Open Relation Mapping Based on Instances and Semantics Expansion

Fang Liu, Shizhu He, Shulin Liu, Guangyou Zhou, Kang Liu, and Jun Zhao

Institute of Automation, Chinese Academy of Sciences
{fliu,szhe,shulin.liu,gyzhou,kliu,jzhao}@nlpr.ia.ac.cn

Abstract. Mining the semantics of open relations is an important task in open information extraction (Open IE). For this task, the difficulty is that the expressions of a specific semantic in free texts are always not unique. Therefore, it needs us to deeply capture the semantics behind the various expressions. In this paper, we propose an open relation mapping method combining the instances and semantic expansion, which maps the open relation mentions in free texts to the attribute name in knowledge base to find the real semanics of each open relation mentions. Our method effectively mines semantic expansion beyond the text surface of relation mentions. Experimental results show that our method can achieve 74.4% average accuracy for open relation mapping.

Keywords: open relation mapping, semantic mining, relation paraphrase.

1 Introduction

In open information extraction, the expressions of a specific semantic relation between two entities are various. For example, we use "X was born in Y", "X's hometown Y" or "X's birth in Y" to express a semantic relation "birthPlace", where X and Y are two named entities. These expressions, like "was born in", " 's hometown" and "'s birth in" are called relation phrases [1,2,3] or relational patterns [1] (we call them open relations or relation mentions in this paper). We notice that there are textual mismatches between relation mentions and the corresponding semantic, which is a big obstacle for semantic discovery in many applications, like natrual language understanding, ontology-based question answering etc. For example, when we ask "Where is the hometown of Yao Ming?", question answering system can give the right answer if it knows the target semantic relation is "birthPlace", then the system will match the attribute value of "birthPlace" for "Yao Ming" from knowledge base. However, if the relation words in question is aforementioned "hometown", and the attribute name stored in knowledge base is "birthPlace", it's difficult to obtain the correct answers since that they are mismatched on the surface forms (detailedly illustrated in Tab.1). Therefore, constructing the mapping between relaion mentions and the semantic relation names stored in KB (shortly named as Open Relation Mapping) is very important and meaningful, which is the focus of this paper.

Formally, the source of open relation mapping is the relation mentions in free texts, and the target is the relations in knowledge base, which are usually human edited attributes (we call it attribute relation later in this paper). Intuitively, a relation mention

R.E. Banchs et al. (Eds.): AIRS 2013, LNCS 8281, pp. 320–331, 2013.

and an attribute can be directly linked if they share the common entity pairs (which are called relation instances). Unfortunately, this assumption is not always correct. As illustrated in Fig.1, ⟨ Yao Ming, fly back to, Shanghai⟩ is mapped to "birthPlace" according to the common instance (⟨ Yao Ming, Shanghai⟩ in Fig.1) assumption, which is obviously incorrect.

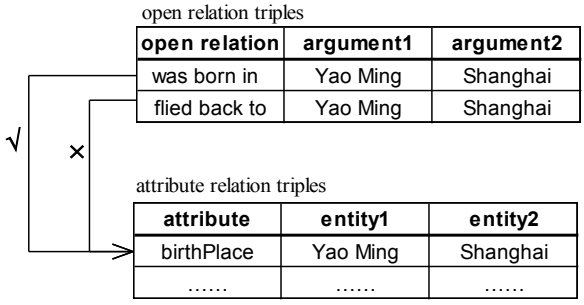

Fig. 1. An example of coincidental match in open relation mapping

Therefore, beside the relation instances, we must further consider the semantic similarity between relation mentions and relation names (attributes) in KB. For example, if we know that "hometown" has the similar semantic with "birthplace", it's easily to construct a mapping between them. However, simply computing this similarity based on surface texts, like edit distance, is insufficient. Tab.1 lists some examples of relations mappings, majority of which are mismatched by simply using surface textual. For example, "received degree in" has not any common words with attribute "yago:graduatedFrom", but they actually should be mapped. Thus, we must mine the deeper semantics behind the texts.

For this aim, we propose a novel method for open relation mapping which considers both of the semantics of relation mentions and relation instances. Our method mainly contains two steps. First, we use relation instances to generate mapping candidates. Second, we compute the semantic similarity between each candidates and the target relation mention. The candidate with similarity score above a threshold will be linked to the relation mention. In specific, we explore external resources of knowledge, like Wikipedia, to mine concepts for the semantic representation. Then, the semantic similarity of relation mention and attribute candidate is computed on this semantic space. We further make semantic expansion for each attribute to improve the mapping performance by using WordNet synset. The behind reason is that directly mining the similarity between attribute candidates and relation mentions may miss much useful information. To demonstrate the effectiveness of the proposed method, we use YAGO [4] as knowledge base, PATTY [1] as open relation dataset. Experimental results on five kinds of relations show that our method can achieve 74.4% average accuracy for open relation mapping, and the proposed semantic expansion method can effectively improve the mapping performance.

Table 1. Some mismatch examples of open relations and attributes

ID	Relation Name in KB (Attribute)	Open Relation
1	yago:wasBornIn	was born in
2	yago:wasBornIn	's hometown
3	yago:wasBornIn	's birth in
4	yago:created	creator of
5	yago:graduatedFrom	received degree in
6	yago:actedIn	played role in films

2 Related Work

In the early days, relation extraction (RE) is relation-driven. Target relations are predefined according to task, then instances are constructed for every target relation to gain relation patterns, and then relation patterns are used to collect more instances which share the same relation. The number of target relations in relation-driven extraction is limited and predefined. With the invention of open information extraction, RE became data-driven. Data-driven RE extracts relation triples from free texts whenever two entities are detected associated by a relation by a trained classifier. The number of relations in data-driven RE is unbounded and the target relations are not known in advance. The data-driven extraction is regarded as "Open Information Extraction"(Open IE) [1,5,6,7,8]. As the relation is not known, additional mapping processing is required to mine the semantics of relation expressions.

Semi-supervised mapping methods like active learning are used to match relation phrases or mentions to domain-specific relations, like NFL-scoring [9] or nutrition domain [10]. Active learning method still requires some human labeling. Later distant supervision [11,12,13] is proposed, and the target domain of mapping is extended to more general domains covered by knowledge base such as YAGO [4], DBpedia [14] etc. Then knowledge-base-supervised method is exploited to map open relations. Takamatsu etc. [12] and Surdeanu etc. [13] present a generative model to model the labeling process of distanct supervision according to statistical feature of instances. Semantics is the key to avoid some wrong mapping, like coincidental match where mapping open relation "flied back to" to attribute "wasBornIn" in light of the provenance of "Yao Ming flied back to Shanghai".

Research about open relations has attracted increasing attentions, as open relation getting more important. Yates and Etzioni [2], Kok and Domingos [15] cluster relation phrases and entities at the same time, further Min et al. [3] loose the constraint that each entity or relation phrase belongs to one cluster to handle polysemy relations. But the above work group synonymy relation phrases into clusters without explicit semantics of each cluster. PATTY [1] constructs a WordNet-style hierarchical taxonomy for binary relation patterns (phrases). It also paraphrases canonical relations in DBpedia and YAGO knowledge bases with relation patterns according to common instances assumption. They did not validate the correctness of the semantics of paraphrases, there are too many noisy mappings and cannot be used directly in applications. For example, "died just""buried in""[[con]] graduated in" are regarded as paraphrase for "yago:wasBornIn".

3 Our Method

In this section, we describe our method in detail. The main framework is illustrated in Fig. 2.

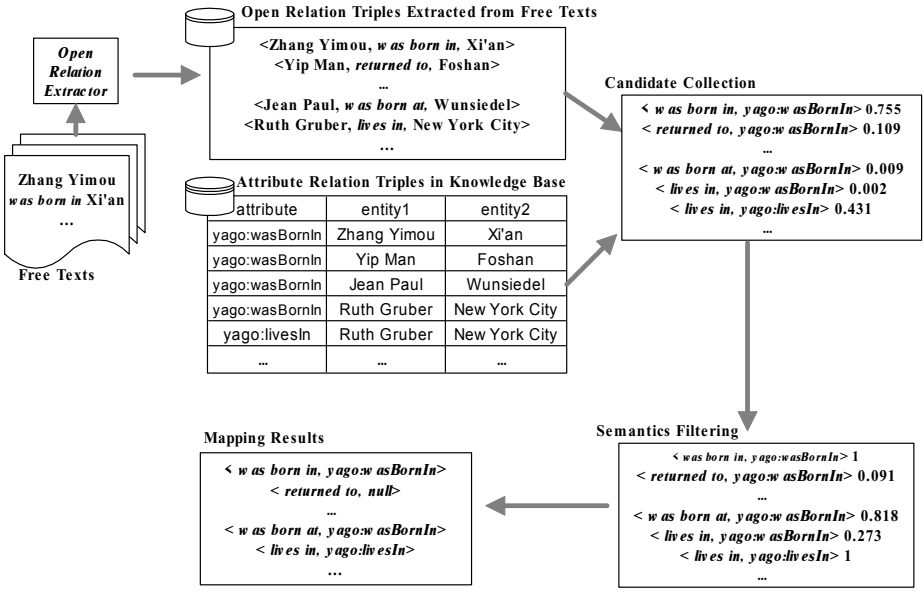

Fig. 2. Flowchart of proposed mapping algorithm

There are two parts of input data. One is relational triples stored in knowledge base taking the form of ⟨ attr, ent1, ent2⟩. where *attr* is the attribute relationship holds between the first entity *ent1* and the second entity *ent2*. Such as ⟨ yago:wasBornIn, Zhang Yimou, Xi'an⟩ etc. The other is relation triples extracted from free text taking the form of ⟨ arg1, open, arg2⟩. *open* is an open relation expressing the relation between the first argument *arg1* and the second argument *arg2*. Such as ⟨ Zhang Yimou, was born in, Xi'an⟩ etc.

The output is the explicit semantics of open relations which is called mapping pair. For example, mapping pair ⟨ was born at, yago:wasBornIn⟩ means "was born at" expresses the semantics of "yago:wasBornIn"; mapping pair ⟨ returned to, null⟩ means "returned to" expresses no semantics defined in knowledge base.

It takes two steps to implement our mapping algorithm: candidate collection and semantics filtering.

Candidate Collection. There are thousands of attributes recorded in knowledge base, YAGO[4] has 72[1] attributes , DBpedia[14] has 48,293[2] . It is time consuming, and also unnecessary to compare every attribute with open relations to eliminate unrelated

[1] http://www.mpi-inf.mpg.de/yago-naga/yago/statistics.html
[2] http://wiki.dbpedia.org/Datasets/DatasetStatistics

candidate attributes. Hence, we use common instances assumption to find candidate attributes as the semantics of every open relation. This process was shown at the upper part in Fig. 2. From the fact that entity pair ⟨ Ruth Gruber, New York City⟩ is shared by open relation " lives in" (Row 5 in open extracted triples in Fig. 2) and two attributes: "yago:wasBornIn" and "yago:livesIn" (Row 4 and 5 in knowledge base triples in Fig. 2), we infer that "yago:wasBornIn" and "yago:livesIn" are candidate attributes for open relation "lives in". We regard attributes which share common instances with open relations as candidate semantics. This assumption may lead to coincidental match error, therefore next we mine semantics similarity to filter out irrelevant attributes.

Semantics Filtering. Statistical features alone, that is the co-occurrence frequency of open relations and attributes may lead to mapping error. In Fig. 2 "returned to" and "yago:wasBornIn" have a higher co-occurrence frequency (0.109) than "was born at" and "yago:wasBornIn" (0.009), but "was born at" has much more closer meaning with "yago:wasBornIn" than "returned to". We filter out unrelated attributes to mitigate this kind of mapping problem by computing semantic similarity between relation mention and its candidates.

After explaining the function of the two steps in our mapping algorithm above, we will give the details of the implementation.

3.1 Candidate Collection

Generating candidate attribute is based on common instances assumption, which requires matching the aforementioned entity1 and entity2 in the relation triple simultaneously. It takes two phrases to collect attribute candidates for every open relation. First is to put binary relations into database, and make index to facilitate subsequent searching. Second is online searching and matching, which read each open extracted relation triples, then search entity1 and entity2 in prepared database. If finding attrib-ute relation triples that have arguments are also entity1 and entity2, we should then save returned attributes as candidate semantics of open relation.

After the above mapping, a series of mapping pairs ⟨open relation, candidate attribute⟩ are collected. As every mapping pair shares at least one common entity pair, we can make a statistics about the number of common instances for every mapping pair and compute the confidence as Eq.1.

$$confidence_{<open,attr>} = \frac{N_{open \cap attr}}{N_{attr}} \tag{1}$$

Where, N_{attr} is the occurrence times of attribute attr in mapping set. $N_{open \cap attr}$ is the occurrence times of mapping pair ⟨open, attr⟩ in mapping set. As shown in Fig. 2, the mapping pair ⟨ was born in, yago:wasBornIn⟩ in first step has a confidence of 0.755, meaning that the probability is 0.755 when an entity pair associated by semantic relation "yago:wasBornIn" is expressed by "was born in".

3.2 Semantics Filtering

In the second step, we compute the semantic similarity between candidate attributes retrieved in the first step and the extracted relation mentions to make filtering. We exploit

empirical threshold to make decisions on whether an open relation can be mapped to an attribute. If similarity value is above a threshold, two strings have mapping relation. Otherwise, they do not. This strategy can map one open relation to multiple semantic relations and map unrelated open relations to null.

We notice *open* in open relation triples and *attr* in relational database are both tokens. For example, relation in knowledge bases is defined like "YAGO:wasBornIn", relation phrases openly extracted are like "'s hometown of" etc. Therefore token-based strings similarity method is required to accumulate the similarity between every two tokens from attribute and relation mention respectively. In our observation, when phrases contains head words of relation, like phrase "' birth in" containing common head word "birth" of "birthPlace", we have very great confidence that "' birth in" indicating relation "birthPlace". According to this feature, we prefer Generalized Mongue-Elkan (GME) proposed by Jimenez et al. [16] as our external similarity, which is computed as follows:

$$sim(A,O) = \left[\frac{1}{|A|} \sum_{\{a_i\} \in A} \left(\max_{\{o_j\} \in O} sim'(a_i, o_j) \right)^m \right]^{\frac{1}{m}} \tag{2}$$

Where, A stands for attribute, O for open relation. When m>2, GME gives greater importance to those pairs of tokens [a_i,o_j] that are more similar, which is the exact feature we need. In our experiment later, m is set to 2. Then the next problem becomes choose internal similarity sim' to meet our mapping requirement which is to capture semantic similarity and immunize to expressions variations.

Semantics Filtering Based on Text Surface. Edit-based string similarity method (like Levenshtein Distance [17] etc.) finds the similarity from textual surface forms in which words share common substrings, like "yago:wasBornIn" and "was born in" in Tab.1-1. It cannot handle different expressions to the same semantics, like "'s hometown" expresses the meaning of "yago:wasBornIn" as shown in Tab.1-2. This method is not fit to open relation mapping.

Semantics Filtering by Using Semantics. Alternately, we employ some lexical resources like experts generated WordNet [18], crowdsources generated Wikipedia etc., which expand the extensional meaning of words to mine implicature. As WordNet labels the semantics relations among words, WordNet-based method could find some morphological changes, such as "born" is synonymous to "birth" (Tab.1-3). It is unable to compute the similarity between words with different part-of-the-speech (POS). The similarity score of "created" and "creator" (example shown in Tab1-4) computed by Jiang&Conarth [19] or Lin [20] methods which exploit the taxonomy of WordNet is all zero. Wikipedia-based methods (WikiMiner [21], WikiRelate! [22], Explicit Semantics Analysis (ESA) [23] etc.) taking advantage of the vast amounts of highly organized knowledge encoded in Wikipedia could mine the similarity between words according to their linkage, hierarchy or co-occurrence information in Wikipedia. However, WikiMiner [21] compares proper nouns, like person or organization name, which share

common inner-linked pages. WikiRelate! [22] compares words that occur in titles of Wikipedia articles. ESA [23] compares words appeared within the text of Wikipedia articles. For our task, ESA is much more reasonable methods, as open relations are textual form of attributes. However, the text relatedness computed by ESA is not specified for open relation mapping. In the Wikipedia Concept space, "graduated from" is more related to "degree" than "received degree in", as "received" introduced a dilution.

Semantics Filtering by Using Semantic Expansion. As existing string similarity methods, like edit/character-based, WordNet-based and Wikipedia- based methods, have their limits in open relation mapping. Therefore, we proposed a semantic similarity computation method combining two lexical resources WordNet and Wikipedia under the strategy of GME [16]. In which, WordNet is used to expand the meaning of attributes words to adapt to various changes of open relations. Wikipedia is used to construct a semantic space which is defined as Wikipedia Concepts. Then the similarity of attribute candidates and relation mentions are computed under that semantic space by ESA tool. ESA stems [24] all substantivals in Wikipedia texts, and then maps them to Wikipedia Concepts. Take "acted" and "played role in films" as an example to illustrate the procedure of our proposed se-mantic similarity computation method (shown in Fig.3).

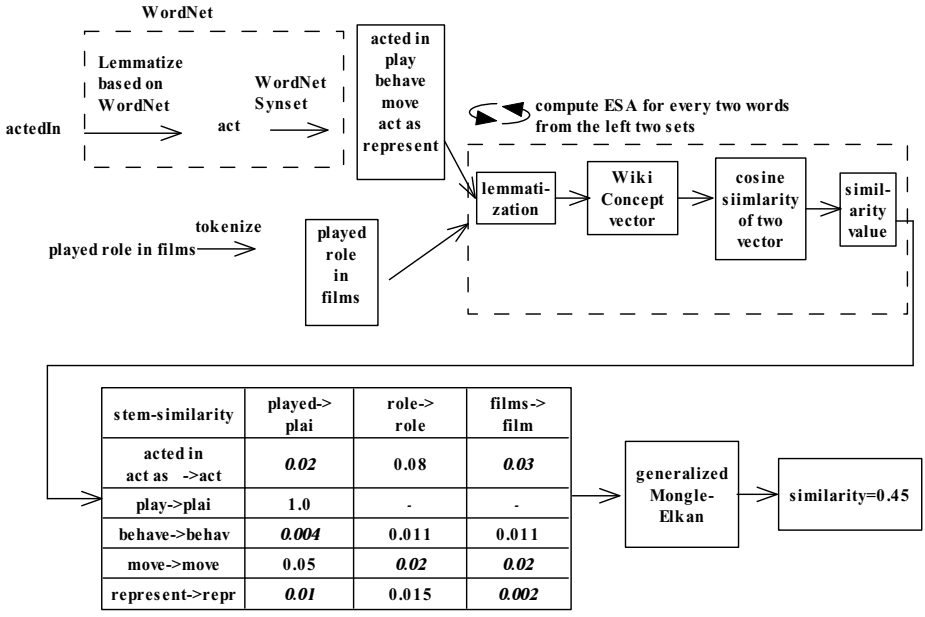

Fig. 3. Take "acted" and "played role in films" as an example to illustrate the flowchart of proposed semantics similarity computation method

The procedure is as follows:

First, tokenize attribute, delete stop words, lemmatize remain words, expand lemmatized words using WordNet synset, construct a string set by adding attribute to WordNet synset. As shown in top left of Fig.3, the yago attribute “actedIn” is lemmatized by WordNet to be “act”. Then a synset of the lemmetized word is collected from WordNet as acted in, play, behave, move, act as, represent. Meanwhile, tokenize the compared open relation, and regard tokenized words as a string set. As we can see from left center of Fig.3, open relation ”played role in films” is tokenized to be played, role, in, films.

Second, compute the word similarity by ESA for every pair word from the two string set, then get a matrix of similarity value. We iteratively pick one word from expanded attribute string set, compare it with every word in tokenized open relation string set using ESA. For example, ESA first stems “played” to “plai”, “acted in” to “act”, and retrieve the Wikipedia Concept vector corresponding to “plai” and “act” respectively. Then compute the cosine similarity between the retrieved vector to ob-tain the semantic relatedness of “played” and “acted in”. Repeat the above process to get a matrix of stem-similarity as shown is lower left.

Third, put the value in matrix into GME [16] (Eq.3), then get a similarity of the two relations. Finally, store the mapping from open relation to attribute when the similarity of the two is larger than the threshold. In our example, substitute the value in matrix into Eq.2, we get the similarity value between “played role in films” and “actedIn” is 0.45.

4 Experiments

We use PATTY[3] as open relation triples input and attribute relation triples in YAGO2[4] as static relation input. The open relations of PATTY are extracted from the New York Times archive (NYT) which includes about 1,800,000 newspaper articles from the years 1987 to 2007. YAGO contains 72 attributes, 10 million entities and 120 million relation facts. The Wikipedia (WKP) data used to train ESA[5] is the English version, which contains about 3,800,000 articles (as of August 3, 2011). All relational triples are stored in a MySQL database. We evaluated mapping quality along five representative attributes: actedIn (who appeared in which show), created(who created which novel thing), graduatedFrom(who graduated from which school), hasAcademicAdvisor(who got academic guidance from who), wasBornIn(who was born in where). “wasBornIn” is the most noisy relation type. “created” has the most transformations of lexicon. “hasAcademicAdvisor” is a much ambiguous and noisy relation. “actedIn” has an acceptable quality only with instance mapping. “graduatedFrom” does not have many changes in surface form without much ambiguity.

To assess the quality of relations mapping, we ranked the open relations mapped to the above mentioned attributes and evaluated the precision of the top 100. Human judgment stated whether an open relation indicated the semantics of its mapped attribute. If so, the mapping pair was labeled as 1. If not, labeled 0. If not sure labeled 0.5.

[3] http://www.mpi-inf.mpg.de/yago-naga/patty/

[4] http://www.mpi-inf.mpg.de/yago-naga/yago/

[5] http://ticcky.github.io/esalib/

To compare the effect of different semantics mining methods, we used the results of Candidate Collection as baseline, adding edit-based analysis, WordNet[6] lexical resource, Wikipedia resource and our proposed method respectively.

1. **Baseline:** This method ranks the mapping results according the confidence computed by Equation 1. This is to evaluate the quality of mapping using only instances without semantics, which is an re-implement of mapping method of PATTY[1].
2. **Edit-based Semantics Mining:** Add edit-based string similarity filtering method to baseline. This method computes the edit distance [17] between open relation and attribute, then transform the distance into a normalized similarity value. The threshold is 0.5.
3. **WordNet-based Semantics Mining:** Add WordNet-based string similarity filtering method to baseline. First, we lemmatize two comparing strings using WordNet, pick the every same POS from their lemmas then compute the similarity using Jiang&Conrath[19], we take the maximum similarity from the similarity set. The threshold is set to 0.5
4. **Wikipedia-based Semantics Mining:** Add Wikipedia-based string similarity filtering method to baseline. Computes the similarity of open relation and attribute by ESA[23]. The threshold is set to 0.05.
5. **Our Combining method:** Our method computes the similarity of two strings by our method combining WordNet and Wikipedia. The threshold is set to 0.05.

Table 2. The precision of top100 results under different methods

precision	actedIn	created	graduated From	hasAcademic Advisor	wasBornIn	average
Baseline	0.725	0.67	0.57	0.31	0.22	0.499
EditDistance	0.66	**0.915**	0.95	0.195	0.42	0.628
WordNet	0.865	0.61	**0.99**	0.485	0.545	0.699
Wikipedia	0.91	0.775	0.825	0.495	0.485	0.698
Proposed	**0.965**	0.815	0.855	**0.575**	**0.51**	**0.744**

As shown in Tab. 2, we can see that for “actedIn”, “hasAcademicAdvisor” these two attributes with various lexical changes in their open relations, adding edit-based semantics mining method decreases the performance with lower precision than baseline. However, adding resources boosts the performance to reach higher precision than baseline, in which our combining method is better than adding single resource methods. For attribute “graduatedFrom” and “created” , there are no much lexical changes in their open expressions, meanwhile, headwords of attributes , like “graduated” for “graduatedFrom”, “created” for attribute “created”, strongly indicate the semantics of attributes. For these kinds of attributes, adding edit-based methods can significantly improve the relation performance mapping precision. Resource-based methods are not as good as edit-based method under top100 precision assessment, but still achieve better quality

[6] `http://wordnet.princeton.edu/`

than baseline for these attributes. For noisy attribute like "wasBornIn", the precision of mapping with only instances is quite low. When adding different semantics mining resource, the quality will get improvement by different degrees.

From the average performance (Col. 7 in Tab. 2), we can see that relation mapping adding semantics is better than mapping with only instances. Resource-based semantics mining method is better than edit-based method. WordNet-based semantics mining method has a comparable quality with Wikipedia-based method. Our combining method outperforms the other semantic mining methods. The average accuracy of our method for open relation mapping achieves 74.4% which is 20% higher than the baseline.

To further evaluate the effectiveness of our proposed method, we compare the top 5 results between PATTY and our method for YAGO attribute "wasBornIn", "created", "actedIn" shown in Tab. 3. From this table, we can see that the semantics filtering strategy of our method can better reduce the coincidental matching error.

Table 3. The top5 results of PATTY and our method for some attributes

	Attribute	PATTY	Our
1	wasBornIn	[[con]] grew up	born in
2		died just	was born [[con]]
3		buried in	is born to
4		was born [[con]] was raised in	[[adj]] was born in
5		[[con]] graduated in	been born
1	created	s album [[num]]	also created for
2		released	in creating
3		released [[prp]] debut album [[det]]	also created
4		[[adj]] studio album [[adj]]	[[det]] created
5		[[adj]] song in	[[mod]] create
1	actedIn	also starred in	also played supporting roles in films [[adj]]
2		starred in [[det]] film	played supporting roles as
3		twice won [[det]]	played supporting [[con]] [[adj]] roles in
4		appeared in [[adj]] film	played supporting roles in
5		also acted [[con]]	played supporting roles

5 Conclusion and Future Work

In order to reduce noisy mappings, we propose a novel open relation mapping method, which combines the instances and semantics, to mine the explicit semantics of open relations, such as hand-editing attribute names in knowledge base. We mine semantics in two steps. In the first step, we use instance information to collect candidate attributes. And in the second step, we filter out the unrelated attributes using our novel semantic similarity method which takes advantage of WordNet and Wikipedia resources. Experimental results show that our method can achieve 74.4% average accuracy for open

relation mapping and the proposed semantic expansion method can effectively improve the mapping performance.

Our future works include two aspects:

1. Handling redirection of entities. There may be more than one spelling of a certain entity in texts. We can get more semantic information by recognizing various mentions to same entity.
2. Trying to find other methods to combine various features. Our present method considers the entity pairs and semantics separately. In the future, we will consider other methods which can capture interaction effects between entity pairs and semantics.

Acknowledgments. This work was supported by the National Natural Science Foundation of China (No. 61070106, No. 61272332 and No. 61202329), the National High Technology Development 863 Program of China (No. 2012AA011102), the National Basic Research Program of China (No. 2012CB316300) and the Opening Project of Beijing Key Laboratory of Internet Culture and Digital Dissemination Research (ICDD2 01201).

References

1. Nakashole, N., Weikum, G., Suchanek, F.: Patty: a taxonomy of relational patterns with semantic types. In: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 1135–1145. Association for Computational Linguistics (2012)
2. Yates, A., Etzioni, O.: Unsupervised resolution of objects and relations on the web. In: Proceedings of NAACL HLT, pp. 121–130 (2007)
3. Min, B., Shi, S., Grishman, R., Lin, C.Y.: Ensemble semantics for large-scale unsupervised relation extraction. In: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 1027–1037. Association for Computational Linguistics (2012)
4. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: A large ontology from wikipedia and wordnet. Web Semantics: Science, Services and Agents on the World Wide Web 6(3), 203–217 (2008)
5. Zeng, D., Lai, S., Zhang, Y.: Open entity attribute-value extraction from unstructured text. In: Proceedings of the 2012 China Conference on Information Retrieval. Jiangxi Normal University (2012)
6. Zhao, J., Liu, K., Zhou, G., Cai, L.: Open information extraction. Journal of Chinese Information Processing 25(6), 98–110 (2011)
7. Yates, A., Cafarella, M., Banko, M., Etzioni, O., Broadhead, M., Soderland, S.: Textrunner: open information extraction on the web. In: Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations, pp. 25–26. Association for Computational Linguistics (2007)
8. Fader, A., Soderland, S., Etzioni, O.: Identifying relations for open information extraction. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 1535–1545. Association for Computational Linguistics (2011)
9. Soderland, S., Roof, B., Qin, B., Xu, S., Etzioni, O., et al.: Adapting open information extraction to domain-specific relations. AI Magazine 31(3), 93–102 (2010)

10. Soderland, S., Mandhani, B.: Moving from textual relations to ontologized relations. In: Proc. AAAI Spring Symposium on Machine Reading (2007)
11. Mintz, M., Bills, S., Snow, R., Jurafsky, D.: Distant supervision for relation extraction without labeled data. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP, vol. 2, pp. 1003–1011. Association for Computational Linguistics (2009)
12. Takamatsu, S., Sato, I., Nakagawa, H.: Reducing wrong labels in distant supervision for relation extraction. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers, vol. 1, pp. 721–729. Association for Computational Linguistics (2012)
13. Surdeanu, M., Tibshirani, J., Nallapati, R., Manning, C.D.: Multi-instance multi-label learning for relation extraction. In: EMNLP-CoNLL 2012, pp. 455–465 (2012)
14. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.G.: DBpedia: A nucleus for a web of open data. In: Aberer, K., et al. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007)
15. Kok, S., Domingos, P.: Extracting semantic networks from text via relational clustering. In: Daelemans, W., Goethals, B., Morik, K. (eds.) ECML PKDD 2008, Part I. LNCS (LNAI), vol. 5211, pp. 624–639. Springer, Heidelberg (2008)
16. Jimenez, S., Becerra, C., Gelbukh, A., Gonzalez, F.: Generalized mongue-elkan method for approximate text string comparison. In: Gelbukh, A. (ed.) CICLing 2009. LNCS, vol. 5449, pp. 559–570. Springer, Heidelberg (2009)
17. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions and reversals. Soviet Physics Doklady 10, 707 (1966)
18. Fellbaum, C.: Wordnet: an electronic lexical database (1998/2010) WordNet is available from, `http://www.cogsci.princeton.edu/wn`
19. Jiang, J.J., Conrath, D.W.: Semantic similarity based on corpus statistics and lexical taxonomy. arXiv preprint cmp-lg/9709008 (1997)
20. Lin, D.: An information-theoretic definition of similarity. In: Proceedings of the 15th International Conference on Machine Learning, San Francisco, vol. 1, pp. 296–304 (1998)
21. Milne, D.N., Witten, I.H.: Learning to link with wikipedia. In: CIKM 2008, pp. 509–518 (2008)
22. Strube, M., Ponzetto, S.P.: Wikirelate! computing semantic relatedness using wikipedia. In: AAAI 2006, p. 1 (2006)
23. Gabrilovich, E., Markovitch, S.: Computing semantic relatedness using wikipedia-based explicit semantic analysis. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence, vol. 6, p. 12 (2007)
24. Porter, M.F.: An algorithm for suffix stripping. Program: electronic library and information systems 14(3), 130–137 (1980)