Literature-based alignment of ontologies

Patrick Lambrix and He Tan and Wei Xu

Department of Computer and Information Science
Linköpings universitet, Sweden

Abstract. In recent years many ontologies have been developed and many of these ontologies contain overlapping information. To be able to use multiple ontologies they have to be aligned. Recently, there is a growing interest to use instance-based methods for ontology alignment and some of these approaches use literature. In this paper we propose and evaluate new strategies for aligning ontologies based on text categorization of literature using support vector machines-based text classifiers, and compare them with existing literature-based strategies. We also compare and combine these strategies with linguistic strategies.

A shorter version of this paper appears in the proceedings of the Third International Workshop on Ontology Matching, 2008.

1 Introduction

In recent years many ontologies have been developed. The benefits of using ontologies include reuse, sharing and portability of knowledge across platforms, and improved documentation, maintenance, and reliability (e.g. [5]). Ontologies lead to a better understanding of a field and to more effective and efficient handling of information in that field. Many of the currently developed ontologies contain overlapping information. For instance, Open Biomedical Ontologies (OBO, http://www.obofoundry.org/) lists 26 different anatomy ontologies (May 2008). Often we would want to be able to use multiple ontologies. For instance, companies may want to use community standard ontologies and use them together with company-specific ontologies. Applications may need to use ontologies from different areas or from different views on one area. Ontology builders may want to use already existing ontologies as the basis for the creation of new ontologies by extending the existing ontologies or by combining knowledge from different smaller ontologies. In each of these cases it is important to know the relationships between the terms in the different ontologies. Further, the data in different data sources in the same domain may have been annotated with different but similar ontologies. Knowledge of the inter-ontology relationships would in this case lead to improvements in search, integration and analysis of data. It has been realized that this is a major issue and some organizations have started to deal with it.

In the remainder of this paper we say that we align two ontologies when we define the relationships between terms in the different ontologies. Currently, there exist a number of ontology alignment systems that support the user to find inter-ontology relationships. For overviews we refer to, e.g., [2, 8, 15, 9] and the Ontology Matching website (http://www.ontologymatching.org/). These systems use different techniques. Recently, there is a growing interest in instance-based methods for ontology alignment. Some of these approaches use literature as instances.

In this paper we slightly generalize the method for literature-based ontology alignment that was proposed in [18] (section 2.2). Further, we propose an instantiation of the method based on text categorization. We propose two algorithms using support vector machines (SVMs) (section 3). We evaluate these algorithms in terms of the quality of the alignment results for the five test cases used in [18] (section 4). We discuss the influence of the literature corpus and compare the two SVM-based algorithms. We also compare these algorithms with the Naive Bayes text classification approach of [18]. Finally, we compare the algorithms with a good text-based approach and discuss the advantages and disadvantages of combining the approaches.

2 Background

2.1 Ontology alignment

Many ontology alignment systems are based on the computation of similarity values between terms in different ontologies and can be described as instantiations of the general framework defined in [9] (figure 1). An alignment algorithm receives as input two source ontologies. The algorithm can include several matchers. These matchers calculate similarities between the terms from the different source ontologies and can be based on knowledge about the linguistic elements, structure, constraints and instances of the ontology. Also auxiliary information can be used. Alignment suggestions are then determined by combining and filtering the results generated by one or more matchers. The pairs of terms with a similarity value above a certain threshold are retained as alignment suggestions. The suggestions are then presented to the user who accepts or rejects them. The acceptance and rejection of a suggestion may influence further suggestions.

2.2 Document-based Ontology Alignment

A method for creating a matcher that uses scientific literature was proposed in [18]. The method builds on the intuition that a similarity measure between concepts in different ontologies can be computed based on relationships between the documents in which they are used.

We give here a slightly generalized version of the method in [18]. It contains the following basic steps.

1. **Generate corpora.** For each ontology that we want to align we generate a corpus of documents.

2. **Generating classifiers.** For each ontology one or more document classifiers

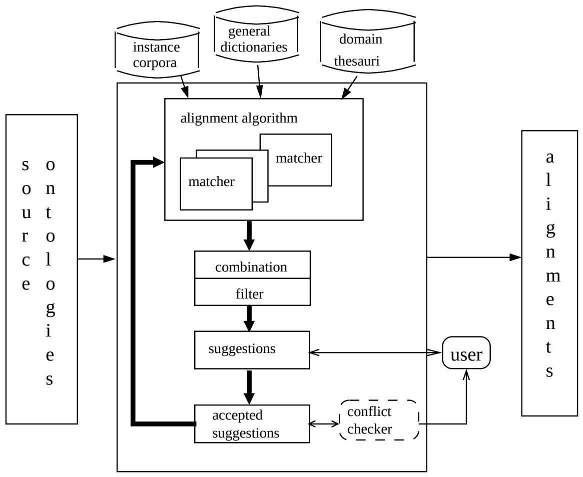

Fig. 1. A general alignment strategy [9].

are generated. The corpus of documents associated to an ontology is used for generating its related classifiers.
3. **Classification.** Documents of one ontology are classified by the document classifiers of the other ontology and visa versa.
4. **Calculate similarities.** A similarity measure between concepts in the different ontologies is computed based on the results of the classification.

2.3 Document-based Ontology Alignment using Naive Bayes text classifiers

In [18] a particular instantiation of this method was implemented and evaluated using test cases involving biomedical ontologies. For step 1 a corpus of PubMed [14] abstracts was generated. PubMed is a service of the National Library of Medicine that includes over 15 million citations from MEDLINE and other biomedical journals. The PubMed version of October 23, 2005 was queried with each concept in the ontologies and the 20, 40, 60, 80 and 100 most recent abstracts for each concept were retrieved. The retrieval system for PubMed did not always find at least 100 abstracts for a concept (even not always 20) and in this case all abstracts were retrieved. There was also no apparent relationship between the location of a concept in the is-a hierarchy and how many abstracts were retrieved for that concept.

In step 2 one Naive Bayes classifier per ontology was generated. The classifiers return for a given document d the concept C in the ontology for which the posterior probability $P(C|d)$ results in the highest value.

In step 3 the Naive Bayes classifier for one ontology was applied to every abstract in the abstract corpus of the other ontology and vice versa.

Finally, in step 4 a similarity value between a concept C_1 from the first ontology and a concept C_2 from the second ontology was computed as:

$$sim(C_1, C_2) = \frac{n_{NBC2}(C_1, C_2) + n_{NBC1}(C_2, C_1)}{n_D(C_1) + n_D(C_2)}$$

where $n_D(C)$ is the number of abstracts originally associated with C, and $n_{NBCx}(C_p, C_q)$ is the number of abstracts associated with C_p that are also related to C_q as found by classifier $NBCx$ related to ontology x.

2.4 Text Categorization using Support Vector Machines

In step 2 of the method described in section 2.2 document classifiers are generated. In [18] one Naive Bayes classifier per ontology was generated which assigns a document to one concept in the ontology. In general, however, a document may be assigned to several concepts and thus we may regard the classification of documents to concepts in an ontology as several binary classification problems, one for each concept in an ontology.

SVMs [19] is a machine learning method that constructs a separating hyperplane in a feature space between two data sets (positive and negative examples) which maximizes the margin between the two sets. The setting can also be generalized to learning from positive and unlabeled examples (e.g. [10]). It has been argued that SVMs are an appropriate learning method for generating text classifiers [6]. They are universal learners, they can learn independent of the dimensionality of the feature space, and they allow for automatic parameter tuning. According to [6] theoretical evidence for the fact that SVMs should perform well for text categorization is given by the facts that (i) text has a highly dimensional feature space (often tf-idf vectors are used) and SVMs have the potential to handle large feature spaces, (ii) text has few irrelevant features and thus large feature spaces need to be handled, (iii) text has sparse feature vectors and algorithms with similar inductive bias are well suited to problems with dense concepts and sparse instances, and (iv) most text categorization problems are linearly separable and SVMs try to find such linear separators. Experimental results also showed that SVMs achieved good performance and are robust.

3 Alignment algorithms

In this section we present an instantiation of the method in section 2.2 based on text classification using SVMs.

3.1 Basic algorithm

1. **Generate corpora.** We used the exact same corpora as in [18].
2. **Generating the classifiers.** For each concept in each ontology an SVM text classifier was generated. We used the LPU [10] system. LPU generates text classifiers based on positive and unlabeled examples. The abstracts retrieved

when querying for a concept were used as positive examples for that concept. Further, for a given concept we used one abstract of each other concept in the same ontology as unlabeled examples.

The SVM text classifier for a concept returns for a given document whether the document is related to the concept. It also returns a value that is positive if the document is classified to the concept and negative otherwise.

3. **Classification.** The SVM text classifier for each concept in one ontology is applied to every abstract in the abstract corpus of the other ontology and vice versa. The classification was done by using the text classifiers generated by LPU within the SVM^{light} system [7]. Observe that a document can be classified to zero, one or more than one concept in an ontology. This is in contrast to the Naive Bayes text classification approach where a document was classified to exactly one concept.

4. **Calculate similarities.** As the last step we compute a similarity between concepts in different ontologies. We define the similarity between a concept C_1 from the first ontology and a concept C_2 from the second ontology as:

$$\frac{n_{SVMC-C_2}(C_1, C_2) + n_{SVMC-C_1}(C_2, C_1)}{n_D(C_1) + n_D(C_2)}$$

where $n_D(C)$ is the number of abstracts originally associated with C, and $n_{SVMC-C_q}(C_p, C_q)$ is the number of abstracts associated with C_p that are also related to C_q as found by classifier $SVMC - C_q$ related to concept C_q.

The pairs of concepts with a similarity measure greater or equal than a predefined threshold are then presented to the user as candidate alignments.

3.2 Alternative: Single classification

In the Naive Bayes text classification approach a document was classified to exactly one concept. We wanted to evaluate whether this has a real influence in the similarity computation. Therefore, we developed an alternative to the basic SVM algorithm where in step 3 a document can be classified to only one concept. We assign a document only to the concept for which its SVM classifier generated the highest positive value for that document. In the case more than one classifier produces the highest positive value, then one of the associated concepts is chosen.

4 Evaluation

We evaluate the proposed algorithms with respect to the quality of the suggestions they generate. We also compare them to the best text classification matcher in [18] as well as to the best text-based matcher implemented in the SAMBO ontology alignment system [9] with respect to the quality of the suggestions. Further, we investigate the combination of the proposed algorithms and the text-based SAMBO matcher.

4.1 Set-up

Test cases. We use the same five test cases as in [18]. For the first two cases we use a part of a GO ontology [3] together with a part of SigO [17]. The first case, *B* (behavior), contains 57 terms from GO and 10 terms from SigO. The second case, *ID* (immune defense), contains 73 terms from GO and 17 terms from SigO. The other cases are taken from the anatomy category of Medical Subject Headings (MeSH, [11]) and the Adult Mouse Anatomy (MA, available from OBO): *nose* (containing 15 terms from MeSH and 18 terms from MA), *ear* (containing 39 terms from MeSH and 77 terms from MA), and *eye* (containing 45 terms from MeSH and 112 terms from MA). Golden standards for these cases were developed by domain experts.
Corpus. We use the same corpus as in [18].
Matchers. In the evaluation we use SVM-based matchers as described in sections 3.1 and 3.2 based on sets of maximum 20 and maximum 100 documents per concept. These matchers are denoted as SVM-20-P, SVM-100-P, SVM-20-S, and SVM-100-S where the number stands for the maximum number of documents per concept in the corpus, and P and S stand for Plural (a document can be classified to several concepts) and Single (a document can be classified to only one concept), respectively. Further, we use the basic (best) matcher from [18], denoted NB and described in section 2.3. We also use the best text-based matcher (TermWN) from [9] which combines edit distance, n-grams and linguistic matching and uses WordNet as a thesaurus to augment the matching by using synonyms and hypernyms.
Double threshold filtering. In one experiment we also use the double threshold filtering technique introduced in [1]. Instead of using one threshold, this filtering technique uses an upper and a lower threshold. Pairs of terms with a similarity value equal or higher than the upper threshold are retained as suggestions. Pairs with a similarity value between the lower and the upper threshold are filtered using information about the structure of the ontologies. The rest is discarded.

4.2 Evaluation results

The different evaluation results are given in tables 1 and 2. The first columns in the tables represent the cases and the number of expected alignments for each case based on the golden standards developed by the domain experts. The expected alignments are a minimal set of suggestions that matchers are expected to generate for a perfect recall. This set does not include the inferred suggestions. Inferred suggestions are counted neither as correct nor as wrong suggestions. An example of an inferred suggestion is that *incus* is a kind of *ear ossicle*. In this case we know that *auditory bone* (MA) is equivalent to *ear ossicle* (MeSH), and *incus* is a kind of *auditory bone* in MA. Then a reasoning system could derive that *incus* is a kind of *ear ossicle*. The second column in the tables represent threshold values. The cells in the other columns contain quadruplets a/b/c/d which represent the number of a) suggestions, b) correct suggestions, c) wrong suggestions and d) inferred suggestions, for a given case, matcher and threshold.

	Th	SVM-20-P	SVM-100-P	SVM-20-S	SVM-100-S	NB-20	NB-100
B	0.4	249/4/165/90	387/4/258/125	2/2/0/0	0/0/0/0	3/2/0/1	4/2/1/1
4	0.5	175/4/108/63	306/4/203/99	2/2/0/0	0/0/0/0	2/2/0/0	2/2/0/0
	0.6	92/4/54/34	225/4/148/73	2/2/0/0	0/0/0/0	2/2/0/0	2/2/0/0
	0.7	52/3/25/24	130/3/79/48	2/2/0/0	0/0/0/0	2/2/0/0	2/2/0/0
	0.8	20/2/10/8	36/0/22/14	2/2/0/0	0/0/0/0	2/2/0/0	1/1/0/0
ID	0.4	526/7/482/37	672/8/592/72	4/3/0/1	2/2/0/0	11/4/4/3	9/6/3/0
8	0.5	344/6/314/24	490/8/433/28	3/2/0/1	0/0/0/0	7/4/0/0	5/5/0/0
	0.6	215/6/195/14	336/8/300/28	2/1/0/1	0/0/0/0	5/4/0/1	2/2/0/0
	0.7	141/5/126/10	222/6/191/25	0/0/0/0	0/0/0/0	2/2/0/0	1/1/0/0
	0.8	75/5/63/7	108/5/93/10	0/0/0/0	0/0/0/0	0/0/0/0	0/0/0/0
nose	0.4	117/7/97/13	155/7/124/24	9/5/4/0	5/5/0/0	7/5/2/0	6/5/1/0
7	0.5	83/7/66/10	120/7/91/22	5/5/0/0	4/4/0/0	6/5/1/0	6/5/1/0
	0.6	61/6/50/5	85/7/60/18	5/5/0/0	2/2/0/0	5/5/0/0	5/5/0/0
	0.7	47/6/39/2	58/6/45/7	4/4/0/0	0/0/0/0	5/5/0/0	5/5/0/0
	0.8	29/6/23/0	34/6/27/1	2/2/0/0	0/0/0/0	4/4/0/0	3/3/0/0
ear	0.4	937/21/828/88	1224/24/1056/144	18/16/2/0	14/12/2/0	20/16/4/0	18/16/2/0
27	0.5	700/21/607/72	957/23/822/112	15/15/0/0	11/10/1/0	18/16/2/0	15/14/1/0
	0.6	487/21/411/55	696/22/590/84	11/11/0/0	1/1/0/0	14/14/0/0	12/11/1/0
	0.7	320/21/262/37	478/22/392/64	5/5/0/0	0/0/0/0	11/11/0/0	11/10/1/0
	0.8	174/20/134/20	278/21/223/34	2/2/0/0	0/0/0/0	5/5/0/0	3/3/0/0
eye	0.4	1588/25/1493/70	2055/25/1926/104	17/16/1/0	7/7/0/0	33/19/13/1	25/18/7/0
27	0.5	1089/25/1009/55	1481/25/1366/90	14/14/0/0	4/4/0/0	20/17/3/0	18/17/1/0
	0.6	667/25/601/41	957/25/860/72	9/9/0/0	0/0/0/0	16/16/0/0	14/14/0/0
	0.7	441/23/386/32	612/24/539/49	2/2/0/0	0/0/0/0	12/12/0/0	10/10/0/0
	0.8	271/23/225/23	344/23/290/31	0/0/0/0	0/0/0/0	5/5/0/0	3/3/0/0

Table 1. SVM plural assignments (SVM-20-P and SVM-100-P), SVM single assignments (SVM-20-S and SVM-100-S), and Naive Bayes (NB-20 and NB-100 from [18]).

Comparison based on different corpus sizes. For SVM with plural assignment (table 1 - columns SVM-20-P and SVM-100-P) we see that SVM-20-P finds at most as many expected alignments as SVM-100-P, sometimes fewer. However, SVM-100-P does not only find more expected alignments, it also gives more suggestions in total, more wrong suggestions and more inferred suggestions.

For SVM with single assignment (table 1 - columns SVM-20-S and SVM-100-S) we observe that SVM-100-S does not find suggestions for high thresholds and for case B also not even for threshold 0.4. It does have almost perfect precision[1]. SVM-20-S gives some more suggestions than SVM-100-S. It gives a few wrong suggestions at threshold 0.4, but no wrong suggestions at higher thresholds. SVM-20-S outperforms SVM-100-S in recall with almost as good precision.

In [18] no relationship was found between the number of abstracts in the corpus and the quality of the suggestions for NB (table 1 - columns NB-20 and NB-100). However, as is the case here, it was observed that the corpus has a large influence on the quality of the suggestions.

Comparison of single and plural assignment. Both for the SVM-100 and the SVM-20 versions the recall for the plural assignment is higher to much higher than the recall for the single assignment. This comes, however, at a cost. The precision for the single assignment algorithms is much higher than for their plural assignment counterparts. We see a real trade-off here: find many expected alignments, but also get many wrong suggestions, or, find few expected alignments, but receive almost no wrong suggestions.

Comparison of NB and SVM. NB is a single assignment algorithm and we therefore compare it with the single assignment versions of SVM. All single assignment algorithms give relatively few suggestions but have high precision. However, NB gives always more suggestions than SVM for the same document corpus and the same threshold. NB also always gives suggestions, except for case ID and threshold 0.8, while this is not the case for the SVM algorithms. SVM-20 does not give suggestions for ID - 0.7 and higher, and eye - 0.8. In addition to these, SVM-100 does not give suggestions for ID - 0.5 and higher, nose - 0.7 and higher, ear - 0.7 and higher, eye - 0.6 and higher, and B with threshold 0.4 and higher. It is clear that the SVM algorithms with single assignment do not perform well with high thresholds.

In general, the NB algorithms have slightly better recall than the SVM algorithms, while the SVM algorithms have slightly higher precision than the NB algorithms. However, even if the recall for NB is better, it is not always the case that the alignments that were found by SVM algorithms were also found by NB.

[1] Precision is used as it is usually defined in information retrieval, i.e. the number of correct suggestions divided by the number of suggestions. As noted before, inferred suggestions are counted neither correct nor wrong. Similarly, recall is defined as the number of correct suggestions divided by the total number of correct suggestions, in this case the expected suggestions.

	Th	TermWN	TermWN+ SVM-100-S	TermWN+ SVM-100-P	TermWN+ SVM-100-P + Double Threshold
B 4	0.4	58/4/22/32	4/4/0/0	156/4/84/68	69/4/26/39
	0.5	35/4/13/18	4/4/0/0	52/4/19/29	26/4/4/18
	0.6	13/4/4/5	0/0/0/0	21/4/7/10	13/4/2/7
	0.7	6/4/0/2	0/0/0/0	7/4/1/2	6/4/1/1
	0.8	4/4/0/0	0/0/0/0	4/4/0/0	4/4/0/0
ID 8	0.4	96/7/66/23	8/6/2/0	302/8/262/32	177/7/165/5
	0.5	49/7/25/17	6/6/0/0	155/7/127/21	86/7/74/5
	0.6	16/5/5/6	2/2/0/0	71/7/48/16	40/7/28/5
	0.7	7/5/2/0	1/1/0/0	19/7/7/5	12/7/5/0
	0.8	6/4/0/2	0/0/0/0	7/5/2/0	7/5/2/0
nose 7	0.4	48/7/37/4	9/7/2/0	80/7/66/7	26/7/19/0
	0.5	28/7/18/3	7/7/0/0	58/7/47/4	21/7/14/0
	0.6	8/6/2/0	6/6/0/0	31/7/47/4	12/7/5/0
	0.7	6/6/0/0	4/4/0/0	11/7/4/0	9/7/2/0
	0.8	6/6/0/0	1/1/0/0	6/6/0/0	6/6/0/0
ear 27	0.4	155/26/110/19	34/25/8/1	585/27/481/77	144/27/103/14
	0.5	99/26/65/8	27/23/4/0	203/26/146/31	64/26/32/6
	0.6	47/26/19/2	17/17/0/0	96/24/64/8	42/24/16/2
	0.7	34/26/8/0	12/12/0/0	55/23/28/4	36/23/11/2
	0.8	28/25/3/0	1/1/0/0	29/21/6/2	29/21/6/2
eye 27	0.4	135/26/100/9	28/23/5/0	643/25/568/50	260/23/223/14
	0.5	74/23/44/7	21/20/1/0	272/25/221/26	144/23/115/6
	0.6	33/22/10/1	16/16/0/0	138/24/101/13	74/22/48/4
	0.7	24/21/3/0	7/7/0/0	54/21/27/6	35/20/11/4
	0.8	22/20/2/0	0/0/0/0	25/21/4/0	25/21/4/0

Table 2. TermWN (from [9]), combination of TermWN and SVM-100-S, combination of TermWN and SVM-100-P, combination of TermWN and SVM-100-P with double threshold filtering.

Comparison with and combination with other matchers. In table 2 we show the quality of the suggestions of TermWN (from [9]), the combinations of TermWN with SVM-100-P and SVM-100-S, as well as the results of the combination of TermWN with SVM-100-P with the double threshold filtering method of [1]. The suggestions for the combinations are determined based on the combination of the similarity values measured by individual matchers using weights, $sim(C_1, C_2) = (\sum_{k=1}^{n} w_k * sim_k(C_1, C_2)) / \sum_{k=1}^{n} w_k$, where sim_k and w_k represent the similarity values and weights, respectively, for the different matchers. In the experiment we used 1 as the weight for each matcher.

When we compare TermWN with the text categorization approaches, we note that TermWN has higher recall than the single assignment approaches. It also has better recall than the plural assignment approaches for the case ear, but for the other cases the recall is similar. TermWN has better precision than the plural assignment algorithms, but worse than the single assignment algorithms.

We also note that all expected alignments, except for 3 alignments for case ear[2], were found by at least one SVM or NB matcher and threshold at least 0.4. TermWN with threshold 0.4 missed 1 expected alignment for ID, 1 for ear and 1 for eye.

The combination of TermWN and SVM-100-S gave perfect results for B and thresholds 0.4 and 0.5. Otherwise, when it gave suggestions, the precision was high. For thresholds 0.4 and 0.5, SVM-100-S worked as a filter on TermWN by removing many wrong suggestions at the cost of no or few correct suggestions. For higher thresholds too many correct suggestions were removed.

For most cases and thresholds the combination of TermWN and SVM-100-P gave better recall than TermWN and better than SVM-100-P. The precision of the combination was higher than the precision for SVM-100-P, but lower than the precision for TermWN. The precision for the combination could be improved to even better precision than TermWN by using the double threshold filtering technique while keeping the recall at the same level for all cases except for ID with threshold 0.4 and the eye case.

5 Related work

Recently, there is a growing interest in instance-based methods for ontology alignment (see, for instance, the publication list on www.ontologymatching.org). Some of these approaches use literature as instances.

Our work can be seen as a continuation of [18] by slightly extending the method and proposing, evaluating and comparing other strategies for different steps in the method.

In [4] the effect of the choice of co-occurrence measures on the performance of instance-based alignment approaches is studied. The study is based on a use case with two ontologies that have been used to annotate publications in the

[2] These alignments are: *maleus* - *malleu*, *vestibular apparatus* - *vestibule*, and *perilymphatic channel* - *cochlear aqueduct*. They were, however, found when combining with TermWN.

National Library of the Netherlands. An important assumption in the study is that doubly annotated instances exist. The similarity measures for concepts that are evaluated are Jaccard, corrected Jaccard, pointwise mutual information, log-likelihood and information gain. Jaccard and corrected Jaccard led to the best results in their case.

One method to compute the similarity between concepts in ArtGen [12] is to use the concept names and compute a similarity between the contexts (1000-character neighborhoods) of all occurrences of the words in a set of domain-specific Web pages.

In FCA-Merge [16] ontologies are merged using a concept lattice which is derived using formal concept analysis based on how documents from a given domain-specific corpus are classified to the concepts in the ontologies using natural language processing techniques.

OntoMapper [13] assigns to each concept a set of abstracts of technical papers taken from ACM's digital library and Citeseer and generates similarity scores matrices for the ontologies that are computed by the Rainbow text classifier. The similarity between the concepts is then calculated based on these two matrices using the Bayesian method.

6 Conclusion

In this paper we have proposed algorithms for aligning ontologies using literature. The methods are based on text categorization using SVMs. We have shown that the corpus has a large influence on the results. Further, there is an important trade-off between the single and plural assignment methods. The plural assignment methods promote recall while the single assignment methods promote precision. Compared to another single assignment text categorization-based method, SVM-based algorithms have slightly higher precision and sightly lower recall. Finally, we compared the SVM-100 approaches with a text-based matcher, TermWN and showed that a combination of TermWN with SVM-based approaches leads to a large gain in precision compared to TermWN and SVM-100-P, while maintaining a high recall.

We are interested in further investigating other combinations of the algorithms. For instance, our results indicate that giving higher weights to single assignment approaches may improve the results for the combinations. Further, domain knowledge and other auxiliary knowledge could be used in different ways to enhance the matchers. We are also interested in looking at other classification algorithms and evaluating the algorithms on other test data.

References

1. Chen B, Tan H and Lambrix P. 2006. Structure-based filtering for ontology alignment. *Proceedings of the IEEE WETICE Workshop on Semantic Technologies in Collaborative Applications*, pp 364-369.
2. Euzenat J and Shvaiko P. 2007. *Ontology Matching.* Springer.

3. The Gene Ontology Consortium. 2000. Gene Ontology: tool for the unification of biology. *Nature Genetics*, 25(1):25-29. http://www.geneontology.org/.
4. Isaac A, van der Meij L, Schlobach S, and Wang S. 2007. An Empirical Study of Instance-Based Ontology Matching. *Proceedings of the 6th International Semantic Web Conference*, LNCS 4825, 253-266.
5. Jasper R and Uschold M. 1999. A Framework for Understanding and Classifying Ontology Applications. *Proceedings of the 12th Workshop on Knowledge Acquisition, Modeling and Management*.
6. Joachims T. 1998. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. *Proceedings of the European Conference on Machine Learning*, LNCS 1398, 137-142.
7. Joachims T. 1999. Making large-Scale SVM Learning Practical. *Advances in Kernel Methods - Support Vector Learning*, B Schölkopf and C Burges and A Smola (eds), MIT-Press. http://svmlight.joachims.org/
8. Kalfoglou Y and Schorlemmer M. 2003. Ontology mapping: the state of the art. *The Knowledge Engineering Review*, 18(1):1-31.
9. Lambrix P and Tan H. 2006. SAMBO - A System for Aligning and Merging Biomedical Ontologies. *Journal of Web Semantics, Special issue on Semantic Web for the Life Sciences*, 4(3):196-206.
10. Liu B, Dai Y, Li X, Lee WS and Yu Ph. 2003. Building Text Classifiers Using Positive and Unlabeled Examples. *Proceedings of the Third IEEE International Conference on Data Mining*, 179-188. http://www.cs.uic.edu/∼liub/LPU/LPU-download.html
11. MeSH, Medical Subject Headings, http://www.nlm.nih.gov/mesh/
12. Mitra P and Wiederhold G. 2002. Resolving terminological heterogeneity in ontologies. *Proceedings of the ECAI Workshop on Ontologies and Semantic Interoperability.*
13. Prasad S, Peng Y and Finin T. 2002. Using Explicit Information To Map Between Two Ontologies. *Proceedings of the AAMAS Workshop on Ontologies in Agent Systems.*
14. PubMed, http://www.ncbi.nlm.nih.gov/pubmed/
15. Shvaiko P and Euzenat J. 2005. A Survey of Schema-based Matching Approaches. *Journal on Data Semantics*, IV:146-171.
16. Stumme G and Mädche A. 2001. FCA-Merge: Bottom-up merging of ontologies. *Proceedings of the 17th International Joint Conference on Artificial Intelligence*, 225-230.
17. Takai-Igarashi T, Nadaoka Y and Kaminuma T. 1998. A Database for Cell Signaling Networks. *Journal of Computational Biology* 5(4):747-754.
18. Tan H, Jakoniene V, Lambrix P, Aberg J and Shahmehri N. 2006. Alignment of Biomedical Ontologies using Life Science Literature. *Proceedings of the International Workshop on Knowledge Discovery in Life Science Literature*, LNBI 3886, 1-17.
19. Vapnik V. 1995. *The Nature of Statistical Learning Theory*. Springer.