Learning Concept Mappings from Instance Similarity

Shenghui Wang[1], Gwenn Englebienne[2], and Stefan Schlobach[1]

[1] Vrije Universiteit Amsterdam
[2] Universiteit van Amsterdam

Abstract. Finding mappings between compatible ontologies is an important but difficult open problem. Instance-based methods for solving this problem have the advantage of focusing on the most active parts of the ontologies and reflect concept semantics as they are actually being used. However such methods have not at present been widely investigated in ontology mapping, compared to linguistic and structural techniques. Furthermore, previous instance-based mapping techniques were only applicable to cases where a substantial set of instances was available that was doubly annotated with both vocabularies. In this paper we approach the mapping problem as a classification problem based on the similarity between instances of concepts. This has the advantage that no doubly annotated instances are required, so that the method can be applied to any two corpora annotated with their own vocabularies. We evaluate the resulting classifiers on two real-world use cases, one with homogeneous and one with heterogeneous instances. The results illustrate the efficiency and generality of this method.

1 Introduction

Motivation. The problem of semantic heterogeneity and the resulting problems of interoperability and information integration have been studied for over 40 years now. It is at present an important hurdle to the realisation of the Semantic Web. 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]. More directly, the current work was motivated by our work in the Cultural Heritage domain, in which we address interoperability problems within the Dutch National Library, and across collections with the Dutch Institute for Sound and Vision.

Previous work. A common way of judging whether two concepts from different ontologies are semantically linked is to observe their *extensional* information [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. Contrarily 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.

A. Sheth et al. (Eds.): ISWC 2008, LNCS 5318, pp. 339–355, 2008.

However, measuring the common extension of concepts requires the existence of sufficient amounts of shared instances, something which is often not the case. Furthermore, it only uses part of the available information, *i.e.*, ignores similarity between instances that have not been doubly annotated. Similarity on the instance-level is often ignored. In this paper we apply a more general *similarity-based extension comparison*, deriving concept mappings from similarity of their instances.

Method. In this paper we formulate matching problems as classification problems and develop a machine learning technique to learn the relationship between the similarity of instances and the validity of mappings between concepts. In many application contexts, information exists about the instances that are annotated. It is therefore possible to compute a measure of similarity between the instances. The main idea of our method is to use this similarity between instances to determine similarity (mappings) between concepts. Unlike previous methods, our method does not rely on the presence of doubly annotated instances.

We extend our previous work [7] in several ways: we apply a more fine-grained measure of instance-similarity by taking the meta-data description of instances into account. This allows us to go even further in two steps: firstly, to apply our method to collections for which no joint instances exist, and secondly, to collections in which the instances are described in different ways (heterogeneous collections).

Research questions. The above described method is based on a number of implicit assumptions, and the purpose of this paper is to evaluate their influence on the quality of the resulting mappings. The most important research questions are:

1. **RQ1:** Are the benefits from feature-similarity of instances in extensional mapping significant when compared to existing methods, such as based on simple co-occurrence information?

More specifically:

2. **RQ2 Joint instances:** Can our approach be applied to corpora for which there are no doubly annotated instances, i.e. for which there are no joint instances.
3. **RQ3 Heterogeneous collections:** Can our approach be applied to corpora in which instances are described in a heterogeneous way? To answer this question we have to answer a more technical question:
   - **RQ3.1 Feature selection:** Can we maintain high mapping quality when features are selected (semi)-automatically?
   - **RQ3.2 Training data:** can we maintain high mapping quality when there is no initial training set available in the first place?

and finally, from a domain perspective:

4. **RQ4 Feature weightings:** Can we make qualitative use of the learned model, more concretely the weightings of importance of similarities?

Experiments. To answer the above research questions we evaluate our method in the context of two real-world cases: (1) collections of books which have been annotated with two thesauri that are to be matched and (2) a collection of books and a "multimedia" collection, both of which have been annotated with their own thesauri, between which the mapping will improve the interoperability across collections.

The first application scenario stems from the Dutch National Library (Koninklijke Bibliotheek, or KB) which requires mappings between two thesauri both used to annotate two *homogeneous* book collections. The second scenario is related to supporting integrated online access of parts of the collections of the Dutch institute of Sound and Vision (Beeld en Geluid, or BG) and the KB, *i.e.*, a mapping between two thesauri, each used for describing a *heterogeneous* collection.

Findings. We show that our method is effective to map both thesauri which are used for homogeneous and thesauri used for heterogeneous collections. It improves significantly over the simple lexical and co-occurrence based method and over one state-of-the-art tool. Moreover, it also works with a disjoint instance space, when no common instances exist. We demonstrate how to use our method when initially no training data is available, *i.e.*, when there are no pairs of concepts for which we know that they should be mapped. This makes this methods generalisable to many other applications. A qualitative analysis on the learning results shows this method can also contribute to achieve a metadata-level interoperability.

Relevance for the Semantic Web. The paper is relevant for the Semantic Web in two aspects: first, as an application of Semantic Web technology (of course, all data and ontologies are represented in SW standards (RDF(S) and SKOS). Secondly, we contribute to the problem of ontology mappings, as our methods can be extended to any ontology mapping problem where information about concepts can be expressed as sets of similarity features.

What to expect from this paper. Beside contributing a novel formulation of the mapping problem and the definition of a mapping method, **instance-similarity mapping**, we provide a thorough empirical evaluation showing that our proposed method improves on the state of the art, even when no initial training data is available, and investigate how it can be generalised when no joint instances are available and the collections are heterogeneous. Finally, as a nice by-product, our method can be used for meta-data schema mapping.

Section 2 introduces our application context and matching problem statement. Section 3 presents the mapping method employed. In Section 4 we describe our experimental setup to validate our research questions before concluding.

2 Application Problems

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

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.[1] These two thesauri are individually used to annotate two book collections in KB. Both thesauri have similar coverage but differ in granularity.

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. 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 instances in both collections are books annotated with the same metadata structure, more specially, using an extension of the Dublin Core metadata standard.[2]

2.2 Heterogeneous Collections with Multiple Thesauri

Our second task is to match the two thesauri (Brinkman and GTT) from the KB to the thesaurus GTAA, which contains 160K concepts and has an average concept depth of 1.30817. The GTAA thesaurus is used to annotate the multimedia collection in the BG. The BG serves as the archive of the Dutch national broadcasting corporations, radio and television programmes that have been broadcast come into the archive continuously. Besides over 700,000 hours of material, the BG also houses 2,000,000 still images and the largest music library of the Netherlands. For our experiments, we used nearly 60K instances from this archive. Each object in the BG collection is annotated by several concepts from the GTAA thesaurus.

Mapping GTAA to one or both of the KB thesauri is very interesting from a Cultural Heritage perspective. 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. Different from the KB case, now the instance meta-data differs significantly.

In both cases, objects (books or multimedia objects) which are annotated by a thesaurus concept are considered as the instances of this concepts. In the next section, we will introduce in details the instance-similarity based mapping technique.

3 Mapping Method: Classification Based on Instance Similarity

Our concept mapping method is based on the similarity between instances, and automatic classification based on some training or seeding mappings. More concretely, we apply the following steps:

1. ontology **concepts are represented as feature vectors** (mostly information about instances, *e.g.* the content of their meta-data fields), as shown in Figure 1.
2. **similarity between two concepts** is represented as a vector of similarities between these features.

[1] Nearly 20K GTT concepts have no parents.

[2] `http://dublincore.org/documents/dces/`

3. a **classifier** learns the relation between instance similarity and concept mappings based on some training data. This classifier then estimates the probability whether an unseen pair of concepts should be mapped or not.

The trained classifier can then be used to determine whether new pairs of concepts should be mapped or not. Let us discuss each of the steps in a bit more detail.

3.1 Representing Concepts as Feature Vectors

It is common to most ontology mapping approaches that properties of concepts are collected, and compared in order to calculate a similarity score between pairs of concepts. In our use-cases, the most prominent knowledge about our concepts are from the books and multimedia objects annotated with the concepts. In [7] we showed already that the information about co-occurrence of instances can provide good mappings. In this paper, we extend this approach by including further information into the mappings process, in our case the metadata about the book and multimedia objects. This has the advantage that we can ignore whether the books are dually annotated, because similarity between the meta-data of the instances also reflects the relatedness of their annotation concepts.

We take the KB case as an example. As Figure 1 shows, books are normally described by their title, creator, publisher, *etc*. These features together represent an individual book instance. For each concept, all its instances are grouped into an integrated representation of the concept, feature by feature. For example, all titles of these books are put together as a “bag of words.” Term frequencies are measured within these bags. Thus, a concept is represented by a set of high-dimensional vectors of term frequencies, one per feature of the instances, which we consider as the features of the concept. When the instances share the same features, the similarity between the corresponding concepts is calculated with respect to each feature, using the cosine similarity between the term frequency vectors of corresponding features. This is the “homogeneous collection” case that we mentioned in our introduction.

Notice that, although in our case the whole corpus is in Dutch, these similarity features can be chosen to be language-independent, *e.g.* ISBN numbers, proper names of creators or actors, publishers, dates, *etc*. This method is therefore usable in a multi-language context.

When no straightforward relationship exists between the metadata of instances, as is the case with heterogeneous collections, we compute the similarity between all pairs of the metadata fields and evaluate which of those are informative; see Section 4.2 for more details. In the end we obtain set of similarity measures, encoded in one vector per pair of concepts, which reflects the similarity of their instances. The classifier deals with this vector only, and sees it as the feature vector of the pair of concepts.

In fact, the feature vector needs not be limited to the instance similarity used here. It is trivial to extend it with the lexical similarity between the concept labels or with structure-based measures of similarity. The classifier would then learn to weigh those appropriately, creating a powerful, integrated solution. However, these measures are not always available, *e.g.* in a multilingual setting or when dealing with ontologies of little structure. We do not include such features in this paper and focus on instance-based features only.

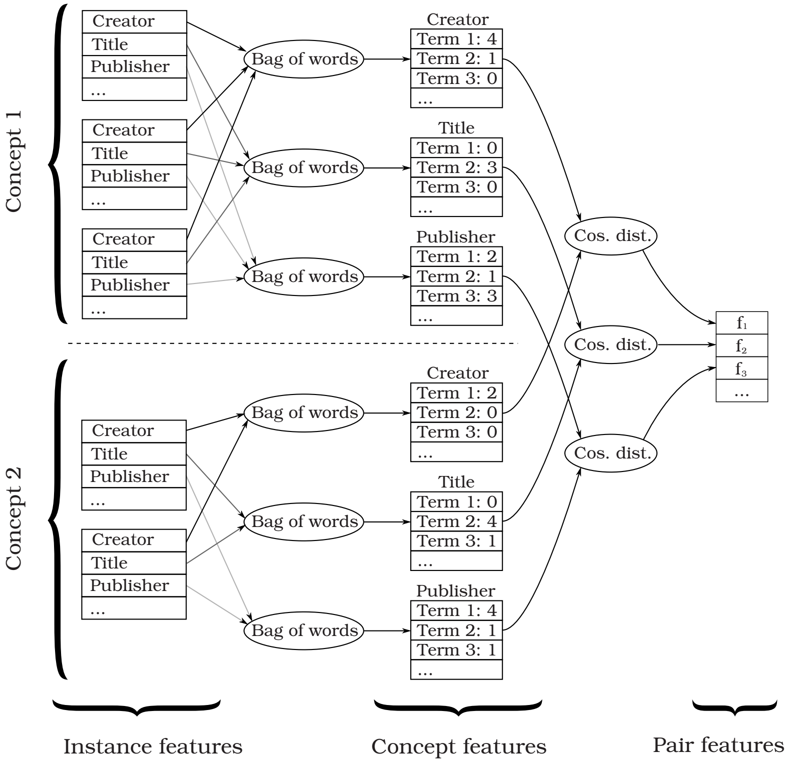


Fig. 1. Feature extraction for a single concept pair. Term frequencies are calculated from the combined features of the instances of the concepts. The cosine distances of these frequency vectors are used as the features of the pair. See text for details.

3.2 Representing Similarity of Concepts

The similarity between a pair i of concepts is measured and represented by a vector $\mathbf{x}^{(i)}$, where each element j of $\mathbf{x}^{(i)}$, denoted $\mathbf{x}_j^{(i)}$, represents the similarity between the features j of the concepts. These similarity vectors can be treated as points in a space. In this "similarity space" of concept pairs, each dimension corresponds to the similarity between features of the concepts. As we know, some points (*i.e.*, some pairs of concepts) correspond to real mappings but some are not. Our hypothesis is that the *label* of a point — which represents whether the pair is a *positive* mapping or *negative* one — is correlated with the position of this point in this space.

Given some known mappings, *e.g.* from a manual selection by experts, our goal is to *learn* this correlation. Therefore the mapping problem is transformed into a classification problem. With already labelled points and the actual similarity values of concepts involved, it is possible to classify a point, *i.e.*, to give it a right label, based on its location given by the actual similarity values.

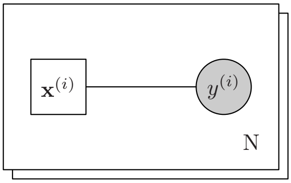


Fig. 2. Graphical representation of the MRF used in this work. The shaded, circular node denotes a hidden, discrete variable and the clear square node denotes the observed, multidimensional variables. This model is repeated N times, once for each data element.

3.3 The Classifier Used: Markov Random Field

We use a Markov Random Field (MRF, [8]) to model the classification-based mapping problem. Let $T = \{ (\mathbf{x}^{(i)}, y^{(i)}) \}_{i=1}^{N}$ be the training set containing N mappings, with, for each given pair of concepts i, a feature vector $\mathbf{x}^{(i)} \in \mathbb{R}^K$ and an associated label $y^{(i)} \in \mathcal{Y}$, where K is the number of features of a pair of instances and $\mathcal{Y}$ is the set of possible values the label can take. Here, the label $y^{(i)}$ is either *positive* or *negative*, although this can be extended to a set of possible mapping relations, such as *exactMatch*, *broadMatch*, *narrowMatch*, *relatedMatch* or *noLink*.

We consider a simple graphical model, consisting of an observed multivariate input $\mathbf{x}$ and a single random variable y (see Figure 2). The input is a vector of the similarity features, the random variable represents the possible values of the label and associated probabilities. We assume the mappings are conditionally independent and identically distributed (*i.i.d.*), conditionally on the observations, and model the conditional probability of a mapping given the input, $p(y^{(i)}|\mathbf{x}^{(i)})$, using a probability distribution from the exponential family. That is:

$$p(y^{(i)}|\mathbf{x}_i, \theta) = \frac{1}{Z(\mathbf{x}_i, \theta)} \exp\big(\sum_{j=1}^{K} \lambda_j \phi_j(y^{(i)}, \mathbf{x}^{(i)})\big), \tag{1}$$

where $\theta = \{ \lambda_j \}_{j=1}^{K}$ are the weights associated to the potential function and $Z(\mathbf{x}_i, \theta)$, called the partition function, is a normalisation constant ensuring that the probabilities of the mutually exclusive labels sum to 1. It is given by

$$Z(\mathbf{x}_i, \theta) = \sum_{y \in \mathcal{Y}} \exp\big(\sum_{j=1}^{K} \lambda_j \phi_j(y^{(i)}, \mathbf{x}^{(i)})\big). \tag{2}$$

The resulting model can be seen as a Conditional Random Field (CRF, [9]) of length zero. Since we assume that the mappings of different concept pairs are independent of each other, the likelihood of the data set for given model parameters $p(T|\theta)$ is given by:

$$p(T|\theta) = \prod_{i=1}^{N} p(y^{(i)}|\mathbf{x}^{(i)}) \tag{3}$$

During learning, our objective is to find the most likely values for θ for the given training data. We can obtain this using Bayes' rule if we assume some *prior probability* distribution $p(\theta)$ on the parameters. We here chose a prior which favours small values, that we model with a normal distribution with zero mean and covariance σ^2 for each λ_i, as this penalises overly flexible models and this reduces over-fitting. The *posterior* probability of θ is then given by

$$p(\theta|T) = \frac{p(T|\theta)p(\theta)}{p(T)}, \tag{4}$$

where $p(T)$ is a normalisation term which does not depend on θ and can therefore be ignored during optimisation. Moreover, since the logarithm is a monotonically increasing function, we can optimise $\log p(\theta|T)$ rather than $p(\theta|T)$. This is simpler, as it involves taking the derivative of a sum rather than of a product over all data points. Ignoring additive constants, which do not affect the derivative, the function we optimise is then:

$$\log p(\theta|T) = \sum_{i=1}^{N} \left[\sum_{j=1}^{K} \lambda_j \phi_j(y^{(i)}, \mathbf{x}^{(i)}) - \log Z(\mathbf{x}^{(i)}) \right] - \sum_{j=1}^{K} \frac{\lambda_j^2}{2\sigma^2}. \tag{5}$$

This function cannot be optimised in closed form because of the logarithm of a sum in the partition function. However it is a convex function which can easily be optimised numerically using any variation of gradient ascent, although (quasi-)Newton methods have proven best suited [10]. We used the limited memory BFGS method to obtain the results presented here [11]. The first derivative of eq. 5 is given by

$$\frac{\partial p(\theta|T)}{\partial \lambda_j} = \sum_{i=1}^{N} \left[\phi_j(y^{(i)}, \mathbf{x}^{(i)}) - \sum_{L \in \{0,1\}} \phi_j(y^{(i)}, \mathbf{x}^{(i)}) p(y^{(i)}|\mathbf{x}^{(i)}, \theta) \right] - \frac{\lambda_j}{\sigma^2} \tag{6}$$

The variance of the prior, σ, is a parameter that has to be set by hand and can be seen as a regularisation parameter which prevents over-fitting of the training data.

Once the model is trained, we use the resulting parameters to compute the probability of a label for a pair of concepts. The decision criterion for assigning a label $y^{(i)}$ to a new pair of concepts i is then simply given by:

$$y^{(i)} = \underset{y}{\operatorname{argmax}}\, p(y|\mathbf{x}^{(i)}) \tag{7}$$

That is, for a given pair of concepts, the label with the highest probability given the pair's feature vector is assigned. Note that in settings where a higher cost is associated with one type of error than the other, another threshold could be set. For example, if the system were used to propose candidates for mapping, missed mappings would be worse than erroneously proposed candidate pairs. In such a case, we could set the system to propose the pair if, say, $p(y = \text{positive}|\mathbf{x}^{(i)}) > 0.3$.

4 Description of the Experiments

The goal of our experiments is to show the effectiveness of our approach in general, and to evaluate the influence of three factors: the existence of joint instances, the use of heterogeneous rather than homogeneous collections for mapping, and how to build a representative training data set when no hand-made initial training set is available.

Table 1. Numbers of positive examples in the training sets

Thesauri	lexical equivalent mapping	non-lexical mapping
GTAA vs. GTT	2720	116
GTAA vs. Brinkman	1372	323

Experimental setup. All of our experiments are set up in the same way. We map two thesauri which are used to annotate instances either from the homogeneous, or from heterogeneous collections.

Training data. Ideally, the training set should be representative enough to model the relation between instance similarity and concept mappings. For the first KB case, we used the manually-built golden standard from [7] as the training set. This set contains a balanced number of positive and negative examples; where all positive mappings are non-lexically but semantically equivalent pairs. For the second BG case, no such handcrafted golden standard is available. Of course, we can build a training set manually, as we did in the GTT and Brinkman case. But this is a very time consuming task, especially when two thesauri to be mapped are very big. To overcome this problem we used two ways to construct a training set automatically.

- *(lexical seeding)* One assumption we take is that concepts with the same label form a valid mapping. Therefore, we applied a simple lexical mapper[3] to select the positive examples. The same number of pairs of concepts are selected at random as negative data from the set of pairs that were not lexically matched. Some true mappings may therefore conceivably be present among the negative training examples, but the probability of this occurring is negligible. A more serious problem, however, is the strong bias towards lexical similarity of this data set.
- *(background seeding)* A way to find non-lexically equivalent concepts which are semantically equivalent is to use "background knowledge" [12]. More concretely, for mapping Brinkman concepts to those of GTAA, we use GTT as a background knowledge. In our previous work, many Brinkman concepts are mapped to GTT concepts using the co-occurrence based techniques [7]; some of these mappings are non-lexical ones. For each Brinkman with no lexical mapping to any GTAA concept but which is mapped to a GTT concept, we check whether the corresponding GTT concept is lexically mapped to a GTAA concept. If there is such a link, then the Brinkman and the GTAA concept are considered as a mapping.

Table 1 lists the size of the training sets built by the above two ways.

Evaluation. We apply two different types of evaluation: standard 10-fold cross-validation, and testing on a specific test set. The former is applied whenever possible as it provides us with an estimate of the reliability of the results. However, as we will see, when evaluating

[3] 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.

how the selection of the training data affects the results, we need to keep the training sets separate and cross validation is not possible.

The quality of our methods is measured quantitatively using the *misclassification rate* or *error rate*, *i.e.*, the number of wrongly classified pairs over the total number of pairs. This is an appropriate measure when the training and test data sets have balanced numbers of positive and negative examples. At the end of Section 4.3, we will briefly discuss the case of more skewed distributions of positive and negative examples.

From a more qualitative point of view, we also analyse the respective importance of the different features, based on the explicit (learned) weights of these features.

4.1 Experiment I: RQ1: Feature-Similarity Based Mapping Versus Existing Methods

The purpose of the following experiment is to compare our new method with existing methods. The task is to map GTT and Brinkman concepts given books from the two KB collections. We compare our approach with the co-occurrence based method detailed in [7], a simple lexical approach, and the state-of-the-art Falcon ontology mapper.[4] An existing golden standard that was built manually by experts, including 747 positive and negative examples, is used for the evaluation.

We compare the following methods:

Falcon: we apply the Falcon mapper to map GTT and Brinkman. We then calculate the error rate by considering all mappings returned by Falcon as classified positive and all the rest pairs as classified negative.

S_{jacc}, the Jaccard similarity between concepts is measured based on 222K dually annotated books. It is defined as the number of books that have been annotated by both concepts over the total number of books that have been annotated by those concepts. As described in [7] we apply a simple adaption to exclude concepts with too few instances.

S_{lex}, the relative edit distance between the labels (including "prefLabel" and "altLabel") of the two concepts is measured and the minimum distance is kept as the lexical similarity of these two concepts.

S_{bag}, where all information was put into a single bag, and similarity is calculated based on one bag of words.

$\{f_1, \ldots, f_{28}\}$ is an instance of our mapping method, where the similarity between each field of the instances was computed separately as depicted in Figure 1.

We train four classifiers on S_{jacc}, S_{lex}, S_{bag} and $\{f_1, \ldots, f_{28}\}$, respectively, and estimate the error-rate using 10-fold cross-validation. When trained on a single feature (*e.g.*, S_{jacc}, S_{lex} and S_{bag}), the classifier simply learns a threshold that separates mapped from non-mapped concepts.

The results are summarised in Table 2: The classifiers based solely on lexical or Jaccard similarity perform slightly better than chance level. Just calculating similarity between the complete information of all instances in S_{bag} is even worse. It is obvious

[4] http://iws.seu.edu.cn/projects/matching/

Table 2. Comparison between existing methods and similarities-based mapping, in KB case

Mapping method	Error rate
Falcon	0.28895
S_{lex}	0.42620 ± 0.049685
S_{jacc80}	0.44643 ± 0.059524
S_{bag}	0.57380 ± 0.049685
$\{f_1, \ldots f_{28}\}$ (our new approach)	0.20491 ± 0.026158

Table 3. Comparison between classifiers using joint and disjoint instances, in KB case

Collections	Testing set	Error rate
Joint instances (original KB corpus)	golden standard (representative)	0.20491 ± 0.026158
	lexical only	0.137871
No joint instances (double instances removed)	golden standard (representative)	0.28378 ± 0.026265
	lexical only	0.161867

that based on this information alone, the classification does not work. Falcon outperforms all three of these methods.

Using the similarity of different fields separately reduces the error rate to a more acceptable level of around 20%, and significantly outperforms all other methods.

4.2 Experiment II: RQ2: Extending to Corpora without Joint Instances

As mentioned in Section 2, there is an overlap between the two collections in KB, so that there are books which are dually annotated, by both GTT and Brinkman concepts. This allows us to apply methods based on co-occurrence to find mappings [7]. However, it is not always the case that two thesauri have joint instances. In this section we evaluate whether our approach can be applied to the case where there are no joint instances, *i.e.*, where there are no doubly annotated instances.

To determine the influence of joint instances on our mapping approach we trained two classifiers: one on the complete KB corpus, and one on the same corpus minus the joint instances, *i.e.* keeping books which are annotated by either GTT or Brinkman concepts, but not by both. For this purpose, we used the same golden standard used above, containing 747 concept pairs for all of which joint instances exist.

We applied two evaluation methods: first, 10-fold cross-validation on the golden standard, and second, testing on a set of purely lexically equivalent concepts. We did this to check whether instance-similarity-based methods could recover the mappings found by lexical techniques.

The results in Table 3 show that inferring the mapping from disjoint instances results in a higher error rate. This is not surprising, as instance co-occurrence information is implicitly distributed among the features, and the existence of joint instances is indeed a more direct indicator of the similarity between concepts. Yet even without joint instances, the classifier performs reasonably well, surely good enough for many applications. This indicates that our method can be extended to situations where joint instances are not available.

On lexically equivalent pairs of concepts the error rate is actually significantly lower than the average error rate tested on the golden standard. This means that by using instances alone, the classifier trained on semantically equivalent concepts works well enough on classifying lexically equivalent mappings. Finding mappings between lexically equivalent concepts tends to be easier than between concepts which do not have lexically equivalent labels: some instance features, such as title or subject, tend to contain words that are closely related to the concept labels. As we will see below, classifiers trained on concept pairs that are lexically matched therefore tend to perform worse when tested on pairs that should be mappings but have no lexically equivalent labels, than the other way around.

4.3 Experiment III: RQ3: Extending to Heterogeneous Collections

The instances used in the previous experiments are books from two collections of KB. These books are described using the same metadata structure, which allows a straightforward measure of similarity. Such a shared structure is however not available when finding mappings between thesauri which are used for different collections with heterogeneous metadata structures. In this section, we evaluate the effect of applying our method to corpora for which instances are described in a heterogeneous way.

In order to work with heterogeneous collections we need to select features to model the similarity space.

RQ3.1: *Feature selection.* The method to construct the pair features outlined in Figure 1 requires corresponding metadata fields. In heterogeneous collections these are most likely not really available. We have some choices to construct a vector of pair features. We will first discuss our options before evaluation the effect on the mapping quality.

1. *(exhaustive combination)* As instances in different collections have different metadata structures, a naïve approach is to ignore the meaning of the fields and to calculate the similarity between all possible pairs of fields exhaustively. In our case, this similarity vector then has 28×38 (1064) dimensions. Theoretically, as the dimensionality of the data grows, an exponentially large number of training examples is required, which is unrealistic in practice. However, the similarity vector in practice can be very sparse — as in our case, fields such as "date" and "creator" do not have any similarity — so that over-fitting might not be too serious.

In order to avoid the potential over-fitting problem, we need to reduce the dimensionality of the data. We have two options:

2. *(manual selection)* To manually select corresponding metadata fields and calculate the similarity between the selected fields. In our case, among 28 fields in KB collection and 38 fields in the BG collection, we found eight shared fields which were therefore chosen for the similarity measuring.
3. *(mutual information)* To select the most informative fields automatically. We computed the mutual information between the label (*positive* or *negative*) and each of

Table 4. Comparison of the performance with different methods of feature selection, using non-lexical dataset

Thesaurus	Feature selection	Error rate	Thesaurus	Error rate
GTAA *vs.* Brinkman	manual selection	0.11290 ± 0.025217	GTAA vs. GTT	0.10000 ± 0.050413
	mutual information	**0.09355** ± 0.044204		**0.07826** ± 0.044904
	exhaustive	0.10323 ± 0.031533		0.11304 ± 0.046738

the exhaustively computed similarities. This indicates how informative the similarity between each combination of instance features is to predict the label. We then select a number of the most informative field combinations: In our case, we arbitrarily selected the 30 most informative fields.

We now investigate how this choice influences the performance of classifiers. We map GTAA to GTT/Brinkman and test the performance using three selections of similarity features. Results are given in Table 4. We see that, due to the nature of our data, the differences in error rate are not statistically significant here. However, the exhaustive enumeration of all feature pairs results in a very high-dimensional feature space. The corresponding classifier therefore much more prone to over-fitting than the other feature selections, not to mention the computational overhead in time and space. Yet, it should also be noted that training with the exhaustive features still only takes around 8 minutes for a set of 4896 examples, on a 2.3GHz Core II processor, as compared to around 15 seconds for the features selected by mutual information. Classification of new pairs is essentially instantaneous.

Automatic selection of features according to mutual information gives the best results, thus answering our third research question: automatic feature selection does result in good mappings. This is also of practical value, as the lower dimensionality of the resulting data leads to reduced computational costs.

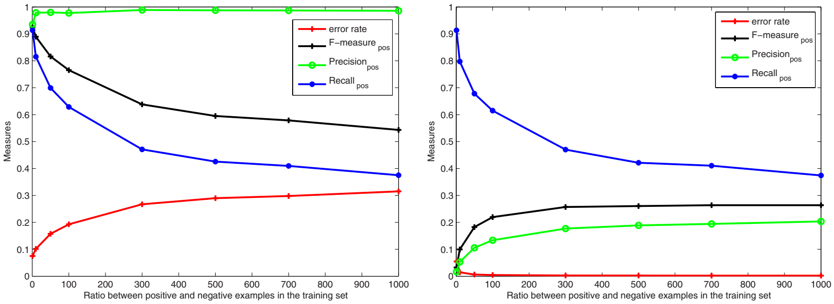
RQ3.2: *Training set.* We mentioned above that the construction of a training set by hand is expensive and often impractical, and that we devised two automatic seeding methods: lexical and background seeding. Background seeding can provide us with semantically valid mappings, but the background knowledge is often hard to obtain. Lexical seeding has the advantage that it is easily applied and that the training-sets can easily be quite large, compared to manually created sets. However, we will have to evaluate how this *biases* the training set and thus the classifier.

In the following experiment, we train two classifiers based on two training sets which were automatically generated by these two methods. Table 5 compares the results, using automatically selected features.

As we mentioned in Section 4.2, when tested on lexical mappings, both classifiers, whether trained on lexical mappings or on non-lexical mappings perform well. However when tested on non-lexical mappings, the classifier trained on lexical mappings performs much worse than the one trained on non-lexical mappings. This indicates that when trained on lexical mappings only, the resulting classifier is biased towards pairs of concepts with lexically similar labels and exhibits a degraded performance on

Table 5. Comparison using different datasets (feature selected using mutual information)

Thesauri	Training set	Test set	Error rate	Thesauri	Error rate
GTAA *vs.* Brinkman	non-lexical	non-lexical	0.09355 ± 0.044204	GTAA *vs.* GTT	0.07826 ± 0.044904
	lexical	non-lexical	0.11501		0.098712
	non-lexical	lexical	0.07124		0.088603
	lexical	lexical	0.04871 ± 0.029911		0.06195 ± 0.008038



(a) testing on 1:1 data

(b) testing on 1:1000 data

Fig. 3. The influence of positive-negative ratios in Brinkman-GTAA mapping

non-lexical mappings. This is an interesting finding, as it suggest that the use of a non-lexical training set is more generic.

We further investigated the influence of the ratio of positive-negative examples during training and testing. We trained on different datasets with the positive-negative ratio varying from 1:1 to 1:1000 and tested on two datasets with 1:1 and 1:1000 positive-negative examples. When training and testing on the data with the same or similar positive-negative ratio, our method performs well, see the left of Fig. 3 (a) and the right of (b). Note, when training on very few positive examples but many negative examples, the error rate could stay very low due to the correct classification of negative examples while the the precision for positive mappings could be low, as the classifier is focused on classifying negative examples and the predictive power on positive examples is therefore not optimised. In practice, the training data should be chosen so as to contain a representative ratio of positive and negative examples, while still providing enough material for the classifier to have good predictive capacity on both types of examples.

4.4 Experiment IV: RQ4: Qualitative Use of Feature Weights

The final set of experiments is a qualitative analysis of the explicit knowledge extracted from the classifiers: the weighting of the features. The training process results in a set of weightings for the features used, *i.e.*, $\theta = \{\lambda_j\}_{j=1}^{K}$, as introduced in Section 3. The value of λ_j reflects the importance of the feature f_j in the process of determining similarity (mappings) between concepts.

When mapping between the GTT and Brinkman thesauri with disjoint instances from the two KB collection, we indeed found fields with descriptive information, such as

Table 6. Examples of informative pairs of metadata fields from the exhaustive feature list

KB fields	BG fields
kb:title	bg:subject
kb:abstract	bg:subject
kb:annotation	bg:LOCATIES
kb:annotation	bg:SUBSIDIE
kb:creator	bg:contributor
kb:creator	bg:PERSOONSNAMEN
kb:Date	bg:OPNAMEDATUM
kb:dateCopyrighted	bg:date
kb:description	bg:subject
kb:publisher	bg:NAMEN
kb:temporal	bg:date

kb:abstract, kb:subject, kb:title, are informative for the classification process. We also find some language-independent fields, such as kb:ISBN and kb:contributor, are similarly important. This indicates this method can be applied in a multi-lingual setting, as long as the features are chosen so as to be language-independent.

When mapping from GTT and Brinkman to GTAA in the heterogeneous BG case, the features are similarities calculated from the exhaustive combination of all metadata fields. By observing the features with large λ values, we can find interesting links between those fields. Basically, we expect a feature in the exhaustive set (which is the Cartesian product between the two instance-feature sets) that corresponds to a high value of λ to indicate that the meta-data fields are related.

As introduced above, eight pairs of fields, such as kb:subject–bg:subject, kb:issued–bg:date, *etc.*, were manually selected. Among the top 30 features with the highest mutual information, about half of these manually selected pairs were present. Investigating the λ values of the features, from the classifier trained on the exhaustive feature set, shows that other pairs of fields, listed in Table 6, are also informative. That is, the similarity between these fields is important to determine the similarity between concepts. This in itself provides useful information for mapping metadata fields, and can also help to achieve interoperability at the meta-data level across different collections.[5]

5 Conclusion

In this paper, we have proposed a machine learning method to automatically use the similarity between instances to determine mappings between concepts from different thesauri/ontologies. This method has the advantage that it does not rely on concept labels or ontology structure. It can therefore be used when other methods fail, *e.g.* in multi-lingual settings or when the ontologies have very little structure.

[5] Similar (and sometimes complementary) results can be obtained using the mutual information as done for feature selection, however this is out of the scope of this paper.

A major improvement of this method, compared to previous instance-based methods, is that it does not require dually annotated instances, *i.e.*, common instances. Instead it uses a more fine-grained similarity at the instance level and uses a classifier to learn the relationship between instance similarity and concept mappings. We have shown that using feature-similarity of instances provides significant improvements when compared to existing methods, such as methods based on the co-occurrence of instances, or on the lexical similarity of concept labels. Moreover, we have demonstrated that our method can be applied when instances are heterogeneous. In our experiments, we have obtained good results when mapping thesauri annotating book and multimedia collections for which the instances are strongly dissimilar and only heterogeneous metadata was available. Finally, the method also works when no initial hand-crafted training mappings are available, as we have shown that using training sets of lexical mappings, or mappings generated using some background knowledge, can still provide high quality results.

A qualitative analysis on the resulting parameters λ allowed us (1) in the homogeneous case, to observe which metadata fields play important roles for mapping decisions; (2), more interestingly, in heterogeneous case, to find some links between metadata fields from different collections. Though a by-product rather than the core of our research, we consider this to be a nice contribution to the field of meta-data mapping.

In the future, we intend to apply our method to other collections, *e.g.* multilingual collections and to investigate integration with other techniques, *e.g.* based on lexical similarity or structure-based methods.

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: Proceedings of the 6th International Semantic Web Conference, Busan, Korea (2007)
8. Kindermann, R., Snell, J.L.: Markov Random Fields and their applications. American Mathematical Society (1980)
9. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proc. 18th International Conf. on Machine Learning, pp. 282–289 (2001)

10. Sha, F., Pereira, F.: Shallow parsing with conditional random fields. In: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, Edmonton, Canada, vol. 1, pp. 134–141. Association for Computational Linguistics (2003)
11. Nocedal, J.: Updating quasi-newton matrices with limited storage. Mathematics of Computation 35, 773–782 (1980)
12. Aleksovski, Z., ten Kate, W., van Harmelen, F.: Ontology matching using comprehensive ontology as background knowledge. In: Shvaiko, P., et al. (eds.) Proceedings of the International Workshop on Ontology Matching at ISWC 2006, CEUR, pp. 13–24 (2006)