Deriving Concept Mappings through Instance Mappings

Balthasar A.C. Schopman, Shenghui Wang, and Stefan Schlobach

Vrije Universiteit Amsterdam

Abstract. Ontology matching is a promising step towards the solution to the interoperability problem of the Semantic Web. Instance-based methods have the advantage of focusing on the most active parts of the ontologies and reflect concept semantics as they are actually being used. Previous instance-based mapping techniques were only applicable to cases where a substantial set of instances shared by both ontologies. In this paper, we propose to use a lexical search engine to map instances from different ontologies. By exchanging concept classification information between these mapped instances, an artificial set of common instances is built, on which existing instance-based methods can apply. Our experiment results demonstrate the effectiveness and applicability of this method in broad thesaurus mapping context.

1 Introduction

The problem of semantic heterogeneity and the resulting problems of interoperability and information integration have been an important hurdle to the realisation of the Semantic Web. Different communities use different ontologies and are unable to intercommunicate easily. Solving matching problems is one step to the solution of the interoperability problem. To address it, the Database and Semantic Web communities have invested significant efforts over the past few years [1,2,3].

Instance-based ontology matching techniques determine the similarity between concepts of different ontologies by examining the extensional information of concepts [4,5], that is, the instance data they classify. The idea behind such instance-based matching techniques is that similarity between the extensions of two concepts reflects the semantic similarity of these concepts. A first and straightforward way is to measure the *common extension* of the concepts — the set of objects that are simultaneously classified by both concepts [6,7]. This method has a number of important benefits. Contrary to lexical methods, it does not depend on the concept labels, which is particularly important when the ontologies or thesauri where written in a multi-lingual setting. Moreover, as opposed to structure-based methods, it does not depend on a rich ontology structure; this is important in the case of thesauri, which often have a very weak, and sometimes even almost flat structure.

However, measuring the common extension of concepts requires the existence of sufficient amounts of shared instances, something which is often not the case.

J. Domingue and C. Anutariya (Eds.): ASWC 2008, LNCS 5367, pp. 122–136, 2008.

Therefore, in this paper, we aim at enriching one ontology by instances from the other ontology which it should be mapped to and vice versa. Such enrichment is carried out through mappings between instances, that is, similar instances should be classified to the same or similar concepts. In this way, an artificial common ontology extension is built so that many current instance-based methods, such as those in [7], can apply.

Research questions. In this paper, we are experimenting to answer the following research questions:

1. Can an ontology be automatically enriched by instances from another ontology using the similarity between instances?
2. Can the artificially built dually classified instances produce reasonable mappings between two enriched ontologies?

Method and experiments. We use the Lucene search engine[1] to match instances from two different ontologies. For each instance i_t of an ontology T, the most similar instance i_s from the to-be-mapped ontology S is automatically classified to the concepts which i_t also belongs to. After the enrichment, we preserve the instances of each concept from both thesauri, which include their original instances and the ones populated from the ontology to be mapped. Based on such artificially built extensional information of concepts, we calculate a similarity (in our case, the simple Jaccard similarity) to measure the overlap between the extensions of two concepts, which in the end leads to mappings between them, *i.e.*, the higher similarity, the higher probability they should be mapped.

We applied this method on two different cases of thesaurus mapping, a special but frequent mapping problem:

1. mapping GTT and Brinkman whose instances are all books from the National Library of the Netherlands. These are homogeneous instances with the same meta-data fields.
2. mapping GTT/Brinkman and GTAA. The latter thesaurus is used to annotate broadcast materials in the Dutch archive for Sound and Vision. These are heterogeneous instances with different meta-data descriptions.

Evaluation. We first measure the quality of the instance mappings, using the first case and then evaluate the concept mappings in both cases to check the effectiveness and applicability of our method to both homogeneous and heterogeneous collections.

Relation to our previous work. In [7], the similarity between two concepts is measured based on the overlap of their instance sets. This method relies on the existence of a set of common instances and therefore limits itself not applicable if there are no common instances. In [8], all instances of each concept are

[1] `http://lucene.apache.org/`

aggregated to form a unified representation of this concept. A probabilistic classifier is trained to model the correlation between the similarity between such aggregated representations and the mapping between two concepts. In this paper, we directly use the similarity between individual instances and assume similar instances should be classified to similar concepts. An artificial set of common instances will be built, based on which the similarity between concepts is measured by applying the methods used in [7]. This is an extension of the work done in [7][2] and in parallel with the learning method used in [8].

The rest of the paper is structured as following: Section 2 describes the two application problems in more details. Section 3 introduce our method of using instance mappings to derive concept mappings, including how to use the lexical search engine, Lucene, to achieve instance mappings. In Section 4, we present the results of our experiments. Section 5 introduces some related work, and finally, Section 5 concludes this paper and discusses the future work.

2 Application Problems

Our research has been motivated by practical problems in the Cultural Heritage domain, an interoperability problem within National Library of the Netherlands (Koninklijke Bibliotheek, or KB), and the problem of unified access to two heterogeneous collections, one from the KB, one from the Dutch archive for Sound and Vision (Nederlands Instituut voor Beeld en Geluid, or BG).

2.1 Homogeneous Collections with Multiple Thesauri

Our first task is to match the GTT and Brinkman thesauri, which contain 35K and 5K concepts respectively. The average concept depths are 0.689606 and 1.03272 respectively.[3] Both thesauri have similar coverage but differ in granularity. These two thesauri are individually used to annotate two book collections in KB: the *Scientific Collection* annotated mainly by GTT concepts and the *Deposit Collection* annotated mainly by Brinkman concepts.

In order to improve the interoperability between these two collections, for example, using GTT concepts to search books annotated only with Brinkman concepts, we need to find mappings between these two thesauri.[4] Among nearly 1M books whose subjects are annotated by concepts from these two thesauri, 307K books are annotated with GTT concepts only, 490K with Brinkman concepts only and 222K with both. The books in both collections are described using the same metadata structure, more specially, using an extension of the Dublin Core metadata standard.[5]

[2] See Section 4.2 for detailed comparison.
[3] Nearly 20K GTT concepts have no parents.
[4] Descriptions of different scenarios of using mappings, the requirements on mappings and various evaluation methods can be found in [9].
[5] `http://dublincore.org/documents/dces/`

2.2 Heterogeneous Collections with Multiple Thesauri

Our second task is to match the Brinkman thesaurus from the KB to the GTAA thesaurus, which is used to annotate the multimedia collection in the BG. The BG serves as the archive of the Dutch national broadcasting corporations. All radio and television programmes that are broadcast by these corporations are continuously added to the archive. Besides over 700K hours of material, the BG also houses 2M still images and the largest music library of the Netherlands. Each object in the BG collection is annotated by one or several concepts from the GTAA thesaurus. The GTAA thesaurus contains 160K concepts in total, including 3868 from the subject facet which are interesting to map with the KB thesauri. The concept hierarchy of the subject facet has an average depth of 1.30817.

Mapping GTAA to one or both of the KB thesauri is very interesting from a Cultural Heritage (CH) perspective, as interoperability across collections has become an urgent practical issue in this domain. For example, one could be interested to search for some broadcasts from the BG about the author of the book he is reading in the KB. Aligning these thesauri with which the collections are annotated provides a promising solution to achieve this interoperability. Different from the KB case, the meta-data structure of instances differs significantly across collections.

In both cases, each of the thesauri to be mapped contains a large amount of concepts, which many current matching tools could not even load. The concepts within the thesaurus are poorly structured or rather in a nearly flat list, which makes the structural matching techniques not really applicable. Luckily, the instances of those concepts are available which allows us to apply instance-based methods, as done in our previous work [7,8]. In this paper, we continue exploring the instance-based method at the meta-data level.

3 Method: From Instance Mappings to Concept Mappings

Our task is to map two thesauri, each of which is used to annotate a collection of objects (books or multimedia materials). Thesaurus concepts are used to annotated the subject of these objects and we consider an object is annotate by a concept as the instance of this concept. Each object may be annotated by multiple concepts, therefore, one object can be the instance of multiple concepts.

On top of their subject feature, instances also have other features, such as title, abstract, creator, *etc*. These features together uniquely represent an instance. All instances are virtually projected into a space where the distance between them can be measured, *e.g.*, using the Euclidean distance in the feature space.

Instances that are close in this space could potentially be classified to similar concepts. Based on this hypothesis, for one concept in one ontology, if instances in the other ontology are similar to its own instances, we can add those instances as

its *virtual* instances. Therefore, these instances can be seen as common instances shared by this concept and those they really belong to, *i.e.*, the concept(s) in the other ontology. Once this artificial set of common instances is built, the existing instance-based methods can be applied to generate concept mappings.

Let us formally describe the (rather simple) idea: let S (for source) and T (for target) be two thesauri we want to map, and I_s and I_t be their finite sets of instances. Let $ann_s(i) = \{C \in S \mid i \in ext(C)\}$ be the annotation of an instance i, which contains a set of concepts from S. These concepts have instance i in their extension $ext(C)$.

Suppose we have a similarity function sim between instances (across I_s and I_t). For each instance $i \in I_s$, we look for an instance $j \in I_t$ which is the most similar to i. That is,

$$j = \operatorname*{argmax}_t sim(t, i).$$

We can now simply add j to the extension of all concepts $C \in ann_s(i)$. The same process is carried out in both directions. This way, we create a virtual dually annotated corpus. This section remains to explain how we calculate the similarity between instances, and to recall how we calculate concept mappings from dually annotated corpora.

3.1 Matching Instances

Based on the above hypothesis, we use the Lucene search engine to achieve instance mappings. Lucene is a high-performance and scalable information retrieval library through which any piece of textual data can be indexed and made searchable. Indexing with Lucene can be divided into three main phases: (i) converting data to text, (ii) analysing the text and (iii) saving the text to an index. We feed instance data in Lucene, stored in the form of a *Document*. A *Lucene document* (LD) consists of a collection of *fields*. Every field contains the content of the corresponding instance features, such as "title," "abstract," "creator," *etc.* Additionally, each instance has a "subject" field which contains the labels or unique identifiers of the concepts they belong to. Lucene allows keyword-based search and search results (on the form of LDs) are collected within Lucene *Hits*. Each LD contained in the Hits, has an associated score value (between 0 and 1) that indicates its similarity to the search key. Lucene scoring schema is based on the Vector Space Model [10] of information retrieval. The benefits of using Lucene are very fast response time, shown in [11], and complexity almost hidden to the users.

The instance matching process is as follows. Let I_s and I_t be the two instance sets of two ontologies, *e.g.*, two book collections annotated by the GTT and Brinkman thesauri. First we populate the Lucene database (Ldb) with a collection I_s. Each instance is stored as a LD with its fields containing information about this instance. Since Lucene operates on a lexical level, we use the textual representations of fields where possible, such as "title," "subject," "abstract," "descriptions," *etc.*

Then we execute a query for every instance i_t in the other collection I_t. The Lucene search engine allows us to search for information in specific fields of the documents, by specifying one or more keywords and one or more Fields to search within. We can use words in, for example, the "title" field of instance i_t as keywords and search the "title" fields of all LDs in the Ldb. It is also possible to carry out cross-field queries. That is, for example, using words in the "title" field to search the "subject" or "abstract" fields, or vice versa. We can also construct queries by concatenating multiple fields. In this construction we create Lucene documents with a single field containing the concatenation of certain fields, for instance the "title" and "subject" fields. Then we execute single-field queries to match these concatenations with each other.

For every query Lucene returns a list of hits, which is ordered by relevance. We take the most similar instance and observe which concepts it belongs to, *i.e.*, concepts in the "subject" field. We then classify instance i_t as an instance of these concepts, by adding these concepts into its "subject" field.[6] The same process is carried out from collection I_s to I_t, *i.e.*, populating the Ldb with collection I_t and enriching the instances of collection I_s. In the end, each instance in both collections will be classified against concepts from both thesauri, which means an artificial set of common instances is created.

If instances in different collections are homogeneously structured, *i.e.*, the same features are available across different collections, such as the two collections in the KB, we can use Lucene to directly map instances. However, in more cases, different collections have different structures to represent/store their instances, such as the different collections in the KB and the BG. Similarly, we can feed different collections to the Lucene database, using their own features. However, we need to specify corresponding query fields in order to run Lucene queries and map instances afterwards. Different from constructing queries for the homogeneous case, when it is clear that two fields are good for query, such as "title" to "title" or "title" to "subject," in the heterogeneous case, we need to anticipate these potentially good pairs of fields. Readers are referred to [8] for different ways of automatically choosing such pairs.

3.2 Matching Concepts

Once the artificial common instance set is built, we can apply existing instance-based techniques to compute mappings between concepts. Our previous work has shown simple measures of similarity between instances suffice to produce sensible mappings [7]. In this paper, we use the Jaccard similarity measure to determine whether two concepts can be mapped or not. Specifically, each concept corresponds to a set of instances which are annotated by this concept. For all possible pairs of concepts, we measure the Jaccard similarity between their instance sets. This is a measure of similarity between the extensional semantics of those concepts. Pairs of concepts with a high Jaccard similarity are considered as a mapping.

[6] By adding concepts into the "subject" field of an instance, this instance will be considered as an instance of each added concepts.

There are two parameters which need to be taken into account:

1. The minimum similarity: a threshold that determines how similar the two concepts must be in order to define a mapping between them.
2. The minimal number of instances shared by two concepts. If a concept has very few instances, using these instances to determine its extensional semantics is not sufficient. Sometimes, it may mislead the similarity judgement. For example, if two concepts each have one instance and by chance this instance is shared. This will result in a Jaccard measure of 1, but it should actually carry less weight than the case that two concepts have 1000 instance in common and a few not.

These two parameters in practice are set in an empirical way. Based on some evaluation criteria, we can set them up to optimise the performance. Note, this is obviously a biased solution, as the evaluation criteria may vary due to different mapping usage scenarios, see [9] for more details.

4 Experiment and Evaluation

We will study the following questions:

1. How good is our method for finding similar instances?
2. How does our proposed method perform on homogeneous data collections?
3. How does our proposed method perform on heterogeneous data collections?

4.1 Evaluation of the Quality of Instance Mappings

In the KB case, we have 222K books which have been previously dually annotated with two thesauri to be matched. This gives us an opportunity to evaluate whether similarity of the descriptions of instances (books) indeed leads to valid instance mappings.

For this purpose, we split the original dually annotated instance set into two parts, noted as I_G and I_B. By hiding the GTT annotation of each book in I_B and the Brinkman annotation of the books in I_G, we created two collections annotated by only one thesaurus.

We first populated the Lucene database (Ldb) with I_G and using the method introduced in Section 3.1, each instance in I_B finds the most similar book in I_G and adopts this book's GTT annotation as its new GTT annotation. Similarly, each book in I_G also borrows the Brinkman annotations from the most similar book in I_B. By comparing its original manually created annotation and this new GTT/Brinkman annotation automatically obtained from the mapped instance, we can evaluate the basic hypothesis of our method.

We calculated the similarity of the original annotations with the artificial ones built from the instance mappings. As the Jaccard similarity is the most common

Table 1. Performance of using different query configurations

Query fields	Sim_a
title	0.244
title, subjects	0.324
title, subjects (cross query)	0.318
title, subjects (concatenated)	0.310

way of comparing sets, we use it again for this purpose,[7] more concretely: the quality of a prediction for each book in I_G and I_B is calculated as following:

$$sim_a = \frac{|S_m \cap S_n|}{|S_m \cup S_n|} \tag{1}$$

where S_m is the manual (original) annotation and S_n is the annotation built by the instance mapping. Clearly, a higher similarity implies a higher chance for this method to produce a reasonable set of common instances. We then take the average of this Jaccard similarity over all dually annotated books as the final measure. Different ways of query configurations, as discussed in Section 3.1, perform differently, shown in Table 1.

From Table 1, we can see that the new annotations obtained by querying the "title" and "subject" fields separately, on average, are the most similar to the original manual ones, with a Jaccard measure of 0.324. This may seem to be a low value, but the following experiments will show that the predicative power of these (now artificially dually annotated) instances is almost as high as of the original ones. It is also worth noting that the values given in Table 1 only refer to the original annotations which are, in our experience, also not necessarily perfect, and often incomplete. This means that this measure may under-estimate the correctness of the new annotations. A proper manual evaluation of this is impossible due to the size of the corpus, and the specialised nature of the annotation task in a library.

4.2 Mapping Thesauri over Homogeneous Collections

In order to evaluate our proposed mapping method for thesauri over homogeneous collections we repeat the experiments of [7] to map Brinkman and GTT, but now based on the full set of instances (not just the doubly annotated corpus). We used Jaccard similarity measure to generate mappings based on instances. We applied this measure on the real singly annotated datasets. In this case, I_G and I_B contain 307K and 490K book instances, respectively.

[7] The reader should not confuse our use of the Jaccard measure to calculate similarity of concepts, and to evaluate the quality of the artificial annotations. Here, the Jaccard similarity measures how similar the artificial annotation is to the original one, while in the former case, the Jaccard similarity measures the overlap of the common extension of two concepts.

Table 2. Comparison with results from the real dually annotated dataset, where the recovered mappings are those found by our method which are also found from the real dually annotated dataset and the percentage in the bracket is the corresponding proportion.

Query fields	Recovered mappings	New mappings
title	426 (31%)	15
title, subjects	549 (40%)	390
title, subjects (cross query)	640 (47%)	564
title, subjects (concatenated)	1140 (84%)	429

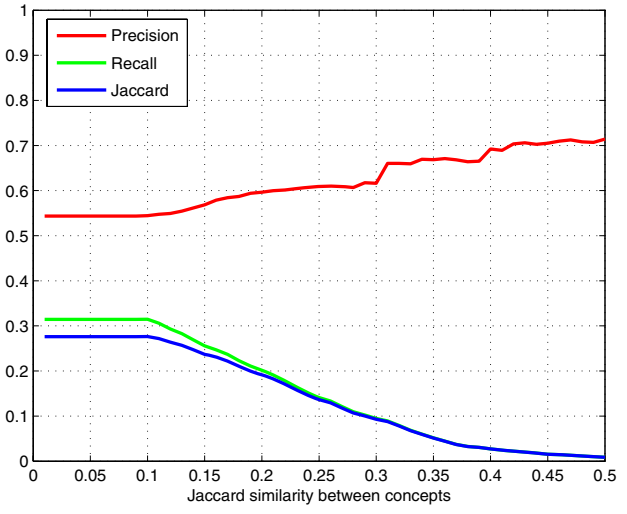
The task to be performed is a book-reindexing scenario, which influences the way the experiments need to be evaluated.[8] The minimal number of instances shared by two concepts is 10 and the lowest threshold was set to 0.001. The performance varies with the choice of threshold, as depicted in Fig. 1. In our case, the Jaccard measure is the one we would like to optimise — a high Jaccard similarity means the translated annotation covers most of its manual annotation without introducing many errors.

We can see from this figure that the optimum Jaccard measure is achieved by taking the threshold around 0.1. Fig. 1 (b) is the performance of mappings generated from the real dually annotated dataset, which can be seen as an upper bound performance of these mappings in this scenario. Mappings generated from the real singly annotated dataset performs at a similar level to what the OAEI'2007 participants did on the same dataset.[9] This is very encouraging, because it indicates that our method does not have the constraints on the existence of the explicit dually annotated instances, and still, performs as well as the state-of-art tools do.

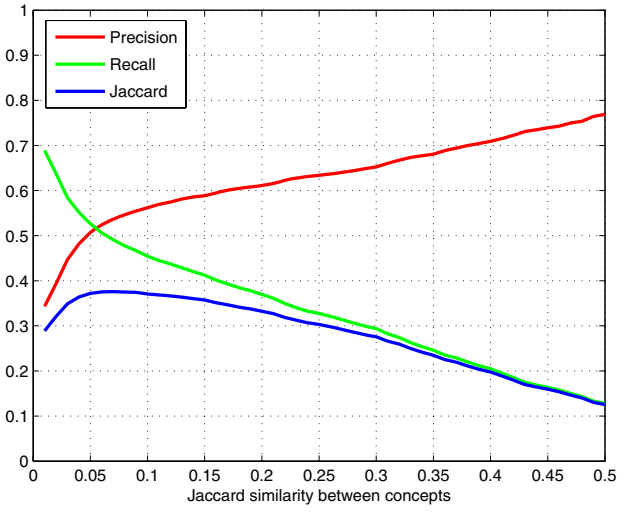
In Table 2, we compared mappings generated from the real singly annotated dataset (*i.e.*, 307K books with only GTT annotations and 490K books with only Brinkman annotations) with those generated from the dually annotated dataset (*i.e.*, 222K books with both Brinkman and GTT annotations). In the base case — generate artificial common instances using concatenated "title" and "subject" — we found 84% pairs which are found from the dually annotated dataset, both using the threshold of 0.1. A manual evaluation has shown that 97% of the 429 new mappings are correct. This comparison confirms that our method can to a large extent recover the mappings generated from a dually annotated dataset if it is available; also the high precision of the new mappings indicates that our method makes use of the information which was not usable before. It means that even when dually annotated instances are available, using our method with singly annotated instances can improve current mapping results.

[8] Technical details can be found in [9].

[9] See `http://oaei.inrialpes.fr/2007/results/library/` for more details of the results of OAEI'2007 participants.



(a) Real singly annotated dataset



(b) Original dual annotated dataset (upper bound)

Fig. 1. Performance evaluated in book-reindexing scenario

4.3 Mapping Thesauri over Heterogeneous Collections

Now we map GTAA with Brinkman using disjoint and heterogeneous collections. As we introduced earlier, GTAA is used for annotating multimedia materials in the Dutch archive for Sound and Vision (BG), Brinkman for books in the KB. In our dataset from the BG collection, there are nearly 60K instances and their

subjects are annotated against 3593 GTAA concepts. The task here is to map these GTAA concepts with 5207 Brinkman concepts.

As we discussed earlier in Section 3.1, we first map instances in order to build an artificial common extension of these concepts. Each collection was fed into the Lucene database and the query fields were set up manually.

We specify queries on concatenated "kb:title" + "kb:subject" and "bg:title" + "bg:subject" +"bg:description." Each KB instance will be classified to one or several GTAA concepts through the mapping between instances, and similarly, each BG instance will be classified to one or several Brinkman concepts. Then, all instances of one concept were put together as the extensional representation of this concept. The Jaccard similarity was measured between the instance sets of all possible pairs of Brinkman and GTAA concepts. All pairs were then ranked by their Jaccard similarity into an ordered list, with the most promising mappings on the top.

Ideally, the generated mappings should be evaluated against a reference alignment for a global view of the precision and recall. Unfortunately, obtaining a complete list of possible mappings is not practically possible. We therefore compare the obtained mappings with results from a lexical mapper and then manually measure the precision of the top K mappings.

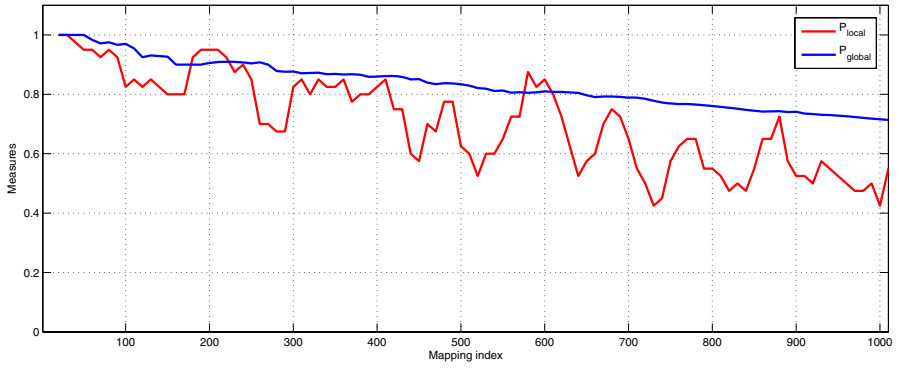
Using a simple lexical mapper,[10] we obtained 1458 lexical equivalent mappings between 5207 Brinkman concepts and 3593 GTAA concepts from the subject facet. One or both concepts of 115 lexical mappings do not have any instances and therefore cannot be measured by our method.

Moving from the top of the ranked list, we measure the proportion of lexical mappings, P_{lexical}, and the coverage over all lexical mappings, C_{lexical}. As Fig. 2 (b) shows, when the Jaccard similarity is relatively high, most of the found mappings are actually lexical mappings. This proportion decreases with the Jaccard similarity. At the Jaccard similarity of 0.05, nearly half of the found mappings are non-lexical pairs. The coverage over all lexical mappings increases slowly up to around 20%. However, from the 273 lexical mappings that have a Jaccard similarity above zero (*i.e.*, there are joint instances) 95.6% are ranked among the top 1000 mappings.

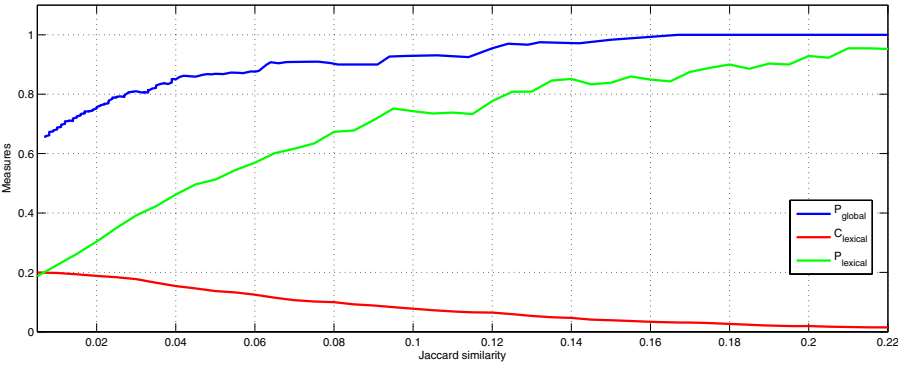
We carried out a manual evaluation on the top 1000 Brinkman–GTAA mappings. The purpose is to check whether the precision decreases and how much it decreases with the increasing number of non-lexical mappings. Among the top 1000 mappings, there are 261 lexically equivalent pairs, which we consider as correct mappings. The remaining 739 non-lexically equivalent pairs were presented to a Dutch-speaker, who judged each pair to be a valid mapping or not.

The evaluation results are analysed as follows. For each 10^{th} mapping in the list, we calculate the precision of all pairs within a window of size 40, 20 to the left and 20 to the right. This gives a local average precision, P_{local}, which is sensitive to its location in the list. It indicates, to some extent, the probability

[10] This Dutch language-specific lexical mapper makes use of the CELEX (`http://www.ru.nl/celex/`) morphology database, which allows to recognise lexicographic variants of a word-form, as well as its morphological components.



(a) Global and local precision



(b) Percentage and coverage of lexical mappings *vs*. the global precision

Fig. 2. Evaluation of mappings between Brinkman and GTAA

for a mapping to be correct in a certain neighbourhood when moving through the ranked list. The precision of all pairs from the top to the current pair, P_{global}, is also calculated, which indicates the precision from a more global view.

According to Fig. 2 (a), the global precision slowly decreases when moving further along the ranked mapping list. Although sensitive to its position in the ranked list, the local average precision also gradually decreases from 100% to 42.5%. A high precision of 71.6% at the 1000^{th} mapping tell us that the quality of the returned non-lexical mappings is quite good. The local average precision at the 1000^{th} mapping is still 42.5%,[11] which means that in this neighbourhood the mappings have an average probability of 42.5% to be correct. Depicted in Fig. 2 (b), with the increasing number of non-lexical mappings (from about 5% to nearly 80%), the precision does not decrease dramatically (from 100% to 71.6%). The nice results here illustrate the effectiveness of our method and its applicability to the heterogeneous case.

[11] In order to achieve this measure, the evaluator judged the 20 mappings after the 1000^{th} one.

Fig. 2 to some extent indicates that lexically equivalent pairs often do not have similar extensional semantics, especially when used in different collections or across domains. This is an indication to consider the reliability or limitations of using lexical mappings in certain applications where extensional semantics play an important roles, such as retrieving or browsing across collections.

5 Related Work

Ontology matching, as a promising solution to the semantic heterogeneity problem, has recently become an interesting and important research problem. Many different matching techniques have been proposed. In order to make use of various properties of ontologies (*e.g.*, labels, structures, instances or related background knowledge), existing matching techniques adopt methods from different fields (*e.g.*, statistics and data analysis, machine learning, linguistics). These solutions share some techniques and attack similar problems, but differ in the way they combine and exploit their results. A detailed analysis of the different matching techniques has been given in [3]. Examples of individual approaches addressing the matching problem and latest development in this area can be found on `www.OntologyMatching.org`.

The most related matching technique to our work is the instance-based methods, also called extensional matching techniques. The idea behind such techniques is that similarity between the extensions of two concepts reflects the semantic similarity of these concepts. Many current mapping tools, such as [12], make very limited use of instances, where instance information are only used complementarity to other techniques. Instance-based method has not been very widely investigated until recently [4,5,6,13], where neural networks, machine learning or statistics were used to model the complex correlation between instances and the semantics of concepts. However, instances are in general simply used as literals and the instance-based similarity normally results from the set operations, such as in [7].

A simple instance based method requires the existence of common instances. However, the explicitly shared instances are often not available, as ontologies in different applications contain similar but different individuals. As a sufficient amount of instance data becomes available, it has been proposed to use machine learning and statistics to grasp the relations between instances themselves. The similarity between instances using their own information, such as the metadata of individuals, has recently been investigated in [8]. The method proposed in this paper is another way of using instance as informative individuals by themselves, instead of treating them only as simple literals.

6 Conclusion and Future Work

In this paper, we propose to use a lexical search engine to map instances from different ontologies. By exchanging concept classification information between these mapped instances, we can generate an artificial set of common instances

shared by concepts from two ontologies, so that existing instance-based methods can apply. By comparing mappings between two thesauri, GTT and Brinkman, generated by explicit dually annotated instances and those by our method using singly annotated datasets, we have shown the feasibility of our method in a homogeneous case. Our experiments of mapping Brinkman and GTAA, using completely different and disjoint collections, have demonstrated this method to be an effective approach and applicable to a broad mapping context, *i.e.*, heterogeneous collections. To the best of our knowledge, this is new, and a very promising step towards effective semantic interoperability between different collections (*e.g.* in the Cultural Heritage domain).

In the future, we will further experiment with different query configurations in the instance mapping step, *e.g.*, the influence of single-field, multi-field and concatenated-field queries on the generated mappings, whether machine learning techniques can help map instances without many manual settings, *etc.*

In the GTT-Brinkman case, a threshold can be decided according to the optimal performance of the obtained mappings in a re-indexing scenario. However, it is not the case for the Brinkman-GTAA case. We will investigate more on how to find such optimisation tasks for deciding the relevant threshold parameters.

Finally, Lucene uses lexical information for answering queries. This hinders our method to be applied in a multi-lingual setting. In the future, we will explore the possibilities to increase the applicability of our method in this direction, such as using an automatic translation tool to reduce the language barrier.

References

1. Rahm, E., Bernstein, P.A.: A survey of approaches to automatic schema matching. VLDB J. 10(4) (2001)
2. Doan, A., Halevy, A.Y.: Semantic integration research in the database community: A brief survey. AI Magazine 26(1) (2005)
3. Euzenat, J., Shvaiko, P.: Ontology Matching. Springer, Heidelberg (2007)
4. Li, W.S., Clifton, C., Liu, S.Y.: Database integration using neural networks: Implementation and experiences. Knowledge and Information Systems 2, 73–96 (2000)
5. Doan, A.H., Madhavan, J., Domingos, P., Halevy, A.: Learning to map between ontologies on the semantic web. In: Proceedings of the 11th international conference on World Wide Web, pp. 662–673 (2002)
6. Ichise, R., Takeda, H., Honiden, S.: Integrating multiple internet directories by instance-based learning. In: Proceedings of the eighteenth International Joint Conference on Artificial Intelligence (2003)
7. Isaac, A., van der Meij, L., Schlobach, S., Wang, S.: An empirical study of instance-based ontology matching. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 253–266. Springer, Heidelberg (2007)
8. Wang, S., Englebienne, G., Schlobach, S.: Learning concept mappings from instance similarity. In: Proceedings of the 7th International Semantic Web Conference (ISWC 2007), Karlsruhe, Germany (to appear, 2007)

9. Isaac, A., Matthezing, H., van der Meij, L., Schlobach, S., Wang, S., Zinn, C.: Putting ontology alignment in context: Usage scenarios, deployment and evaluation in a library case. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) ESWC 2008. LNCS, vol. 5021, pp. 402–417. Springer, Heidelberg (2008)
10. Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. Commun. ACM 18, 613–620 (1975)
11. Pirro, B., Talia, D.: An approach to ontology mapping based on the lucene search engine library. In: Proceedings of the 18th International Conference on Database and Expert Systems Applications (DEXA 2007), Regensburg, Germany, pp. 407–411 (September 2007)
12. Hu, W., Qu, Y.: Falcon-AO: A practical ontology matching system. Journal of Web Semantics (2007)
13. Dhamankar, R., Lee, Y., Doan, A., Halevy, A., Domingos, P.: iMAP: Discovering complex semantic matches between database schemas. In: Proceedings of the ACM International Conference on Management of Data (SIGMOD), pp. 383–394 (2004)