2011 International Conference on Computer Communication and Management
Proc .of CSIT vol.5 (2011) © (2011) IACSIT Press, Singapore

A New Similarity Measure for Instance Data Matching

Kamran Zamanifar [1] and Farinaz Alamiyan [1 +]
[1] Department of Computer Science
Islamic Azad University-Najaf Abad Branch, Najaf Abad, Iran

Abstract. Instance based ontology matching is a new approach which uses the extensions of concepts to solve matching problems. Instance matching is the task of recognizing similarity between instances with different ontology descriptions for the same real world objects. It is crucial in this ontology matching area like other data integration issues, and the quality of its algorithm can extremely affect the performance of this way of matching because it can change the overlap of extensional information; that is, common instances of concepts among different ontologies. In this paper, we propose a distributional instance similarity measure and study how the choice of this measure improves the quality of instance mapping in ontology matching scenario.

Keywords: ontology matching, instance matching, similarity measure.

1. Introduction

The task of detecting similar resources is vital in web data integrity; and semantic web needs this identification to get semantic interoperability because it is a distributed and heterogeneous environment, too. In the context of semantic web, resources which referred to real world objects and typed according to an ontology, usually in RDF description, are named instances. Whereas identifying mapping between ontologies and their large data sets of instances manually is a time consuming and enormous task, researchers have developed many sophisticated algorithms and tools which contain various methods such as machine learning, graph analysis and background knowledge in concept mapping, and use different measures to find similar instances. The scale of instance matching is usually much larger than ontology matching. As a result, the efficiency of instance matching strategies becomes a critical issue, and complicated algorithms can't be served unless algorithmic improvements are made.

Instance similarity measures usually compute the similarity of two instances without attending other existent instances. In our current work, we introduce another way of detecting similar instances by using an information theoretic measure, Jensen-Shannon divergence (JSD) and by computing the difference between probability distributions of instances.

2. Related Works

2.1. Related works in instance matching

Euzenar and Shvaiko [1] have prepared an overview of ontology matching systems which utilize similarity of instances. Ontology Alignment Evaluation Initiative or briefly OAEI [2] has posed contests of ontology matching yearly since 2004. One of the youngest match tasks is the instance matching test which

+ Corresponding author, Tel.: +00983112257990; fax: +00983112736217.
E-mail address: Alamiyan@sco.iaun.ac.ir.

has occurred since 2009. So, some participants have developed systems that can match instances too, for example:

- RIMOM
  The Risk Minimization based Ontology Mapping (RIMOM) [3] is a dynamic ontology matching system which uses multiple strategies. Each strategy is based on different ontological information. It proposes a mechanism to find proper strategies or their combination in respect of features of ontologies. The result of ontology matching is used to map instances as useful background knowledge, and utilizes instance type information to eliminate the defects of their own ontology.
- ASMOV
  The Automated Semantic Mapping of Ontologies with Validation (ASMOV) [4] has mechanisms for instance matching based on the same principles as for ontology matching. It computes similarity of two entities by analyzing four features include lexical elements (id, label, and comments), relational structure (ancestor-descendant hierarchy), internal structure (property restrictions for concepts; types, domains, and ranges for properties; data values for individuals), and extension (instances of classes and property values).Then It combines these strategies with weighted sum to find a single value. These values form three matrices, one each for concepts, properties and individuals which help to provide a pre-alignment. Finally, semantic verification is served to improve this alignment.
- FBEM
  Entity Name System (ENS) [5] is a system of entity recognition. This system uses free-form entity description and does not need a specific kind of schema. As a result, this system is only used in instance matching tasks. The ENS can never recognize anything about the type of entity or the way of entity description it is dealing with because it does not have a formal model. It presents an approach called Feature Based Entity Matching (FBEM) to remove ENS shortcomings. This approach defines a function as a similarity measure that combines a suitable string similarity measure with some other functions, and implements a vocabulary to maintain in run time memory.

2.2. Related works in similarity measure

One research that focuses on instance-based ontology matching can be found in [6]. The authors use annotated data to compute similarities between terms, and illustrate that choosing appropriate dissimilarity measures is an important part of this kind of mapping. They consider different dissimilarity measures in their case, and compare the results for each measure. They perceive that a slightly modified variant of the well known Jaccard coefficient gives the best results.

The approach BOOTSTRAP-JS [7] uses JSD to active learning based on algorithms Bootstrap-LV and ACTIVEDECORATE to improve sample selection for optimizing class probability estimates. The results of this research show that these measures can improve classification accuracy and JSD can be more useful when the objective is improving class probability estimates.

The Jaccard coefficient is compared to JSD in an instance based ontology matching domain in [8]. The two concept similarity measures are tested by matching Wikipedia categories with user-generated tags.

3. Instance Matching Problem

In this problem, there are two data sets of instances, D_1 and D_2, which belong to the same or different ontologies. An instance matching algorithm uses these data sets as its input, and detects all instances like i_2 in D_2 that seem to be similar to each instance like i_1 in D_1. These similarities are shown by one value in set of $\{0,1\}$, where 0 denotes that two instances are different and 1 denotes they are equivalent and describe the same real world object.

There are some challenges in instance matching problem which are discussed in [9] and classified in three major groups as follows:

- Data value differences

1. Real data usually tangle with some human mistakes like typo-graphical errors in strings values, integer and date values, etc. People often insert a wrong character or number, omit some important ones or make mistakes in the position of two adjacent characters.
2. Multiple ways can be used to represent the same data in different sources. For example, someone's name can be "Brian, De Palma" in one source and "De Palma, Brian" in another one.

- Structural heterogeneity
  1. Instances can have various descriptions for each property despite different or even same schema. Instance properties can be filled with all possible and allowable values or they can stay empty.
  2. Instances can be represented in different property descriptions. In particular, a data type property in one source can be defined as an object property.
  3. Properties can be defined in variant aggregation criteria, for example, the name of a person can appear in a single data type property or separately into multiple properties such as, first name, last name.
  4. Some instances scrimmage missing values specification, for example, one source has three values for a property and another has just one.
- Logical heterogeneity
  1. Sometimes two instances belong to different concepts but they refer to the same real world object. These concepts can be subclasses of the same super-class or one of them can be the subclass of another.
  2. Two instances can have identical property values but can belong to disjoint classes. In these cases, instance matching algorithm is expected not to detect these instances as similar ones.
  3. Sometimes two individuals have instantiation on different classes which have an implicit class hierarchy declaration that makes those individuals identical.
  4. Two similar instances can belong to classes some properties of whose are implicitly defined through some class restrictions.

The quality of an instance mapping algorithm depends on the measures utilized to find similarity. Of course, each measure needs some algorithmic requirements, and the efficiency of that measure depends on satisfying those requirements. Our current instance matching algorithm can recognize most of the above data value differences and structural heterogeneity cases by using our proposal measure. But it can't control logical heterogeneity because it does not have any reasoning technique.

4. Measure Requirements and Setup

JSD is a measure of the “distance” between two probability distributions [10] which can also be generalized to measure the distance (similarity) between a finite number of distributions [11]. JSD is a natural extension of the Kullback-Leibler divergence (KLD) to a set of distributions. KLD is defined between two distributions, and the JSD of a set of distributions is the average KLD of each distribution to the mean of the set. Unlike KLD, JSD is true metric and is nonnegative and bounded.

If we want to use JSD as a dissimilarity measure in instance matching scenarios, we should first provide a distribution for each instance in both data sets. As we explained the instance matching scenario in the previous section, suppose i_1 is one of instances of source data set (D_1) and instantiates on one or more classes of source ontology (that can be target ontology, too). We want to find its similar instances (like i_2) in target data set (D_2). We should first educe major information of this instance, for example data type property values, the label and comment and even (optionally for some data sets) the information of the instances related to the underlying one through object properties. All values are tokenized into words, and transform all words to lowercase, unimportant words like "the", "and", "of", "at", etc which add little or no semantics are removed. Then, we employ stemming on remaining words to generate the features. These features form a set $W_1 = \{w_1, w_2, \ldots, w_m\}$, where m is the number of features.

Now, we should calculate the distribution of i_1 and all instances of target data set on W_1. Each occurrence of each member of W_1 in i_1 is used as the occurrence of instance i_1 in that member of W_1. So, if word w_1 appears three times in instance i_1, it means that the total occurrences of i_1 on w_1 is 3. Let $n(w_k, i)$contain the number of occurrences of instance i on word w_k , thereupon $n(i) = \sum_k n(w_k, i)$ denotes the total number of

occurrences of instance i on all $w_k \epsilon W_1$ and $N(w) = \sum_k n(w, i_k)$ denotes the total number of instance occurrences in w.

There are two salient points in this step of our measure setup. Whereas we have two data sets in an instance matching scenario, using D_2 as source data set instead of D_1 has different results. This gives us three options to consider:

1. For each instance like i_1 of D_1, generate a separate feature set (W_1) and find the distribution specification of each instance of D_2 according to it.
2. Change source data set to D_2 and do the pervious option.
3. Generate feature set ($W_1 \cup W_2$) using i_1, i_2 both.

Selecting option 1 or 2, make our measure to work like a directional measure i.e the result of mapping from D_1 to D_2 and from D_2 to D_1 will be different, option 3 can remove this direction. Another point is that we can also use one or both of data sets as a target data set. In the rest of this paper, we use W as our feature set, D_1 as source data set and D as target data set (that can be D_2 or $(D_1 \cup D_2)$ or $(\{i_1\} \cup D_2)$).

Most measures used in instance matching scenarios compute the co-occurrence of instances i_1 and i_2 to judge about their similarity, e.g Jaccard coefficient measure, but taking into account other instances that co-occur with i_1 and i_2 can be useful, too. It can give us valuable information because if two instances co-occur often with the same instances, they are more probable to be similar, even if their mutual co-occurrence is not very high. Imagine that one author has some publications such as books and articles; one instance (i_1) in source data set introduces one of these publications for example, an article. We need to find its similar instances in a target data set. Suppose the goal similar instance (i_2) in target data set does not have a value for article title property but contains its author's information and there is another instance (i_3) in target data set, too which seems similar to source instance but has different authors. Now, the question is how can we find the true similar instance? Considering i_1 and i_2 separately has a lower similarity value than comparing them against other instances, but the case of i_1 and i_3 has an inverse result.

So, for each instance i we compute the co-occurrence probabilities with each instance of target data set, i.e for each word w which appears in instance i we compute the probability that another instance i' contains it, too, weighted with the importance of instance i for w. By this computation, for each instance, we find a probability distribution over all instances. This approach is similar to the setup in [8, 12]. We use conditional probability distributions to complete this setup, as follows:

$$Q_i(w) = n(w, i)/n(i) \quad \text{on } W \tag{1}$$

$$q_w(i) = n(w, i)/N(w) \quad \text{on } D \tag{2}$$

$Q_i(w)$ is word distribution of i. The probability of existence of word w in instance i that we interpret as the probability that a randomly selected occurrence of instance i contains w. In a similar manner, $q_w(i)$ is instance distribution of word w. It means the probably of a randomly selected word occurrence of w has appeared in instance i. We use the simple distribution p_z which is defined as

$$p_z(i) = \begin{cases} 1 & if\ i = z \\ 0 & otherwise \end{cases} \tag{3}$$

We can find the average co-occurrence distribution like [8] by using a two step evolution in a Markov chain that shows the connection between instances in D and the members of W. It is given by

$$\sum_{w,i'} q_w(i) Q_{i'}(w) p_z(i') = \sum_w q_w(i) Q_z(w) = \bar{p}_z(i) \tag{4}$$

After providing this setup, we can take a standard information theoretic dissimilarity measure between probability distributions in order to compare instances. We choose JSD dissimilarity measure that is defined as

$$JSD(p||q) = \frac{1}{2} KLD(p||\frac{p+q}{2}) + \frac{1}{2} KLD(q||\frac{p+q}{2}) \tag{5}$$

Where p and q are two distribution and KLD measures for distributions x and y defined as

$$KLD(x||y) = \sum_{k=1}^{n} x_k \log\left(\frac{x_k}{y_k}\right) \quad (6)$$

Where n is the number of members of D. In our case, the dissimilarity between two instances is computed by

$$Dis(i_1, i_2) = JSD(\bar{p}_{i_1}, \bar{p}_{i_2}) \quad (7)$$

Where $\bar{p}_{i_1}$ is the co-occurrence distribution of source instance and $\bar{p}_{i_2}$ is the co-occurrence distribution of target instance, both over D and W.

The more the dissimilarity of two instances converges to zero, they will be more similar.

5. Future Work

Computing the distributions $\bar{p}_z$ has a high time complexity and it is a disadvantage for our approach. So, our research continues on how to improve the performance of this approach and ignore some computations with less harm on the quality of mapping. We'll also try to extend our approach to achieve ability of detecting logical heterogeneity of different data sets which are explained in section 3. We are going to prepare some experiments to consider our measure behaviour when applies on different data sets, and to compare our results with others'.

6. Conclusion

In this paper, we introduced a similarity measure in order to use in instance matching scenarios. This measure finds the similarity of two instances that belong to different data sets based on the distribution of co-occurrence of aforesaid source and target instance with other instances in target data set. This measure can detect similar instances which have a small mutual co-occurrence if there are a number of co-occurring instances that are common between them.

We utilize this approach to achieve better precision and recall factors in instance matching results, but we should play up that providing the proper feature set (W) seriously affects the result. If the members of this set add more semantics and represent source instance more precisely, we can have an alignment with higher quality. So, we propose using this measure beside best stemmer methods and other useful techniques, and choosing the data set with better and more complete description as source data set.

7. References

[1] J. Euzenat, and P. Shvaiko. Ontology matching. Springer-Verlag, Heidelberg (DE), 2007.

[2] Ontology Alignment Evaluation Initiative (2009), <http://oaei.ontologymatching.org>.

[3] J. Li, J. Tang, Y. Li, and Q. Luo. RIMOM: A dynamic multi-strategy ontology alignment framework. *IEEE Transaction on Knowledge and Data Engineering*. Aug 2009, 21(8):1218–1232.

[4] Y. Jean-Mary, E.P. Shironoshita, and M. Kabuka. Ontology matching with semantic verification. *Web Semantics: Science, Services and Agents on the World Wide Web*. 2009, 7(3): 235-251.

[5] P. Bouquet, H. Stoermer, C. Niederee, and A. Mana. Entity Name System: The backbone of an open and scalable web of data. *Proc of the IEEE International Conf on Semantic Computing, ICSC 2008*, CSS-ICSC, IEEE Computer Society. 2008, 554–561.

[6] A. Isaac, L. van der Meij, S. Schlobach, and S. Wang. An empirical study of instance based ontology matching. *ISWC/ASWC*. 2007, 253–266.

[7] P. Melville, S. M. Yang, M. Saar-Tsechansky, and R. Mooney. Active learning for probability estimation using Jensen-Shannon divergence. *The 16th European Conf on Machine Learning (ECML)*, Porto, Portugal. October 2005, 268-279.

[8] C. Wartena, and R. Brussee. Instanced-based mapping between thesauri and folksonomies. *ISWC'08*. 2008.

[9] A. Ferrara, D. Lorusso, S. Montanelli, and G. Varese. Towards a benchmark for instance matching. 2008.

[10] T. M. Cover, and J. A. Thomas*, Elements of Information theory*.Wiley, New York, NY, 1991.

[11] I. Dhillon, S. Mallela, and R. Kumar. Enhanced word clustering for hierarchical classification. *Proc. of 8th ACM Intl. Conf. on Knowledge Discovery and Data Mining*. 2002.

[12] H. Li, and K. Yamanishi. Topic analysis using a finite mixture model. *Inf. Process. Manage.* 2003, 39(4): 521–541.