Application of Textual Corpus in Ontology Matching

Besat Kassaie, Alireza Vazifedoost, and Maseud Rahgozar

Abstract—**Ontology matching or finding similarities between concepts of different ontologies, has many applications today. One automatic approach for finding these similarities is by leveraging machine learning techniques. In this paper we propose a new method in which a text corpus is used as the source of knowledge in conjunction with a machine learning method to find matchings between two ontologies**.

Index Terms—**Ontology matchin, machine learning, text corpus.**

I. Introduction

The aim of ontology matching is to find similarities or matches between concepts of different ontologies. There are many new applications which need a sort of ontology matching. Examples are semantic web applications, multi agent systems, applications mash up and so on. One may be interested in either finding lexical similarity or semantic similarity, but in the both cases, the result of such a matching process can be useful for relating distinct ontologies. This enables us to reuse existing ontologies in new applications.

A wide range of methods for ontology matchinghas been proposed. Among all, methods based on lexical similarities and structural similarities [1]-[4] are more observed. However, the methods in which we are more involved are instance based similarity methods. The main objective in these methods is to find similarity of two concepts based on similarity of their instances. Application of textual corpus and machine learning in instance based ontology matching are not new concepts [5]-[7].

Research works,presented in [5] and [6] were mainly focused on detecting similarities between instances based on machine learning methods. They used Na ïve Bayes learning method[5] and Jaccard similarity formula [6] for finding the rate of similarities between different concepts. The main idea investigated was to find similaritiesbetween different concepts based on the size of intersections between their instances by using Jaccard similarity formula.

A possible drawback of those works is the techniques they use for picking features in their machine learning steps. Actually in those approaches, the features are selected from a very restricted domain of information which is provided inside the considered ontologies. We believe that, the restricted amount of this knowledge about an instance affect the quality of learning.

On the other hands, in [7], the main idea is to select informative features from the documents which are relevant to the concepts of the considered ontologies. This would extend the knowledge about the concepts for having a better judgment about their similarities.

In our proposed approach we extract the features for learning step from a text corpus which is in the same domain of the considered ontologies.However, our work has fundamental differences with [7] regarding the methods used for extraction of features from documents. Firstly, in [7] a handmade corpus is used while we use an online search engine as the provider of corpus which makes our solution less dependent to human intervention. Secondly, in [7] simply the tf.idf parameter of each term is considered as the value of that term as a feature while we use the co-occurrence value of each term with a concept as the value of corresponding feature. We believe that, this is a more informative value for a feature because tf.idf of a term says about its importance in comparison with other terms for discriminating a document in a corpus.While, the co-occurrence value determines how much a feature is helpful in describing a concept which is more closeto ontology matching objective. The last but not the least difference of our approach with [7] is that while both methods are using Jaccard formula, we use machine learning techniques for comparing the instances of two concepts. This helps us to count correctly the semantically identical instances which may have different labels as the shared instances of two concepts. On the other hand, [7] uses the Jaccard formula by regarding facial labels of instancesthat may lead to wrong decisions.

In subsequent sections, first we give details about our method to involve a text corpus in ontology matching problem. Then, in the next section we explain about our implementation and present our results and, after all, we evaluate those results and give discussions about them in the final section.

II. Method

Every concept in ontology represents some entity of knowledge in the domain of that ontology. These concepts are related together by means of some relationships. Two more important relations of anontology may be is-a and part-of relationships. In this paper, we consider the sub concepts of a concept A which are related to it by an is-a relation as the instances of the concept A.

Here, we assume that the similarity value of two concepts A and B from different ontologies is related to the size of shared instance set of those concepts. We use the Jaccard [8] formula to calculate this similarity,:

Manuscript received June 9, 2012; revised August 1, 2012.

Besat Kassaie is with Islamic Azad University Science and Research Branch, Tehran, Iran (e-mail: Besat_k@yahoo.com)

Maseud Rahgozar is with ontrol and Intelligent Processing Center of Excellence, School of Electrical and Computer Engineering, University College of Engineering, University of Tehran, Tehran, Iran (e-mail: Rahgozar@ut.ac.ir)

Alireza Vazifedoost is with chool of Electrical and Computer Engineering, University College of Engineering, University of Tehran, Tehran, Iran (e-mail: Vazifehdst@ut.ac.ir)

$$Jaccard\text{–}Sim(A,B)=\frac{P(A\cap B)}{P(A\cup B)}=\frac{P(A,B)}{P(A,B)+P(A,\bar{B})+P(\bar{A},B)} \quad (1)$$

This formula consists of several constituent but the way of calculating them is very similar. We explain one of them as a sample. P(A,B) is calculated by using the following equation:

$$P(A,B)=\frac{N(U_1^{A,B})+N(U_2^{A,B})}{N(U_1)+N(U_2)} \quad (2)$$

The above equation has multiple parts too and, this is where we leverage machine learning to estimate the value of those parts. For example if we considerN (U 1A, B)it would be the number of instances in ontology O1which could belong to both concepts A (from O1) and B (from O2). In the same way has the N (U 2A,B)same meaning when it comes to ontology O2. For calculating N (U 1A,B)we need to count the number of instances which are owned by both A and B. If we get an instance like s from ontology O1, finding its attachment to the set of instances of concept A is trivial because both of them are in the same ontology. But, the problem becomesmore intricate when we want to find its membership to the set of instances of Concept B from the second ontology. Making it clear, we have to notice that the instance s may be not exactly in the set of current instances of concept B, but we could admit such a membership because of possibly a vast similarity of its properties with the properties of current instances of Bthat make it a potential instance of B. Although, this can go in the opposite way, in which we reject the ownership of an instance even thought there exists facial similarities like string matches, between an instance of B and S.

Therefore, after determining the ownership of an instance in the first ontology to a concept A we need a way to decide on its belonging to a concept B in the second ontology. This is where we use machine learning techniques. We know about the current instances of concept B and we know their features. Therefore in terms of machine learning we can train a machine by using these instances to discriminate between two categories. The first category (C1) includes the concepts which can be placed as an instance of concept B and the second category (C2) which contains the other concepts not association to the set of instances of B. This process is shown in Fig. 1. After this training we can give this machine an arbitrary instance s from the first ontology to examine its belonging to the concept B. if this examination is successful we can count it as a member of shared instances as is shown in Fig. 2.

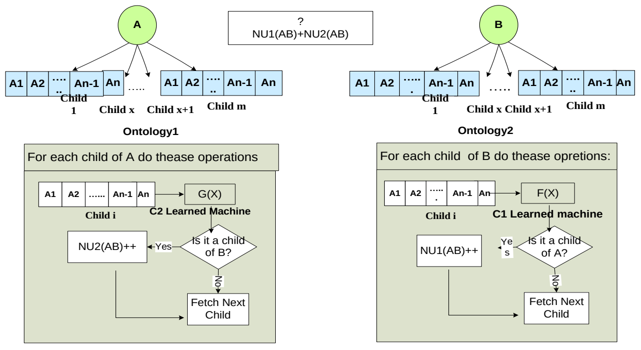

Fig. 1. Learning Process for concept C1 of first ontology and C2 from second ontology

In the previous works, different features of concepts have been used during learning process. For example in [6] two learners were used. Each of them uses different features of concepts and a third learner combines their results. The first learner uses the number of existing words in the textual content of a concept which is a brief description of that concept, and the second one, uses the complete name of a concept which is constructed by placing all names of concepts from the root to that concept consecutively as a feature in learning process.

Also in [6] which is an extension to the work in [5], some features like complete name was used but beside that, learning based on nearest neighbourhood was applied too. This was a try to use textual content in matching process. In that way, the similarities of documents in which different concept are occurred is used as a clue for similarity of concepts themselves.

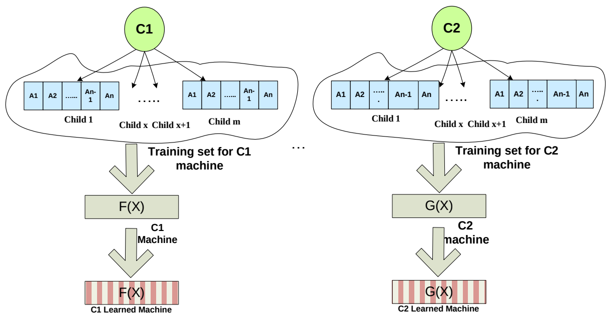

Fig. 2. An example which shows how a constituent of the Jaccard similarity is calculated by using machine learning

In the current paper the new idea is to extract features of concepts from a textual corpus. In our context the entities which are considered as a feature are themselves terms inside the corpus. Then the value of each feature (term), would be the amount of its co-occurrence with a concept of the ontology. This value could be accurately given by this equation:

$$Co-occurenceValue(A,B)=\frac{N_d(A,B)}{N_d(A)} \quad (3)$$

In this equation, N_d (*A*, *B*) is the number of documents from the corpus in which both concept A from ontology and the term B from corpus are co-occurred. Also the *Nd* (*A*) is the number of documents in which the concept A comes alone. Here, the idea is that, more co-occurrence suggests stronger relationship between a concept and a feature (term).

Having a table of concepts and their features now we can use several methods of machine learning. However, we used the simple yet effective method of Naïve Bayes.The Bayes classifier based on Bayes theory calculates the fitness probability of for example a concept to a category. Formally, in this context if we suppose *X* represents a vector of feature values of a concept like (*x*1, *x*2,.., *xk*) then we are looking for the conditional probability *P* (*H*|*X*) in which *H* is a random variable that shows probability of belongness of a concept to a category c. This conditional probability could be rewritten as (4):

$$P(H=c|X)=\frac{P(X|H=c)P(H=c)}{P(X)} \quad (4)$$

In this formula $P(H)$ is the probability of happening class c. This formula is the base of Na ïve Bayes method. It can be proven that by regarding some assumptions like discretized features' values and by considering probability independence and if we consider just two categories, the above formula could be written such as equation (5):

$$g(X)=\sum_{i=1}^{d} x_i \log\left[\frac{p_i(1-q_i)}{q_i(1-p_i)}\right]+\sum_{i=1}^{d}\log\left(\frac{1-p_i}{1-q_i}\right)+\log\left[\frac{p(1)}{1-p(1)}\right] \tag{5}$$

If $g(x) > 0$ then X would belong to category 1 and if $g(x) < 0$ to the category 2. Also, d is the number of features, and pi and qi are calculated in the following way:

$$p(x_i = 1|1) \triangleq p_i$$
$$p(x_i = 0|1) \triangleq 1 - p_i$$
$$p(x_i = 1|2) \triangleq q_i$$
$$p(x_i = 0|2) \triangleq 1 - q_i \quad i = 1, \ldots\ldots, d$$

So the learning problem is reduced to finding these coefficients from the training set which could easily be done.

III. Implementation and Results

We did experimentations by two ontologies (numbered 101 and 304)from OAIE (Ontology Alignment Initiative Evaluation) [9] benchmark ontologies which are both in bibliographic domain with overall 75 concepts in both of them.

For creating corpus we used Google search engine [10]. We queried the Google per each concept of ontologies by following expression:

ConceptName + BibliographicInformation + filetype:doc

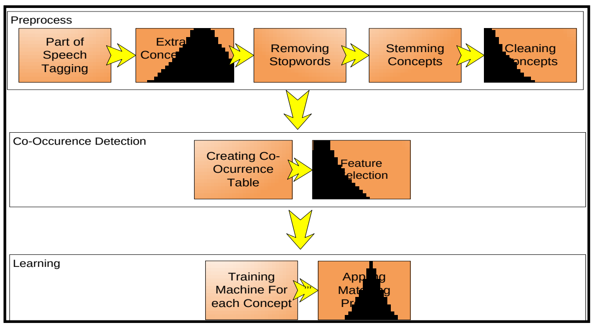

Fig. 1. The overall implemented process

And finally the first top 50 ranked returned documents was selected to be included in corpus. In this fashion, we tried to find documents which are related to bibliographic domain and it is assumed that these top ranked documents by Google are relevant to the domain. Lastly, we had a corpus consisting of 505 documents with size of 16 MB. We considered three steps for alignment task which is shown in figure 3. The first step is preprocessing. In this step, the documents ofcorpus would be prepared for our purpose. Here we take advantage of GATE [11] for preprocessing task.In the next step we first find the co-occurencies of terms in corpus and the concepts of ontologies. Among all the terms (features) extracted from corpus for which we have calculated the co-occurrence value with ontologies' concepts, we have to select just a subset. The reason is both avoiding time complexity of program and purging less useful features from learning process.

We considered two parameters for selecting among features. The first parameter named ZeroParam and the second one is Entropy. We will describe them in detail at next:ZeroParam: the selected features need to be useful regarding all of concepts not just some of them. Some concepts have co-occurency with just a limited number of features and if at least some of those feature don't exist in the selected set we would end up with concepts which have zero value for most of their features. This makes learning process very imprecise. Therefore, we need to avoid the condition in which many concepts have just zero value for many of features in selected set. This leads us to specify a factor of importance for features of every concept. This factor shows how much important is each feature which has non-zero value regarding a concept. So we proposed the following formula for calculating this importance for every concept:

$$ConceptZeroParam(C) = \frac{Count of all features}{Count of NonZeroFeatures} \tag{6}$$

The numerator of fraction is the number of all features, and the denominator is the number of features which have a non-zero value for concept c. This parameter shows how important is a typical feature with non-zero value for a concept. Then in one piece, the ZeroParam of a feature would be calculated from the formula (7):

$$FeatureZeroParam(f) = \sum_{c \in Concepts} ConceptZeroParam(c) \tag{7}$$

where value of (f) is not zero

This is a simple summation over all ConceptZeroParams where the feature f has a non-zero value. This way we can have a clue for selecting features which have non-zero value for most of concepts.

Entropy: It seems that a good feature for machine learning is the one which takes more different values per concepts. Therefore, it is a better idea to select features which have more disarray in their values. For quantifying this disarray we use Entropy formula in the following order:

$$Entropy\,(f) = \sum_{d \in Distinct\ \ Values\ \ of\ \ f} P(d)\, log_2 \frac{1}{P(d)} \tag{8}$$

TABLE I: A Sample of Results Achieved by Machine Learning Method

Concept1	Concept2	Similarity
reference	Entry	0.33015873
book	Composite	0.55
informal	Informal	0.537378115
part	Part	0.457746479
academic	mastersthesis	0.493055556
academic	Phdthesis	0.493055556
misc	Misc	1
report	deliverable	0.493055556
report	Techreport	0.493055556
motionpicture	motionpicture	1
journal	Journal	0.986486486
conference	Journal	0.486666667
address	Entry	0.025345622
institution	Institution	0.473333333
institution	School	0.473333333
institution	Publisher	0.473333333

In this equation, P(d) is the probability of occurring a specific value d among all values which are taken by a feature f. based on this equation taking more different values by a feature f, bring about more Entropy.

Now based on these two mentioned parameters we define a new parameter named, FeatureQuality which is determined in the following way for each of features:

$$FeatureQuality\,(f) = Entropy(f) * FeatureZeroParam(f) \quad (9)$$

IV. Evaluation and Discussion

For evaluating purpose, we calculated three well-known measures in this field, i.e. recall, precision and f-measure [12], [13]. We give the definition of each of parameters in ontology alignment context. For a given alignment A and a reference alignment R, Precision of A is:

$$P(A,R) = \frac{|A \cap R|}{|A|} \quad (10)$$

|A ∩ R| is the number of matchs we found correctly and |A| is the total number of detected matchs.

A reference alignment is a standard one which contains accepted matchings. We use a standard alignment reference for these two ontologies provided by OAIE in year 2008 [14]-[23]. We also used the standard tools of this conference for calculating precision, recall and f-measure. It needs to mention that we hadn't a chance to participate in the contest of OAIE in this year, so we are just using their results and tools and our results are not evaluated by conference.

The Recall measure is defined as well in following equation:

$$R(A,R) = \frac{|A \cap R|}{|R|} \quad (11)$$

Here, |R|is the total number of matchings, that could be found in the reference alignment.

Also the f-measure which combines precision and recall is defined as follow:

$$M_\alpha(A,R) = \frac{P(A,R).R(A,R)}{(1-\alpha)\times P(A,R) + \alpha \times R(A,R)} \quad (12)$$

Here, we considered α as 0.5.

The result of experiments is shown in a diagram in figure 4. There you can see our results beside the results of other participants. Also in table 2 you can see the percentage of destruction or improvement of our system in compare with other systems. The destruction or improvement (DoI) is defined as following:

$$DoI = \frac{theresultofProposedSystem \quad - TheresultofSystemX}{theresultofSystemX} \times 100 \quad (13)$$

Comparing with other systems the average DoI for precision,recall and f-measure are 14.1,51.97 and 31.21 percents respectively, based on the data in Table 2.

Finally, the features with the maximum amount of FeatureQuality would be selected for participating in learning process. We used 500features in our experiments.

Another issue which has to be noticed during implementation is the way we make the set of children. Here we can imagine two options. In the first case, per each concept we just pick those children which have a direct is-a relation with that concept. But in the second case, we opt for the whole subtree which is rooted in that concept. The result of experiment change based on these two selections, but in this paper we choose just the first option.

Finally, in the last step most similar concepts of two ontologies would be considered as a match. For each of those matches we assign a number based on the level of confidence we have to the match. Here we used the similarity value given by Jaccard formula as the confidence value. A sample of matches is presented in table 1.

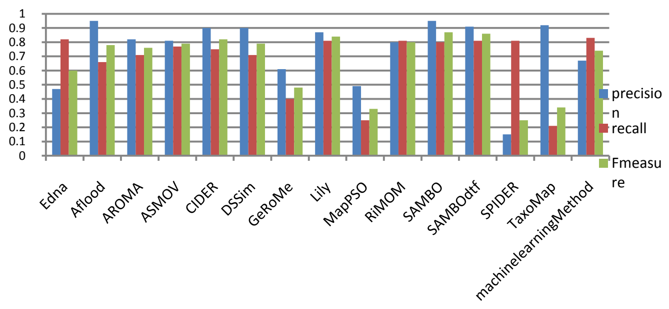

Fig. 1. Comparison results with other systems of OAIE in 2008 [14-23]

TABLE II: Evaluation Result

Systems	Measures	DoI	Systems	Measures	DoI
Edna	precision	+68.09	***Lily***	precision	-9.20
	recall	-3.66		recall	-2.47
	fmeasure	+31.67		fmeasure	-5.95
AFlood	precision	-16.84	***MapPSO***	precision	+61.22
	recall	+19.7		recall	+216
	fmeasure	+1.28		fmeasure	+139.39
AROMA	precision	-3.66	***RiMOM***	precision	-1.25
	recall	+11.27		recall	-2.47
	fmeasure	+3.95		fmeasure	-1.25
ASMOV	precision	-2.47	***SAMBO***	precision	-16.84
	recall	+2.6		recall	-1.25
	fmeasure	0		fmeasure	-9.2
CIDER	precision	-12.22	***SAMBOdtf***	precision	-13.19
	recall	+5.33		recall	-2.47
	fmeasure	-3.66		fmeasure	-8.14
DSSIM	precision	-12.22	***SPIDER***	precision	+426.67
	recall	+11.27		recall	-2.47
	fmeasure	0		fmeasure	+216
GeRoMe	Precision	+29.51	***TaxoMap***	precision	-14.13
	Recall	+97.5		recall	+276.19
	Fmeasure	+64.58		fmeasure	+132.35

As a conclusion, in this research we are looking for finding semantic relations between concepts and the resultsseems to be promising. As an example the match found between concepts Reference and Entrywhich have no lexical similarities while they are both roots of their corresponding ontologies, so the matching result is semantically correct. However, we have to notice that, the current work was approved small ontologies, so the number of instances used for each concept in learning process, was limited. Also, the OAIE benchmark doesn't pay enough attention to semantically related conceptswhich explains why our system couldn't get the best place between all systems. But already our approach seems to have a good performance among other methods.

References

[1] W. W.Cohen, P. Ravikumar, and S. E.Fienberg, “A comparison of String distance metrics for name-matching tasks,”in *proc. of II Web*, 2003.

[2] T. B.Rose, D. Kuntz, and F. Gandon, “On ontology matching problems (for building a corporate semantic web in a multi-communicates organization),” in *Proc. of 6th International Conference on Enterprise Information Systems(ICEIS)*, 2004, pp. 236-243.

[3] R. Dieng and S. Hug, “Comparison of personal ontologies represented through conceptual graphs,” in *Proc.of ECAI*, 1998, pp. 341-345.

[4] A. Maedche and S. Staab, “Measuring Similarity between Ontologies,” in *Proc. of the European Conference on Knowledge Acquisition and Management (EKAW)*, 2002, pp. 251-263.

[5] P. Natarajan, “A Machine Learning Approach to Ontology Matching,”*Technical report, unpublished*, 2005.

[6] D. AnHai, M. Jayant, D. Robin, D. Pedro, and H. Alon, “Learning to match ontologies on the semantic web,” *The VLDB journal*, vol. 12, no. 4, pp. 303–319, November 2003.

[7] S. Xiaomeng and G. J. Atle, “An information retrieval approach to ontology mapping,” *Data & Knowledge Engineering*, vol. 58, no. 1, Application of natural language to information systems (NLDB04), pp. 47-69, July 2006.

[8] L. Feiyu, “State of the art: automatic ontology matching,” *Research Report. School of Engineering*, Jönköping: Tekniska Högskolan, 2007.

[9] Benchmark test library. [Online]. Available: http://oaei.ontologymatching.org/2008/benchmarks

[10] Google. [Online]. Available: http://www.google.com.

[11] H. Cunningham, Y. Wilks, and R. Gaizauskas, “GATE-a General Architecture for Text Engineering,” in *Proc. of the 16th Conference on Computational Linguistics(COLING-96)*, Copenhagen,1996.

[12] J. Euzenat, M. Ehrig; and R. García-Castro, “Towards a methodology for evaluating alignment and matching algorithms,” *Technical Report.Ontology Alignment Evaluation Initiative (OAEI)*, May 2005.

[13] J. Makhoul, F. Kubala, R. Schwartz, and R. Weischedel. “Performance measures for information extraction,” in *Proc. of DARPA Broadcast News Workshop*, Herndon. 1999.

[14] M. S. Hanif and M. Aono, “Alignment results of Anchor-Flood algorithm for OAEI-2008,” in *Proc. of OM*, 2008.

[15] J. David, “AROMA results for OAEI 2008,” in *Proc. of OM*, 2008.

[16] Y. R. Jean-Mary and M. R. Kabuka, “ASMOV: results for OAEI 2008,” in *Proc. of OM*, 2008.

[17] G. Jorge and M. Eduardo, “Ontology matching with CIDER: evaluation report for the OAEI 2008,” in *Proc. of OM*, 2008.

[18] N. Miklos, V. V. Maria, and S. Piotr, “DSSim results for OAEI 2008,” in *Proc. of OM*, 2008.

[19] Q. Christoph, G. Sandra, and K. David, “Results of GeRoMeSuite for OAEI 2008,” in *Proc. of OM*, 2008.

[20] W. Peng and X. Baowen, “Lily: ontology alignment results for AEI 2008,” in *Proc. of OM*, 2008.

[21] B. J. Urgen and H. Jan, “MapPSO results for OAEI 2008,” in *Proc. of OM*, 2008.

[22] Z. Xiao, Z. Qian, Li Juanzi, and Tang Jie , “RiMOM results for OAEI 2008,” in *Proc. of OM*, 2008.

[23] L. Patrick, T. He, and L. Qiang, “SAMBO and SAMBOdtf results for the Ontology Alignment Evaluation Initiative 2008,” in *Proc. of OM*, 2008.