

A Semi-supervised Learning Approach for Ontology Matching

Zhichun Wang^(✉)

College of Information Science and Technology, Beijing Normal University,
Beijing, China
zawang@bnu.edu.cn

Abstract. Ontology matching is the task of finding correspondences between semantically related entities in different ontologies, which is a key solution to the semantic heterogeneity problem. Recently, several supervised learning approaches for ontology matching have been proposed, which outperform traditional unsupervised approaches. The existing learning based approaches treat the similarity values of matchers as normal numerical features, and need a lot of training examples. In this paper, we propose a semi-supervised learning approach for ontology matching. Our approach needs a small set of training examples, and exploit the dominant relation of similarity metrics to enrich the training examples. A label propagation algorithm is used to determine the matching results. Experimental results show that our approach can achieve good matching results with a few training examples.

Keywords: Ontology matching · Semi-supervised learning · Heterogeneity

1 Introduction

An ontology provides a vocabulary describing a domain of interest, it plays an important role in sharing and reusing knowledge among software agents. Recently, a large amount of ontologies have been created by different individuals and organizations, especially in the domain of Biology and Semantic Web. Ontologies within the same specific domain could be heterogeneous; for example, entities with the same meaning may have different names and descriptions, entities having the same name may also have different meanings.

Ontology matching is the task of finding correspondences between semantically related entities of different ontologies [8], which is critical important for solving the semantic heterogeneity problem. Currently, many ontology matching approaches have been proposed in recent years. Typically, an ontology matching approach first uses several matchers (usually utilize different types of information, such as entities' labels, entities' instances, ontology taxonomy structures) to compute the similarity between ontology entities, and then makes decision based on similarity values of entity pairs. A simple strategy is to sum the similarity

values of each entity pair in a linear weighted fashion, and select a proper threshold to distinguish matching and non-matching pairs. However, given a matching situation, it is difficult to determine the right weights for each matcher. Some sophisticated weighting strategy such as Harmony [14], and Local Confidence [11] are proposed to adaptively determine the weights. But there is no single strategy always return the best results according to experimental results in [15].

Finding matching pairs in completely unsupervised way can not always get desired results, hence, several machine learning based ontology matching methods have been proposed to get more accurate matching results. Using a set of validated matching pairs as training examples, matching learning based methods train regression [3] or classification [7] models to infer the right matches from all the candidate matching pairs. Comparing with unsupervised approaches, machine learning based approaches usually get better results. However, we have found that two aspects in these approaches can be further improved: first, training predicting models needs a certain amount of training examples, which are not always available; second, existing methods only treat the similarity values merely as numeric features, without taking their essential characteristics into account.

In this paper, we propose a semi-supervised learning approach for ontology matching. Given a small set of validated matching entity pairs, our approach first exploit the dominant relations in the similarity space to enrich positive training examples. After getting more training examples, a graph based semi-supervised learning algorithm, label propagation [19], is employed to classify the rest candidate entity pairs into matched and non-matched groups. We also define several constraints to adjust the probability matrix in label propagation algorithm, which help to improve the quality of matching results.

This paper is structured as follows. First, we introduce the ontology matching problem and the label propagation algorithm. Then we present the method of training examples enrichment. After that we introduce the implementation of constrained semi-supervised learning algorithm for ontology matching. Finally, we evaluate the proposed algorithm on matching tests from OAEI¹.

2 Ontology Matching

An ontology is a formal specification of a shared conceptualization, which provides a vocabulary describing a domain of interest [8]. Different from schemas in the database domain, ontologies contain more semantic metadata making the entities in ontologies self-describing. The major components of an ontology are classes, properties, instances and axioms. Classes represent sets of entities or ‘things’ within a domain; they can be organized into a hierarchy. Properties describe various features and attributes of concepts and constraints on these attributes, which can also be organized into a hierarchy. Instances are ‘things’

¹ <http://oei.ontologymatching.org/>

represented by the concepts. Axioms are assertions in the form of logic to constrain values for classes or properties. Formally an ontology is represented as a 6-tuple [13]:

$$O = \{C, P, H^C, H^P, I, A^O\} \quad (1)$$

where C and P denote classes and properties, H^C and H^P are the hierarchy of classes and properties, I is a set of instances and A^O is a set of axioms. Ontologies can be described by several standard languages, such as the Web Ontology Language (OWL) and Resource Description Framework Schema (RDFS).

The goal of ontology matching is to find correspondences between semantically related entities in different ontologies. In this paper, we only focus on finding the equivalent relations. Here we use the term of ‘entity’ to refer class, property or instance in an ontology. Given a source ontology O_S , a target ontology O_T , and an entity e_i in O_S , the procedure to find the semantically equivalent entity e_j in O_T is called ontology matching, denoted as M . Formally, for each entity $e_i \in O_S$, the ontology matching M can be represented as [16]:

$$M(e_i, O_S, O_T) = e_j \quad (2)$$

3 Semi-supervised Learning for Ontology Matching

In this paper, we formulate the ontology matching problem as a binary classification problem. For each candidate matching, we first use multiple matchers to compute several similarities between the entity pair, and use these similarities as the features of candidate matching. Then we use a semi-supervised learning algorithm, label propagation, to classify candidate matchings to correct matchings and incorrect matchings. We present the details of our algorithm in this section.

3.1 Similarity Metrics and Matchers

There are a lot of ontology information of entities on which comparison can be made to compute similarities, such as label, comment and instance; [8] presents a detailed list of them. In order to compute the similarity, edit-distance and vector-based similarity are two popular similar metrics for ontology matching:

Edit-distance: Edit distance between two strings is the minimal cost of operations (insertion, replacement and deletion of characters) to be applied to one of the strings in order to obtain the other one. Given two strings s and t , the edit-distance-based similarity between them is defined as:

$$S_e(s, t) = 1 - \frac{|\{ops\}|}{\max(|s|, |t|)} \quad (3)$$

where $|\{ops\}|$ indicates the number of operations, $|s|$ and $|t|$ represent the number of characters in s and t respectively.

Vector-based similarity: Vector-based similarity is calculated between two documents in the Vector Space Model (VSM) using the TF/IDF technique. Before computing the similarity, documents are first transformed into a weighted feature vector, where the elements in the vector are weights assigned to words by TF/IDF. For a word i in document j , the weight of the word is computed as

$$\omega_{ij} = tf_{ij} \cdot \lg \frac{N}{df_i} \quad (4)$$

where tf_{ij} is the number of occurrences of i in j , df_i is the number of documents containing i , N is the total number of documents. Then the similarity of two documents d and k is computed as the cosine value between their feature vectors:

$$S_v(d, k) = \frac{\sum_{i=1}^M \omega_{id} \cdot \omega_{ik}}{\sqrt{\sum_{i=1}^M \omega_{id}^2} \cdot \sqrt{\sum_{i=1}^M \omega_{ik}^2}} \quad (5)$$

where M is total number of distinct words in all documents.

Based on the above two similarity metrics, we construct the following four matchers in our proposed approach.

(1) Name-based strategy

$$M_{name}(e_1, e_2) = S_e(label(e_1), label(e_2)) \quad (6)$$

where $label(e)$ is the value of `rdfs:label` of an entity, if there is no information for `rdfs:label`, $label(e)$ corresponds to the last segment of the ontology entity's URI.

(2) Metadata-based strategy

$$M_{meta}(e_1, e_2) = S_v(meta(e_1), meta(e_2)) \quad (7)$$

where $meta(e)$ is a set of words composed by combining the values of `rdfs:label` and `rdfs:comment` of entity e .

(3) Instance-based strategy

$$M_{inst}(e_1, e_2) = S_v(inst(e_1), inst(e_2)) \quad (8)$$

If e is a class, $inst(e)$ is all metadata of the instances belonging to class e ; if e is a property, $inst(e)$ corresponds to all lexical values of property e .

(4) Attribute-based strategy

$$M_{att}(e_1, e_2) = S_v(att(e_1), att(e_2)) \quad (9)$$

where $att(e)$ denotes the values of data type properties of e .

3.2 Label Propagation

Label propagation is an effective graph based semi-supervised learning algorithm. The basic idea behind label propagation is to first build a graph in which each node represents a data point and each edge is assigned a weight usually

computed as the similarity between data points, then propagate the class labels of labeled data to neighbors in the built graph in order to make predictions [17].

Let $L = \{(x_1, y_1), \dots, (x_l, y_l)\}$ be the labeled data, $y \in \{1, \dots, C\}$, and $U = \{x_{l+1}, \dots, x_{l+u}\}$ the unlabeled data, let $n = l + u$. The basic label propagation algorithm builds a graph with the following weights:

$$\omega_{ij} = \exp\left(-\frac{\|x_i - x_j\|^2}{\alpha^2}\right) \quad (10)$$

where α is a bandwidth hyper-parameter. Then it computes a $n \times n$ probabilistic transition matrix P

$$P_{ij} = P(i \rightarrow j) = \frac{\omega_{ij}}{\sum_{k=1}^n \omega_{ik}} \quad (11)$$

where P_{ij} is the probability of transit from node i to j . Let Y_L be a $l \times C$ label matrix which represents the labels of the labeled data, and $f = (f_L, f_U)^T$ be a $n \times C$ matrix represents the soft labels for all the nodes. The label propagation algorithm runs the following process repeatedly until convergence [18]:

- Propagate $f \leftarrow Pf$
- Clamp the labeled data $f_L = Y_L$
- Repeat from step 1 until f converges

Finally, unlabeled data $x_i \in U$ is classified to class $c \in C$ if $f_{ic} > \delta$, where δ is a threshold usually can be set to 0.5.

3.3 Matching Decision by Label Propagation

Give two ontologies O_1 and O_2 , a small set of training matchings, our algorithm first predicts some matchings based on the monotonicity of similarities; then it adds the predicted matchings to the training matchings and feeds these matchings to the label propagation algorithm, which decides the rest matchings.

Enriching Training Examples

Definition 1. Given two entity pairs $\langle e_1, e_2 \rangle$ and $\langle g_1, g_2 \rangle$, let $m = \langle m_1, m_2, \dots, m_k \rangle$ and $n = \langle n_1, n_2, \dots, n_k \rangle$ are similarities of $\langle e_1, e_2 \rangle$ and $\langle g_1, g_2 \rangle$ returned by k matchers, we say that $\langle e_1, e_2 \rangle$ dominates $\langle g_1, g_2 \rangle$, denote $\langle e_1, e_2 \rangle \succeq \langle g_1, g_2 \rangle$, if $m_i \geq n_i$ for all $1 \leq i \leq d$, and exist one $1 \leq j \leq d$ that $m_j > n_j$.

Definition 2. Given two ontologies O_1 and O_2 , a entity pair $\langle e, e' \rangle$, $e \in O_1$ and $e' \in O_2$, is a positive matching if e is equivalent to e' , is a negative matching if e is not equivalent to e' .

In our proposed algorithm, the user should initially verify a small set of positive matchings as training examples. Generally, more training examples enable the algorithm make more accurate predictions of the unknown matchings. Here, we make use of the dominant relations between entity pairs to infer more positive

matchings. Algorithm 1 formalizes the process of enriching positive matchings, its basic idea is that the dominator of a positive matching is also a positive matching. Given a small set of positive matchings, we first use Algorithm 1 to enrich the training set, and then feed the enlarged training set to the label propagation algorithm.

Algorithm 1. Positive matchings enrichment

Input:

Two ontologies O_1 and O_2
A set of positive matchings E

Output:

A new set of positive matchings E'

```

1:  $E' \leftarrow E$ 
2: for all  $\langle e, e' \rangle \in E$  do
3:   if  $\exists \langle g, g' \rangle \succeq \langle e, e' \rangle, g \in O_1, g' \in O_2$  then
4:      $E' \leftarrow E' + \langle g, g' \rangle$ 
5:   end if
6: end for

```

When enriching the training examples, we also add a virtual positive matching v to the training set. The similarity values of v are all set to 1. We assume that if all matchers return similarity values equal to 1 for an entity pair, then this entity pair is a positive matching.

(2) *Adjust the propagation matrix*

We build a kNN graph for the label propagation algorithm, in which each node only connects to the k-nearest neighbors. After obtaining the propagation matrix by Eq. 11, we adjust the propagation matrix based on some prior knowledge to make the predictions more accurate. Here, we use the following constraints (or axioms) to adjust the propagation matrix:

- An entity from O_1 can only match one entity from O_2 , and vice versa;
- No crisscross matching is allowed.
- Two entities match if their *owl:sameAs* or *owl:equivalentClass*, *owl:equivalentProperty* entities match.

Correspondingly, we make the following adjustment to the propagation matrix:

- For each entity pair $i = \langle e, e' \rangle, e \in O_1$ and $e' \in O_2$, set the transition probability $P_{ij} = 0$, where $j = \langle e, g \rangle, g \in O_2 / \{e'\}$, or $j = \langle g, e' \rangle, g \in O_1 / \{e\}$.
- Let $sub(e)$ and $super(e)$ denote the sub entities and super entities of e respectively. For each entity pair $i = \langle e, e' \rangle, e \in O_1$ and $e' \in O_2$, set the transition probability $P_{ij} = 0$, where $j = \langle g, g' \rangle, g \in sub(e)$ and $g' \in super(e')$, or $g \in super(e)$ and $g' \in sub(e')$.

- For two entity pairs $i = \langle e, e' \rangle$ and $j = \langle g, g' \rangle$, set the transition probability $P_{ij} = 1$ if there are *owl:sameAs* or *owl:equivalentClass*, *owl:equivalentProperty* relations between e and g , or between e' and g' .

4 Evaluation

4.1 Datasets

We evaluate our approach on four matching tasks from OAEI'2010 Benchmark dataset. These tasks are #301, #302, #303 and #304, which involve matching four real biography ontologies with one common ontology. In each matching task, there are more than 100 classes and properties in the ontologies. We evaluate our approach on these tasks because the ontologies are all realistic ones, the participants of OAEI can not get satisfying results on these tasks.

4.2 Performance Metrics

We use precision, recall, and F1-Measure to measure the performance of our proposed approach, which are defined as follows:

Precision (p): It is the percentage of correctly discovered matchings in all discovered matchings.

$$p = \frac{|A \cap T|}{|A|} \quad (12)$$

where A is the set of discovered matchings, and T is the set of ground truth matchings.

Recall (r): It is the percentage of correctly discovered matchings in all correct matchings.

$$r = \frac{|A \cap T|}{|T|} \quad (13)$$

Table 1. Experimental results

Tasks		#301	#302	#303	#304
0 % training	Pre.	0.90	0.96	0.90	0.92
	Rec.	0.71	0.58	0.80	0.79
5 % training	Pre.	0.91	0.96	0.90	0.95
	Rec.	0.75	0.60	0.80	0.89
10 % training	Pre.	0.93	1.00	0.90	0.96
	Rec.	0.75	0.60	0.81	0.89
15 % training	Pre.	0.94	1.00	0.91	0.96
	Rec.	0.75	0.61	0.83	0.92
20 % training	Pre.	0.96	1.00	0.91	0.97
	Rec.	0.77	0.63	0.85	0.95

F1-Measure ($F1$): F1-Measure considers the overall result of precision and recall.

$$F1 = \frac{2pr}{p + r} \quad (14)$$

4.3 Results

In the experiments, we run four group of experiments on each matching task. In these four group of experiments, we use 0 %, 5 %, 15 %, 20 % randomly selected matchings in the reference alignment as the training examples respectively. In each group of experiments, we run our algorithm ten times and each time on a different sample of training examples. When using 0 % training examples, we only use the virtual positive matching as training example.

Table 1 lists the results on four matching tasks. For each matching tasks, we list the average precision and recall on different samples of training examples.

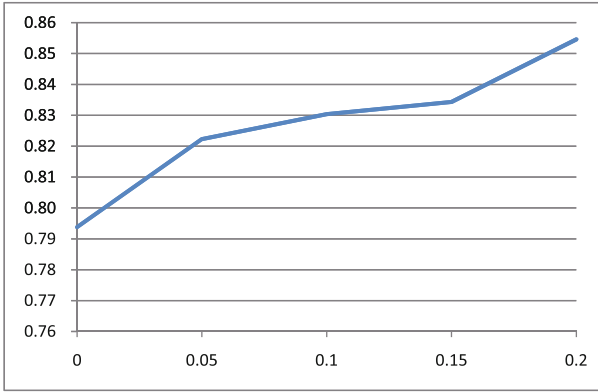


Fig. 1. F-Measure on #301 matching task

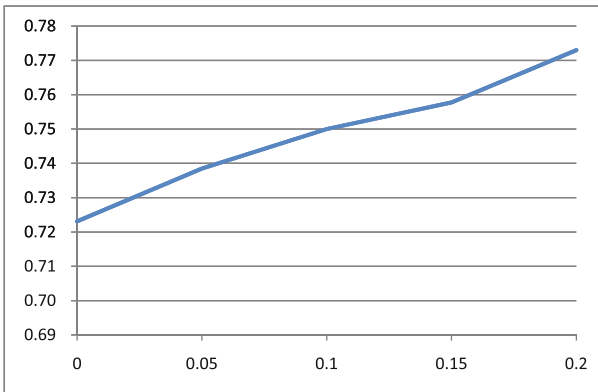


Fig. 2. F-Measure on #302 matching task

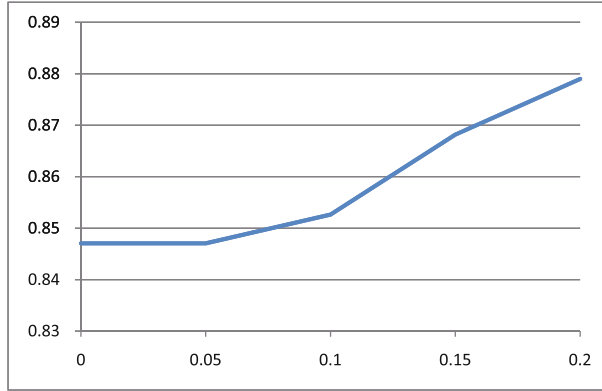


Fig. 3. F-Measure on #303 matching task

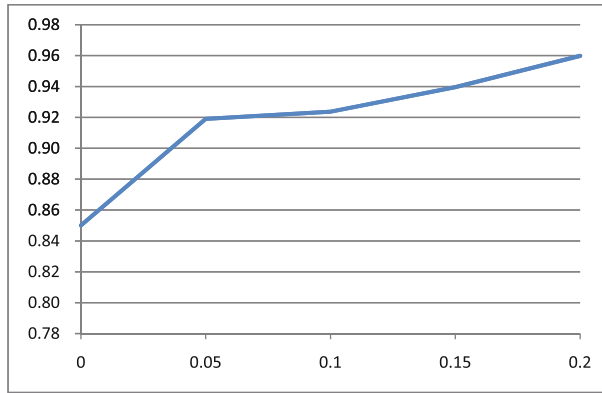


Fig. 4. F-Measure on #304 matching task

Figures 1, 2, 3 and 4 show the increments of F-measures as the number of training examples increase. It can be observed that if we use no training examples, the algorithm can also get a acceptable results. As we increase the number of training examples, the F-measures also increase. Therefore, our proposed algorithm can help improve the quality of matching results with training examples.

5 Related Work

In this section, we review related work on ontology matching.

5.1 Non-learning Based Approaches

COMA [1] is a schema matching tool, which provides a library of matching algorithms. COMA allows the user to use different algorithms and combination strategies, such as MAX, MIN, AVG and SIG, but users should select the strategies and combination method according to the matching problem. ASMOV [12] is an automated ontology matching system that combines a comprehensive set of element-level and structure-level measures of similarity with a technique that uses formal semantics to verify whether computed correspondences comply with desired characteristics. ASMOV computes four kinds of similarities, including a lexical similarity, two structural similarities and an external similarity. It combines these similarities using weighted sum, where the weights are adjusted based on static rules. PRIOR+ [14] is an adaptive ontology mapping approach, which contains three major modules, i.e. the IR-based similarity generator, the adaptive similarity filter, weighted similarity aggregator, and the neural network based constraint satisfaction solver. Both the linguistic and structural similarities of the ontologies are computed in a vector space model, and then are aggregated by an adaptive method based on their harmonies. RiMOM [13] is a dynamic multistage ontology matching framework. It uses both lexical and structural information of ontologies to compute similarity between entities. In order to adaptively combine multiple matching strategies, RiMOM estimates the similarity characteristics for each matching task by computing two factors: the label similarity factor and the structural similarity factor. UFOME [9] is an ontology matching software framework, which provides a library of matchers and a strategy prediction module which suggests individual matchers should be used in a specific task. The strategy prediction module computes three coefficients of two ontologies, including lexical affinity coefficient, structural affinity coefficient, and exploiting affinity coefficient. Based on these coefficients UFOME suggests the optimal weights value for different matchers. The strategy prediction in UFOME is quite similar to RiMOM’s weighting strategy. Falcon-AO [10] is a similarity-based ontology matching system. It employs three matching strategies, i.e., V-Doc, I-Sub and GMO. V-Doc builds a virtual document for each entity, and then measures their similarity in a vector space model. I-Sub computes similarities between strings attached to different entities. GMO explores structural similarity based on a bipartite graph.

5.2 Learning Based Approaches

FOMA [5] achieves high-quality results by using a matching learning method to draw classification rules for combining results of different similarity metrics. Our approach is an unsupervised one therefore it does not need training examples. Nevertheless, it still achieves good performance in various matching tasks. GLUE [2] is a well-known machine learning based ontology matching system. It employs a set of base learners and then combine their predictions using a metalearner. Base learners apply machine learning techniques to compute the joint probability distributions of every entity pair, then use the joint distributions to compute

the Jaccard similarity. The metalearner uses manually set weights to combine the base learners' predictions via the weighted sum. Eckert et al. [4] propose to use matching learning techniques to train a classifier that decides whether two elements from different ontologies should be linked by an equivalence relation based on the output of different matchers. They do not make any assumptions about the matchers, which can be simple similarity metrics or composite matching systems. The machine learning algorithm is also not particularly specified, it can be Decision Tree or Naive Bayes. APFEL [6] is another ontology matching system based on machine learning method. It computes similarities using various ontology information, and then trains a decision tree to predict the equivalence relations between the entities of two ontologies. Different from other machine learning based approaches, APFEL first uses an unsupervised approach to find some correct matchings, then uses these matchings as training examples. Most recently, [3] has proposed an ontology matching approach which interact with user feedback. Their algorithm uses logistic regression to aggregate similarities from different matchers. The training examples also are used to decide the degree of structural propagation.

6 Conclusion

In this paper, we propose a semi-supervised learning approach for ontology matching. Our approach uses four matchers to compute similarities between entities and then employs the label propagation algorithm to make the matching decisions. In order to improve the quality of matching result, we first explore the dominant relations in the similarity space to infer more training examples, and also adjust the propagation portability matrix based on domain knowledge. Experimental results show our proposed approach can effectively improve the matching results with training examples.

Acknowledgement. The work is supported by NSFC (No. 61202246), NSFC-ANR (No. 61261130588), and the Fundamental Research Funds for the Central Universities (2013NT56).

References

1. Do, H.-H., Rahm, E.: Coma: a system for flexible combination of schema matching approaches. In: Proceedings of the 28th International Conference on Very Large Data Bases (VLDB '02), pp. 610–621 (2002)
2. Doan, A., Madhavan, J., Dhamankar, R., Domingos, P., Halevy, A.: Learning to match ontologies on the semantic web. *VLDB J.* **12**, 303–319 (2003)
3. Duan, S., Fokoue, A., Srinivas, K.: One size does not fit all: customizing ontology alignment using user feedback. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) *ISWC 2010, Part I. LNCS*, vol. 6496, pp. 177–192. Springer, Heidelberg (2010)

4. Eckert, K., Meilicke, C., Stuckenschmidt, H.: Improving ontology matching using meta-level learning. In: Aroyo, L., Traverso, P., Ciravegna, F., Cimiano, P., Heath, T., Hyvönen, E., Mizoguchi, R., Oren, E., Sabou, M., Simperl, E. (eds.) ESWC 2009. LNCS, vol. 5554, pp. 158–172. Springer, Heidelberg (2009)
5. Ehrig, M.: Foam - framework for ontology alignment and mapping; results of the ontology alignment initiative. In: Proceedings of Integrating Ontologies Workshop (2005)
6. Ehrig, M., Staab, S., Sure, Y.: Framework for ontology alignment and mapping
7. Ehrig, M., Staab, S., Sure, Y.: Bootstrapping ontology alignment methods with apfel. In: Special Interest Tracks and Posters of the 14th International Conference on World Wide Web, WWW '05, New York, NY, USA, pp. 1148–1149. ACM (2005)
8. Euzenat, J., Shvaiko, P.: *Ontology Matching*, 1st edn. Springer, New York (2007)
9. Giuseppe, P., Talia, D.: Ufome: an ontology mapping system with strategy prediction capabilities. *Data Knowl. Eng.* **69**(5), 444–471 (2010)
10. Hu, W., Falcon-ao, Y.Q.: A practical ontology matching system. *Web Semant. Sci. Serv. Agents World Wide Web* **6**(3), 237–239 (2008)
11. Isabel, F.P.A., Cruz, F., Stroe, C.: Efficient selection of mappings and automatic quality-driven combination of matching methods. In: Workshop on Ontology Matching, pp. 49–60 (2009)
12. Jean-Mary, Y.R., Shironoshita, E.P., Kabuka, M.R.: Ontology matching with semantic verification. *Web Semant. Sci. Serv. Agents World Wide Web* **7**(3), 235–251 (2009)
13. Li, J., Tang, J., Li, Y., Luo, Q.: Rimom: a dynamic multistrategy ontology alignment framework. *IEEE Trans. Knowl. Data Eng.* **21**(8), 1218–1232 (2009)
14. Mao, M., Peng, Y., Spring, M.: An adaptive ontology mapping approach with neural network based constraint satisfaction. *Web Semant. Sci. Serv. Agents World Wide Web* **8**(1), 14–25 (2010)
15. Peukert, E., Mamann, S., Knig, K.: Comparing similarity combination methods for schema matching. In: GI Jahrestagung (1)'10, pp. 692–701 (2010)
16. Tang, J., Li, J., Liang, B., Huang, X., Li, Y., Wang, K.: Using bayesian decision for ontology mapping. *Web Semant.* **4**(4), 243–262 (2006)
17. Wu, M.: Label propagation for classification and ranking. Ph.D. thesis, East Lansing, MI, USA. AAI3282228 (2007)
18. Zhu, X.: Semi-supervised learning with graphs. Ph.D. thesis, Pittsburgh, PA, USA. AAI3179046 (2005)
19. Zhu, X., Ghahramani, Z., Lafferty, J.: Semi-supervised learning using Gaussian fields and harmonic functions. In: ICML, pp. 912–919 (2003)

The Semantic Web and Web Science

8th Chinese Conference, CSWS 2014, Wuhan, China,

August 8-12, 2014, Revised Selected Papers

Zhao, D.; Du, J.; Wang, H.; Wang, P.; Ji, D.; Pan, J. (Eds.)

2014, XII, 252 p. 96 illus., Softcover

ISBN: 978-3-662-45494-7