy preference.

The Privacy Preference Ontology (PPO) [10] provides a light-weight vocabulary enabling Linked Data creators to describe fine-grained privacy preferences for restricting (or granting) access to specific data. PPO can be used for instance to restrict part of a FOAF user profile only to users that have similar interests. It provides a machine-readable way to define settings such as "Provide my phone number only to DERI colleagues" or "Grant write access to this picture gallery only to people I've met in real-life".

As we deal with Semantic Web data, a privacy preference (Figure 1), defines: (1) which resource, statement or named graph to restrict access to ; (2) the conditions to refine what to restrict; (3) the access control type; and (4) a SPARQL query, known as an `AccessSpace` containing a graph pattern representing what

must be satisfied by the user requesting information. The access control type is defined by using the Web Access Control (WAC)[5] vocabulary which defines the `Read` and `Write` access control privileges (for reading or updating data).

2.2 Use Case

As mentioned in section 1, current social networks provide minimum privacy settings such as granting privileges to all people belonging to one's social graph to access his/her information. Suppose a social network which provides users to specify which information can be accessed by specific users not necessarily in one's social graph, for instance having similar interests. Although applications are being developed to export user information from closed social networks into RDF, the privacy settings are platform dependent such that the privacy settings cannot be reused on other platforms. Moreover, privacy preferences cannot make use of other platform's information, for instance, defining a privacy preference that restricts access to users from one platform and grants users from another platform. Therefore, a system is required that provides users to create fine-grained privacy preferences described using PPO which can be used by different platforms. This system will provide users to be fully in control who can access their personal information and who can access their published RDF data. Additionally, the user can set privacy preferences to control which data can be used by recommender systems or other applications.

3 A Formal Model for the Privacy Preference Ontology (PPO)

As portrayed in figure 1, a PPO-based privacy preference consists of: (1) `Restrictions`; (2) `Conditions`; (3) `Access Control Privileges` and; (4) `Access Spaces`. This section presents the associated formal model for PPO.

3.1 Defining the Classes and Properties of PPO

Definition 1: Restrictions. A `restriction` applies to a *Resource*, a *Statement* or a *Named Graph* (Fig. 1), where:

- A *Resource* (instance of `rdfs:Resource`) is identified by its own URI;
- A *Statement* consists of a $< subject, predicate, object >$ triple, each being instances of `rdfs:Resource`[6];
- A *Named Graph* consists of (1) a name denoted by a URI, and (2) a set of statements (an RDF graph) mapped to this name [4].

[5] WAC — `http://www.w3.org/ns/auth/acl`
[6] Including literals

Let St be a statement, U a URI, S be a subject, P a predicate, O an object, NG a named graph and A an access control privilege. Let $Subject(U, St)$ mean that U is subject of St, $Predicate(U, St)$ mean that U is a predicate of St, $Object(U, St)$ mean that U is an object of St, $RDFGraph(St, NG)$ mean that St is contained within the RDF graph of NG and $AssignAccess(U, A)$ mean that A is assigned to U.

Restricting access to a resource is defined as follows.

$$\forall St(AssignAccess(U,A) \wedge (Subject(U,St) \vee Predicate(U,St) \vee Object(U,St)) \Rightarrow AssignAccess(St,A)) \quad (1)$$

In other words, restricting access to a resource restricts access to all statements involving that resource as subject, predicate or object.

Restricting access to a statement is defined as follows.

$$\forall St((AssignAccess(S,A) \wedge AssignAccess(P,A) \wedge AssignAccess(O,A)) \wedge (Subject(S,St) \wedge Predicate(P,St) \wedge Object(O,St)) \Rightarrow AssignAccess(St,A)) \quad (2)$$

Restricting access to a named graph is defined as follows.

$$\forall St(AssignAccess(NG,A) \wedge RDFGraph(St,NG) \Rightarrow AssignAccess(St,A)) \quad (3)$$

In other words, restricting access to a Named Graph restricts access to all statements within that Graph.

Definition 2: Conditions. A `condition` defines whether what is being restricted has:

- a resource's URI identified as a statement's subject or object;
- an instance of a class which is defined as a statement's subject or object;
- a statement contains a particular literal as a value and;
- a statement that contains a particular property.

Let St be a statement, U a URI, C a class and A an access control privilege. Let $Subject(U, St)$ mean that U is subject of St, $Object(U, St)$ mean that U is the object of St, $RDFType(U, C)$ mean that U rdf:type C and AssignAccess(U,A) mean that A is assigned to U.

The condition resource as subject is defined as follows.

$$\forall St(AssignAccess(U,A) \wedge Subject(U,St) \Rightarrow AssignAccess(St,A)) \quad (4)$$

The condition resource as object is defined as follows.

$$\forall St(AssignAccess(U,A) \wedge Object(U,St) \Rightarrow AssignAccess(St,A)) \quad (5)$$

The condition class as subject is defined as follows.

$$\forall St(AssignAccess(C,A) \wedge RDFType(U,C) \wedge Subject(U,St) \Rightarrow AssignAccess(St,A)) \quad (6)$$

The condition class as object is defined as follows.

$$\forall St(AssignAccess(C,A) \wedge RDFType(U,C) \wedge Object(U,St) \Rightarrow AssignAccess(St,A)) \quad (7)$$

Definition 3: Access Control Privilege. An `access control privilege` defines the `read` and/or `write` privilege (defined by the WAC), and it is defined as:

$$AccessControl = \{read,write\}. \quad (8)$$

Definition 4: Access Space. An `Access Space` contains an access query that is executed to check whether a requester satisfies specific attributes. An access space can have multiple queries and therefore, it can be defined as the set:

$$AccessSpace = \{accessquery_1,...,accessquery_n\}. \quad (9)$$

3.2 Defining a Privacy Preference

Definition 5: A Privacy Preference. A privacy preference is the set of all the sets `Restrictions`, `Conditions`, `Access Control Privilege` and `Access Space` and it is defined as:

$$PrivacyPreference \subseteq Restrictions \cup Conditions \cup AccessControl \cup AccessSpace. \quad (10)$$

3.3 Applying Privacy Preferences

A privacy preference applies when requested information matches with the restricted statement(s), resource(s) and/or named graph(s). This is defined as follows. Let St be a requested statement, R a requested resource, NG a requested named graph and P a privacy preference. Let $ApplyPrivacyPreference(P)$ mean that P is applied, $Statement(St, P)$ mean that St is a restricted statement in P, $Resource(R, P)$ mean that R is a restricted resource in P and $NamedGraph(NG, P)$ mean that NG is a restricted named graph in P. Then:

$$\forall P((Statement(St,P) \vee Resource(R,P) \vee NamedGraph(NG,P)) \Rightarrow ApplyPrivacyPreference(P)) \quad (11)$$

The relationship between restrictions and conditions consists of a mapping from restricted statements RS to condition statements CS, which this mapping is defined as $M : RestrictedStatements(RS) \mapsto ConditionStatements(CS)$. `IF` M = `false THEN` ¬ApplyPrivacyPreference(P).

However, there are situations where restrictions are not defined but only conditions are defined within a privacy preference. In this case, the mapping

is performed between the RequestedInformation(RI) and the ConditionStatements(CS). This mapping is defined as *M : RequestedInformation(RI)* $\mapsto$ *ConditionStatements(CS)*. `IF` M = `true THEN` ApplyPrivacyPrefence(P). Therefore, applying a privacy preference based on the mapping between restricted or requested statements and condition statements is defined as: $\forall$*PM(P)* $\rightarrow$ *ApplyPrivacyPreference(P)*.

The access space query Q is executed on the requester's authenticated information. `IF` AccessSpace(Q) = `true THEN` AccessControl(A) defined in the privacy preference is granted to the requester. `IF` AccessSpace(Q) = `false THEN` the requester is $\neg$AccessControl(A).

4 PPO in-use: Implementing *MyPrivacyManager*

This section presents *MyPrivacyManager*[7], a privacy preference manager for the Social Semantic Web. It was developed to validate PPO and the formal model, i.e. to implement the creation of privacy preferences for RDF data described using PPO, and make sure the preferences are applied when requesting information, to filter requested data. Although *MyPrivacyManager* is designed to work with any Social Semantic Data[8], we will focus on defining privacy preferences for FOAF profiles. With FOAF profiles, our aim is to illustrate how the formal model can be applied to create privacy preferences and how personal information can be filtered based on such preferences.

Figure 2 illustrates the *MyPrivacyManager* architecture, which contains: (1) WebID Authenticator: handles user sign-on using the FOAF+SSL protocol; (2) RDF Data Retriever and Parser: retrieves and parses RDF data such as FOAF profiles from WebID URIs; (3) Privacy Preferences Creator: defines privacy preferences using PPO; (4) Privacy Preferences Enforcer: queries the RDF data store to retrieve and enforce privacy preferences; (5) User Interface: provides users the environment whereby they can create privacy preferences and to view other user's filtered FOAF profiles; and (6) RDF Data store: an ARC2[9] RDF data store to store the privacy preferences[10]. The implementation and functionality of these modules are explained in more detail in this section.

MyPrivacyManager employs the federated approach whereby everyone has his/her own instance of *MyPrivacyManager*. As opposed to the majority of Social Web applications which are centralised environments whereby the companies offering such services have the sole authority to control all user's data, this federated approach ensures that everyone is in control of their privacy preferences [1]. Moreover, users can deploy their instances of *MyPrivacyManager* on whichever server they prefer.

[7] Screencast online – http://vmuss13.deri.ie/myprivacymanager/screencast/screencast.html
[8] Consists of Social Web data formatted in RDF or any other structured format
[9] ARC2 — `http://arc.semsol.org`
[10] Although ARC2 was used for the implementation of MyPrivacyManager, any RDF store can be used.

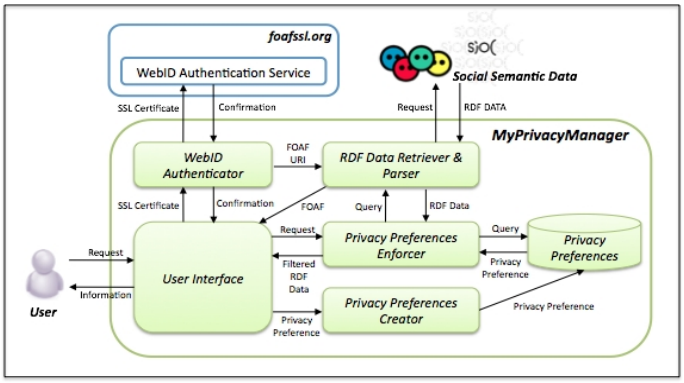

Fig. 2. MyPrivacyManager Architecture

4.1 Authentication with the WebID protocol

The WebID protocol [12] provides a mechanism whereby users can authenticate using FOAF and SSL certificates.

The WebID protocol implemented in *MyPrivacyManager* uses the libraries provided by foaf.me[11] which calls the WebID authentication mechanism offered by the FOAF+SSL Identity Provider Service[12]. This provides a secure delegated authentication service that returns back the WebID URI of the user which links to the FOAF document of the user signing in. If the identity service does not return back the WebID, then it means that the authentication has failed.

Once the user is authenticated, *MyPrivacyManager* matches the WebID URI with the WebID URI of the owner of that instance. If the owner is signed in, then the interface provides options where the user can create privacy preferences. On the other hand, if the user signed in is a requester, then the FOAF profile of the owner of that particular instance is requested. The *Privacy Preferences Enforcer* module is called (described later in this section) to filter the FOAF profile according to the privacy preferences specified by the owner of that instance.

4.2 Creating Privacy Preferences

MyPrivacyManager provides users an interface to create privacy preferences for their Social Semantic Data. The interface displays (1) the profile attributes extracted from the user's FOAF profile which the user can specify what to share in the first column and (2) other attributes (extracted from the user profile) in the second column for the user to specify who can access the specific shared information; – as illustrated in the screenshot in figure 3.

The system provides profile attributes (extracted from the user's profile) which the user can share classified as follows: (1) Basic Information consisting

[11] foaf.me — `http://foaf.me/`
[12] foafssl.org — `http://foafssl.org/`

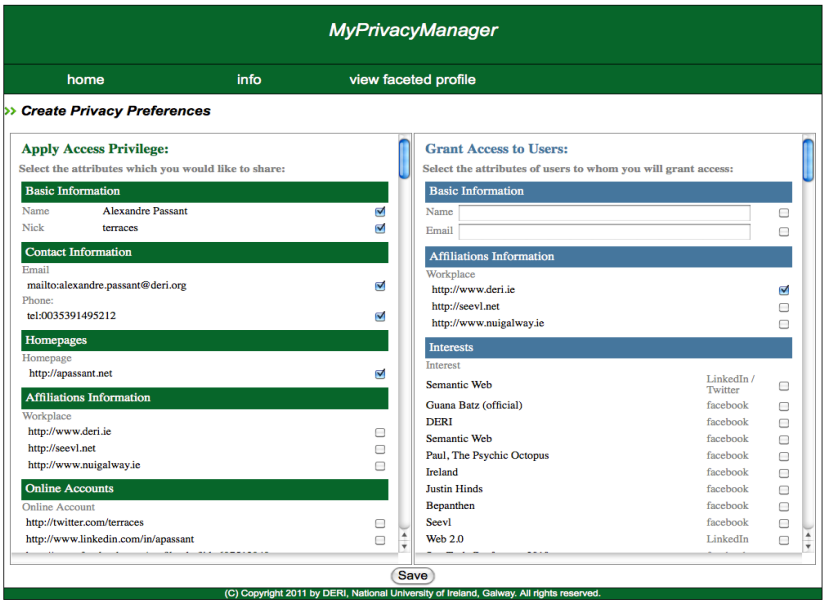

Fig. 3. The interface for creating privacy preferences in MyPrivacyManager

```
PREFIX ppo: <http://vocab.deri.ie/ppo#> .
PREFIX ex: <http://vmuss13.deri.ie/> .
ex:preference1 a ppo:PrivacyPreference;
 foaf:maker <http://foaf.me/ppm_usera#me>;
 dc:title "Restricting access to my personal information";
 dc:created "2011-06-01T13:32:59+02:00";
 ppo:appliesToStatement :Statement1;
 :Statement1
  rdf:subject <http://vmuss13.deri.ie/foafprofiles/terraces#
      me> ;
  rdf:predicate <http://xmlns.com/foaf/0.1/name>;
  rdf:object "Alexandre Passant";
 ppo:appliesToStatement :Statement2;
 :Statement2
  rdf:subject <http://vmuss13.deri.ie/foafprofiles/terraces#
      me> ;
  rdf:predicate <http://xmlns.com/foaf/0.1/nick>;
  rdf:object "terraces" ;
ppo:assignAccess acl:Read;
ppo:hasAccessSpace [
  ppo:hasAccessQuery
    "ASK { ?x foaf:workplaceHomepage <http://www.deri.ie> }"
       ].
```

Fig. 4. Privacy Preference described using PPO created in *MyPrivacyManager*

of the name, age, birthday and gender; (2) Contact Information consisting of email and phone number; (3) Homepages; (4) Affiliations consisting of the website of the user's work place; (5) Online Accounts such as Twitter, LinkedIn and Facebook user pages; (6) Education that contains the user's educational achievements and from which institute such achievements where obtained; (7) Experiences consisting of job experiences which include job title and organisation; and (8) Interests which contain a list of user interests ranked according to the calculated weight of each interest.

The attributes, extracted from the FOAF profile, which the user can select which to whom to share information must have are categorised as follow: (1) Basic Information containing fields to insert the name and email address of specific users; (2) Affiliations to share information with work colleagues; and (3) Interests to share information with users having the same interests.

Once the user selects which information to share and to whom, he/she clicks on the save button for the system to generate automatically the privacy preference using PPO. Figure 4 illustrates an example of a privacy preference described using PPO and created from *MyPrivacyManager* that restricts access to a person's name and nick name to those users who are work colleagues. Although reification is used, we intend to use named graphs in order to reduce the number of statements.

4.3 Requesting and Enforcing Privacy Preferences

MyPrivacyManager provides users to view other people's FOAF profile based on privacy preferences by logging into third party's instance. On the contrary of common Social Networks which are public by default, *MyPrivacyManager* enforces a private by default policy. This means that if no privacy preferences are set for a profile or for specific information, then this is not granted access to be viewed. In the near future, *MyPrivacyManager* will be modified to provide a feature where users can select which default setting they wish to enforce – public or private.

The sequence in which privacy preferences are requested and enforced is performed as illustrated in figure 5 which consists of: (1) a requester authenticates to another user's MyPrivacyManager instance using the WebID protocol and the system automatically requests the other user's FOAF profile; (2) the privacy preferences of the requested user's FOAF profile are queried to identify which preference applies; (3) the access space preferences are matched according to the requester's profile to test what the requester can access; (4) the requested information (in this case, FOAF data) is retrieved based on what can be accessed; and (5) the requester is provided with the data he/she can access.

MyPrivacyManager handles each privacy preference separately since each preference may contain different access spaces. Once the system retrieves the privacy preferences, for each preference it tests the access space queries with the requester's FOAF profile. If the access space query on the requester's FOAF profile returns true, then the privacy preference is considered, however, if it returns false, then that particular privacy preference is ignored. Since the access

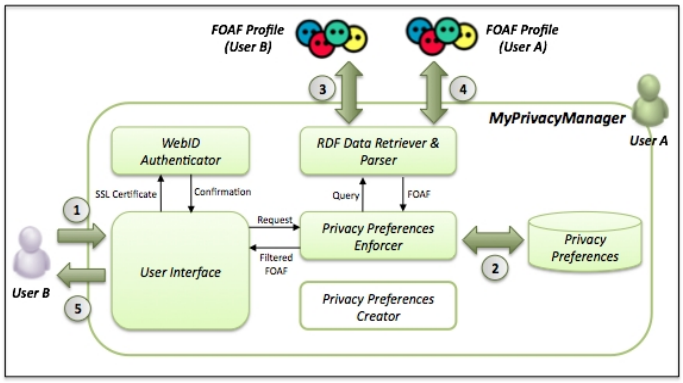

Fig. 5. The sequence of requesting third party FOAF profiles

space can contain more than one access query, in the case when one access query returns true and the other false, then by default the system enforces that the access space is true. The system then processes the restrictions and conditions defined in the privacy preference.

The system will formulate the restrictions and conditions as a group graph pattern. This group graph pattern from each privacy preference will be used to create a SPARQL query and the result from this query will be the filtered FOAF profile that can be accessed by the requester. The group graph pattern constructed from each privacy preference are combined using the keyword `UNION` within the same SPARQL query. Once the SPARQL queries are formalised, the access control privilege is assigned to the user. However, currently the system only accepts the `acl:Read` property since its purpose is to view the filtered FOAF documents of other users.

5 Related Work

The Web Access Control (WAC) vocabulary[13] describes access control privileges for RDF data. This vocabulary defines the `Read` and `Write` access control privileges (for reading or updating data) as well as the `Control` privilege to grant access to modify the access control lists (ACL). This vocabulary is designed to specify access control to the full RDF document rather than specifying access control properties to specific data contained within the RDF document. As pointed out in [9], the authors observe that protecting data does not merely mean granting access or not to the full RDF data but in most cases, users require more fine-grained privacy preferences that define access privileges to specific data. Therefore, fine-grained privacy preferences applied to RDF data using our solution create a mechanism to filter and provide customised RDF data views that only show the specific data which is granted access.

[13] WAC — `http://www.w3.org/ns/auth/acl`

The authors in [8] propose a privacy preference formal model consisting of relationships between objects and subjects. Objects consist of resources and actions, whereas subjects are those roles that are allowed to perform the action on the resource. Since the privacy settings based on this formal model combine objects and actions together, this requires the user to define the same action each time with different objects rather than having actions separate from objects. Thus, this method results in defining redundant privacy preferences. Moreover, the proposed formal model relies on specifying precisely who can access the resource. Our approach provides a more flexible solution which requires the user to specify attributes which the requester must satisfy.

The authors in [3] propose an access control framework for Social Networks by specifying privacy rules using the Semantic Web Rule Language (SWRL) [14]. This approach is also based on specifying who can access which resource. Moreover, this approach relies that the system contains a SWRL reasoner. In [5] the authors propose a relational based access control model called `RelBac` which provides a formal model based on relationships amongst communities and resources. This approach also requires to specifically define who can access the resource(s).

In [11] the authors propose a method to direct messages, such as microblog posts in SMOB, to specific users according to their online status. The authors also propose the idea of a `SharingSpace` which represents the persons or group of persons who can access the messages. The authors also describe that a `SharingSpace` can be a dynamic group constructed using a SPARQL `CONSTRUCT` query. However, the proposed ontology only allows relating the messages to a pre-constructed group.

In [7] the authors propose a system whereby users can set access control to RDF documents. The access controls are described using the Web Access Control vocabulary by specifying who can access which RDF document. Authentication to this system is achieved using the WebID protocol [12] which provides a secure connection to a user's personal information stored in a FOAF profile [6]. This protocol uses FOAF+SSL techniques whereby a user provides a certificate which contains a URL that denotes the user's FOAF profile. The public key from the FOAF profile and the public key contained in the certificate which the user provides are matched to allow or disallow access. Our approach extends the Web Access Control vocabulary to provide more fine-grained access control to the data rather than to the whole RDF document.

6 Conclusion and Future Work

In this paper we presented a formalisation of the PPO that can be used as a model whilst creating privacy preferences for any structured data. Since structured data can be used easily by other platforms taking advantage of Semantic Web technologies, privacy preferences described using the PPO can be utilised by

[14] SWRL — `http://www.w3.org/Submission/SWRL/`

any system that implements the formal model. Moreover we presented *MyPrivacyManager* which implemented the formal model of PPO in order to demonstrate how to create privacy preferences for Social Semantic Data, primarily focusing on user profiles described using FOAF. *MyPrivacyManager* also demonstrates how data is filtered on the basis of these privacy preferences.

Similar to all prototype systems, further enhancements is required to enrich *MyPrivacyManager*. It will be extended to demonstrate how data from current Social Networks such as Facebook can be filtered based on privacy preferences defined in PPO. Furthermore, since *MyPrivacyManager* assumes that the requester's information is trustworthy, the system will be extended to incorporate methodologies on how to assert the trustworthiness of requesters.

References

1. C. Au Yeung, I. Liccardi, K. Lu, O. Seneviratne, and T. Berners-Lee. Decentralization: The Future of Online Social Networking. In *Proceedings of the W3C Workshop on the Future of Social Networking Position Papers,'08*, 2008.
2. D. Boyn and E. Hargittai. Facebook privacy settings. Who cares? *First Monday*, 15(8), August 2010.
3. B. Carminati, E. Ferrari, R. Heatherly, M. Kantarcioglu, and B. Thuraisingham. A Semantic Web Based Framework for Social Network Access Control. In *Proceedings of the 14th ACM Symposium on Access Control Models and Technologies, SACMAT '09*, 2009.
4. Carroll, Jeremy J. and Bizer, Christian and Hayes, Pat and Stickler, Patrick. Named graphs, provenance and trust. In *Proceedings of the 14th international conference on World Wide Web, WWW'05*, 2005.
5. F. Giunchiglia, R. Zhang, and B. Crispo. Ontology Driven Community Access Control. *Trust and Privacy on the Social and Semantic Web, SPOT'09*, 2009.
6. B. Heitmann, J. Kim, A. Passant, C. Hayes, and H. Kim. An Architecture for Privacy-Enabled User Profile Portability on the Web of Data. In *Proceedings of the 1st International Workshop on Information Heterogeneity and Fusion in Recommender Systems, HetRec '10*, 2010.
7. J. Hollenbach and J. Presbrey. Using RDF Metadata to Enable Access Control on the Social Semantic Web. In *Proceedings of the Workshop on Collaborative Construction, Management and Linking of Structured Knowledge, CK'09*, 2009.
8. P. Kärger and W. Siberski. Guarding a Walled Garden Semantic Privacy Preferences for the Social Web. *The Semantic Web: Research and Applications*, 2010.
9. A. Passant, P. Kärger, M. Hausenblas, D. Olmedilla, A. Polleres, and S. Decker. Enabling Trust and Privacy on the Social Web. In *W3C Workshop on the Future of Social Networking)*, 2009.
10. O. Sacco and A. Passant. A Privacy Preference Ontology (PPO) for Linked Data. In *Proceedings of the Linked Data on the Web Workshop, LDOW2011*, 2011.
11. M. Stankovic, A. Passant, and P. Laublet. Directing status messages to their audience in online communities. In *Proceedings of the 5th International Conference on Coordination, Organizations, Institutions, and Norms in Agent Systems*, 2010.
12. H. Story, B. Harbulot, I. Jacobi, and M. Jones. FOAF + SSL : RESTful Authentication for the Social Web. *Semantic Web Conference*, 2009.

Performance Measures for Multi-Graded Relevance

Christian Scheel, Andreas Lommatzsch, and Sahin Albayrak

Technische Universität Berlin, DAI-Labor, Germany
{christian.scheel,andreas.lommatzsch,sahin.albayrak}@dai-labor.de

Abstract. We extend performance measures commonly used in semantic web applications to be capable of handling multi-graded relevance data. Most of today's recommender social web applications offer the possibility to rate objects with different levels of relevance. Nevertheless most performance measures in Information Retrieval and recommender systems are based on the assumption that retrieved objects (e. g. entities or documents) are either relevant or irrelevant. Hence, thresholds have to be applied to convert multi-graded relevance labels to binary relevance labels. With regard to the necessity of evaluating information retrieval strategies on multi-graded data, we propose an extended version of the performance measure average precision that pays attention to levels of relevance without applying thresholds, but keeping and respecting the detailed relevance information. Furthermore we propose an improvement to the NDCG measure avoiding problems caused by different scales in different datasets.

1 Introduction

Semantic information retrieval systems as well as recommender systems provide documents or entities computed to be relevant according a user profile or an explicit user query. Potentially relevant entities (e. g. users, items, or documents) are generally ranked by the assumed relevance, simplifying user's navigation through presented results. Performance measures evaluate computed rankings based on user-given feedback and thus allow comparing different filtering or recommendation strategies [9].

The most frequently used performance measures in semantic web applications are the *Precision* ($P = \frac{\text{number of relevant items in the result set}}{\text{total number of items in the result set}}$) and the *Mean Average Precision* (MAP) designed to compute the Average Precision over sorted result lists ("rankings"). The main advantage of these measures is that they are simple and very commonly used. The main disadvantage of these measures is, that they only take into account binary relevance ratings and are not able to cope with multi-graded relevance assignments.

One well accepted performance measure designed for handling multi-graded relevance assignments is the *Normalized Discounted Cumulative Gain* (NDCG) [3,8]. From one position in the result list to another the NDCG focuses on the gain of information. Because the information gain of items in the result list

on the same level of relevance is constant, it is possible to swap the positions of items belonging to the same relevance level without changing the performance measure. The advantage of *NDCG* is that it applies an information-theoretic model for considering multiple relevance levels. Unfortunately, the NDCG measure values depend on the number of reference relevance values of the dataset. Thus, NDCG values computed for different datasets cannot be directly be compared with each other.

An alternative point of view to multi-graded relevance was used in the TREC-8 competition [2]. Instead of multiple relevance levels, probabilities for measuring the relevance of entities were used. As performance measure the *Mean Scaled Utility* (SCU) was suggested. Since the SCU is very sensitive to the applied scaling model and the properties of the queries, the SCU measure should not be used for comparing different datasets [2].

Due to the fact, that binary relevance performance measures *precision* and *mean average precision* are commonly used, a promising approach is to extend these binary measures to be capable of handling multi-graded relevance assignments. Kekäläinen et al. [5] discuss the possibility to evaluate retrieval strategies *"on each level of relevance separately"* and then *"find out whether one IR method is better than another at a particular level of relevance"*. Additionally it is proposed to weight different levels of relevance according to their gain of importance. Kekäläinen et al suggest a generalized precision and recall, which contributes to the level of relevance importance, but does not consider the position of an item in the retrieved result list.

In our work we extend the measures *Precision* and *MAP* to be capable of handling multiple relevance levels. The idea of looking at the performance of each level of relevance separately is carried on. An extension of MAP is proposed where strategies can be evaluated with user given feedback independent from the number of used relevance levels. We refer to this extension of MAP as μMAP. Additionally, we introduce an adaptation of the NDCG measure taking into account the number of relevance levels present in the respective reference datasets.

The paper is structured as follows: In the next section we explain the dataset used for benchmarking our work. We explain the performance measure *Average Precision* and show how data has to be transformed in order to compute the *Average Precision*. In Section 3 we propose an extension to *Average Precision* allowing us to handle multi-graded relevance assignment without changing the original ratings. After introducing our approach we evaluate the proposed measures for several Retrieval algorithms and different datasets (Section 4). Finally we discuss the advantages and disadvantages of the new measures and give an outlook to future work.

2 Settings and Methods

For evaluating the performance of a computed item list, a reference ranking is needed (or the items must be rated allowing us to derive a reference ranking). The

reference ranking is expert defined or provided by the user. It can be retrieved explicitly or implicitly [4]. Very popular is the 5-star rating allowing the user to rate entities or items on a scale from 0 to 4, meaning five levels of relevance.

For analyzing and comparing the properties the proposed evaluation measures, we deploy an artificial data set and a real-world dataset, providing three relevance levels. We assume that the reference item ratings stand for ratings coming from human experts and that the test rankings stand for the item list coming from different prediction strategies. We discuss several different types of reference ratings: In each evaluation setting the optimal item list based on the reference ratings should achieve the performance value 1.0.

2.1 Artificial Dataset

We create an artificial dataset and analyze how changes in the dataset influence the measured result quality. For this purpose, we compute the performance of 100 different test item lists for each given reference ranking considering different performance measures.

Test Ratings We create items list ("test rankings") by pair-wise swapping the item of an *optimal* item list ("reference ranking"), see Fig. 1. Swapping means that two rated items in the ranking change their positions. The best test ranking is the one for that no items have been swapped. The performance of the obtained item list decreases with increasing number of swapped item pairs.

The analyzed 100 test rankings differ in the number of the swapped pairs: In the first test ranking (0) we do not swap any item pair, in the last test ranking (99) we randomly swap 99 item pairs. How much the performance decreases per swap depends on the relevance levels' distance of the swapped items. Hence, an evaluation run for each number of switches includes 100 test ranking evaluations to average the results.

Uniformly Distributed Reference Ratings There are four different kinds of reference rankings which differ in the number of relevance levels. Each reference

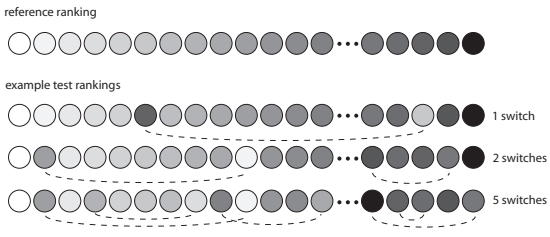

Fig. 1. The figure visualizes the creation of test rankings. Starting with the reference ranking (used for the evaluation) randomly selected item pairs are swapping. The created test rankings differ in the number of swapped pairs.

ranking contains 100 elements which are uniformly distributed among 2, 10, 20, or 50 levels of relevance (see Fig. 2).

Non-Uniformly Distributed Reference Ratings In contrast to the reference rankings used in the previous paragraph, we consider reference rankings consisting of non-uniformly rated items making use of 2, 10, 20, or 50 levels of relevance (see Fig. 3). In other words, the probabilities (that a relevance level is used in the reference ranking) differ randomly between the relevance levels. Moreover, some relevance levels may not be used. Hence, this dataset is more realistic, because users do not assign relevance scores uniformly.

2.2 OHSUMED Dataset

The OHSUMED dataset provided by the Hersh team at Oregon Health Sciences University [1] consists of medical journal articles from the period of 1987–1991 rated by human experts, on a scale of three levels of relevance. Our evaluation is based on the OHSUMED dataset provided in LETOR [6]. The items ("documents") are rated on a scale of 0, 1, and 2, meaning *not relevant*, *possibly relevant* and *definitely relevant*. As in the *Non-Uniformly Distributed Reference Ratings* the given relevance scores are not uniformly distributed.

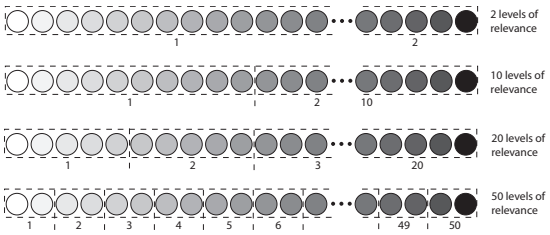

Fig. 2. The Figure visualizes datasets having an almost similar number of items assigned to every relevance level ("uniform distribution of used relevance levels"). The number of relevance levels varies in the shown datasets (2, 10, 20, 50).

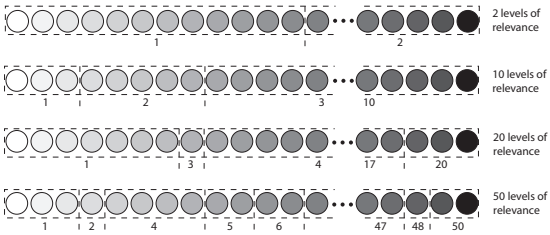

Fig. 3. The Figure shows datasets featuring a high variance in the number of items assigned to a relevance level ("non-uniform distribution of used relevance levels"). There are relevance levels having no items assigned.

Test Ratings The OHSUMED dataset in LETOR provides 106 queries and 25 strategies assigning relevance scores to each item in result set for a respective query. Due to the fact that some the provided strategies show a very similar behavior, we limit the number of evaluated strategies to eight (OHSUMED id 1, 5, 9, 11, 15, 19, 21, 22) enabling a better visualization of the evaluation results.

User-Given Ratings The OHSUMED dataset provides expert-labeled data based on a three level scale. Because there is no real distance between *not relevant*, *possibly relevant* and *definitely relevant*, we assume 1 as distance of successive levels of relevance as the assigned scores 0, 1, and 2 in the dataset imply.

Approximated Virtual-User Ratings The OHSUMED dataset provides three relevance levels. Because fine-grained ratings enable a more precise evaluation, authors believe that soon there will be datasets available with higher number of relevance levels. Until these datasets are available a trick is applied, replacing user's ratings with relevance scores calculated by computer-controlled strategies. The computer calculated relevance scores are treated as "user-given" reference ratings. In our evaluation we selected the OHSUMED strategies *TF of the title* (resulting in 9 different relevance levels) and *TF-IDF of the title* (resulting in 158 different relevance levels) as "virtual" reference users. Both rating strategies show a very strong correlation; the Pearson's correlation coefficient of the relevance assessments is 0.96. The availability of more than three relevance levels in the reference ranking allows us to evaluate ranking strategies with multi-graded relevance assignments. The two strategies treated as reference rating strategies are also considered in the evaluation. Thus, these strategies should reach an evaluation value of 1.

2.3 Performance Measures

There are several performance measures commonly used in information retrieval and recommender systems, such as *precision*, *Area Under an ROC curve* or *rank of the first relevant document (mean reciprocal rank)*. Additionally, the mean of each performance measure over all queries can be computed to overcome the unstable character of some performance measures.

In this section we focus on the popular performance measures *Average Precision* (AP) [10] and *Normalized Discounted Cumulative Gain* (NDCG) [3]). Unlike AP, NDCG can handle different numbers of relevance levels, due to the fact that NDCG defines the information gain based on the relevance score assigned to a document.

Average Precision The average precision of an sorted item ("document") list for a query q is defined as

$$AP_q = \frac{\sum_{p=1}^{N} P@p \cdot rel_q(p)}{R_q} \tag{1}$$

where N denotes the number of the items in the evaluated list, $P@p$ the precision at position p, and R_q the number of relevant items with respect to q. $rel_q(p)$ is a binary function describing if the element at position p is relevant (1) or not (0). A higher AP value means that more relevant items are in the heading of the result list. Given a set of queries, the mean over the AP of all queries is referred to as MAP.

When there are more than two relevance levels, these levels have to be assigned to either 0 or 1. A threshold must be applied, separating the relevant items from the irrelevant items. For later use, we denote AP_q^t as the AP with threshold t applied. AP_q^t is calculated by

$$AP_q^t = \frac{\sum_{p=1}^{N} P@p \cdot rel_q^t(p)}{R_q^t} \quad \text{with} \quad rel_q^t(p) = \begin{cases} 1, & rel_q(p) \geq t \\ 0, & rel_q(p) < t \end{cases} \tag{2}$$

where R_q^t defines the number of results so that $rel_q^t(p)$ is 1.

Normalized Discounted Cumulative Gain For a query q, the normalized discounted cumulative gain at position n is computed

$$\text{NDCG@}n(q) = \text{N}_n^q\text{DCG} = N_n^q \sum_{i=1}^{n} \frac{2^{\text{gain}_q(i)} - 1}{\log(i+1)} \tag{3}$$

where $\text{gain}_q(i)$ denotes the gain of the document at position i of the (sorted) result list. N_n is a normalization constant, scaling the optimal DCG@n to 1. The optimal DCG@n can be retrieved by calculating the DCG@n with the correctly sorted item list.

3 Extending Performance Measures

The need to apply thresholds makes the measures AP and MAP not applicable for multi-graded relevance data. NDCG supports multi-graded relevance data, but the sensitivity to the choice of relevance levels prevents the comparison of NDCG values computed based on different datasets. Hence, for a detailed evaluation based on datasets having multiple relevance levels, both MAP and NDCG have to be adapted.

3.1 Extending Average Precision

In the most commonly used evaluation scenarios, the relevance of items is a binary function (returning "relevant" or "irrelevant"). If the reference dataset provides more than two relevance levels, a threshold is applied which separates the documents into a set of relevant items and a set of irrelevant items. The example in Table 1 illustrates how levels of relevance affect the calculation of the measure AP The example shows a sorted list of items (A ... H). The relevance of

Table 1. The Table shows an example of calculating the average precision for a given item list (each item is rated base on scale of 5 relevance levels). Dependent on the applied threshold t, items are handled as relevant (+) or irrelevant (–). Thus the computed AP depends on the threshold t.

i	A	B	C	D	E	F	G	H	
$rel(i)$	1	0	3	3	2	0	1	4	AP
$t = 5$	–	–	–	–	–	–	–	–	0.000
$t = 4$	–	–	–	–	–	–	–	+	0.125
$t = 3$	–	–	+	+	–	–	–	+	0.403
$t = 2$	–	–	+	+	+	–	–	+	0.483
$t = 1$	+	–	+	+	+	–	+	+	0.780
$t = 0$	+	+	+	+	+	+	+	+	1.000
								mean	0.465
								mean of 5>t>0	0.448

each item is denoted on a scale from 0 to 4 (5 relevance levels). For calculating the precision, a threshold t must be applied, to separate "relevant" from "irrelevant" items. The threshold $t = 0$ implies that all documents are relevant. We refer to this threshold as the irrelevant threshold. In contrast to $t = 0$, applying the threshold $t = 5$ leads to no relevant documents. Table 1 illustrates that the threshold t strongly affects the computed AP. To cope with this problem, we propose calculating the performance on each relevance level, and then computing the mean. This ensures that higher relevance levels are considered more frequently than lower relevance levels. The example visualized in Table 1 shows that item H having a relevance score of 4 is considered relevant more often than all other items.

We refer to this approach as μAP, and μMAP if the mean of μAP for several result lists is calculated. For handling the case that the not all relevance levels are used in every result list and that the "distance" between successive relevance levels is not constant, μAP has to be normalized.

$$\mu AP_q = \frac{1}{\sum_{t \in L} d^t} \sum_{t \in L} \left(AP_q^t \cdot d^t \right) \tag{4}$$

where AP_q^t denotes the average precision using the threshold t, and L a set of all relevance levels (meaning all thresholds) used in the reference ranking. d^t denotes the distance between the relevance level t_i and t_{i-1} if $i > 1$ (and t if $i = 0$). The following example demonstrates the approach: Given a set dataset based on three relevance levels (0.0, 0.3, 1.0), the threshold $t = 0.3$ leads to the $AP^t = 0.3 - 0.0 = 0.3$. The threshold $t = 1.0$ leads to $AP^t = 1.0 - 0.3 = 0.7$.

3.2 The Normalized Discounted Cumulative Normalized Gain

In contrast to MAP, NDCG is designed for handling multiple relevance levels. Unfortunately NDCG does not consider the scale used for the relevance scores.

Table 2. The Table shows how the mapping of relevance scores to relevance levels influences the NDCG measure. In the first example the gain represents an equal match from ratings to relevance levels, in the second example the relevance level is twice the value of the rating, and in the third example the gain of both previous examples is normalized.

	i	A	B	C	D	E	F	G	H	
	rel(i)	1	0	3	3	2	0	1	4	mean
example one:	gain	1	0	3	3	2	0	1	4	
gain equals rel(i)	gain_{opt}	4	3	3	2	1	1	0	0	
	dcg	3.32	3.32	14.95	24.96	28.82	28.82	29.93	45.65	
	dcg_{opt}	49.83	64.50	76.13	80.42	81.70	82.89	82.89	82.89	
	ndcg	0.07	0.05	0.20	0.31	0.35	0.35	0.36	0.55	0.28
example two:	gain	2	0	6	6	4	0	2	8	
gain equals rel(i) $\cdot$ 2	gain_{opt}	8	6	6	4	2	2	0	0	
	dcg	9.97	9.97	114	204	224	224	227	494	
	dcg_{opt}	847	979	1083	1105	1109	1112	1112	1112	
	ndcg	0.01	0.01	0.11	0.19	0.20	0.20	0.20	0.44	0.17
example three:	gain	0.25	0	0.75	0.75	0.5	0	0.25	1	
gain is normalized	gain_{opt}	1	0.75	0.75	0.5	0.25	0.25	0	0	
with ngain (Equ. 5)	dcg	0.63	0.63	1.76	2.74	3.27	3.27	3.48	4.53	
	dcg_{opt}	3.32	4.75	5.88	6.48	6.72	6.94	6.94	6.94	
	ndcg	0.19	0.13	0.30	0.42	0.49	0.47	0.50	0.65	0.39

Thus, computed NDCG values highly depend on the number of relevance levels making it impossible to compare NDCG values between different datasets. Table 2 illustrates this problem.In the first example the NDCG is calculated as usual. In the second example, the number of relevance levels is doubled, but the number of assigned scores as well as the number of used levels of relevance is equal to the first example. This doubling leads to a smaller NDCG compared to the first example, even though no rated element became more or less relevant to another element. In the third example, the gain of example one is normalized and the NDCG is calculated. It can be seen that the normalization solves the inconsistency. A normalization of the gain overcomes the problem of incomparable performance values for data with relevance assignments within a different number of relevance levels. We define the *Normalized Discounted Cumulative Normalized Gain* (NDCNG) at position n as follows:

$$\text{NDCNG@}n(q) = N_n^q \sum_{i=1}^{n} \frac{2^{\text{ngain}_q(i)} - 1}{\log(i+1)}, \ \text{ngain}_q(i) = \begin{cases} \dfrac{\text{gain}_q(i)}{m_q} & , m_q > 0 \\ 0 & , m_q \leq 0 \end{cases} \quad (5)$$

where m_q is the highest reachable gain for the query q ("normalization term"). If there is no relevant item, m_q is set to 0 assuming that irrelevant items are rated with 0. All rating are ≥ 0; relevant items have relevance scores > 0. If these assumptions do not apply, the relevance scores must be shifted so that the irrelevant level is mapped to the relevance score 0.

4 Evaluation

Evaluation based on the Artificial Dataset We evaluated the proposed performance measures on the artificial dataset introduced in Section 2.1. Fig. 4 shows the mean of 100 evaluation runs with uniformly distributed relevance scores. From left to right the number of false pair-wise item preferences increases, and hence the measured performance decreases.

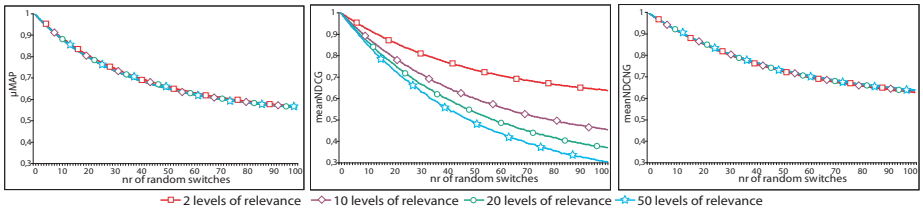

Fig. 4. The evaluation (average over 100 test runs) with the artificial dataset based on uniformly distributed reference ratings shows that in contrast to NDCG, the measures μMAP and NDCNG do not depend on the number of relevance levels.

Fig. 4 shows that in contrast to NDCG, the measures μMAP and NDCNG do not depend on the number of relevance levels. μMAP and the NDCNG calculate the same performance values for similar test rankings. The proposed performance measures explicitly consider the number of relevance levels. This is very important since the common way of applying a threshold to a binary-relevance-based performance measure often leads to a constant performance for item lists differing in the order of items assigned to different relevance levels.

The second evaluation focuses on the analysis how unused relevance levels influence the performance measures. This evaluation is based on the non-uniformly distributed artificial dataset introduced in Section 2.1. Fig. 5 shows that neither μMAP nor NDCNG are affected by the number of items per rank or by the number unused relevance levels.

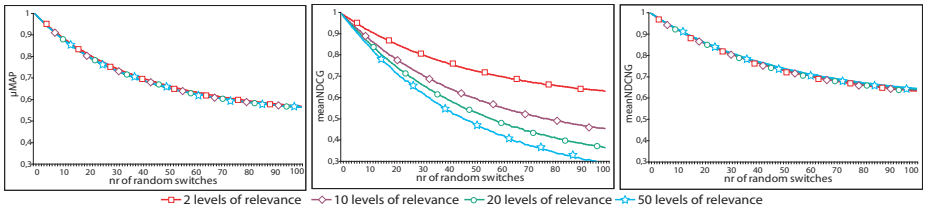

Fig. 5. The evaluation (average over 100 test runs) with the artificial dataset based on a non-uniformly distributed reference ratings shows that NDCG highly depends on the number of relevance levels whereas μMAP and NDCNG do not.

Evaluation based on the OHSUMED Dataset The OHSUMED dataset (introduced in Sec. 2.2) uses three different relevance levels. Fig. 6 visualizes the measured performance of selected retrieval strategies using μMAP, the mean NDCG and the mean NDCNG. Since the OHSUMED dataset uses two relevance levels, the μMAP is the mean of the MAP computed applying the thresholds $t = 1$ and $t = 2$.

A virtual user which is in fact strategy 1 (*TF of title*) provides the relevance assessments for the evaluation presented in Fig. 7. Strategy 1 assigns ordinal scores to 9 different levels of relevance. On this data, μMAP is the mean of 8 different MAP values.

The measured results show, that the measure μMAP evaluates the retrieval strategy 1 (as expected) with a performance value of 1.0, so does the mean NDCG and the mean NDCNG. All the considered evaluation measures agree that the retrieval strategy 9 is most similar to strategy 1, which makes sense, since strategy 9 is computed based on the *TF-IDF of title* and strategy 1 is computed based on *TF of title*. The main difference between both retrieval strategies is the number of used relevance levels: Strategy 1 assigns ordinal relevance scores (using 9 different relevance levels); strategy 9 assigns real values (resulting in 158 relevance levels). The distance between these relevance levels varies a lot.

When applying strategy 9 as reference rating strategy, the need for taking into account the distance between the relevance levels (Equ. 4) can be seen. Several very high relevance scores are used only once; lower relevance scores are used much more frequently. Fig. 8 shows the advantages of the NDCNG compared to "standard" NDCG. The comparison of the mean NDCG in Fig. 7 with the mean NDCG in Fig. 8 reveals that the NDCG is affected by the number of relevance levels. Since the strategies 1 and 9 show a very similar performances in both figures, the other strategies are evaluated with disproportionate lower performance values in Fig. 8 although both reference rating strategies assign similar relevance ratings. The μMAP and the proposed mean NDCNG values do not differ much in both evaluations due to the fact the these measures are almost independent from the number of relevance levels.

5 Conclusion and Discussion

In this paper we introduced the performance measure μAP that is capable to handle more than two levels of relevance. The main advantages of the approach is that it extends the commonly used performance measures *precision* and *Mean Average Precision*. μAP is fully compatible with the "traditional" measures, since it delivers the same performance values if only two reference relevance levels exist in the dataset. The properties of the proposed measures have been analyzed on different datasets. The experimental results show that the proposed measures satisfy the defined requirements and enable the comparison of semantic filtering strategies based on datasets with multi-graded relevance levels. Since μAP is based on well-accepted measures, only a minor adaptation of theses measures is required.

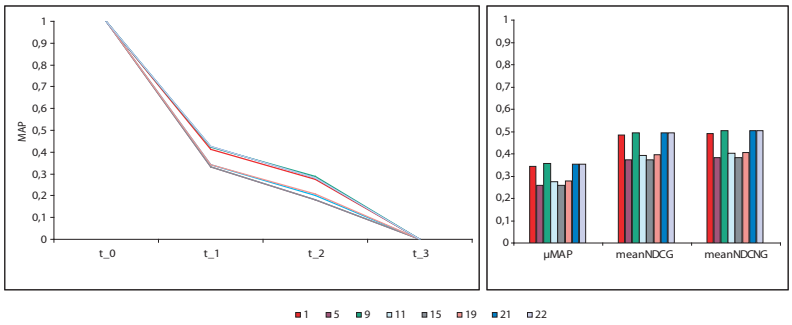

Fig. 6. Performance of selected strategies (OHSUMED id 1, 5, 9, 11, 15, 19, 21, 22). On the left side the mean average precision for each threshold t and on the right side, μMAP, the mean NDCG, and the mean NDCNG value are presented.

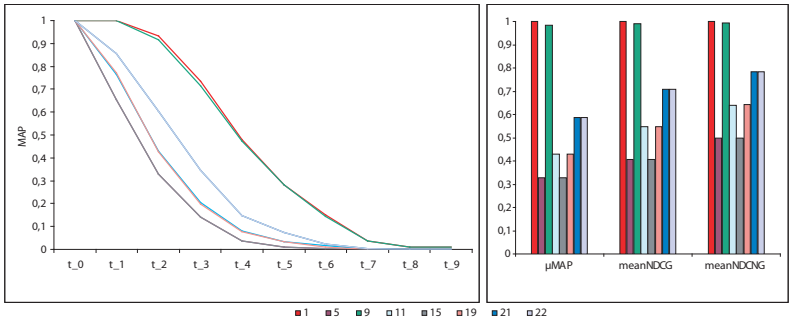

Fig. 7. Performance of selected strategies (OHSUMED id 1, 5, 9, 11, 15, 19, 21, 22), taking strategy TF of title (OHSUMED id 1, 9 levels of relevance) as approximated virtual user.

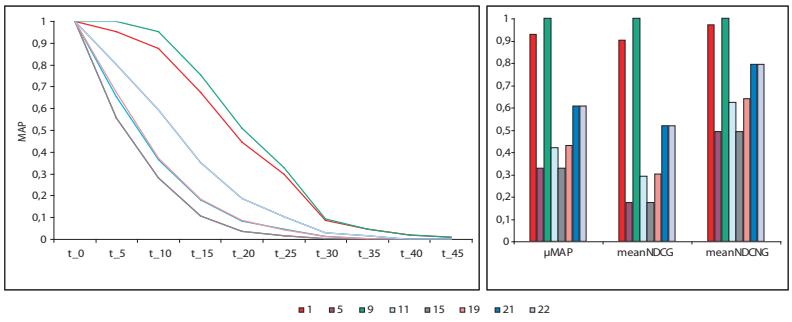

Fig. 8. Performance of selected strategies (OHSUMED id 1,5,9,11,15,19,21,22), taking strategy TF-IDF of title (OHSUMED id 9, 158 levels of relevance) as approximated virtual user.

Additionally, we showed in this paper that the NDCG measure is sensitive to the number of relevance levels in a dataset making it impossible to compare the performance values computed for datasets with a different number of relevance levels. To overcome this problem, we suggest an additional normalization ensuring that the number of relevance levels in the dataset does not influence the computed performance values. Our evaluation shows that NDCNG assigns similar performance values to recommender strategies that are almost similar except that different numbers of relevance levels are used. In the analysis, we demonstrated that high gain values (caused by a high number of relevance levels) lead to incommensurately low NDCG values. Since typically the number of relevance levels differs between the data sets the NDCG values cannot be compared among different data sets. Thus, the gain values per level of relevance must be limited. An additional normalization solves this problem.

Future Work As future work, we plan to use the measures μAP and NDCNG for evaluating recommender algorithms on additional datasets with multi-graded relevance assessments. We will focus on movie datasets such as EACHMOVIE [7] (having user ratings on a discrete scale from zero to five), and movie ratings from the Internet Movie Database (IMDB)[1] (having user ratings on a scale from one to ten).

References

1. W. Hersh, C. Buckley, T. J. Leone, and D. Hickam. Ohsumed: an interactive retrieval evaluation and new large test collection for research. In *SIGIR '94*, pages 192–201, New York, NY, USA, 1994. Springer-Verlag New York, Inc.
2. D. A. Hull and S. Robertson. The trec-8 filtering track final report. In *In The 8th Text Retrieval Conference (TREC-8), NIST SP 500-246*, page 35Ũ56, 2000. http://trec.nist.gov/pubs/trec8/t8_proceedings.html.
3. K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, 2002.
4. T. Joachims and F. Radlinski. Search engines that learn from implicit feedback. *Computer*, 40(8):34–40, 2007.
5. J. Kekäläinen and K. Järvelin. Using graded relevance assessments in ir evaluation. *Journal of the American Society for Information Science and Technology*, 53(13):1120–1129, 2002.
6. T.-Y. Liu, T. Qin, J. Xu, W. Xiong, and H. Li. LETOR: Benchmark dataset for research on learning to rank for information retrieval. In *SIGIR Workshop: Learning to Rank for Information Retrieval (LR4IR 2007)*, 2007.
7. P. McJones. Eachmovie collaborative filtering data set. Available from http://research.compaq.com/SRC/eachmovie/, 1997.
8. M. A. Najork, H. Zaragoza, and M. J. Taylor. Hits on the web: how does it compare? In *SIGIR '07*, pages 471–478, New York, NY, USA, 2007. ACM.
9. C. J. Van Rijsbergen. *Information Retrieval, 2nd edition.* Dept. of Computer Science, University of Glasgow, 1979.
10. E. M. Voorhees and D. K. Harman, editors. *TREC: Experiment and Evaluation in Information Retrieval.* MIT Press, 2005.

[1] http://www.imdb.com/

Classifying Users and Identifying User Interests in Folksonomies

Elias Zavitsanos[1], George A. Vouros[1], and Georgios Paliouras[2]

[1]Department of Information and Communication Systems Engineering
University of the Aegean, Greece
izavits@iit.demokritos.gr, georgev@aegean.gr
[2]Inst. of Informatics and Telecommunications, NCSR Demokritos, Greece
paliourg@iit.demokritos.gr

Abstract. This paper presents a probabilistic method for classifying folksonomy users to folksonomy sub-domains and identifying their particular interests. In particular, we propose a method for mining topic hierarchies that may reveal either the collective or the user-specific conceptualization of those domains, as these are reflected by users' tags. We then propose two alternatives for identifying users' interests in the domains: The first exploits users' tags directly, and the second exploits users' specific conceptualizations of each domain. Both approaches use the collective domain conceptualizations as "reference", to which users' tags and conceptualizations are compared. The proposed statistical method is parametric-less and does not require any prior knowledge or external resources. We apply the proposed method on the Del.icio.us online bookmarking system and we provide experimental results.

1 Introduction

Collaborative tagging systems, also known as *Folksonomies*, comprise *content* (objects, resources), *annotations* (tags) and *users*. Popular examples of folksonomies include Del.icio.us, Flickr and CiteULike. Although the term "folksonomy" is based on the term "taxonomy", that implies a hierarchy, folksonomies constitute flat organizations of tags and resources: They do not include semantic relations between tags or any representation of tags' intended meaning. Folksonomy tags depend totally on the interests, preferences, conceptualization, nomenclature, whim and personal style of users. Therefore, there is a great potential for acquiring knowledge about folksonomy users by exploiting their tags ([5], [12], [11], [6], [18]), introducing a number of interesting challenges and opportunities in the context of Web 2.0 and its bridge to the Semantic Web.

In this context, the issues that this paper addresses are the following: (a) Automated identification of conceptualizations of domains, by exploiting the tags users introduce to resources related to those domains, and (b) exploitation of tags and induced conceptualizations for identifying individual users' interests to specific domains and topics. Numerous entities and organizations can make use of such capabilities: For advertisement, for recommendation and for educational

purposes, identifying users' interests, preferences and needs is of high value. Moreover, the collaborative tagging systems themselves could be improved by these capabilities, guiding the users to specific topics of interest, and of course, influencing the future tagging activity.

To address these challenging issues, we present a probabilistic method for classifying folksonomy users to folksonomy sub-domains, mining users' interests and conceptualizations. In particular, the contributions made in this work are as follows: (a) The automated induction of topic hierarchies from tags, in a statistical and parametric-less way, without requiring any external resources or prior knowledge, using the method proposed in [19]. These hierarchies of latent topics represent (collective or user-specific) conceptualizations of domains. (b) The use of collective topic hierarchies for classifying and identifying particular interests of users to the specific domains, by means of two alternative methods. The first exploits users' tags directly, and the second builds and exploits a user-specific conceptualization of each domain. Both approaches use the collective conceptualizations as "references", to which users' tags and conceptualizations are compared. It must be emphasized that all the above methods are performed in an unsupervised and language agnostic way, without requiring training data for each user. In our experiments we use datasets gathered from Del.icio.us, a popular online bookmarking system that offers collaborative tagging of bookmarks.

2 Related Work

Regarding the induction of hierarchies from folksonomies, the work in [7] presents a method for converting a corpus of tags into a taxonomy. The corpus is represented as frequency vectors of tags. A similarity function is defined between vectors and then a threshold is established to prune irrelevant values. Finally, for a given dataset, a tag similarity graph is created exploiting the notion of graph centrality. Starting from this graph, a hierarchy is induced.

The work in [15] proposes the application of Sanderson's probabilistic subsumption model [13] to tag sets in order to induce a hierarchy of tags from Flickr. The method adjusts the statistical thresholds of the subsumption model and adds filters in order to control the highly idiosyncratic Flickr vocabulary. The aim is to eventually produce a hierarchy of tags.

Since folksonomies are actually triples, the authors in [14] present a formal model of folksonomies as a tripartite hyper-graph and explore possible projections of the folksonomy into two dimensions, in order to apply association rule mining methods and mine the relations between tags. Doing so, they manage to create a graph of tags connected with edges that represent mined rules.

The work in [8] uses formal concept analysis to build tag hierarchies from tags of the blogosphere. The main assumption is that if a blog has relationships with other blogs, these blogs will use a similar sets of tags.

Regarding the clustering approaches reported in [18] and [21], our aim is not to cluster the tags per se, but to identify the latent topics that reveal the content of tag chains: Since tags are introduced by users, latent topics reflect

users' conceptualizations. Additionally, tags may contribute to different topics with different proportions, and topics are represented as probability distributions over the tag space.

To a greater extent than existing approaches, in this paper we present a fully automated, parametric-less method for learning a hierarchy of latent topics from tag chains without the use of external resources or any prior knowledge. It must be pointed out that we consider the proposed method to be complementary to the approaches reported in [12] and [6], since it can be applied to different projections of the information concerning tags. However, further work is necessary to thoroughly compare the different approaches using commonly agreed datasets and evaluation criteria.

Regarding the construction of users' profiles, or the classification of users based on their interests in folksonomies, the work in [3] presents a framework that depends on external ontologies in order to build users' profiles, given their tagging activity and navigation in a folksonomy. A predefined ontology defines the concepts that are required to build a user profile. Profiles are exploited for recommendation purposes by a reasoner. The whole framework depends heavily on external ontologies and resources that are being used used for matching tags with elements of the domain ontology.

The work in [9] aims to cluster users based on their tagging activity. For a particular domain of interest, the main idea is to find the urls and the users that have labeled those urls with the tags in the domain. For each domain, a cluster of users is generated, comprising users with similar interests.

Moreover, the authors in [4] propose a method for generating and maintaining user profiles in a tag-based manner. The basic idea is to relate a user with a set of tagged objects and store them in an intermediate user profile. The representation of the user profile is based on the tags associated with the objects. Based on the user profile, recommendations can take place, since the tags define the interests of the users.

The work in [16] proposes an architecture for building user profiles by exploiting folksonomies, in four steps: (a) user account identification, (b) harvesting of user tags, (c) tag filtering to identify synonyms and deal with misspellings, and (d) profile building by matching tags to Wikipedia categories.

Finally, the work in [1] aims to exploit users' tags and additional knowledge inferred from the expertise profiles of other users to infer user's expertise. Our approach is rather generic: It induces and exploits collective and user-specific topic hierarchies, aiming to the classification of users to specific domains and to the identification of users' specific interests to these domains.

Therefore, to a greater extent than the existing approaches, the aim of this paper is to classify the users to specific domains, according to their interests, in an automated and unsupervised way, identifying also their specific interests to topics of these domains, given their tags. This is done with a probabilistic topic modeling approach, in order to avoid pitfalls related to surface appearance of tags.

3 The Del.icio.us Datasets

We provide experimental results using datasets compiled from the Del.icio.us collaborative tagging system: Crawling Del.icio.us we have gathered the tag chains (sets of tags related to a specific resource) of resources regarding a specific "domain". E.g. a tag chain for a resource in `http://www.Del.icio.us.com/popular/programming` includes the tags *python*, *c*, *compiler*, *performance*, *programming*. The crawler takes as input a single tag characterizing a domain (e.g. **programming**), and a number specifying the depth of the crawling process. For instance, for depth equal to 0, only the tag chains of the first page for the input tag are gathered. For depth equal to 1, the tag chains of the first page are gathered, and next, for each tag of each tag chain, the tag chains of the first page of that tag are also gathered. The above crawling process is done without considering the individual users tagging the resources. We are also interested in harvesting the tag chains of specific users. For this purpose we follow a similar process as that for harvesting the tag chains of a domain. Again, the depth parameter must be specified in order to retrieve the user-specific tag chains. These datasets provide all the necessary information in order to induce users'-specific conceptualizations and further classify the users to the domains. Obviously, deep crawling results in introducing tags that tend to be not closely related to the domain, thus introducing noise. For instance, starting with the tag **programming**, if the crawler reaches a depth of 4, then resources that are not closely related to programming appear, introducing also irrelevant tags. Having said that, we must point out that the specification of a domain with a single tag (as it is done here) is done without any loss of generality, since in case we were using a set of tags, the crawling process would consider the resources that have been tagged with all tags specified. However, starting the whole process from a single tag makes the whole task of inducing topic hierarchies more difficult, since a lot of heterogeneity is introduced in the set of tags gathered by the crawling process.

We have been running the crawler for a two-month period, for four domains, delineated by the following tags: **design**, **software**, **programming** and **web**. We have crawled Del.icio.us for each of these domains and for depth values 0 to 3. For each domain, we have created a corpus of documents. In particular, each tag chain is treated as a separate ("virtual") document. Regarding individual users, for the above time period, and for the aforementioned depths, we have gathered the tag chains of 300 users for each of the four domains. Table 1 summarizes the characteristics of the compiled dataset. The first column indicates the domain of interest. The next four columns indicate the different depths of crawling. Each cell shows the number of documents per data set. The last column indicates the number of users.

4 The Proposed Method

The proposed method is based on computing hierarchical topic models for specific domains. These models, constructed by exploiting the gathered tags, represent conceptualizations of those domains.

Table 1. Crawled data: Number of virtual documents per crawling depth and number of users.

Domain	d 0	d 1	d 2	d 3	Users
design	12	125	1511	4678	300
software	7	122	1417	2476	300
programming	7	84	931	1993	300
web	7	146	1073	5510	300

The dataset gathered for each domain (described in Section 3) constitutes the input for building the domain topic hierarchy. This process consists of two main steps: The first step creates a document - tag matrix of frequencies. The tags of the corpus constitute the features of the vector representation, whose values are the frequencies of the tags occurring in the documents. This matrix is the input to the second step, which induces the topic hierarchy. This is further described in detail below. We have to point out that the introduced method may skip this first step and instead use a domain ontology: In this case, the ontology concepts must be transformed to distributions over the common term space of the ontology and the crawled tags [20]. This is a subtlety that we plan to consider in our future work.

4.1 Hierarchy Learning

The proposed hierarchical learning algorithm is based on the Hierarchical Dirichlet Process (HDP) priors [17], as shown in figure 1b.

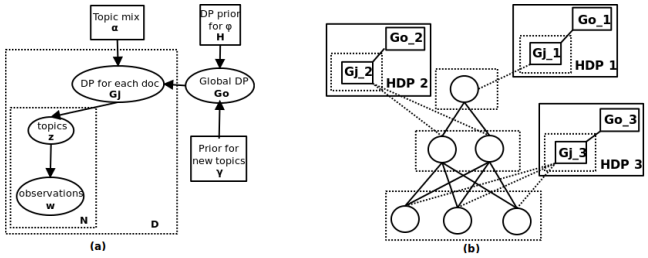

Fig. 1. (a): The HDP model. Assuming a corpus of D "virtual" documents, each of length N, there is a DP G_j for each document to draw tag distributions and a global, higher-level DP (G_0) that maintains the global distribution of tag distributions. (b): The hierarchy learning model. There is a HDP associated at each level, connected to all topics of that level.

A document (tag chain) consisting of N words (tags) is assumed to have been generated by a number of K latent topics. These topics have been drawn by a Dirichlet Process base measure G_j which in turn has been drawn from a Global Dirichlet Process G_0 that applies to the whole corpus of documents, assuring the

sharing of topics among documents (Fig. 1a). The topics maintain a multinomial probability distribution over the words of the term space of the corpus (i.e. over the space of tags in the corpus). Thus, according to the generative process, to generate a document (i.e. a tag chain of a specific resource), topics are selected according to some probability, and then, for each selected topic, a word (tag) is selected from that topic, according again to some probability. In our case, where the corpus is given at hand, we perform the reverse process: The inference of the latent topics. We are interested therefore in the process where the model computes the topics and their hierarchical relations.

The proposed learning method, besides the fully automated estimation of the topic hierarchy per domain, supports the inference of the depth of the hierarchy, and by inheriting the characteristics of the HDP model, estimates the number of nodes at each level. All the above factors make the learning of the topic hierarchy completely parametric-less without relying on external resources or prior knowledge of the domain of interest.

In particular, according to the proposed method (Fig. 1b), at each level, there is a DP (G_j) for each document and a global DP (G_0) over all the DPs for that level. Therefore, each level of the topic hierarchy is associated with a HDP. An important characteristic of this approach is that the number of topics of each level is automatically inferred, due to the non-parametric Bayesian nature of the HDP, and it allows the topics at each level to be shared among the documents in the dataset.

The dataset provides the observations, i.e. the occurrence of tags, for the inference of the latent hierarchy. The process starts by inferring the lowest level of the hierarchy, i.e. the leaves. During this process, tags are assigned to leaf topics. Having inferred the leaf topics, their mixture proportions for the documents is known. In other words we can infer which topics have contributed, and to what degree, to the "generation" of each document. This type of inference has been used for the classification of user-specific virtual documents (tag chains) described in the following section.

Furthermore, the assignment of a tag to a specific topic constitutes the observation for the inference of the next level up. At the next levels up, following the same procedure, each inferred topic maintains a distribution over the tags of the virtual documents and over the topics at the level below. Therefore, each internal node or topic maintains a distribution over tags and over subtopics. The procedure is repeated until it converges to a single topic, which serves as the root of the hierarchy. The sampling scheme that we propose for the taxonomy learning method is described in Algorithm 1. More details may be found in [19].

Regarding the induced hierarchies, these contain hierarchical relations among topics rather than tags. Induced topics may index documents (i.e. tag chains) even if their constituent tags do not actually appear in a document, if this is consistent with the major patterns of association in the data. Doing so, synonym terms may end up in the same topic, and a polysemous term may exist in several topics.

User Classification based on Maximum Likelihood: Having computed the collective topic hierarchy for each of the domains, the first users' classification

Algorithm 1 Estimation of latent topic hierarchy.

```
DATA: Document - Tag matrix of frequencies
RESULT: Estimated topic hierarchy
set M=number of documents
set V=vocabulary size
estimate leaf topics K
set T = K
while | T |> 1 do
  // transform document space
  set M = K
  set input=MxV matrix of frequencies
  estimate topics K of next level up
  set T = K
end while
```

alternative computes the log-likelihood of each hierarchical model, given the documents (i.e. the dataset) of each user. The user is classified to the model that has the maximum likelihood, since it is assessed that this is the model that is able to "generate" the dataset of that user. The log-likelihood of the models is measured by using the Left-to-Right Sequential sampler [2]. It must be pointed out that as a consequence of this computation, the log likelihood of the specific topics that may have generated users' document are also computed: Doing so, the interest of users to specific domain topics is revealed.

User Classification based on Hierarchy Comparison: The second alternative for user classification, in conjunction to the computation of the collective models, creates a topic hierarchy for each user, using as input the user's tag chains. The process is the same as the one used for computing the collective conceptualization of each domain, as explained in Subsection 4.1. Then, the classification process continues as follows: having the collective model of each domain and the domain model of a particular user, the topic hierarchies are compared and the corresponding user is classified to the domain whose model is "closest" to the user's model. Closeness is measured by the metrics described in the following paragraphs. In order to compare two hierarchical topic models, we use the DMA distributional alignment method proposed in [20]. This method is mainly used for evaluating learned ontologies with respect to a gold standard. The main idea is to align the two ontologies, and based on the matchings to derive some scores that are inspired by the notions of Precision, Recall and F-measure. In our case, we treat the collective hierarchy as the gold one and the user-specific hierarchy as the learned one. The extensive experimental tests in [20] show that this method succeeds to reflect the deviation between the two hierarchies, taking also into account the differences between the hierarchies' structures and the deviations of the induced topics. It is also shown that the effectiveness of the alignment computed by this method is comparable to that of state of the art methods. Therefore, this method constitutes a firm basis for classifying users by exploiting domain-specific hierarchical topic models. Again, as a consequence of this comparison between models, topics in the collective model are compared to

user-specific topics: Doing so, the interest of users to specific domain topics is revealed.

Briefly, the DMA alignment method proceeds as follows. Given that all nodes in both hierarchical models are represented as multinomial probability distributions over the tags of the dataset, the method proceeds to compute a common term (tag) space. This contains the union of the tags that the two models comprise. The nodes of the two hierarchies are now transformed to distributions over the common term space. Then, the collective topic hierarchy is compared to the user-specific hierarchy, by comparing the topics from the two hierarchies. For the computation of the similarity SD between different topics we have used the Total Variational Distance Measure (TVD) specified in Equation (1) and ranging in $[0, 1]$.

$$TVD = \frac{1}{2} \sum_{i} \mid P(i) - Q(i) \mid \tag{1}$$

In Equation (1), $P(\cdot)$ and $Q(\cdot)$ are multinomial probability distributions over tags in the compared topics. Therefore, the matching scheme compares the distributional representations of topics and finds the best correspondences between topics. Finally, Matching Precision MP, Matching Recall MR and the Matching F-measure MF[1] provide an assessment of user's topic hierarchy "closeness" to the collective topic hierarchy. The formulae for these measures are given in Equations (2), (3) and (4).

$$MP = \frac{1}{M} \sum_{i=1}^{M} (1 - SD_i) PCP_i \tag{2}$$

$$MR = \frac{1}{M} \sum_{i=1}^{M} (1 - SD_i) PCR_i \tag{3}$$

$$MF = \frac{(\beta^2 + 1) MP * MR}{(\beta^2 MR) + MP} \tag{4}$$

In Equations (2) - (4), M is the number of matchings between topics in both induced hierarchies. The PCP and PCR (*Probabilistic Cotopy Precision and Recall*) factors in Equations (2) and (3) respectively, are influenced by the notion of Semantic Cotopy [10]. The cotopy set of a topic C is the set of all its direct and indirect super and subtopics, including also the topic C itself. Thus, for a matching i, of a topic T in the user-specific hierarchy and a topic C in the collective hierarchy, PCP_i is defined as the number of topics in the cotopy set of T matched to topics in the cotopy set of C, divided by the number of topics participating in the cotopy set of T. For the same matching i, PCR_i is defined as the number of topics in the cotopy set of T matched to topics in the

[1] Originally, these measures are called P, R and F values, but since we use these standard measures for the evaluation of the proposed methods, we have renamed them .

cotopy set of C, divided by the number of topics participating in the cotopy set of C. Values of the MP, MR and MF measures close to 1 indicate that the user-specific topic hierarchy is close to the collective one, while values close to 0 indicate the opposite.

5 Empirical Evaluation

The empirical evaluation of the proposed methods concerns the classification of different users into four main domains: **design**, **programming**, **software** and **web**. This process constitutes a multi-class classification problem (in the sense that we have more than two classes) that we address in an unsupervised way. We provide quantitative results in terms of Precision, Recall and F-measure per domain, for both alternatives of user classification described in Section 4: Precision is defined as the ratio of the number of users correctly classified to a domain to the total number of users that are classified to that domain. Recall is the ratio of the number of users correctly classified to that domain to the number of users that should have been classified to that domain. The F-measure is the harmonic mean of Precision and Recall. The experiments have been performed using a 3 GHz PC with one core. In the worst case, the learning of the hierarchy of the collective conceptualizations requires approximately 32 minutes, while the classification task requires less than 5 minutes. The CPU intensive task corresponds to the learning of the hierarchies, which depends on the size of the dataset (i.e. the number of the tags).

We provide experimental results using the datasets for crawling depth (i.e. the depth for gathering the user-specific and user-independent tag chains per domain) equal to 1. We do consider this crawling depth for two reasons: (a) Tags gathered from greater depths result to hierarchies that contain certain portions maybe from different domains. (b) Given the tags of users and the computed collective conceptualizations, we have asked three external evaluators to classify the users into the four categories (**design**, **programming**, **software** and **web**) in order to use this classification as the ground truth. Gathering tags from depths greater than 1 would make the set of virtual documents per user much larger and the topics in the hierarchies would be more as well: This would make the job of evaluators much more harder and thus, error-prone.

To show an example of the induced hierarchies, Figure 2 illustrates the hierarchy for the domain **web**, using the dataset compiled for crawling depth equal to 1, as well the induced hierarchies of two users: One belonging to that domain and one that is not. The figure shows the estimated latent topics with the four most probable words from their multinomial probability distributions.

The evaluators have agreed for the classification of 285 different users per domain. Each user was classified to only one domain (multi-label classification, in the sense that a user may belong to more than one domain at the same time, is left for future work). For evaluation purposes, all 1140 users were put in a single directory. The aim is to classify each of these users in one of the four categories in an unsupervised way. Having said that, we must point out that the

evaluation method is rather strict, since the classification problem is handled as a multi-class classification problem.

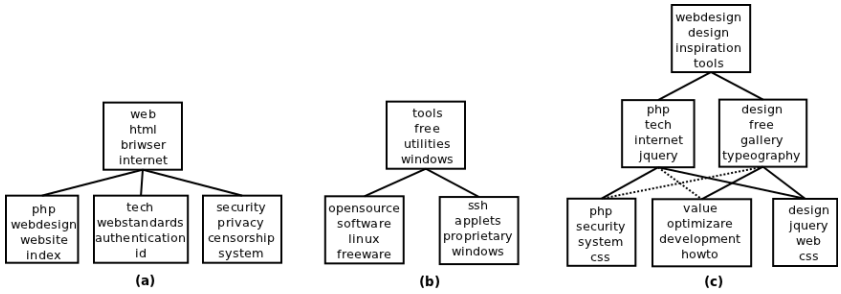

Fig. 2. The induced hierarchies (a) for a user belonging in the domain "web", (b) for a user that does not belong to that domain, and (c) for the domain "web" for crawling depth=1.

Table 2 provides experimental results for both classification alternatives. Regarding the classification process based on the log-likelihood, we observe that the F-measure ranges between 0.80 and 0.90, while the *Accuracy* of this method is equal to 0.865. We observe that the effectiveness of this classification method for the domains **programming** and **software** is lower than that reported for the other domains. This is so, since these two domains share many tags and it is rather difficult to classify users to one of them: This is something also experienced by the evaluators.

Table 2. Evaluation results for the two classification approaches.

	LogLikelihood Approach			DMA		
Domain	Precision	Recall	F-measure	Precision	Recall	F-measure
Design	0.98	0.82	0.89	0.91	0.80	0.85
Programming	0.99	0.67	0.80	0.96	0.68	0.80
Software	0.75	0.97	0.85	0.75	0.96	0.84
Web	0.83	1.0	0.90	0.82	0.92	0.87

One could choose to treat this task as a binary classification process, since we have a distinct model for each class (domain). In other words, one could choose to assess the method by measuring the classification accuracy per domain if as no other classes existed. That is, to assess for instance whether a particular user is classified to a specific domain (i.e. binary classification) without penalizing mis-classification to other domains. In that case, for all domains the precision would be equal to 1.0, boosting accordingly the corresponding F-measures to 0.90 for the domain **design**, 0.80 for the domain **programming**, 0.98 for the domain **software** and 1.0 for the domain **web**.

Regarding the second classification method (DMA) we observe that the F-measure ranges between 0.80 and 0.87, with *Accuracy* equal to 0.841. Again, if we address classification as a binary problem, then the precision of each domain would be equal to 1.0, and the F-measures would become 0.89 for the domain **design**, 0.81 for the domain **programming**, 0.98 for the domain **software** and 0.96 for the domain **web**. In order to increase the distinctive power of this classification alternative we experimented with different values of the parameter β of the MF measure so as to give more emphasis to the MR measure: This is motivated by the fact that by comparing the model of each user with the collective one, we prefer having a large number of correspondences between topics, rather than having few precise correspondences. By increasing β, we have observed a significant improvement of the evaluation results. Specifically, when setting $\beta = 2$, the F-measure of the domain **design** increases to 0.92, while the F-measure of the domain **web** increases to 0.93. The *Accuracy* of the method for this setting is equal to 0.876. Finally, when setting $\beta = 3$, the evaluation results are further improved. In particular, the F-measure of the domain **design** increases to 1.0, the domain **programming** increases to 0.82, the domain **software** increases to 0.861 and **web** is close to 1.0. The *Accuracy* of the method for this setting is equal to 0.92.

In order to gain a better insight on how the two proposed classification alternatives are related, we performed the following experiment: For each of the users that were classified correctly by both classification methods, we measured the log-likelihood of each topic in the collective model of each domain. This task computes the likelihood of each domain topic to index the tag chains of a user. The soft clustering that is performed during the learning of the hierarchy imposes that a tag chain may have been generated by more than one topic, with different proportions. Experimental results showed that in case a user U is classified under a specific domain D (i.e. the user's documents are indexed by some of the topics in the hierarchy for D), then all these topics in the domain hierarchy correspond to topics in the user's topic hierarchy. These topics show the particular interests of users to the specific domains.

6 Conclusions

Folksonomies are rapidly gaining momentum in the context of the Social Web. In this paper we presented methods for classifying the users of a folksonomy into hierarchical models that are induced by folksonomy data corresponding to a specific domain of interest. Specifically, given a set of tags, the proposed method is able to create a hierarchical topic model for a particular domain. From this point, two alternatives were proposed for user classification. One, based on the log-likelihood of the collective models to generate/index users' tag chains, and another, based on computing correspondences between the induced, collective hierarchical models and the user-specific induced models. Initial evaluation results provided illustrate the behavior of the proposed methods in both approaches of classification. We have observed promising results that suggest further investigation towards the direction of user classification in folksonomies.

Future plans include multi-label classification experiments, the application of the method to larger datasets with more users and for specific communities of users and experimentation with various probability matching schemes regarding the user classification based on the hierarchy comparison method.

References

1. A. Budura, D. Bourges-Waldegg, and J. Riordan. Deriving expertise profiles from tags. In *CSE (4)'09*, 2009.
2. W. Buntine. Estimating Likelihoods for Topic Models. In *ACML*, 2009.
3. F. Carmagnola, F. Cena, L. Console, O. Cortassa, C. Gena, A. Goy, I. Torre, A. Toso, and F. Vernero. Tag-based user modeling for social multi-device adaptive guides. *User Modeling And User-Adapted Interaction*, 18(5):497–538, 2008.
4. J. Diederich and T. Iofciu. Finding communities of practice from user profiles based on folksonomies. In *EC-TEL 2006 Workshop Proceedings*, 2006.
5. S. Golder and B. Huberman. Usage patterns of collaborative tagging systems. *Journal of Information Science*, 32(2):198–208, 2006.
6. H. Halpin, V. Robu, and H. Shepard. The dynamics and semantics of collaborative tagging. In *SAAW'06*, 2006.
7. P. Haymann and H. Garcia-Molina. Collaborative creation of communal hierarchical taxonomies in social tagging systems. In *Technical Report*, 2006.
8. H. Kim, S. Hwang, and H. Kim. Fca-based approach for mining contextualized folksonomy. In *Proc. of the ACM Symposium on Applied Computing*, 2007.
9. X. Li, L. Guo, and Y. Zhao. Tag-based social interest discovery. In *WWW*, 2008.
10. A. Maedche and S. Staab. Measuring similarity between ontologies. In *EKAW*, 2002.
11. A. Mathes. Folksonomies - cooperative classification and communication through shared metadata, December 2004.
12. P. Mika. Ontologies are us: A unified model of social networks and semantics. *Journal of Web Semantics*, 5(1):5–15, 2007.
13. M. Sanderson and B. Croft. Deriving concept hierarchies from text. In *ACM Conf. of the Special Interest Group in Information Retrieval*, pages 206–213, 1999.
14. C. Schmitz, A. Hotho, R. Jschke, and G. Stumme. Mining association rules in folksonomies. In *Data Science and Classification*, pages 261–270, 2006.
15. P. Schmitz. Inducing ontology from flickr tags. In *Collaborative Web Tagging Workshop*, 2006.
16. M. Szomszor, H. Alani, I. Cantador, K. O'Hara, and N. Shadbolt. Semantic modelling of user interests based on cross-folksonomy analysis. In *ISWC*, 2008.
17. Y. Teh, M. Jordan, M. Beal, and D. Blei. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 2006.
18. H. Wu, M. Zubair, and K. Maly. Harvesting social knowledge from folksonomies. In *Proc. of the 17th Conference on Hypertext and Hypermedia*, 2006.
19. E. Zavitsanos. *Learning Ontologies from Text Collections and Evaluating them Against Gold Standards*. PhD Thesis, University of the Aegean, 2009.
20. E. Zavitsanos, G. Paliouras, and G. A. Vouros. Gold standard evaluation of ontology learning methods through ontology transformation and alignment. *TKDE*, `http://doi.ieeecomputersociety.org/10.1109/TKDE.2010.195` (to appear).
21. M. Zhou, S. Bao, X. Wu, and Y. Yu. An unsupervised model for exploring hierarchical semantics from social annotations. In *ISWC/ASWC2007*, 2007.

User Modeling for the Social Semantic Web

Till Plumbaum[1], Songxuan Wu[2], Ernesto William De Luca[1], and Sahin Albayrak[1]

[1] Technische Universität Berlin, DAI Labor, Berlin, Germany
{till,ernesto.deluca,sahin}@dai-lab.de
[2] Otto-von-Guericke Universität Magdeburg, Magdeburg, Germany
wusongxuan@gmail.com

Abstract. How can we make use of the personal information a single user is spreading all over the Social Web every day? In this paper we investigate what is needed from a user model point of view to support user data sharing and aggregation to enhance personalization and recommendation services. We present a study of 17 social applications to define requirements and attributes for a common user model that allows sharing of user data and analyze what is needed to enhance user model aggregation approaches. As a result, we present a comprehensive user model especially fitted to the needs of the Social Web. Furthermore, we present a WordNet for the user modeling domain as part of the user model to support user model aggregation.

Keywords: User modeling, Social Web, Semantic Web, User Model Aggregation

1 Introduction

Every day, people in the Social Web create 1.5 billion pieces of information on Facebook, over 140 million tweets on Twitter, upload more than 2 million videos on YouTube and around 5 million of images to Flickr[3]. This huge amount of social data attracts researchers who want to use it to learn more about user preferences and interests, and enhance recommendation and personalization systems. What most current system have in common is that they use data from a single application and depend on sufficient user information (user behavior or ratings) to produce good results [1, 2]. By using the distributed personal information a single user produces on a daily base, and by building a holistic model of the user, personalization and recommendation quality can be further enhanced. But, for this holistic model the distributed user data has to be aggregated across applications. This idea is not new, it has existed since the 90's where different research initiatives proposed generic user modeling servers that build a central structure to manage and share user information [9, 10]. These approaches could

[3] http://www.scribbal.com/2011/04/infographic-how-much-daily-content-is-published-to-twitter-facebook-flickr/

not succeed because of their static, predefined user models while application-based user models strongly differ in the information they need to know about a user (as we will show in Section 3). Another reason for the failure was that applications do not want to lose control over their data, thus, a central storage was not wanted. New trends from the Semantic Web can provide a remedy. Instead of having a central server, ontology based user models are proposed to support data aggregation and sharing. Thus, applications can keep their data but use a common "language" to model the information. While semantic technologies help to overcome technical problems, the main questions remain: What user information must a semantic model contain with focus on the Social Web? What requirements must a model fulfill to support data sharing and aggregation?

In this paper, we want to give answer to those questions by analyzing user models from different Social Web applications and draw conclusions about the diversity and type of user information that such a generic user model should have. We therefore discuss existing work and motivate a semantic Social Web User Model (SWUM). Requirements and structure of SWUM will be introduced in Section 3 and is based on the extensive analysis of 17 Social Web applications in Section 3.1 and 3.2. In Section 3.3 we also carefully investigate what is needed to enable an easy, automated, aggregation process. To give a better understanding of the intended use of the SWUM we present a use case in Section 4. The main contributions of this paper are an extensive analysis of requirements of today's Social Web applications regarding stored user data and the introduction of a new Social Web user model that is:

- generally adapted to the needs of Social Web applications and
- that allows an easy data sharing between applications.

2 Related Work

Until the turn of the millennium, most personalization and recommendation research focused on user information available in one application and how to use this information to enhance personalization quality. With the influence of the Social Web, or Web 2.0, and the fact that user information is highly distributed over several applications, research started to explore cross-system personalization approaches. This research can be roughly classified into two major directions [11]:

- A centralized approach with standardized models that aggregate the distributed user information and build the basis for cross-system information transfer.
- A decentralized approach where dedicated software components transfer user information from one application's representation to another.

The work presented in this paper is in alignment with the first direction, the centralized approach. This approach can also be subdivided into two aggregation strategies. The first strategy proposes the use of standardized user models which all involved applications must agree on. The second strategy deals with

the mediation of different user model representations using meta-models that connect user data from one application with data from another application, in the same domain, or across domains. The standardization approach involves no computational effort to aggregate data as all data already is in the same format. An effort in this direction is the General User Modeling Ontology (GUMO) created by Heckman et al. [7]. GUMO is a comprehensive user model that intends to cover all aspects of a user's life. The user dimensions covered range from contact information and demographics over abilities, personality right up to special information like mood, nutrition or facial expressions. GUMO is at the time of this writing the most comprehensive generic user modeling ontology. Another approach that came up with the Web 2.0 is the Friend-of-a-Friend (FOAF) ontology. FOAF is a lightweight model that is integrated on the website, the application's user interface, using RDFa. FOAF covers basic user information like contact information, basic demographics and allows to specify some social relations like group membership or "knows" relations to other FOAF profiles. GUMO, which represents the most generic user model, covers only some parts of information that are needed for the Social Web. Especially the *Interest* dimension (in music, books, etc.) and user information like accounts for different Social Web applications, which are crucial, as we show in Section 3, are completely missing. FOAF, which is designed for a Web use, is too simplified. FOAF has a "knows" relationship, which defines a social relation, but the type of the relation remains unclear. Also no user needs and goals can be defined, which is part of many social applications as we will see in Section 3.

The second strategy is to build meta-models that allow defining how application-dependent user data corresponds to user data from another application. This has the advantage that applications are not forced to adopt a predefined generic user model and can rely on their own model. In [13], the authors present an aggregation ontology which gives applications the possibility to define a model, which describes how information in different profiles is related and how data can be aggregated. Furthermore, the ontology not only allows to define relations between data in different application models but also to define the overlap, the similarity, of the modeled information. So it is possible to define that the field "interests" in one application and the field "music interests" in another, is related but only to a certain degree as "music interests" is only subset of "interests". In [16], van der Sluijs et al. present the Generic User model Component (GUC) which builds a central component where all applications have to subscribe to and describe their user model via a schema defining the data structure of the user models for different applications. The authors also suggest the possibility to use different matching and merging techniques to map input schemas and create a merged schema as the union of the input schemata and to construct combined ontologies of the application schemata. While the meta-model approach seems to be a more practical one but to achieve a semantic and syntactic interoperability, the big disadvantage is that is needs a lot of effort to connect all the different user models. This work currently has to be done manually or semi-manually and must be repeated for every new application user model.

To summarize: both strategies and the presented related work have shortcomings. Because of the big differences, regarding the covered user information and representation forms in different applications, the development of a commonly accepted ontology, covering all aspects of user modeling for all domains seems not feasible. The meta-model approach, without automatic aggregation mechanisms, is only applicable in small settings where only a few applications are connected and not for the Social Web. We therefore propose a middle way: We need a new 'common' user model that combines aspects of the presented approaches and focuses on a special domain, the Social Web. Also, the user model should support automatic aggregation by defining a structure that allows finding relations between different user model concepts and allows for a flexible extension of the model.

3 Requirements for a Social Web User Model

To define a user model for the domain of the Social Web, we first have to understand the demands of social web applications on user models. Therefore, we did an extensive survey of the modeled user information of 17 well-known Social Web applications. The list of analyzed applications is shown in Table 1. The applications were chosen because of their size and level of awareness (number of users, global distribution). To be able to consider local differences, we also included applications that are strong in only one or two regions (Orkut in South America, Lokalisten and StudiVZ in Germany). We also selected Social Web applications from different kinds of domains, photo- and video-sharing platforms, short-message services, social networks, etc. To decide if the user information stored by an application is of importance, we picked at least two Social Web applications from the same domain.

Table 1. List of 17 social applications that we analyzed for the requirements analysis

Facebook	`http://www.facebook.com`	Myspace	`http://www.myspace.com`
Windows Live	`http://home.live.com`	YouTube	`http://www.youtube.com`
Flickr	`http://www.flickr.com`	Yahoo	`http://de.yahoo.com`
Picasa Web	`http://picasa.google.com`	StudiVZ	`http://www.studivz.net`
Digg	`http://www.digg.com`	Yelp	`http://www.yelp.com`
Lokalisten	`http://www.lokalisten.de`	Orkut	`http://orkut.com`
Identi.ca	`http://identi.ca`	LinkedIn	`http://www.linkedin.com`
Vimeo	`http://www.vimeo.com`	Xing	`http://www.xing.com`
LastFM	`http://www.last.fm`		

For each evaluated application, we collected the type of information and the internal attribute name. Table 2 shows the type of user information and where the information was found on the Web page. The internal attribute names, used

by each application are particularly important as they are later used to define and name the attributes of the Social Web User Model (SWUM).

Table 2. Evaluation example for Yahoo: User information, attribute name and where the information was found on the Web page.

IU Name	Source code ID	Found on
Name	name	Registration Page
Firstname	firstname	Registration Page
Surname	secondname	Registration Page
Gender	gender	Registration Page
Birthday	birthdaygroup	Registration Page
Country	country	Registration Page
Postal Code	postalcode	Registration Page
Yahoo! ID and Email	yahooid	Registration Page

To be able to create our SWUM, we first have to decide which type of information, which user model dimensions, should be part of the model and which attributes in the different dimensions should be supported.

3.1 User Model Dimensions

After collecting all the information, the first step is to determine the user model dimensions that our user model has to cover. As shown in GUMO, a lot of dimensions exist, but not all of them are required in the context of the Social Web. Several dimension are mentioned and discussed in the literature. We present a consolidated taxonomy that bases on [17, 6, 8, 9, 3] and builds the basis for the selection of needed dimensions for our model:

- *Personal Characteristics* (or Demographics) range from basic information like gender or age to more social ones like relationship status.
- *Interests and Preferences* in an adaptive system usually describe the users interest in certain items. Items can be e.g. products, news or documents.
- *Needs and Goals*: When using computer systems, users usually have a goal they want to achieve. Such goals can be to satisfy an information need or to buy a product. The plan to reach such goals is for example to support users by changing navigation paths or reducing the amount of information to a more relevant subset.
- *Mental and Physical State* describe individual characteristics of a user like physical limitations (ability to see, ability to walk, heartbeat, blood pressure, etc.) or mental states (under pressure, cognitive load).
- *Knowledge and Background* describe the users knowledge about a topic or system. It is used in educational systems to adapt the learning material to the knowledge of a student, display personalized help texts or tailor descriptions to the technical background of a user. The knowledge and background is a

long-term attribute on the one hand but can differ and change from session to session depending on the topic. Knowledge and background about certain topics can increase or decrease over time [3].
- *User Behavior*: The observation and analysis of user behavior is usually a preliminary stage to infer information for one of the previous mentioned dimensions. It can also serve for direct adaptation like using interaction history to adapt the user interface to common usage patterns of the user.
- *Context*: In computer science context generally refers to "any information that can be used to characterize the situation of an entity" [4], but the discussion about what context actually is, is still ongoing[5]. In the area of user modeling, the term context focuses on the users environment (e.g. Location or Time, or devices the user interacts with) and human characteristics. Human characteristics describe Social Context, Personal Context and overlap with the *Mental and Physical State* dimension).
- *Individual Traits* refer to a broad range of user features that define the user as an individual. Such features can be user characteristics like introvert or extrovert or cognitive style and learning style.

Based on this user taxonomy, we checked all 17 applications if they cover these dimensions. Fig. 1 shows that social applications only cover some dimensions. All of the applications maintain *Personal Characteristics* and most of them also use *Interests and Preferences* information. Not used at all are the dimensions *Individual Traits* and *Mental and Physical State* which are more used in educational systems than in Social Web applications [3].

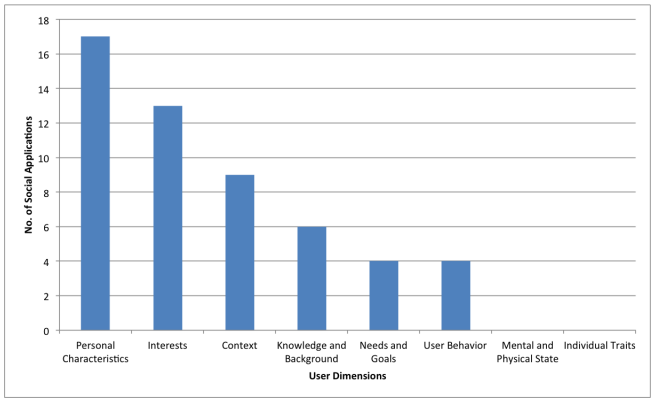

Fig. 1. Number of applications storing user information in the different user dimension categories.

The usage of *Knowledge and Background* and *Context* depends on the focus of the social application. Social business applications, like LinkedIn or Xing, support the *Knowledge and Background* dimension as users can enter their college

degree, areas of profession, etc. The support for the dimension *User Behavior* is not easy to work out, as user behavior usually is an implicit feature and not displayed on the user profile page of an application. It can be assumed, though, that almost all applications track user behavior on their site. A positive exception is "Google Dashboard" [4] where a user gets an easy overview of the stored personal information e.g. previous search behavior. The *User Behavior* dimension, although it is an important piece of adaptation and personalization, is to complex to be part of a generic Social Web User Model. For this purpose we recommend a specialized approach with an extra user behavior ontology as presented in [14, 12]. *Context* is an important area as the latest research shows and of importance for a Social Web User Model [15]. However, not all forms of context can be considered as a part of a Social Web User Model. The analysis showed that the Social Context and Location is of importance and therefore those subdimensions of context are part of SWUM. The importance of the context Time also seems of interest, but did not show up in our analysis.

From this analysis it follows that a main requirement for Social Web user model is, that it has to cover the user dimensions *Personal Characteristics, Interests, Knowledge and Behavior, Needs and Goals and Context (Social Context, Location)*. Accordingly, these dimensions are part of our SWUM.

3.2 User Model Attributes

After we selected the dimensions to be covered, we have to define the attributes that the user model should support.

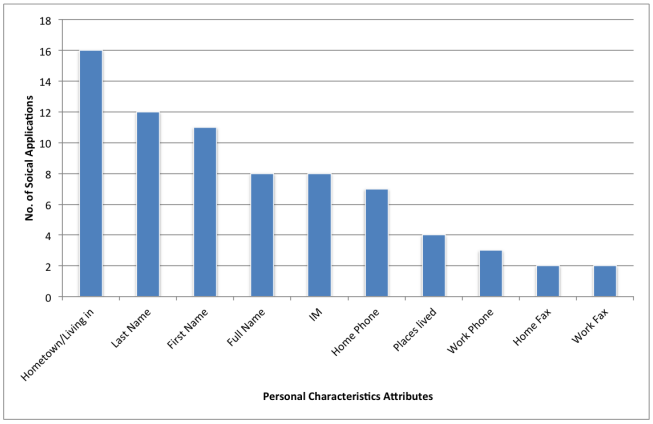

Fig. 2. Attributes of the *Personal Characteristic* dimension and how often they occur in the different applications.

[4] https://www.google.com/dashboard

The procedure for the attribute selection is similar to the procedure used to select the dimensions. We checked the different attributes of the different applications. Fig. 2 gives an example for the*Personal Characteristic* dimension. It shows an excerpt of the attributes and how often they occur in the analyzed social applications. In this way, we selected a set of attributes for each dimension. An example for the *Personal Characteristic* dimension is shown in Fig. 3. The *Personal Characteristic* is divided into two main concepts namely Demographics and Contact Information. The concept Location is a helper concept to model locations and link certain information, e.g. places lived, to it.

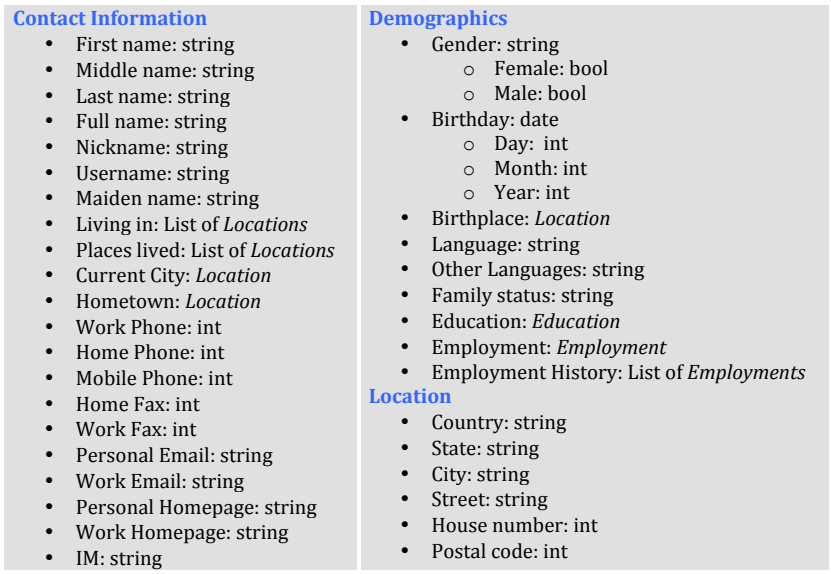

Fig. 3. SWUM attributes for *Personal Characteristic* dimension.

3.3 A User Model Word Net

An important outcome of the attribute distribution analysis was that often similar information is stored by most applications, but in differently named attributes, e.g. name (Yahoo) and real_name (LastFM) or homepage (LastFM) and website (Flickr). This problem of attribute name heterogeneity complicates a possible aggregation using a Meta-Model strategy. To cover that problem, we decided to extend our model with a WordNet like lexicon called User Model Word Net (UMWN). WordNet defines word sense relations between words. If a word represents a user attribute, the relatedness between different attributes can be acquired easily. However, many user attributes are not defined in WordNet. Moreover, many terms in WordNet are useless for user profile aggregation.

Hence, the standard WordNet does not help, thus, we designed a reduced WordNet, specialized to serve the user profile aggregation and initially based on the attribute distribution of our analysis. The decision to use a WordNet based structure comes from the fact, that WordNet has a flexible and well-defined lexicon schema, which is publicly known and accepted. The user model terms can be linked to each other accurately by using the properties defined in WordNet. An example is depicted in Fig. 4 where the word sense relations for name and date are shown.

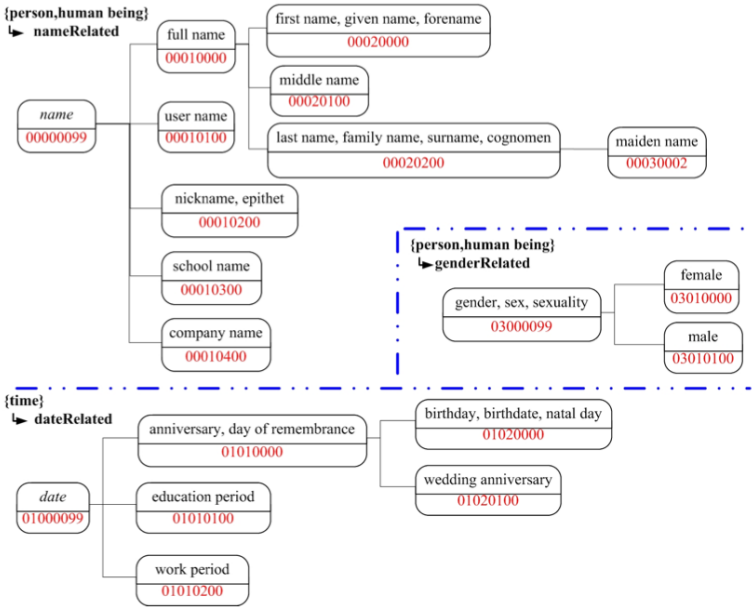

Fig. 4. User Model WordNet relations.

The UMWN is an important step for an automatized aggregation of different user models. It defines different types of word relations. The "Name" concept describes the relations between different types of name attributes that can occur in a user model. The concept "full name" consists of different subclasses like "first name", which has several synonyms ("given name" or "forename"). UMWN is stored in RDF(s)/OWL. Using ontology structures has the advantage that such a model is not static and can be easily extended. Our UMWN is extensible, towards not only to the individuals, but also to the schema of UMWN. Because of the highly distributed and heterogeneous user information in different user models, extensibility is an important feature. The UMWN contains currently ca. 520 syn sets where around 200 are unique in the User Model WordNet and not part of

the common WordNet. It also contains over 100 antonyms and homonyms and 200 meronyms.

4 Use Case: Profile Aggregation with the SWUM

To outline the intended usage and functionality of the SWUM (which includes the UMWN) we want to exemplary explain the steps needed to aggregate a Facebook user model and a LastFM user model. The aggregation is a two-step process which we want to explain by the example of the website/homepage attribute shown in Fig. 5. First step is to connect the LastFM attributes to the SWUM (see Fig. 5a). The LastFM user model has the attribute "homepage" which can be directly linked to the SWUM, with a concept match of 100%. The Facebook profile (Fig. 5b) contains the attribute "website" which is also part of our SWUM and thus, the attribute can also be linked to the SWUM without any extra effort.

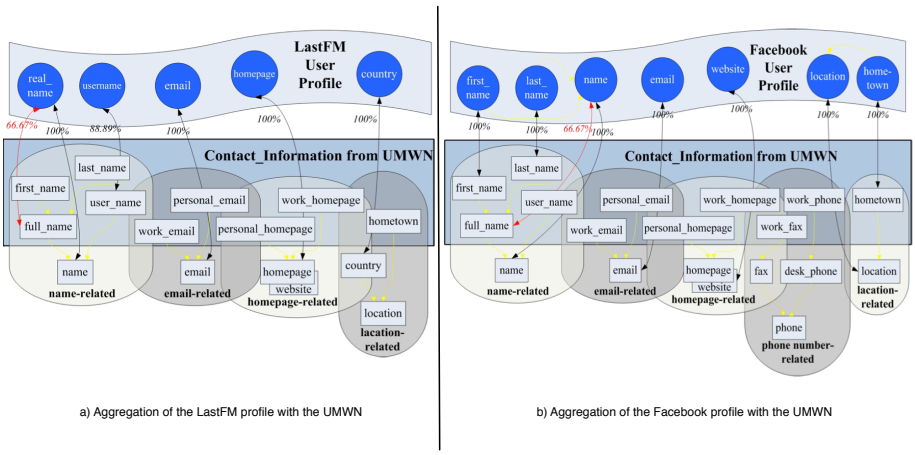

Fig. 5. First step of the aggregation process. Figure a) shows how the attributes of LastFM and the SWUM/UMWN are connected. Figure b) depicts the connections of the Facebook profile.

The second step is then to directly connect the LastFM and Facebook user model as shown in 6. Based on the previously shown aggregation, connecting both models is straightforward. Revisiting the homepage/website example, these attributes can be directly linked because of the UMWN. The UMWN defines a synonym relation between the concepts "homepage" and "website", thus the LastFM and Facebook attribute can be directly linked with a match of 100%.

The aggregation of attributes that are not part of the SWUM can be done not only using the attribute name but also using the attribute content. So could an

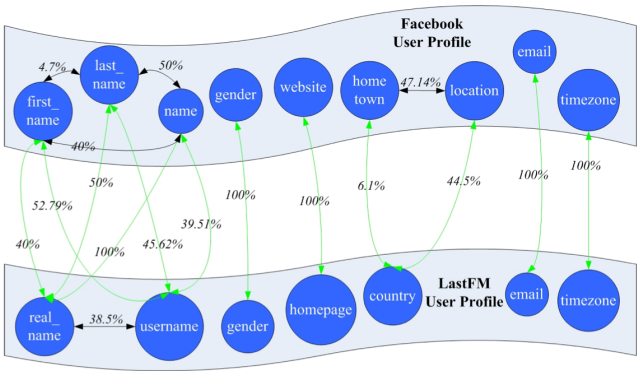

Fig. 6. Aggregated LastFm and Facbook profiles.

analysis show that the LastFM attribute "real_name" often contains the users' full name and thus a connection with the SWUM/UMWN attribute "full_name" can be done. Or the missing attributes can be added to the SWUM which is easy to do as it is a flexible RDF/OWL structure.

5 Conclusion and Outlook

In this paper we wanted to answer the question what are the requirements of the Social Web for a user model to profit from the available distributed user information. We present a new user model, the Social Web User Model (SWUM) that is fitted to the needs of the Social Web. We therefore conducted an extensive analysis of 17 social applications and to specify requirements, which dimensions and attributes are needed, for a Social Web user model. Based on this analysis we defined the dimensions a Social Web user model must cover and explained how the decision process was conducted. The analysis showed, that a Social Web user model only needs to cover certain dimensions of the user, namely *Personal Characteristics, Interests, Knowledge and Behavior, Needs and Goals and Context (Social Context, Location)*. We also presented the procedure to define the attributes of such a Social Web user model. To cover the problem of attribute heterogeneity throughout different social applications, we also equipped our model with a reduced WordNet that is especially tailored to the area of user modeling, the User Model Word Net (UMWN). The complete SWUM and UMWN model is based on RDF/OWL and thus easy to extend and reuse.

References

1. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. Trans. on Knowledge and Data Engineering 17, 734–749 (2005)

2. Bilenko, M., White, R.W.: Mining the search trails of surfing crowds: identifying relevant websites from user activity. In: Proceedings of the 17th international conference on World Wide Web. pp. 51–60. WWW '08, ACM (2008)
3. Brusilovsky, P., Millan, E.: User models for adaptive hypermedia and adaptive educational systems. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) The Adaptive Web: Methods and Strategies of Web Personalization, chap. 1, pp. 3–53. Springer-Verlag, Berlin Heidelberg New York (2007)
4. Dey, A.K.: Understanding and using context. Personal and Ubiquitous Computing 5, 4–7 (2001)
5. Dourish, P.: What we talk about when we talk about context. Personal Ubiquitous Comput. 8, 19–30 (February 2004)
6. Heckmann, D.: Ubiquitous User Modeling. Akademische Verlagsgesellschaft Aka GmbH, Berlin (2006)
7. Heckmann, D., Schwarzkopf, E., Mori, J., Dengler, D., Krner, A.: The user model and context ontology gumo revisited for future web 2.0 extensions. In: Proceedings of the Int. Workshop on Contexts and Ontologies: Representation and Reasoning. CEUR Workshop Proceedings, vol. 298. CEUR-WS.org (2007)
8. Jameson, A.: Modelling both the context and the user. Personal and Ubiquitous Computing 5, 29–33 (2001)
9. Kobsa, A.: Generic user modeling systems. User Modeling and User-Adapted Interaction 11, 49–63 (March 2001)
10. Korth, A., Plumbaum, T.: A framework for ubiquitous user modeling. In: Information Reuse and Integration, 2007. IRI 2007. IEEE International Conference on Information Reuse and Integration. pp. 291–297 (August 2007)
11. Kuflik, T.: Semantically-enhanced user models mediation: Research agenda. In: 5th international workshop on ubiquitous user modeling (January 2008)
12. Plumbaum, T., Lommatzsch, A., Luca, E.W.D., Albayrak, S.: Serum: Collecting semantic user behavior for improved news recommendations. In: UMAP 2011, Poster and Demo Session, Girona, Spain (2011)
13. Plumbaum, T., Schulz, K., Kurze, M., Albayrak, S.: My personal user interface: A semantic user-centric approach to manage and share user information. In: HCI International 2011 (2011)
14. Plumbaum, T., Stelter, T., Korth, A.: Semantic web usage mining: Using semantics to understand user intentions. In: UMAP '09: Proceedings of the 17th International Conference on User Modeling, Adaptation, and Personalization. pp. 391–396. Springer-Verlag, Berlin, Heidelberg (2009)
15. Said, A., Berkovsky, S., De Luca, E.W.: Putting things in context: Challenge on context-aware movie recommendation. In: Proceedings of the Workshop on Context-Aware Movie Recommendation. pp. 2–6. CAMRa '10, ACM, New York, NY, USA (2010)
16. van der Sluijs, K., Houben, G.J.: A generic component for exchanging user models between web-based systems. Int. J. Continuing Education and Liflong Learning Vol. 16 Nos. 1/2, 64–76 (2006)
17. Sosnovsky, S., Dicheva, D.: Ontological technologies for user modelling. Int. J. Metadata Semant. Ontologies 5(1), 32–71 (2010)

Personalization in Skipforward, an Ontology-Based Distributed Annotation System

Malte Kiesel[1] and Florian Mittag[2]

[1] DFKI GmbH, Kaiserslautern, Germany
[2] University of Tübingen, Germany
malte.kiesel@dfki.de / florian.mittag@uni-tuebingen.de

Abstract. Skipforward is a distributed annotation system allowing users to enter and browse statements about items and their features. Items can be things such as movies or books; item features are the genre of a movie or the storytelling pace of a book. Whenever multiple users annotate the same item with a statement about the same feature, these individual statements get aggregated by the system. For aggregation, individual user statements are weighted according to a competence metric based on the constrained Pearson correlation, adapted for Skipforward data: A user gets assigned high competence with regard to the feature in question if, for other items and the same feature type, he had a similar opinion to the current user. Since the competence metric is dependent on the user currently viewing the data, the user's view of the data is completely personalized. In this paper, the personalization aspect as well as the item and expert recommender are presented.

1 Introduction

Rating and recommendation web platforms have become important and ubiquitous nowadays. Typically, these platforms support collaborative filtering; users can rate items and are recommended items that people who liked the same items gave a high rating as well. This works fine for many cases and many domains; drawbacks are that in-depth explanations of recommendations cannot be given, and that the user has little control over the actual recommendation process. On the other hand, there is content-based filtering, which recommends items based on the features the user presumably likes. E.g., "We recommend song X since that song features prominent drums that you seem to like". Pandora.com is an example for such a system. This approach does not have the shortcomings of collaborative filtering outlined above; however, getting the content-based annotations needed for the recommendation process is costly, as this typically requires trusted experts.

Skipforward [4] pursues a hybrid approach—in terms of [6], it is a semantic recommender system pursuing an active item-based approach. Every ontology-based statement or *feature (instance)* Skipforward uses consists of a link to the item it refers to, a feature type[3], applicability value (+1: The feature applies to

[3] Technically, every user statement is an RDF instance of a subclass of the Skipforward `Feature` class.

the item, -1: The feature does *not* apply to the item), confidence value (0..1), and a plain text comment. The applicability value, in traditional recommender terms, corresponds to a user rating with regard to an item and a feature type. In the following, we avoid the term "rating" since this term implies item liking which does not quite fit in our case.

Skipforward's simple basic data model allows quite thorough annotation of items and provides rich metadata for recommender and other functionality. Conflicting annotations do not break the system; a competence metric is used for weighting individual statements for aggregated views on the system's data. The competence metric is the foundation for much of Skipforward's recommendation functionality, which not only includes an item recommender that finds similar items or items fitting some user-chosen feature profile, but also an expert recommender, and annotation recommenders (functionality that helps annotating items).

2 Components of Interest

For an overview of most Skipforward components, also see [4] and the Skipforward website[4] which also includes a screencast and online demo. In the following, we will describe the building blocks of the system: its top-level ontology, domain ontologies, the user interface, and recommender functionality.

2.1 Ontologies used in Skipforward

We have a number of requirements the top level ontology (coined *Skipinions*) shall be able to handle.
The ontology should be able to represent user opinions of items such as books, movies, etc. – we solve this by providing an `Item` and a `Feature` class. For example, book features could be "Thriller (Genre)" or "Fast-paced writing style". Every `Item` can be associated to a `Feature` by using the `Item`'s `hasFeature` property.
The facts databases of multiple users should be easy to merge. Fact databases can be just copied together using this approach. Smushing of items and features is done using the `owl:sameAs` predicate.
Provenance of statements needs to be tracked. We solve this by assigning an individual namespace to every user. Then, the URI of every instance created by this user has to use this namespace. This also fits nicely with Linked Open Data principles.
Plain text comments should be supported. Internationalized plain text comments can be added to `Feature`s.
It should be possible to explicitly dissent with an opinion of another user. This is implemented by the `applicability` property on every `Feature`. Applicability -1 means "this feature does not apply to this item", applicability $+1$ means

[4] `http://skipforward.opendfki.de/`

"this feature applies to this item", implementing the Open World Assumption. Additionally, any `Feature` can point to another `Feature` instance, implementing discussion threading.
Marking an opinion as uncertain should be possible. This is implemented by the `confidence` property on every `Feature`. Together with `applicability`, this forms a Dempster-Shafer-like approach.
The amount of noise seen by users should be kept minimal. The feature hierarchy as presented by the system is created by the owner of the respective namespace so arbitrary changes of the feature hierarchy are not possible. This is both a limitation and a feature of the system. It is limitating insofar as users cannot create new feature classes on the fly. On the other hand, systems that implement this (i.e., most normal tagging systems) show that a *lot* of entropy enters the system otherwise. We try to keep this noise limited to the feature instance level where it can be handled in a coherent manner.

The basic top level ontology structure can be seen in Figure 1.

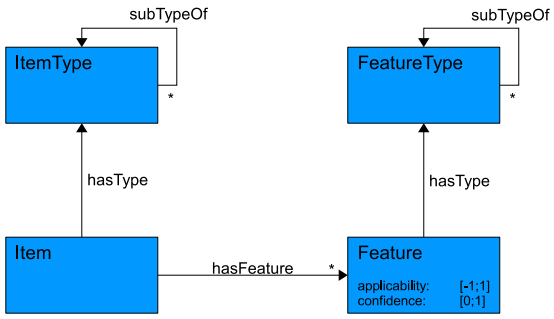

Fig. 1. Class diagram of Items and Features

Domain ontologies that subclass the `Item` and `Feature` classes as well as the `hasFeature` property are used for annotating actual items.

Currently, within Skipforward multiple domains are covered. Apart from ontologies that model features of board games and a corresponding set of instances, we mainly use *DBTropes* [3]. DBTropes.org is a wrapper of TVTropes.org, a wiki describing works of fiction by associating features—known as "Tropes"—to these works. The focus of TV Tropes is providing content-based annotations (as opposed to more technical information as, for example, supplied by IMDb.com), with a definite emphasis on fun and entertainment aspects. DBTropes extracts the information contained in the TV Tropes wiki, and publishes it as Linked Data. The implicit data model used in the TV Tropes wiki matches the data model used in Skipforward quite well. DBTropes uses Skipforward ontologies as its output format, and the main Skipforward application can consume this data directly. We use this data mainly as a source for feature types and the hierar-

chy within feature types. As of September 2011, the complete DBTropes data consists of about 10.000.000 RDF statements describing 22.000 items, 22.000 feature types, and 1.750.000 feature instances.

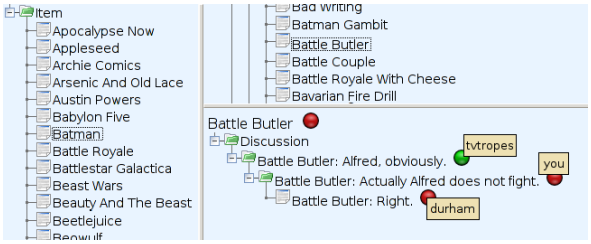

Fig. 2. Browsing DBTropes data in the Dojo-based Skipforward frontend.

2.2 Skipforward user interface

The Skipforward system is implemented as a web application. This allows running it easily in the background and in remote scenarios. Currently, we experiment with two frontends that serve slightly different purposes: A Dojo[5]-based Ajax UI, mainly used for annotating items, and a standard HTML template-driven UI that was built for better scalability and easier extensibility. The template HTML interface is mainly used for browsing and viewing additional information from the item and expert recommender components. In Figure 2, a small part of the data made available by DBTropes is shown, as visualized by the Skipforward UI implemented using the Dojo framework[6]. The left pane lists items (here, the movie *Batman* is selected); the upper right pane displays available feature types (here, the type *Battle Butler* is selected); the lower right pane shows instances of the selected feature type (i.e., users expressing opinions about one item with regard to one feature type). The uppermost (red) circle in the lower right pane shows the weighted average of applicability of the feature type with user opinions weighted according to their trust value. Here, three (threaded) user opinions for the feature *Battle Butler* in the movie *Batman* are available in the system. TV Tropes users stated that the feature *is* present for *Batman* (green circle: feature present) while the current user and the user *Durham* disagreed (red circle: feature not present). Note that the aggregated circle is deep red and not just a normal average of the individual user opinions (which would result in a light red or neutral tone). This leads us to the competence metric and weighting of user opinions.

[5] `http://dojotoolkit.org/`
[6] Note that the screenshot has been shortened for clarity.

2.3 Competence metric

The competence metric in Skipforward is based on user similarity with regard to feature types. This is different from traditional content-based filtering as it takes into account opinions of different users; it also differs from standard collaborative filtering which does not support multiple opinions concerning different feature types per item. It is calculated with a modified variant of the constrained Pearson correlation [7] shown in Formula 1. Here, u_x denotes user x, $r_{x,i}$ denotes the applicability value user x assigned to item i for feature type t, I_{xy} is the set of co-rated items of users x and y, and $w_{xy,i}$ denotes the combined confidence of the statements concerning feature type f of users x and y and item i. Note that these calculations need to be repeated for each Skipforward feature type.

$$sim_t(u_x, u_y) = \frac{\sum_{i \in I_{xy}} w_{xy,i} r_{x,i} r_{y,i}}{\sqrt{\sum_{i \in I_{xy}} w_{x,i} (r_{x,i})^2 \sum_{i \in I_{xy}} w_{y,i} (r_{y,i})^2}} \quad (1)$$

For efficient calculation, an incremental algorithm has been implemented, only recalculating similarity values on changes, and only locally. $sim_t(u_x, u_y)$ is used for weighting user statements for aggregated features. Aggregated features represent all user statements concerning one feature type and one item. They are used in the user interface and in recommenders. To compute the competence metric, the following algorithm is used.

Algorithm 2.1: CalculateAllCorrelations()

for each $i \in I$
  $copyFeaturesToCache(i)$
  for each $u \in U$
    $doInferencing(u, i)$
for each $t_f \in T_f$
  $calculateMean(localuser, t_f)$
  for each $u \in U$
    $calculateMean(u, t_f)$
    $calculateCorrelation(localuser, u, t_f)$

To compute aggregated features per item and feature type according to the competence metric, another algorithm is used: For all feature types an item is annotated with, the aggregated feature for a specific feature type is represented by a weighted sum of the individual features' applicability for this feature type. Applicability weight is the respective user's competence regarding the feature type.

In effect, this means that for the aggregated feature, statements made by people who have been assigned a low competence value influence the outcome less than statements made by people with a high competence value.

2.4 Item recommender

The item recommender shows items similar to the current item (Figure 3). It is based on a similarity metric comparing aggregated features assigned to the two items in question. For the standard item recommender user interface shown on each item page, this is a straightforward distance metric comparing feature applicability. Additionally, there is also an advanced recommender which lets users freely select and weight individual feature types, implementing recommendation channels.

Similar items

Item	Similarity	Matches
Natural State	0.67	Fast story pac
Far & Deep	0.52	Fast story pac
Here We Are, Falling Through Shadows	0.45	Fantasy, Revi
Orchestral Manoeuvers in the Dark Matter	0.42	Fast story pac
Freedom(TM)	0.40	Fast story pac
Intellectual Property	0.39	Fast story pac
The Stories of Ibis	0.34	Fast story pac
When the Thorns are the Tips of Trees	0.33	Fantasy, Revi
After Everything Woke Up	0.32	Fast story pac
Fool's Errand	0.32	Fast story pac

Fig. 3. List of recommended items including similarity and feature type matches.

2.5 Annotation recommender

The annotation recommender is a utility for annotating items quickly. It shows a list of feature types that the current item has not yet been annotated with. Internally, for generating this list of feature types, item recommender output is reused. The annotations present for recommended items are compared with the annotations present for the current item. Any feature types for that a statement exists for the recommended item but not for the current item is presented to the user for quick annotation (Figure 4, right side). Therefore, annotating using the annotation recommender quickly improves the quality of results given by the item recommender.

Algorithm 2.2: GetRecFeatureTypes($curItem, recItems$)

for each $i \in recItems$
$recFeatureTypes.add(getFeatureTypes(i) \backslash getFeatureTypes(curItem))$

2.6 Expert recommender

In Figure 4 (left side), the HTML interface representing the expert recommender is shown. The expert recommender works on a per-feature type basis, recommending users who expressed similar opinions concerning a selected Skipforward feature type compared with the current user ($sim_t(u_x, u_y)$ in Formula 1). Since the current user agreed with the user *Durham* and disagreed with TV Tropes (cf. Figure 2), TV Tropes was assigned a smaller weight for aggregation of the feature type *Battle Butler* than *Durham*.

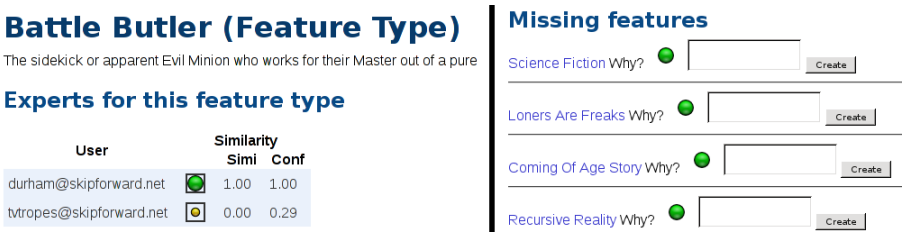

Fig. 4. Expert recommender for one feature type (left) — list of feature types recommended to annotate the current item with (right)

3 Related and Future Work

Skipforward is a unique amalgam of different technologies. Part of its functionality can be found in other systems; for example, *Revyu.com* [2] allows users to submit reviews which can be tagged with keywords. Absolute ratings can be given to items. Metadata is available as RDF/Linked Data; however, the tagging-based approach gives relatively shallow metadata only. In contrast to Skipforward, (formalized) discussions about annotations are not supported, and there is no personalization.

DBin [8] is similar to Skipforward but more generic and heavyweight. For example, it comes with its own messaging API, a plug-in architecture for its user interface, and needs dedicated metadata servers and a Java client whereas in Skipforward no server component is needed.

In terms of recommendation functionality, Skipforward implements a semantic hybrid filtering model. Similar approaches are discussed in [5] (the recommendation channels of Skipforward are similar to the *Collaboration via content* approach outlined in that paper) and [1] (clustering users based on domain concepts they are interested in—possible but not implemented in Skipforward yet).

Most tasks we want to pursue in the future are concerned with improving annotations and providing better recommendations. According to [5], recommender

approaches similar to those used in Skipforward in general cope well with annotation sparsity. However, the current implementation of the annotation recommender does not encourage overlapping annotations. I.e., several users should create feature instances for the same feature type and item to supply the competence metric with input, but the annotation recommender does not address this currently. We addressed this problem by using feature inference so far (i.e., inference using the feature type hierarchy is carried out), but this does not completely solve the problem. We plan to modify the annotation recommender accordingly. Another approach would be to introduce another recommender that explicitly targets annotation overlap to improve the competence metric.

A user study in the books domain will be carried out soon. A number of user interface improvements and additional statistics and recommender functionality will be added during that course.

4 Acknowledgments

This work has been supported by the German Federal Ministry of Education and Research (BMBF) in the context of the iGreen project (01IA08005A).

References

1. Cantador, I., and Castells, P. Multilayered semantic social network modeling by ontology-based user profiles clustering: Application to collaborative filtering. 2006, pp. 334–349.
2. Heath, T., and Motta, E. Revyu.com: A reviewing and rating site for the web of data. 2008, pp. 895–902.
3. Kiesel, M., and Grimnes, G. A. DBTropes—a linked data wrapper approach incorporating community feedback. In *EKAW 2010 Demo and Poster Abstracts. International Conference on Knowledge Engineering and Knowledge Management (EKAW-10), 17th International Conference on Knowledge Engineering and Knowledge Management, October 11-15, Lisbon, Portugal* (10 2010), J. V. O. Corcho, Ed., -. Best Poster.
4. Kiesel, M., and Schwarz, S. Skipforward—a lightweight ontology-based peer-to-peer recommendation system. In *International Semantic Web Conference (Demo)* (2008), C. Bizer and A. Joshi, Eds., vol. 401 of *CEUR Workshop Proceedings*, CEUR-WS.org.
5. Pazzani, M. J. A framework for collaborative, content-based and demographic filtering. *Artif. Intell. Rev. 13*, 5-6 (1999), 393–408.
6. Peis, E., del Castillo, J. M. M., and Delgado-López, J. A. Semantic recommender systems. analysis of the state of the topic. online, 2008.
7. Shardanand, U., and Maes, P. Social information filtering: algorithms for automating "word of mouth". In *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems* (New York, NY, USA, 1995), ACM Press/Addison-Wesley Publishing Co., pp. 210–217.
8. Tummarello, G., and Morbidoni, C. Collaboratively building structured knowledge with dbin: from del.icio.us tags to an rdfs folksonomy. *Workshop on Social and Collaborative Construction of Structured Knowledge at 16th International World Wide Web Conference (WWW2007)* (2007).

User's Food Preference Extraction for Personalized Cooking Recipe Recommendation

Mayumi Ueda*, Mari Takahata**, and Shinsuke Nakajima***

Kyoto University*
Yoshida Nihonmatsu-cho, Sakyo-ku, Kyoto, Kyoto 606–8501, Japan
Kyoto Sangyo University**
Motoyama, Kamigamo, Kita-Ku, Kyoto-City 603–8555, Japan
mayumi@mm.media.kyoto-u.ac.jp,* g0846741@cc.kyoto-su.ac.jp,**
nakajima@cse.kyoto-su.ac.jp***

Abstract. There are many websites and researches that involve cooking recipe recommendation. However, these websites present cooking recipes on the basis of entry date, access frequency, or the recipe's user ratings. They do not reflect the user's personal preferences. We have proposed a personalized recipe recommendation method that is based on the user's food preferences. For extracting the user's food preferences, we use his/her recipe browsing and cooking history. In this paper, we present a method for extracting the user's preferences. In the experimental results, extracting the user's favorite ingredients were detected with a 60 to 83% of precision. And extracting the unfavorite ingredients were detected with 14.7% of precision, and 58% of recall. Furthermore, the F-measure value for extraction of favorite ingredients was 60.8% when we focused on the top 20 ingredients.

Key words: user's food preferences, preference extraction, recipe recommendation, cooking and browsing history

1 Introduction

As a result of the lifestyle-related disease epidemic, dietary life is now attracting attention. Good eating habits are important for maintaining a healthy life. However, menu planning requires one to take various factors into consideration, such as the nutritional value, food in stock, food preferences, and cost. Thus, people need to expand a lot of effort toward planning their daily menu. Against this background, a number of cooking websites comprising various food recipes have recently been launched, such as Cookpad[1] and Yahoo! Recipe[2]. Many people refer to these websites when planning their menu. Cookpad contains 900,000 recipes and has 10,000,000 monthly users[3]. This data reflects the high demand for recipe-providing services. However, these websites do not reflect user's preferences and conditions, although these two factors need to be considered if the goal is to provide high-satisfactory recipes.

Furthermore, several researches on cooking recipe recommendation for menu planning support have been conducted in the past. Mino et al. propose a method

that takes the user's schedule into consideration[4]. This paper defines the evaluation value of either the intake or consumption calories that are assigned to each event in the user's schedules. Karikome et al. propose a system that helps users plan nutritionally balanced menus and visualize their dietary habits[5]. Their system calculates the nutritional value of each dish, and records this information in the form of a dietary log. Next, the system recommends recipes foster sound nutrition. Freyne et al. show the results of their investigation in which they compare three recommendation strategies: content-based, collaborative, and hybrid[6].

In these circumstances, we have proposed a recipe recommendation method based on the user's food preferences[7]. Our method breaks recipes down into their ingredients, and scores them on the basis of the frequency of use and specificity of the ingredients. Furthermore, our proposed system does not recommend dishes that are similar to the food the users have eaten over the past few days on the grounds that people do not want to eat similar dishes iteratively. Moreover, our system does not require any particular action on the user's past to reflect his/her food preferences: it estimates the user's food preferences automatically through his/hers recipe browsing and cooking history. In this paper, we present a method for extracting the user's preferences.

This paper is structured as follows. Section 2 describes the method of scoring recipes and extracting user's preferences. Section 3 shows experimental results, using precision and recall. Section 4 shows concludes the paper.

2 Scoring Recipes and Extracting User's Preferences

In the recent years, concern over various health issues, such as lifestyle-related diseases and diets, has been growing. It has also been noted that picky eating is one of the main reasons causing these health issues. However, people do not want to eat food that they dislike even if it perfectly addresses their nutritional needs. They hope to derive essential nutrition solely from their favorite foods. We conducted questionnaire to the 20 men and women in their 20s to 40s to survey the key considerations for menu planning. According to the results of the questionnaire, people consider the following elements (1) food preferences, (2) nutritional balance and calories, (3) ingredients they have in stock or they can procure easily, (4) easily to cook, and (5) mood. Therefore, in this paper, we focus on (1) food preferences and try to extract user's food preferences.

2.1 Preferences for Ingredients

We express the user's food preferences I_k by using in the form of the following Eq.(1).

$$I_k = I_k^+ + I_k^- \tag{1}$$

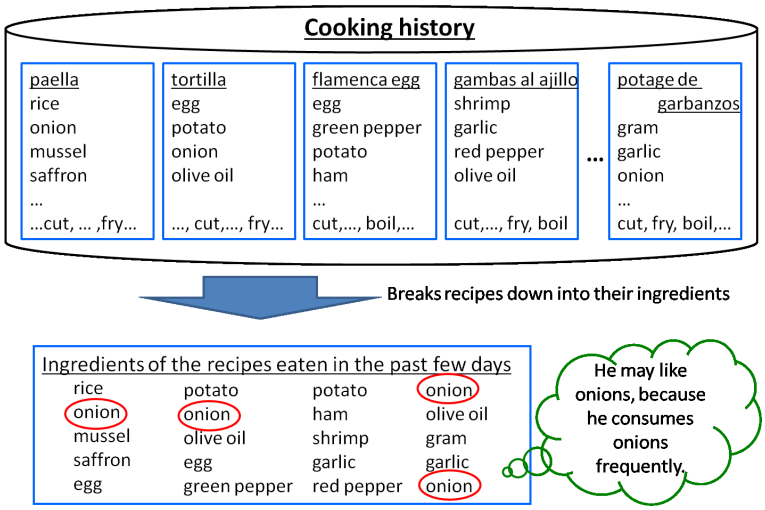

Fig. 1. Extracting the favorite ingredients using cooking history.

User's favorite ingredients Fig.1 shows the key idea behind estimating user's favorite ingredients by his/her cooking history. Our method considers the ingredient that the user eats repeatedly as his/her favorite ingredients. It breaks recipes down into their ingredient as the outset and calculates *the score of ingredients* I_k^+ by incorporating the frequency of use of the ingredients in the dishes that the target user has eaten(FF_k: Foodstuff Frequency) as well as the specificity of ingredients(IRF_k: Inverted Recipe Frequency) into Eq.(2). This equation is based on the idea of TF-IDF.

$$I_k^+ = FF_k \times IRF_k \tag{2}$$

For estimating the user's favorite ingredients by using *the frequency of use of ingredient* k (FF_k), we utilize *the simple frequency of use of ingredient* k (F_k) during a definite period D, as shown in Eq.(3).

$$FF_k = \frac{F_k}{D} \tag{3}$$

Then, we calculate —it the specificity of ingredient k (IRF_k) using *the total number of recipe* (M) and *the number of recipes that contain ingredient* k (M_k), as shown in Eq.(4).

$$IRF_k = log\frac{M}{M_k} \tag{4}$$

User's disliked ingredients We consider that user's food preferences are also influenced by his/her disliked ingredients. We estimate the user's disliked ingredients, by considering the ingredients in the recipes that he/she has never

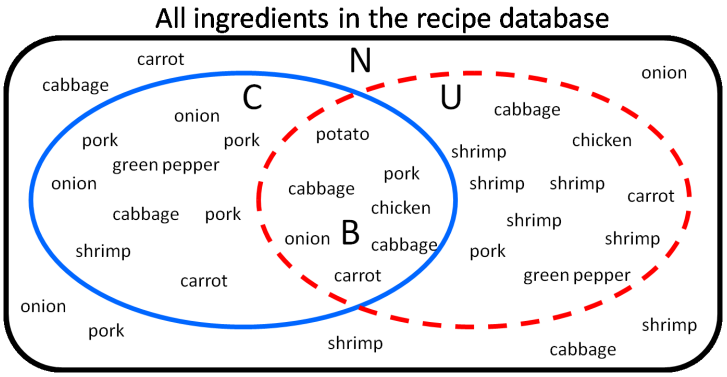

Fig. 2. Extracting the disliked ingredients using browsing and cooking history.

cooked, even if he/she has browsed the recipe details. Fig.2 shows the estimating method for user's disliked ingredients through the user's recipe browsing and cooking history. N corresponds to the set of ingredients in the recipes that the user has not browse. C corresponds to the set of ingredients in the recipes that the user has cooked over the past few days. U corresponds to the set of ingredients in the recipes that the user has not cooked, even if he/she has browse them completely. For example, "shrimp" in Fig.2 corresponds to the user's disliked ingredient. We calculate *the score of disliked ingredient* k (I_k^-) in Eq.(5).

$$I_k^-(x) = \begin{cases} 0 & (0 < \frac{2|U_k|}{|A_k|} \leqq 0.5) \\ (\frac{2|U_k|}{|A_k|} - 1)^x & (0.5 < \frac{2|U_k|}{|A_k|} \leqq 1) \end{cases} \tag{5}$$

$|U_k|$ denotes the presence of ingredient k in U and $|A_k|$ denotes the presence of ingredient k in the recipe database. We should investigate the ratio or frequency of the user's avoidance of the ingredient that he/she dislikes, because he/she will use the ingredients that he/she does not like. x denotes the ratio or frequency of avoiding the ingredients, and we plan to verify x through some preliminary experiments.

2.2 Recipe Scoring

Our method scores cooking recipes in accordance with the estimation results regarding favorite/disliked ingredients, and then provides recipes in decreasing order of the scores. In general, people do not like eating dishes similar to those they have eaten in the past few days. Therefore, our method weights recipes to avoid the repetition of similar dishes. *The score of cooking recipes* are defined as shown in Eq.(6).

$$Score(R) = \sum_{k \in R} I_k - \alpha \sum_{d=1} (w_d \cdot sim\,(R, R_d)) \tag{6}$$

d denotes the weight for avoiding repeating similar dishes iteratively. $sim\,(R, R_d)$ denotes the similarities between the considered recipe R and the recipe of the dish eaten d days ago R_d. The weight w_d for avoiding similar dishes eaten d days ago is defined as shown in Eq.(7).

$$w_d = 1 - \frac{d-1}{7} \qquad (1 \leqq d \leqq 7) \tag{7}$$

3 Evaluation of the Accuracy of Extracting User's Food Preference

3.1 Experimental Condition

In order to verify the extracting accuracy of the user's food preferences, we conducted simple experiments. We used 100 recipes extracted from Cookpad[1] that is a most popular recipe search website in Japan. We used randomly selected recipes categorized as main dish.

We conducted experiment as follows.

1. We present a list of 10 recipe titles to subjects.
2. He/She chooses recipes which he/she would like to browse completely, such as ingredients, procedures, and so on.
3. He/She chooses one recipe that he/she would like to cook.
4. Repeat this sequence(Step 1 to 3) 10 times.

We gathered each user's browsed recipes and recipes he/she would like to cook through the above procedure. 6 men and women in their 20s to 40s participated in this experiment as subjects.

We calculated the specificity of ingredient k (IRF_k) in the target 100 recipes, using Eq.(4). Table 1 shows the examples of IRF_k. Furthermore, we collected the labeled data as the user's preferences via questionnaire. Responses are coded on a 6-point scale, ranging from "love" to "hate".

3.2 Evaluation of Extracting the Favorite Ingredients

We evaluated the extraction accuracy of the user's favorite ingredients. We extracted individual user's favorite ingredients by calculating I_k^+, using Eq.(2). For evaluating the accuracy, we calculated precision, recall, and F-measure for top N ingredients which sorted by I_k^+. The results are shown in Table 2 and Fig.3. The precision, recall, and F-measure were calculated by the number of extracted user's favorite ingredients in the top N (E) and the number of user's favorite ingredients via questionnaire (Q), as follows.

$$Precision = \frac{E}{N}$$
$$Recall = \frac{E}{Q}$$

Table 1. Examples of the specificity of ingredient k (IRF_k)

ingredient	IRF_k	ingredient	IRF_k	ingredient	IRF_k
pumpkin	1.70	oyster	1.52	white wine	1.10
cucumber	1.70	eggplant	1.40	mayonnaise	1.01
burdock root	1.70	parsley	1.40	tomato	1.00
konjac	1.70	honey	1.40	sesame oil	0.96
snow crab	1.70	beef	1.40	carrot	0.92
green pepper	1.70	shrimp	1.30	butter	0.92
yellowtail	1.70	bacon	1.30	milk	0.92
lettuce	1.70	Japanese radish	1.22	mushroom	0.89
yam	1.70	minced meat	1.22	egg	0.77
soy milk	1.70	potato	1.15	pork	0.74
pickled plum	1.70	miso	1.15	ginger	0.68
chinese chive	1.52	lemon	1.15	garlic	0.57
bean sprout	1.52	tofu	1.15	cibol	0.57
salmon	1.52	cabbage	1.10	chicken	0.54
Colocasia esculenta	1.52	cheese	1.10	onion	0.44

Table 2. Elicitation accuracy of user's food preferences

N	Recall	Precision	F-measure
1	0.045	0.833	0.086
3	0.124	0.778	0.213
5	0.196	0.733	0.309
10	0.410	0.767	0.535
15	0.521	0.656	0.580
20	0.609	0.607	0.608

$$F\text{-}measure = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \quad (N = 1, 3, 5, 10, 15, 20) \tag{8}$$

As shown in Table 2, when we focused on the only top one ingredient ($N = 1$), our method extracted with precision of 83.3%. However, at the same time, the recall was only 4.5%. When we focused on the top 20 ingredients ($N = 20$), the recall was increased to 61%. Since the average number of individual user's favorite ingredients was 19.2, the recall was not enough value when we focused on a few ingredients such as $N = 1, 3, 5$. It was found from the result that even if our system focused on the top 20 ingredients ($N = 20$), the precision did not reduce very much. Furthermore, the highest value of F-measure, at this experiment, was 60.8% when we focused on the top 20 ingredients (Table 2). Therefore, the results show that our system should focus on the top 20 ingredients sorted by I_k^+ for recipe recommendation.

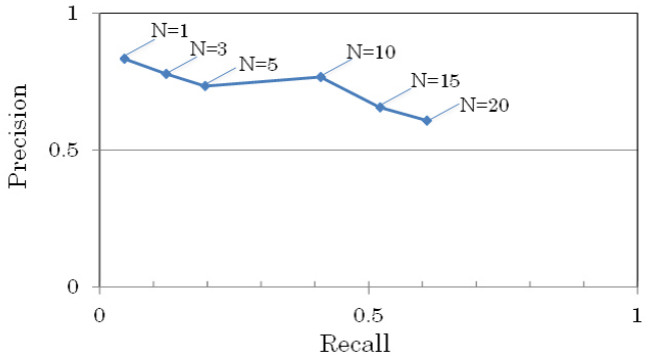

Fig. 3. The precision and recall for extracting the favorite ingredients.

3.3 Evaluation of Extracting the Disliked Ingredients

We evaluated the accuracy of extracting user's disliked ingredients. We extracted individual user's disliked ingredients by calculating I_k^-, using Eq.(5). The precision of the extracting disliked ingredients was 14.7%, and the recall was 58.3%. In this experiment, we considered that the ingredients which were contained only in U, were the user's disliked ingredients. In other words, in this experiment, our method estimated the user's disliked ingredients which were in the recipes that he/she has never cooked, even if he/she has browsed the complete recipe.

$$disliked\ ingredients = \{U \cap \overline{C}\} \tag{9}$$

In this experiment, we could not extract satisfactory accuracy for disliked ingredients, because of the lack of the number of experiments. Since the number of experiments was not enough, our method determined that the ingredient which happened to include in U was his/her disliked ingredient. We consider that our method can improve the accuracy of extracting favorite ingredients, by increasing the number of experiments.

4 Conclusion

In this paper, we presented a method for extracting the user's food preferences for recipe recommendation. Our method estimates a user's preferences from his/her past actions, such as through their recipe browsing and menu planning history. For extracting the preferences, our method breaks recipes down into their ingredients and scores the recipes using the frequency and specificity of ingredients. Since our method can estimate the preferences through their browsing and cooking history, the user convey his/her preferences to the system without having to carry out any particular operation. Furthermore, the user can convey the changes in his/her preferences to the system on a daily basis.

In order to verify the extracting accuracy of the user's food preferences, we conducted simple experiments. In the experimental results, extracting the user's favorite ingredients were detected with a 60 to 83% of precision. And the F-measure was 60.8% when we focused on the top 20 ingredients. Since the average number of user's favorite ingredients was 19.2, our system should focus on the top 20 ingredients sorted by I_k^+ to score recipes for recommendation. And extracting the disliked ingredient were detected with 14.7% of precision, and 58% of recall. In this time, we could not extract satisfactory accuracy. However, we consider that our method can improve the accuracy of extraction, by increasing the number of experiments.

As future work, we plan to consider the ingredient ontology for estimating favorite/disliked ingredients. Furthermore, we plan to investigate the various weights. For example, we plan to verify x in Eq.(5), the ratio or frequency of avoiding the ingredients, through experiments. Moreover, we want to investigate the length of time for which the system should avoid recommending similar dishes for d in Eq.(6),(7), and the degree of similarity that the system should consider while refraining from recommending similar dishes for α in Eq.(6).

Acknowledgments

This work is supported in part by Kyoto University Global COE Program: Informatics Education and Research Center for Knowledge-Circulating Society (MEXT Global COE Program) , and the MEXT Grant-in-Aid for Scientific Research(C) (#23500140).

References

1. Cookpad. http://cookpad.com/, (Accessed 12 August 2011)
2. Yahoo! Recipe. http://recipe.gourmet.yahoo.co.jp/, (Accessed 12 August 2011)
3. Choose dishes using websites of cooking. (in Japanese), http://otona.-yomiuri.co.jp/news/news110118_03.htm, (Accessed 12 August 2011)
4. Yoko Mino and Ichiro Kobayashi. Recipe Recommendation for a Diet Considering a User's Schedule and the Balance of Nourishment. Proceedings of IEEE International Conference on Intelligent Computing and Intelligent Systems 2009, pp.383-387. 2009.
5. Shihono Karikome and Atsushi Fujii. A System for Supporting Dietary Habits: Planning Menus and Visualizing Nutritional Intake Balance. Proceedings of the 4th International Conference on Ubiquitous Information Management and Communication, pp.386-391. 2009.
6. Jill Freye and Shlomo Berkovsky. Intelligent Food Planning: Personalized Recipe Recommendation. In IUI(2010) 321-324.
7. Mayumi Ueda, Mari Takahata and Shinsuke Nakajima. Recipe Recommendation Method Based on User's Food Preferences. Proceedings of the IADIS International Conference on e-Society 2011, pp.591-594. 2011.

Finding similar research papers using language models

Germán Hurtado Martín[1,2], Steven Schockaert[2], Chris Cornelis[2], and Helga Naessens[1]

[1] Dept. of Industrial Engineering, University College Ghent, Ghent, Belgium
[2] Dept. of Applied Math. and Comp. Science, Ghent University, Ghent, Belgium

Abstract. The task of assessing the similarity of research papers is of interest in a variety of application contexts. It is a challenging task, however, as the full text of the papers is often not available, and similarity needs to be determined based on the papers' abstract, and some additional features such as authors, keywords, and journal. Our work explores the possibility of adapting language modeling techniques to this end. The basic strategy we pursue is to augment the information contained in the abstract by interpolating the corresponding language model with language models for the authors, keywords and journal of the paper. This strategy is then extended by finding topics and additionally interpolating with the resulting topic models. These topics are found using an adaptation of Latent Dirichlet Allocation (LDA), in which the keywords that were provided by the authors are used to guide the process.

Keywords: Similarity, Language modeling, Latent Dirichlet allocation

1 Introduction

Due to the rapidly growing number of published research results, searching for relevant papers can become a tedious task for researchers. In order to mitigate this problem, several solutions have been proposed, such as scientific article recommender systems [2, 8] or dedicated search engines such as Google Scholar. At the core of such systems lies the ability to measure to what extent two papers are similar, e.g. to find out whether a paper is similar to papers that are known to be of interest to the user, to explicitly allow users to find "Related articles" (as in Google Scholar), or to ensure that the list of search results that is presented to the user is sufficiently novel and diverse [3]. To find out whether two articles are similar, content-based approaches can be complemented with collaborative filtering techniques (e.g. based on CiteULike.org or Bibsonomy.org) or citation analysis (e.g. PageRank, HITS, etc.). While the latter are well-studied, content-based approaches are usually limited to baseline techniques such as using the cosine similarity between vector representations of the abstracts.

Comparing research papers is complicated by the fact that their full text is often not publicly available, and only the abstract along with some document features such as keywords, authors, or journal can be accessed. The challenge

thus becomes to make optimal use of this limited amount of information. Hurtado et al. [7] investigated the impact of individual document features within the vector space model. Their main conclusion was that baseline methods using only the abstract could not be improved significantly by enriching them with other features. Language modeling techniques, however, have been shown to perform well for comparing short text snippets [6, 10].

Our goal in this paper is therefore to explore how language models can be used to compare research paper abstracts, how they can best make use of the other document features, and whether they are a more reasonable choice than a vector space model based approach for this task. In particular, we combine two ideas to address these questions. On the one hand, we consider the idea of estimating language models for document features such as keywords, authors, and journal, and derive a language model for the article by interpolating them (an idea which has already proven useful for expert finding [11]). On the other hand, we apply LDA to discover latent topics in the documents, and explore how the keywords can help to improve the performance of standard LDA.

2 Research paper similarity

In this section we review and introduce several methods to measure article similarity, based on the information commonly available for a research paper: abstract, keywords, authors, and journal.

2.1 Vector space model

The similarity of two papers can easily be measured by comparing their abstracts in the vector space model (method *abstract* in the result tables): each paper is represented as a vector, in which each component corresponds to a term occurring in the collection. To calculate the weight for that term the standard tf-idf approach is used, after removing stopwords. The vectors $\mathbf{d_1}$ and $\mathbf{d_2}$ corresponding to different papers can then be compared using standard similarity measures such as the cosine (*cos*), generalized Jaccard (*g.jacc*), extended Jaccard(*e.jacc*), and Dice (*dice*) similarity, using them as in [7]. Alternatively, we also consider a vector representation where the abstract is completely ignored, and where there is one component for each keyword, with the weights calculated analogously as in the tf-idf model (method *keywords*).

In [7] an alternative scheme for using the keywords has been proposed, which does not ignore the information from the abstract. This scheme was referred to as explicit semantic analysis (ESA) since it is analogous to the approach from [4]. The idea is, departing from a vector $\mathbf{d}$ obtained by method *abstract*, to define a new vector representation $\mathbf{d_E}$ of this paper, with one component for every keyword k appearing in the collection. The weights of $\mathbf{d_E}$'s components are defined by $w_k = \mathbf{d} \cdot \mathbf{q_k}$, where the vector $\mathbf{q_k}$ is formed by the tf-idf weights corresponding to the concatenation of the abstracts of all papers to which keyword k was assigned. This method is called *ESA-kws* in our experiments below. Similar

methods are considered in which vector components refer to authors (*ESA-aut*) or to journals (*ESA-jou*). For efficiency and robustness, only authors are considered that appear in at least 4 papers in the *ESA-aut* method, and only keywords that appear in at least 6 papers in the *ESA-kw* method. Higher thresholds would exclude too many keywords and authors, while lower thresholds would result in a high computational cost due to the large number of values in each vector.

2.2 Language modeling

A different approach is to estimate unigram language models [9] for each document, and calculate their divergence. A document d is then assumed to be generated by a given model D. This model is estimated from the terms that occur in the abstract of d (and the rest of the abstracts in the collection). Using Jelinek-Mercer smoothing, the probability that model D generates term w is given by:

$$P^*(w|D) = \lambda P(w|d) + (1-\lambda)P(w|\mathcal{C}) \tag{1}$$

where $\mathcal{C}$ is the whole collection of abstracts. The probabilities $P(w|d)$ and $P(w|\mathcal{C})$ are estimated using maximum likelihood, e.g. $P(w|d)$ is the fraction of occurrences of term w in the abstract of document d. Once the models D_1 and D_2 corresponding to two documents d_1 and d_2 are estimated, we measure their difference using the well-known Kullback-Leibler divergence, defined by

$$KLD(D_1||D_2) = \sum_w D_1(w) log \frac{D_1(w)}{D_2(w)} \tag{2}$$

If a symmetric measure is desired, Jensen-Shannon divergence could alternatively be used.

Language model interpolation The probabilities in the model of a document are thus calculated using the abstracts in the collection. However, given the short length of the abstracts, we should make maximal use of all the available information, i.e. also consider the keywords k, authors a, and journal j. In particular, the idea of interpolating language models (also used for example in [11]), which underlies Jelinek-Mercer smoothing, can be generalized:

$$P^*(w|D) = \lambda_1 P(w|d) + \lambda_2 P(w|k) + \lambda_3 P(w|a) + \lambda_4 P(w|j) + \lambda_5 P(w|\mathcal{C}) \tag{3}$$

with $\sum_i \lambda_i = 1$. In order to estimate $P(w|k)$, $P(w|a)$, and $P(w|j)$, we consider an artificial document for each keyword k, author a and journal j corresponding to the concatenation the abstracts where k, a and j occur, respectively. Then, the probabilities are estimated using maximum likelihood, analogously to $P(w|d)$. Since a document may contain more than one author and one keyword, we define $P(w|k)$ and $P(w|a)$ as:

$$P(w|k) = \frac{1}{n}\sum_{i=1}^{n} P(w|k_i) \tag{4}$$

$$P(w|a) = \frac{1}{m}\sum_{j=1}^{m} P(w|a_j) \tag{5}$$

where n and m are the number of keywords and authors in the document.

Latent Dirichlet Allocation Two conceptually related abstracts may contain different terms (e.g. synonyms, misspellings, related terms), and may therefore not be recognized as similar. While this is a typical problem in information retrieval, it is aggravated here due to the short length of abstracts. To cope with this, methods can be used that recognize which topics are covered by an abstract. The idea is that topics are broader than keywords, but still sufficiently discriminative to yield a meaningful description of the content of an abstract. This topical information is not directly available; however, it can be estimated by using Latent Dirichlet Allocation (LDA) [1].

The idea behind LDA is that documents are generated by a (latent) set of topics, which are modeled as a probability distribution over terms. To generate a document, a distribution over those topics is set, and then, to generate each word w in the document, a topic z is sampled from the topic distribution, and w is sampled from the word distribution of the selected topic. In other words, the set of distributions ϕ over the words in the collection and the set of distributions θ over all the topics must be estimated. To do so, we use LDA with Gibbs sampling [5]. We can then estimate these probabilities as:

$$P(w|z) = \hat{\phi}_z^{(w)} = \frac{{n_z}^{(w)} + \beta}{{n_z}^{(\cdot)} + W\beta} \quad (6) \qquad P(z|t) = \hat{\theta}_z^{(d)} = \frac{{n_z}^{(d)} + \alpha}{{n_\cdot}^{(d)} + T\alpha} \quad (7)$$

where t is the LDA model obtained with Gibbs sampling, W is the number of words in the collection, and T is the number of topics. Parameters α and β intuitively specify how close (6) and (7) are to a maximum likelihood estimation. The count ${n_z}^{(w)}$ is the number of times word w has been assigned to topic z, while ${n_z}^{(d)}$ is the number of times a word of document d has been assigned to topic z. Finally, ${n_z}^{(\cdot)}$ is the total number of words assigned to topic z, and ${n_\cdot}^{(d)}$ is the total number of words of document d assigned to any topic. All these values are unknown a priori; however, by using Gibbs sampling they can be estimated.

To find the underlying topics, the LDA algorithm needs some input, namely the number T of topics to be found. Based on preliminary results, we set $T = K/10$, where K is the number of keywords that are considered. The topics that are obtained from LDA can be used to improve the language model of a given document d. In particular, we propose to add $P(w|t)$ to the right-hand side of (3), with the appropriate weight λ. $P(w|t)$ reflects the probability that term w is generated by the topics underlying document d. It can be estimated by considering that:

$$P(w|t) = \sum_{i=1}^{T} P(w|z_i) \times P(z_i|t) \quad (8)$$

This method is referred to as *LM0* in the result tables.

The method *LM0* can be improved by taking advantage of the keywords that have been assigned to each paper. In particular, we propose to initialize the topics by determining T clusters of keywords using k-means. Then, a document c is created for every cluster. This artificial document is the concatenation of

the abstracts of all papers to which some keyword from the cluster was assigned. Once these documents c are made, initial values for the parameters ${n_z}^{(w)}$, ${n_z}^{(d)}$, and ${n_z}^{(\cdot)}$ in (6) and (7) can be retrieved from them: ${n_z}^{(w)}$ is initialized with the number of occurrences of w in artificial document c_z, ${n_z}^{(d)}$ with the number of words of document d occurring in c_z, and ${n_z}^{(\cdot)}$ with the total number of words in c_z. Parameter ${n_\cdot}^{(d)}$ is independent from the clustering results as it takes the value of the total number of words in document d. We furthermore take $\alpha = 50/T$ and $\beta = 0.1$. Subsequently, we can either work directly with these initial values (i.e., use them in (6) and (7): method *LM1* in the results tables), or we can apply Gibbs sampling (method *LM2*). In this last case, the values resulting from the clustering are only used to initialize the sampler, as an alternative to the random values used normally.

3 Experimental evaluation

3.1 Experimental Set-Up

To build a test collection and evaluate the proposed methods, we downloaded a portion of the ISI Web of Science[3], consisting of files with information about articles from 19 journals in the Artificial Intelligence domain. These files contain, among other data, the abstract, authors, journal, and keywords freely chosen by the authors. A total of 25964 paper descriptions were retrieved, although our experiments are restricted to the 16597 papers for which none of the considered fields is empty.

The ground truth for our experiments is based on annotations made by 3 experts. First, 100 articles with which at least one of the experts was sufficiently familiar were selected. Then, using tf-idf with cosine similarity, the 30 most similar articles in the test collection were found for each of the 100 articles. Each of those 30 articles were manually tagged by the expert as similar or dissimilar. To evaluate the performance of the methods, each paper $\mathbf{p}$ is thus compared against 30 others[4], some of which are tagged as similar. Similarity measures can then be used to rank the 30 papers, such that ideally the papers similar to $\mathbf{p}$ appear at the top of the ranking. In principle, we thus obtain 100 rankings. However, due to the fact that some of the lists contained only dissimilar articles, and that sometimes the experts were not certain about the similarity of some items, the initial 100-article set was reduced to 89 rankings. To evaluate these rankings, we use mean average precision (MAP) and mean reciprocal rank (MRR).

3.2 Results

Tables 1 and 2 summarize the results of the experiment. The λ configurations in the first column of Table 2 are shown in the order $\lambda_{abstract}$, $\lambda_{keywords}$, $\lambda_{authors}$,

[3] http://apps.isiknowledge.com

[4] During the annotation process it was also possible to tag some items as "Don't know" for those cases where the expert had no certainty about the similarity. These items are ignored and therefore some papers are compared to less than 30 others.

Table 1. Vector space model

	MAP				MRR			
	cos	*dice*	*e.jacc*	*g.jacc*	*cos*	*dice*	*e.jacc*	*g.jacc*
abstract	0.492	0.492	0.492	0.543	0.689	0.689	0.689	0.725
keywords	0.496	0.497	0.495	0.481	0.736	0.741	0.738	0.715
ESA-kws	0.544	0.544	0.543	0.537	0.701	0.701	0.701	0.717
ESA-aut	0.553	0.553	0.553	0.554	0.726	0.726	0.726	0.727
ESA-jou	0.414	0.414	0.414	0.422	0.574	0.574	0.574	0.581

Table 2. Language modeling

	MAP			MRR		
λ-config.	LM0	LM1	LM2	LM0	LM1	LM2
0.9 0 0 0 0	0.596	0.596	0.596	0.759	0.759	0.759
0 0.9 0 0 0	0.533	0.533	0.533	0.732	0.732	0.732
0 0 0.9 0 0	0.520	0.520	0.520	0.664	0.664	0.664
0 0 0 0.9 0	0.294	0.294	0.294	0.395	0.395	0.395
0 0 0 0 0.9	0.464	0.365	0.522	0.620	0.500	0.693
0.7 0 0 0 0.2	0.611	0.602	0.622	0.754	0.771	0.766
0.5 0 0 0 0.4	0.611	0.604	0.636	0.764	0.769	0.778
0.2 0 0 0 0.7	0.570	0.607	0.613	0.731	0.764	0.759
0.7 0.2 0 0 0	0.593	0.593	0.593	0.786	0.786	0.786
0.5 0.4 0 0 0	0.575	0.575	0.575	0.781	0.781	0.781
0.2 0.7 0 0 0	0.544	0.544	0.544	0.736	0.736	0.736
0.4 0.1 0 0 0.4	0.651	0.650	**0.671**	0.820	0.821	0.820
0.1 0.4 0 0 0.4	0.582	0.560	0.593	0.754	0.754	0.775
0.4 0.4 0 0 0.1	0.581	0.578	0.588	0.768	0.775	0.782
0.3 0.3 0 0 0.3	0.606	0.591	0.618	0.783	0.788	0.797
0.3 0.1 0.1 0.1 0.3	0.635	0.617	0.654	0.784	0.780	0.784
0.4 0.1 0.1 0 0.3	0.633	0.620	0.656	0.779	0.779	0.800
0.4 0.1 0 0.1 0.3	0.657	0.644	0.667	**0.834**	0.814	0.824

$\lambda_{journal}$, λ_{topics}. We fixed the sum of these weights to 0.9, and set the general smoothing factor (λ_5 in (3)) to 0.1.

The main conclusion that we can draw from these results is that language models are indeed capable of yielding a substantial improvement over all of the vector space approaches. The first block of Table 2 summarizes the results obtained with language models that only use one of the features. We find that language models which only use the abstract significantly[5] improve the performance of the most traditional vector space methods (*abstract*). Models uniquely based on other features can perform slightly better than *abstract*, but these

[5] In this work we consider an improvement to be significant when $p < 0.05$ for the paired t-test.

improvements were not found to be significant. However, these results are still useful as an indication of the amount of information contained in each of the features: language models based exclusively on keywords or on authors perform comparable to the method *abstract*. Using topics only yields such results when *LM2* is used, while the information contained in the journal is clearly poorer.

In the second block of Table 2 we examine different combinations of two features: abstract with topics on the first three lines, and abstract with keywords on the last three. These results confirm that the abstract contains the most information, and should be assigned a high weight. On the other hand, we can observe how the topics, when combined with the abstract, yield a better MAP score. In particular, the MAP score for the LM2 configuration on the second (resp. third) line of the second block are significantly better than the LM2 score on the fifth (resp. sixth) line.

The third block of Table 2 shows the results of combining abstract and topics, with keywords, authors, and journal. It is clear that giving a small weight to keywords is beneficial, as it leads to the highest scores, which are significantly better than all configurations of the second block. For authors and journal, however, we do not find a substantial improvement. In Fig. 1 we further explore the importance of the abstract and the topics. We set the weight of the keywords to a fixed value of 0.1, and the remaining weight of 0.8 is divided between abstract and topics. What is particularly noticeable is that ignoring the abstract is penalized stronger than ignoring the topics, but the optimal performance is obtained when both features are given approximately the same weight.

Finally, we can note from Table 2 that *LM1* cannot improve *LM0*, but clear differences in MAP scores can be observed between *LM0* and *LM2*. These latter differences are significant for all configurations in the third block, and the three first configurations in the second block.

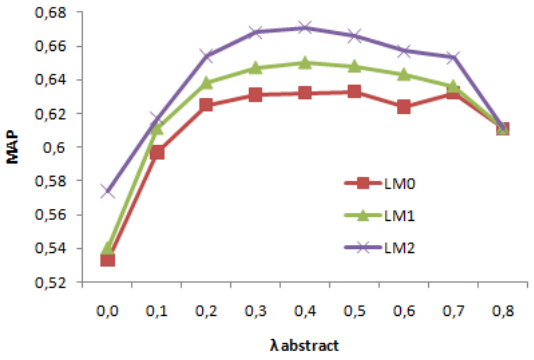

Fig. 1. Importance of abstract vs. topics

4 Conclusion

We have shown how language models can be used to compare research paper abstracts and how their performance for this task can be improved by using other available document features such as keywords, authors, and journal. In particular, language models have proven more suitable in this context than any of the vector space methods we considered. We have also explored how LDA could be used in this case to discover latent topics, and a method has been proposed to effectively exploit the keywords to significantly improve the performance of the standard LDA algorithm.

References

1. D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.
2. T. Bogers and A. van den Bosch. Recommending scientific articles using CiteULike. In *Proc. of the 2008 ACM Conf. on Recommender Systems*, pages 287–290, 2008.
3. C. L. Clarke, M. Kolla, G. V. Cormack, O. Vechtomova, A. Ashkan, S. Büttcher, and I. MacKinnon. Novelty and diversity in information retrieval evaluation. In *Proc. of the 31st Annual International ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 659–666, 2008.
4. E. Gabrilovich and S. Markovitch. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In *Proc. of the 20th International Joint Conf. on Artifical Intelligence*, pages 1606–1611, 2007.
5. T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proc. of the National Academy of Sciences*, 101(Suppl. 1):5228–5235, 2004.
6. L. Hong and B. D. Davison. Empirical study of topic modeling in Twitter. In *Proc. of the First Workshop on Social Media Analytics*, pages 80–88, 2010.
7. G. Hurtado Martín, S. Schockaert, C. Cornelis, and H. Naessens. Metadata impact on research paper similarity. In *Proc. of the 14th European Conf. on Research and Advanced Technology for Digital Libraries*, pages 457–460, 2010.
8. S. M. McNee, I. Albert, D. Cosley, P. Gopalkrishnan, S. K. Lam, A. M. Rashid, J. A. Konstan, and J. Riedl. On the recommending of citations for research papers. In *Proc. of the 2002 ACM Conf. on Computer Supported Cooperative Work*, pages 116–125, 2002.
9. J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proc. of the 21st Annual International ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 275–281, 1998.
10. X. Quan, G. Liu, Z. Lu, X. Ni, and L. Wenyin. Short text similarity based on probabilistic topics. *Knowledge and Information Systems*, 25:473–491, 2010.
11. J. Zhu, X. Huang, D. Song, and S. Rüger. Integrating multiple document features in language models for expert finding. *Knowledge and Information Systems*, 23:29–54, 2010.

A Dimensionality Reduction Approach for Semantic Document Classification

Oskar Ahlgren, Pekka Malo, Ankur Sinha, Pekka Korhonen & Jyrki Wallenius

Aalto University School of Economics
P.O. Box 21210, FI-00076 AALTO, FINLAND

Abstract. The curse of dimensionality is a well-recognized problem in the field of document filtering. In particular, this concerns methods where vector space models are utilized to describe the document-concept space. When performing content classification across a variety of topics, the number of different concepts (dimensions) rapidly explodes and as a result many techniques are rendered inapplicable. Furthermore the extent of information represented by each of the concepts may vary significantly. In this paper, we present a dimensionality reduction approach which approximates the user's preferences in the form of value function and leads to a quick and efficient filtering procedure. The proposed system requires the user to provide preference information in the form of a training set in order to generate a search rule. Each document in the training set is profiled into a vector of concepts. The document profiling is accomplished by utilizing Wikipedia-articles to define the semantic information contained in words which allows them to be perceived as concepts. Once the set of concepts contained in the training set is known, a modified Wilks' lambda approach is used for dimensionality reduction by ensuring minimal loss of semantic information.

1 Introduction

Most information retrieval systems are based on free language searching, where the user can compose any ad hoc query by presenting a list of keywords or a short phrase to describe a topic. The popularity of phrase-based methods can largely be explained by their convenience for the users. However, the ease of usage comes with a few drawbacks as well. While exploring a new topic or searching within an expert domain with specialized terminology it can be surprisingly hard to find the right words for getting the relevant content. To cope with the ambiguity of the vocabulary, concept-based document classification techniques have been proposed, as concepts by definition cannot be ambiguous. However, the use of concepts instead of keywords is only part of the solution. If the filtering methods rely on vector-space models of documents and concepts, a dimension reduction technique comes in handy. Instead of training the classifiers using the entire concept-base, the learning of filtering models is improved by restricting the space to those concepts that are most relevant for the given task.

In this paper, we introduce, Wilks-VF, a light-weight concept selection method inspired by Wilks' lambda to reduce the curse of dimensionality. In Wilks-VF the

document classification task is carried out in the following three stages: 1) Once the user has supplied a training sample of relevant and irrelevant documents, a semantic profiler is applied to build a document-concept space representation. The semantic knowledge is drawn from Wikipedia, which provides the semantic relatedness information. 2) Next, the Wilks' lambda based dimension reduction method is used to select concepts that provide the best separation between relevant and irrelevant documents. 3) Finally, the value function framework proposed by Malo et al. [1] is employed to learn a classification rule for the given topic.

The main contribution of the Wilks-VF, as compared to the existing literature, is a light-weight concept selection method, where a clustering based Wilks' lambda approach is used to equip the methodology for on-line usability. Evaluation of the framework's classification performance is carried out using the Reuters TREC-11 corpus. The result is then benchmarked with other well-known feature selection methods. As primary performance measures we use F-Score, precision and recall. The obtained results are promising, but the work is still preliminary and further evaluation with other corpora needs to be carried out.

2 Related Work

During the last decade, the role of document content descriptors (words/phrases vs. categories/concepts) in the performance of information retrieval systems has piqued considerable interest [2]. Consequently, a number of studies have examined the benefits of using concept hierarchies or controlled vocabularies derived from ontologies and folksonomies [3] [4] [6]. In particular, the use of Wikipedia as a source of semantic knowledge has turned out to be an increasingly popular choice, see e.g. [7] [8] [9] [10] [11] [12]. The use of value function in preference modeling is well founded in the fields of operations research and management science [13] [14] [15] [16] [17]. There the purpose is to develop interactive methods for helping the users to find preferred solutions for complex decision-making problems with several competing objectives. The existence of a value function which imitates a decision maker's choices makes the two problems very similar. The essential difference between decision making problems and document classification is the high-dimensionality of information classification problems, which leads to concerns about the ability of value function based methods to deal with large number of attributes.

To alleviate the curse of the dimensionality problem which is often encountered in classification tasks, such as document filtering, a number of feature selection techniques have been proposed [18] [19] [20] [21] [22] [23] [24]. For an extensive overview of the various methods, see e.g. Fodor [25]. However, most of these techniques are designed for general purposes, whereas the approach suggested in this paper is mainly suited for concept-selection task.

3 Wilks-VF Framework

This section describes the steps of the procedure and the implementation of the framework. First, we describe the Wikipedia based document indexing procedure, where every document is transformed into a stream of concepts. Next, we present the dimensionality reduction approach utilizing Wilks' lambda, and finally we describe an efficient linear optimization method for learning the value function based document classifier.

3.1 Document profiling

The document profiling approach used in this paper is similar to the technique adopted by Malo et al.[1], where each document is profiled into a collection of concepts. To illustrate this idea, consider the example in Fig. 1. The text in the figure on the left-hand-side is transformed into a vector of concepts by the profiler. The profiler is implemented as a two-stage classifier, where disambiguation and link recognition are accomplished jointly to detect Wikipedia-concepts in the documents and each concept corresponds to a Wikipedia article. On the right-hand-side (under concept space), a small network is shown, corresponding to the central concepts found in the document. In addition to the concepts directly present in the document, the network displays also some other concepts that are specified in the Wikipedia link structure. As discussed by [7] [8] [9] [10] [11] [12] the link structure can be used for mining semantic relatedness information, which is useful for constructing concept-based classification models.

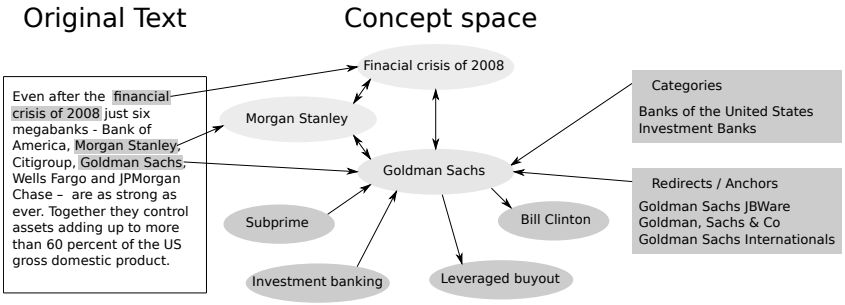

Fig. 1. Wikipedia link structure example

In this paper, we used the concept-relatedness measure described by Malo et al.[1], which in turn is inspired by the Normalized Google Distance approach proposed by Cilibrasi and Vitanyi [26]. In the following definition we introduce the concept relatedness measure and thereafter discuss its usage in Sect. 3.3.

Definition 1. *Concept relatedness: Let C denote concept-space and c_1 and c_2 be an arbitrary pair of Wikipedia-concepts. If $C_1, C_2 \subset C$ denote the sets of all articles that link to c_1 and c_2, respectively, the concept-relatedness measure is given by the mapping c-rel: $C \times C \rightarrow [0, 1]$,*

$$c\text{-}rel(c_1, c_2) = e^{-ND(c_1,c_2)},$$

where the ND measure is $ND(c_1, c_2) = \frac{\log(\max|C_1|.|C_2|)-\log(|C_1 \cap C_2|}{\log(|C|)-\log(\min(|C_1|,|C_2|)}$.

3.2 Dimension Reduction with Wilks' lambda

Wilks' lambda is used to identify the concepts which best separate the relevant documents from the irrelevant ones. Once the documents have been profiled into a matrix where each row represents a document and each column represents a concept, Wilks' lambda tests whether there are differences between the means of the two identified groups of subjects (relevant and irrelevant documents) on a number of dependent variables (concepts). A large difference in the means indicates that the chosen concept can be used to distinguish a relevant document from an irrelevant one [27].

Wilks' lambda statistic. Let $X \in \mathbb{R}^{N \times |\hat{C}|}$ denote the document-concept matrix, where N is the number of documents in the training set and $|\hat{C}|$ is the number of different concepts found in the documents. The matrix can be decomposed into two parts according to the relevance of the documents

$$X = \begin{bmatrix} X_R \\ X_{IR} \end{bmatrix},$$

where X_R and X_{IR} are the collections of profiles corresponding to the relevant documents and the irrelevant ones respectively. The dimensionality reduction procedure is based on the assumption that the profile means, $\bar{x}_R$ and $\bar{x}_{IR}$ in the two document groups are different. If $\bar{x}_R = \bar{x}_{IR}$, none of the concepts are able to differentiate between relevant and irrelevant concepts. The hypothesis $H_0 : \bar{x}_R = \bar{x}_{IR}$ can be tested by the principle of maximum likelihood, using a Wilks' lambda statistic, where T denotes the total centered cross product and W denotes the within groups cross product

$$\Lambda = \frac{|W|}{|T|} = \frac{|X_{\mathrm{R}}^T H X_{\mathrm{R}} + X_{\mathrm{IR}}^T H X_{\mathrm{IR}}|}{|X^T H X|}.$$

In the above equation, Λ follows the F-distribution and can be tested with the approximation developed by Rao [5].

Additional information. In order to choose the concepts that provide the best separation between the two groups, we employ Wilks' lambda to evaluate the information content of the concepts. Let

$$T = \begin{bmatrix} T_{11}\ T_{12} \\ T_{21}\ T_{22} \end{bmatrix} \text{ and } W = \begin{bmatrix} W_{11}\ W_{12} \\ W_{21}\ W_{22} \end{bmatrix}$$

be the block-representation of the total and between groups matrices, where $T_{11} = T_{11(q,q)}$ and $W_{11} = W_{11(q,q)}$ refer to those variables, q, that are included

into the model. For simplicity, we can assume that the variables in the model have indices $1, 2, \ldots, q$. For this purpose, we introduce the decomposition of Wilks' lambda into two parts: the information content of the selected concepts Λ_1 and the information content of the remaining concepts $\Lambda_{2,1}$:

$$\Lambda = \Lambda_1 \Lambda_{2,1} = \frac{|W_{11}|}{|T_{11}|} \frac{|W_{22} - W_{21} W_{11}^{-1} W_{12}|}{|T_{22} - T_{21} T_{11}^{-1} T_{12}|}.$$

The parts of the decomposition can be interpreted as follows:

1. if $\Lambda_1 \approx 1$, then variables $i = 1, 2, \ldots, q$ are not able to separate the groups
2. if $\Lambda_1 << 1$, then variables $i = 1, 2, \ldots, q$ separate the groups very well
3. if $\Lambda_{2,1} \approx 1$, then variables $i = q+1, q+2, \ldots, p$ are not able to provide additional information
4. if $\Lambda_{2,1} << 1$, then variables $i = q+1, q+2, \ldots, p$ contain at least some additional information

Selection heuristic. Motivated by the Wilks' lambda statistic, we now introduce the following heuristic for concept selection:

1. **Initiation**: Let $M = \{1, 2, \ldots, |C|\}$ denote the index set of all concepts and let $N = \varnothing$ be the collection of selected concept indices.
2. **Ranking**: For every concept index $i \in M$, compute $\lambda_i = w_{ii}/t_{ii}$ and sort the index set M in ascending order according to $(\lambda_i)_{i \in M}$ values. The smaller the λ_i, the better the separation power of the concept.
3. **Selection**: For the sorted index set M, choose q concepts with smallest λ-values. Denote this set as M_q and write $M = M \setminus M_q$ and $N = N \cup M_q$. Construct cross-product matrices W_{ii} and T_{ii}, in such a way that they correspond to N selected concept indices.
4. **Evaluation**: Test Λ_1 and $\Lambda_{2,1}$. If the test based on $\Lambda_{2,1}$ indicates no remaining information, then stop.
5. **Update**: For the remaining indices $j \in M$, compute $\lambda_j = (w_{ii} - \boldsymbol{w}_{j1} W_{11}^{-1} \boldsymbol{w}_{1j}) / (t_{jj} - \boldsymbol{t}_{j1} T_{11}^{-1} \boldsymbol{t}_{1j})$. This step is carried out to remove the effect of the already selected variables. Then the execution moves to Step 2 and the process is repeated. In practice choosing a large q (i.e. several concepts are selected at once), leads to a quick termination of the algorithm. which is preferable for on-line use. That is, for most topics a single iteration should give sufficiently good results.
6. **Output**: The collection of selected concepts corresponding to the index set N.

3.3 Value function based classification

In the Wilks-VF framework, the utility or value of each document is obtained based on a combination of the individual attributes (i.e. concepts). Learning a filtering rule in this system is in essence equal to finding the optimal parameters for the user's value function. The process used in this paper is formalized as follows:

Definition 2. *Value Function: Let D denote the space of profiled documents, where each $d \in D$ is a vector of Wikipedia concepts. A value function representing the user's preference information is defined as mapping $V : D \to \mathbb{R}$, given by*

$$V(d) = \sum_{c \in C_N} \mu(c, d) w(c),$$

where $w(c) \in [-1, 1]$ denotes the weight of concept c and C_N is the set of selected concepts from the dimension reduction step. The function $\mu : C_N \times D \to \{0, 1\}$ determines the presence of a concept in the document by the rule

$$\mu(c, d) = \begin{cases} 1 \text{ if } d\text{-}rel(c, d) \geq \alpha \\ 0 \text{ otherwise} \end{cases}$$

where $d\text{-}rel(c, d) = \max_{\bar{c} \in d} c\text{-}rel(c, \bar{c})$ is a document-concept relatedness measure.

Let $\mathbb{D}^{(R)}$ and $\mathbb{D}^{(IR)}$ denote the set of relevant and irrelevant documents originally supplied by the user. The parameters of the value function are determined by solving the following linear optimization problem. Maximize ϵ subject to

$$V(d^{(R)}) - V(d^{(IR)}) \geq \epsilon$$
$$\forall d^{(R)} \in \mathbb{D}^{(R)}, d^{(IR)} \in \mathbb{D}^{(IR)}$$

A positive weight indicates a relevant concept and a negative an irrelevant one. When $\epsilon > 0$, the obtained value function is consistent with the user's preferences. Based on the Wilks-VF a simple document classification rule is obtained by choosing a suitable cutoff [1]. If a document's value is above the cutoff, it is considered relevant.

4 Experiments and Results

In this section, we present the results of the Wilks-VF method. The method has been tested on Reuters TREC-11 newswire documents, which is a collection of news stories from 1996 to 1997. The collection is divided into 100 subsets or topics. All documents belonging to a topic are classified as either relevant or irrelevant to the given topics. These topics are further partitioned into a training and an evaluation set. The purpose of the training set is to generate a search query, which is then applied onto the evaluation set in order to evaluate its performance.

The results from all 100 topics are reported together with five benchmark methods in Table 1. As benchmarks, we consider the following commonly applied feature selection techniques: Gain ratio [18], Kullback−Leibler divergence [18], Symmetric uncertainty [19], SVM based feature selection [20] and Relief [22] [23]. The performance is recorded in terms of precision and recall. F-Score is used to combine the two measures as: F-Score = (2 × Precision × Recall)/(Precision + Recall). The reported performance measures are calculated as averages over all topics. In the experiment we set $q = 10$.

Model	F-Score	Recall	Precision	Model	F-Score	Recall	Precision
GR	0.2785	0.2930	0.3902	SVM	0.3680	0.4073	0.4215
Symmetry	0.3235	0.3578	0.4110	Relief	0.3785	0.4568	0.4068
KLD	0.3391	0.3977	0.3984	Wilks-VF	0.3990	0.5184	0.4011

Table 1. Model comparison

As can be seen from the table, the Wilks-VF method is competitive in terms of F-Score. When searching for reasons, it appears that the performance differences are largely explained by recall levels. The recall for Wilks-VF is considerably better than other methods, as can be observed in Table 1. Differences across precision are however, smaller for different methods. Thisproposed by Cilibrasi and Vitanyi means that the advantage of Wilks-VF is its ability to retrieve relevant instances.

5 Conclusions

The paper discusses an important aspect in document classification, i.e. dimensionality reduction. Dimensionality reduction is ubiquitous in various fields and has been widely studied. In this paper, we have specialized a well known Wilks lambda procedure for document classification. The novelty introduced in the approach is a cluster based concept selection procedure which ensures that all the concepts which are significant for classification are selected. The dimensionality reduction procedure has been integrated with a recently suggested value function approach which makes the overall system computationally less expensive to an extent that the methodology can be developed for on-line usage. The empirical results computed on the Reuters TREC-11 corpus show that the Wilks-VF approach is efficient when compared with other widely used methods for dimensionality reduction.

References

1. Malo P., Sinha A., Wallenius J. & Korhonen P.: Concept-based Document Classification Using Wikipedia and Value Function. Journal of American Society of Information Science and Technology (2011) to appear
2. Rajapakse R. & Denham M.: Text retrieval with more realistic concept matching and reinforcement learning. Information Processing and Management (2006) vol. 42, 1260-1275
3. Kim H.: ONTOWEB: Implementing an ontology-based web retrieval system. Journal of American Society for Information Science and Technology (2005) vol. 56, no. 11. 1167-1176
4. Kim H.: Toward Video Semantic Search Based on a Structured Folksonomy: Journal of the American Society for Information Science and Technology (2011) 478-492
5. Rao C, Linear Statistical Inference. Wiley, NYC, NY (1973)
6. Pera M., Lund W. & Ng Y.-K.: A Sophisticated Library Search Strategy Using Folksonomies and Similary Matching. Journal of the American Society for Information Science and Technology (2009) vol. 60, 1392-1406

7. Malo P., Siitari P. & Sinha A.: Automated Query Learning with Wikipedia and Genetic Programming. Artificial Intelligence (2010) Conditionally accepted
8. Gabrilovich E. & Markovitch S.: Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis. In Proc. IJCAI-07 (2007) 1606-1611
9. Malo P., Siitari P., Ahlgren O., Wallenius J. & Korhonen P.: Semantic Content Filtering with Wikipedia and Ontologies, 10th IEEE International Conference on Data Mining Workshops 2010, Los Alamitos, CA, USA: IEEE Computer Society (2010) 518-526
10. Ponzetto S. & Strube M.: Knowledge Derived From Wikipedia For Computing Semantic Relatedness. Journal of Artificial Intelligence Research (2007) vol. 30, 181-212
11. Milne D. & Witten I.: Learning to link with Wikipedia. Proc. CIKM, (2008)
12. Medelyan O., Milne D., Legg C. & Witten I.: Mining meaning from Wikipedia. International Journal of Human-Computer Studies (2009) vol. 67, 716-754
13. Korhonen P., Moskowitz H. & Wallenius J.: A progressive algorithm for modeling and solving multiple-criteria decision problems, Operations Research (1986) vol. 34, no. 5, 726-731
14. Korhonen P., Moskowitz H., Salminen P. & Wallenius J.: Further developments and test of a progressive algorithm multiple-criteria decision making, Operations Research (1993) vol. 41, no. 6, 1033-1045
15. Deb K., Sinha A., Korhonen P., & Wallenius J.: An interactive evolutionary multi-objective optimization method based on progressively approximated value functions, IEEE Transactions on Evolutionary Computation (2010), vol. 14, no. 5, 723-739
16. Zionts S. & Wallenius J.: An interactive programming method for solving the multiple criteria problem. Management Science (1976) vol. 22, 656-663
17. Roy A., Mackin P., Wallenus J., Corner J., Keith M., Schmick G. & Arora H.: An interactive search method based on user preferences. Decision Analysis (2009) vol. 5 203-229
18. Abeel T., Van de Peer Y.& Saeys Y.: Java-ML: A Machine Learning Library. Journal of Machine Learning Research 10 (2009) 931-934
19. Yu L. & Liu H.: Feature Selection for High-Dimensional Data: A Fast Correlation-Based Filter Solution. Proceedings of the Twentieth International Conference on Machine Learning (2003) 856-863
20. Guyon I., Weston J., Barnhill S. & Vapnik V.: Gene selection for cancer classification using support vector machines. Machine Learning. (2002) 46:389-422
21. Burges C.: A Tutorial on Support Vector Machines for Pattern Recognition. Kluwer Academic Publishers, Boston
22. Kira K. & Rendell L.: A Practical Approach to Feature Selection. Ninth International Workshop on Machine Learning (1992) 249-256
23. Robnik-Sikonja M. & Kononenko I.: An adaptation of Relief for attribute estimation in regression. Fourteenth International Conference on Machine Learning (1997) 296-304
24. Harris E.: Information Gain Versus Gain Ratio: A Study of Split Method Biases. The MITRE Corporation (2001)
25. Fodor I.: A Survey of Dimension Reduction Techniques. U.S. Department of Energy (2002)
26. Cilibrasi R. & Vitnyi P. M. B.: The Google Similarity Distance. IEEE Trans. Knowl. Data Eng. (2007) 19(3): 370-383
27. Friedman H. P. & Rubin J.: On Some Invariant Criteria for Grouping Data. Journal of the American Statistical Association (1967) vol. 62, no. 320 , 1159-1178

This volume is published and copyrighted by:

Marco de Gemmis
Ernesto William De Luca
Tommaso Di Noia
Aldo Gangemi
Michael Hausenblas
Pasquale Lops
Thomas Lukasiewicz
Till Plumbaum
Giovanni Semeraro

ISSN 1613-0073