Generalizing Matching Knowledge using Active Learning

Anna Primpeli
supervised by Christian Bizer
Data and Web Science Group
University of Mannheim
anna@informatik.uni-mannheim.de

ABSTRACT

Research on integrating small numbers of datasets suggests the use of customized matching rules in order to adapt to the patterns in the data and achieve better results. The state-of-the-art work on matching large numbers of datasets exploits attribute co-occurrence as well as the similarity of values between multiple sources. We build upon these research directions in order to develop a method for generalizing matching knowledge using minimal human intervention. The central idea of our research program is that even in large numbers of datasets of a specific domain patterns (matching knowledge) reoccur, and discovering those can facilitate the integration task. Our proposed approach plans to use and extend existing work of our group on schema and instance matching as well as on learning expressive rules with active learning. We plan to evaluate our approach on publicly available e-commerce data collected from the Web.

1. INTRODUCTION

Data integration is a long standing and very active research topic dealing with overcoming the semantic and syntactic heterogeneity of records located in the same or separate data sources [3]. While early work focused on integrating data from small numbers of datasets in a corporate context, there is an increasing body of research on integrating large numbers of datasets in the Web context, where an increased level of heterogeneity exists on both the instance and schema-level.

The matching approaches dealing with the task of integrating large numbers of datasets can be categorized by the addressed integration scenario. One scenario is the N:1, in which multiple datasets are matched against a central source; for instance, web tables against DBpedia [8] or product entities against a central catalog. The second scenario is the N:M, in which datasets are matched with each other without the help of an intermediate schema or a knowledge base.

Proceedings of the VLDB 2017 PhD Workshop, August 28, 2017. Munich, Germany.
Copyright (c) 2017 for this paper by its authors. Copying permitted for private and academic purposes.

Widely used matching systems such as COMA [2] indicate the need of rich matcher and aggregator libraries in order to solve different types of heterogeneity and find correspondences. A major finding of our group on integrating small numbers of datasets is that specific matchers and aggregators deriving from such rich libraries as well as property specific data normalization techniques can be combined into high quality, domain specific matching rules [5]. Such rules achieve a twofold goal: firstly, they give an insight into the current task by encoding matching knowledge, and secondly they achieve high quality correspondences by adapting to the nature of every matching scenario.

Research on integrating large numbers of datasets has shown that it is valuable to exploit attribute co-occurrence in the schema corpus as well as the similarity of data values not only between a central data source and a single external data source, but also the similarities of data values between multiple sources [4, 10]. A weak spot that can be observed in these approaches is that the employed data normalization techniques, similarity functions, and matching rules are not customized for the different types of entities and thus produce lower quality results than customized techniques.

The proposed research program builds upon this work and aims as its first goal to investigate the extent to which it is possible to generalize matching knowledge in order to improve matching quality in large-scale N:1 and N:M matching situations. The rationale for this approach is that typical patterns reoccur among entities of a certain domain. An example of such a pattern would be *"When matching entities representing the product type phones, it is effective to compare the attributes [brand, producer] using the following preprocessing methods [tokenization, lowercasing], the following similarity function [Levenshtein distance] and the following threshold [0.75]"*.

In most matching situations, collaboration between humans and machines is helpful in order to judge corner cases or to boot-strap matching with a certain amount of supervision [12]. Previous work of our group on guiding collaboration between humans and machines on small-scale matching scenarios has shown that using active learning can produce high quality matching results even with a small amount of human interaction [6]: The employed active learning approach was evaluated against six different datasets reaching between 0.8 and 0.98 F1 score after asking the human annotator to label ten pairs of entities as positive or negative matches. Building upon this work and extending certain steps of the active learning process, we formulate the second goal of the thesis, which is steering human attention in

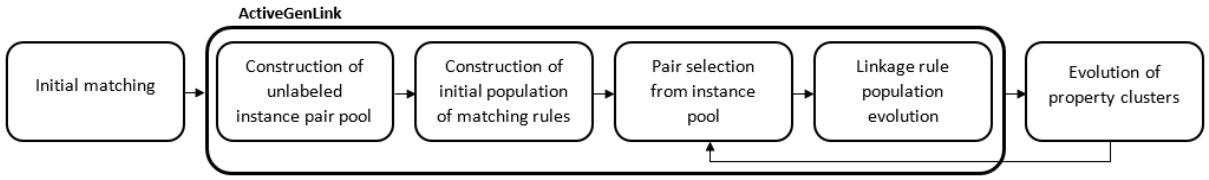

Figure 1: Proposed matching approach pipeline

large-scale matching situations with the aim to learn relevant, high quality matching knowledge.

Summing up, in the context of this work, we aim to answer the following research questions:

- Are domain specific patterns transferable within large-scale matching situations?
- How can we maximize the benefit of human supervision with respect to the discovery of those patterns?

In order to answer the above stated research questions, we will experiment with the N:1 and N:M matching scenarios using datasets created in the context of the Web Data Commons project[1] such as web tables, schema.org data and product data.

The remainder of this paper is organized as follows. Section 2 describes the proposed matching approach. Section 3 presents the data which we plan to use for evaluation. Finally, Section 4 gives a short overview of our workplan.

2. PROPOSED MATCHING APPROACH

This section gives an overview of the current plan for generalizing matching knowledge using active learning for the N:1 matching scenario. The planned approach involves three main steps. The first step is matching on the instance and schema-level with the goal to generate a preliminary set of schema correspondences which forms the basis for later refinement. Next, we build upon the concepts of the *ActiveGenLink* algorithm [6], an active learning approach based on genetic programming, which uses human supervision to evolve matching rules applicable on the instance-level. The final step is the refinement of the schema correspondences based on the instance-level matching results. The two last steps are iterated until the desired accuracy or the maximum amount of questions to the human annotator have been reached.

Figure 1 shows the steps of the matching process which will be the main guideline of this research. Below the individual steps are further explained, and the related state-of-the-art work upon which we build our proposed approach is presented together with our suggested methodological contributions.

2.1 Initial matching

The first step of our algorithm involves matching on the instance and schema-level with the goal to generate an initial set of correspondences which will be refined in the next steps of the algorithm. To achieve this, we employ existing techniques for large-scale matching.

For the N:1 scenario we use the T2K matching algorithm which involves cycles of instance and schema matching and achieves an F1 score of 0.8 on the instance-level and 0.7 on the schema-level for the task of matching web tables to DBpedia [11].

The resulting schema correspondences of this step are grouped into clusters, with each cluster representing a specific property such as *product name* or *brand*. The motivation behind property clusters is that matching information concerning one property can further affect the other elements of the cluster, as it will be later explained in Section 2.6.

In this step, we plan to reuse existing work to form our baseline for further improvement using active learning.

2.2 Construction of unlabeled instance pair pool

The second step involves the construction of an unlabeled pool of pairs of instances that are potential candidates for labeling by the user. Considering the complexity involved with matching large-scale data as well as our goal for creating generalized matching rules, the unlabeled instance pair pool should be constructed on the basis of two guidelines: computational space reduction and preservation of matching knowledge.

To achieve computational space reduction we propose an indexing and a blocking technique. We use three types of information to build an index value out of every instance: the instance name, the attribute labels and the attribute values using words or n-grams. After indexing, we make sure that the candidates for the unlabeled instance pair pool hold valuable information while eliminating the rest of them. To ensure this, different pair characteristics are evaluated. Such possible entity characteristics aside being likely matches, would be if the involved entities are described by many frequent properties and if they are head or tail entities, based on how often they occur in the data corpus.

After defining such informativeness criteria, we linearly scan over the entity names of the central source and we generate a pair if it is considered informative. Next, the generated pair is added in the unlabeled instance pair pool.

Thus, in this step we need to discover which characteristics make a pair a good candidate for the instance pair pool and how to combine them in order to draw the line between informative and non-informative pairs.

2.3 Construction of initial population of matching rules

As a next step, the initial linkage rules are created. We build upon the linkage rule definition introduced in [6]. A linkage rule is defined as a combination of different operators having a tree representation that gradually transforms with the evolution of the *GenLink* algorithm, a variation of the genetic algorithm [5]. A linkage rule contains the following set of operators:

[1]http://webdatacommons.org/

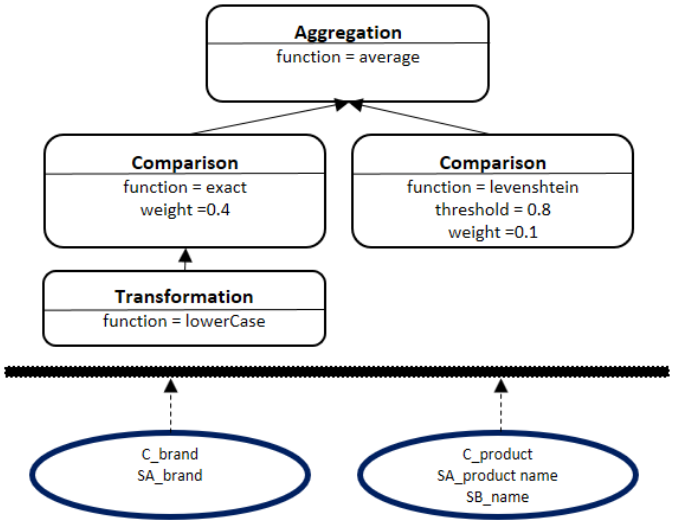

Figure 2: An example matching rule

(a) Property Operator: Selects the values of a property.

(b) Transformation Operator: Defines the transformation functions for the selected property values. Such transformations may be: case normalization, address standardization, stop-word removal, and structural transformations such as segmentation and extraction from values from URIs [5].

(c) Comparison Operator: Defines the distance metric and threshold that should be used to compare the selected values.

(d) Aggregation Operator: Defines the way to combine the results of the different comparison operators of the previous level.

The difference in our setting is that the property operators do not refer to specific properties but to property clusters, as introduced in Section 2.1. Thus, when a rule is applied to a specific instance pair from the pool of unlabeled pairs, the property operator checks if both entities contain a property which is part of any property cluster. If this is the case, the property operator outputs a pair of values. Otherwise it outputs an empty set of values. The functionality of the other operators remains the same.

Figure 2 shows an example rule of our matching approach. In the illustrated example the property operator selects the clusters that represent the *product brand* and the *product name* properties. In the next level, the values of the property *brand* are lowercased and a specific comparison operator is defined. Based on the weights and threshold the comparison operators normalize the similarity score to the range [0,1]. Finally, the results of the comparison operators are aggregated into a single score using the average aggregator which finally decides if the pair is a positive or a negative correspondence.

2.4 Pair selection from instance pool

In this step a pair of instances is selected from the instance pool and presented to the human annotator who provides a label as matching or non-matching. The goal of this step is to define a query strategy that selects the most informative pair to be labeled, thus minimizing the human involvement in the whole process.

For this we build on the three query strategies employed by [6]: 1. *query by committee* evaluates the unlabeled pairs against the current population of matching rules and selects the pair that causes the biggest disagreement, 2. *query by divergence* selects one pair out of every group in the similarity space, thus considering pairs which convey different similarity patterns, and 3. *query by restricted committee* uses the query by committee strategy but only considers the disagreements between the top K optimal matching rules of the current population.

Our strategy will build upon the existing ones and further clarify which other criteria should be considered in order to maximize the benefit of the selected pair. One possible direction of our query strategy could be to prefer pairs that contain many properties so that information about a bigger variety of properties can be learned after a pair has been annotated. In addition, the usage of a mixture between head and tail entities can prove effective in revealing information concerning the whole domain and not focus only on the frequent entities. Another possible component of our query strategy could be the characteristics of the properties of the selected pairs. Such characteristics might be frequency and the size of the cluster to which they belong. The rationale behind using those features is that if the answer of the human annotator gives further insight for a centroid of a property cluster then other properties may be indirectly affected, as described more detailed later in Section 2.6, thus leading to a faster convergence of the algorithm.

2.5 Linkage rule population evolution

In this step, we exploit the information provided in the previous step by the human annotator as supervision to evolve the population of matching rules. The goal is to gradually generate more customized and accurate matching rules which evaluate correctly against the labeled set of pairs. To achieve this we use the *GenLink* algorithm [5].

GenLink evolves the population of linkage rules in two steps, selection and transformation. Firstly, the matching rules are evaluated on the basis of their fitness on the current set of labeled pairs and selected using tournament selection [7]. The selected rules are then transformed by applying certain crossover operations. More specifically, a crossover operator accepts two linkage rules and returns an updated linkage rule that is built by recombining parts of the parents. In our setting we use the specific set of crossover operations of *GenLink*: *transformation*, *distance measure*, *threshold*, *combine operators*, *aggregation function*, *weight*, and *aggregation hierarchy*.

2.6 Evolution of property clusters

In the final step of our approach, the evolution of the property clusters which preserve the schema-level correspondences takes place. The goal is to gradually improve the matching accuracy on the schema-level whilst exploiting the information on the instance-level.

To achieve this we select the top rules of the current linkage rule population based on their fitness score and apply them on the unlabeled candidates. Possible metrics for calculating the fitness score are F1 or Matthews correlation coefficient in the case that the set of reference links is un-

balanced. As a result, we retrieve instance-level correspondences which we use as input for duplicate-based schema matching with the goal to refine the property clusters.

We follow the approach of DUMAS (Duplicate-based Matching of Schemas) [1] for improving schema correspondences based on instance-level duplicates. In their work, they evolve the meta-level similarities gradually by calculating similarities on the instance-level using the SoftTFIDF measure and then solving the transformed problem as a bipartite weighted matching one. In every iteration, schema-level matches are either confirmed, doubted, or rejected.

After calculating the schema-level similarities, the property clusters of our setting are refined. We will investigate how the move of one element from a cluster may affect the rest of the related elements. The indirect effects may be a result of frequent co-occurence or strong similarity. For example, consider the property cluster setting $C_1 = \{A, B, C\}$ and $C_2 = \{D, E\}$. Assuming that property A matches to properties D and E, we move A to cluster C_2. If we additionally know that property B is very close on the similarity space to property A, then B follows A thus formulating the final state of property clusters as: $C_1 = \{C\}$ and $C_2 = \{A, B, D, E\}$.

2.7 Convergence and output

The process iterates by selecting a new pair from the unlabeled instance pair pool, evolving the linkage rules, and further property cluster refinement as described in Sections 2.4, 2.5 and 2.6. The cycle of iterations terminates when either the evolved linkage rules achieve the desired fitness score or the maximum number of questions to the user has been reached.

The outputs of the proposed matching approach are instance and schema-level correspondences as well as generalized matching knowledge deriving from the linkage rules with the best fitness score. In the N:1 matching scenario the acquired knowledge can be used to annotate the knowledge base with rules concerning attribute relevance for matching, appropriate similarity functions, data normalization transformations, aggregation functions, and thresholds. In the N:M matching scenario we aim to exploit the resulting matching knowledge rules to annotate the implicit, mediated schema created through holistically matching entities.

3. EVALUATION

We plan to evaluate our approach on e-commerce data we already created in the context of the Web Data Commons project. The dataset contains 13 million product-related web pages retrieved from the 32 most frequently visited websites. We have manually annotated 500 electronic product entities and created a product catalog with 160 products of the same electronic categories. The total number of correspondences contained in our gold standard is 75,000, of which 1,500 are positive [9].

Other possible use cases for evaluating our approach would be web tables, linked open data and schema.org annotations. Web Data Commons provides the WDC Web Tables Corpus[2], the largest non-commercial corpus of web tables deriving from 1.78 billion HTML pages with 90.2 million relational tables.

[2]http://webdatacommons.org/webtables/

4. WORKPLAN

The outlined research program is currently in its first year. As an initial step towards accomplishing the goals defined in the context of this work we will focus on the N:1 matching scenario by applying the steps presented in Section 2. Next we will move on to the N:M matching scenario for which special indexing and blocking techniques need to be defined in order to deal with the increased complexity. Finally, granting that our proposed approach meets our goals, we aim to enhance existing knowledge bases by annotating them with matching knowledge.

5. REFERENCES

[1] A. Bilke and F. Naumann. Schema matching using duplicates. In *Proc. of the 21st Int. Conf. on Data Engineering*, pages 69–80. IEEE, 2005.

[2] H.-H. Do and E. Rahm. COMA: a system for flexible combination of schema matching approaches. In *Proc. of the 28th Int. Conf. on Very Large Data Bases*, pages 610–621. VLDB Endowment, 2002.

[3] A. Halevy, A. Rajaraman, and J. Ordille. Data integration: The teenage years. In *Proc. of the 32nd Int. Conf. on Very large data bases*, pages 9–16. VLDB Endowment, 2006.

[4] B. He and K. C.-C. Chang. A holistic paradigm for large scale schema matching. *SIGMOD Record*, 33(4):20, 2004.

[5] R. Isele. *Learning Expressive Linkage Rules for Entity Matching using Genetic Programming*. PhD thesis, University of Mannheim, 2013.

[6] R. Isele, A. Jentzsch, and C. Bizer. Active learning of expressive linkage rules for the web of data. In *Proc. of the 12th Int. Conf. on Web Engineering*, pages 411–418. Springer, 2012.

[7] J. R. Koza, M. A. Keane, M. J. Streeter, W. Mydlowec, J. Yu, and G. Lanza. *Genetic programming IV: Routine human-competitive machine intelligence*, volume 5. Springer Science & Business Media, 2006.

[8] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. Mendes, S. Hellmann, M. Morsey, P. Van Kleef, S. Auer, and C. Bizer. DBpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal*, 2014.

[9] P. Petrovski, A. Primpeli, R. Meusel, and C. Bizer. The WDC gold standards for product feature extraction and product matching. In *Proc. of the 17th Int. Conf. on Electronic Commerce and Web Technologies*, pages 73–86. Springer, Cham, 2016.

[10] E. Rahm. The case for holistic data integration. In *Proc. of the 20th Advances in Databases and Information Systems Conf.*, pages 11–27. Springer, 2016.

[11] D. Ritze, O. Lehmberg, and C. Bizer. Matching HTML tables to DBpedia. In *Proc. of the 5th Int. Conf. on Web Intelligence, Mining and Semantics*, page 10. ACM, 2015.

[12] M. Stonebraker, D. Bruckner, I. F. Ilyas, G. Beskales, M. Cherniack, S. B. Zdonik, A. Pagan, and S. Xu. Data Curation at Scale: The Data Tamer System. In *Proc. of the Conf. on Innovative Data Systems Research*, 2013.