Entity Translation with Collective Inference in Knowledge Graph

Qinglin Li[1], Shujie Liu[2(✉)], Rui Lin[3], Mu Li[2], and Ming Zhou[2]

[1] Shanghai Jiaotong University, Shanghai, China
v-lqing@microsoft.com
[2] Microsoft Research Asia, Beijing, China
{shujliu,muli,mingzhou}@microsoft.com
[3] Harbin Institute of Technology, Harbin, China
linrui@mtlab.hit.educ.n

Abstract. Nowadays knowledge base (KB) has been viewed as one of the important infrastructures for many web search applications and NLP tasks. However, in practice the availability of KB data varies from language to language, which greatly limits potential usage of knowledge base. In this paper, we propose a novel method to construct or enrich a knowledge base by entity translation with help of another KB but compiled in a different language. In our work, we concentrate on two key tasks: 1) collecting translation candidates with as good coverage as possible from various sources such as web or lexicon; 2) building an effective disambiguation algorithm based on collective inference approach over knowledge graph to find correct translation for entities in the source knowledge base. We conduct experiments on movie domain of our in-house knowledge base from English to Chinese, and the results show the proposed method can achieve very high translation precision compared with classical translation methods, and significantly increase the volume of Chinese knowledge base in this domain.

Keywords: Knowledge base · Machine translation · Collective learning

1 Introduction

Knowledge bases are structured databases that store facts concerning entities and relationships about the world, and they are widely used in NLP applications, such as question answering systems, search engines, and expert systems [12,3,2,14]. Knowledge base construction is attracting more and more attention from researchers in both academia and industry. Most knowledge bases are mainly compiled in one language, making it hard to adapt them in applications of other languages. For example, in freebase[1], there are 23M entities, but only 1% of them are Chinese entities, which limits the application of freebase for Chinese language processing. In order to extend the application of a knowledge

[1] http://www.freebase.com/

J. Li et al. (Eds.): NLPCC 2015, LNAI 9362, pp. 49–63, 2015.
DOI: 10.1007/978-3-319-25207-0_5

base, in this paper, we propose a method to translate a knowledge base from one language to another.

In contrast to conventional machine translation of sentences, knowledge base translation poses two particular challenges. For sentence translation, translation equivalence for phrases can be extracted from a bilingual corpus that is mined from the web. But for knowledge base translation, many entities are rare words/phrases (for example, *Hobbit* and *Philomena*, which are in movie names), and as such cannot be covered by this method. We collect translation candidates from multiple sources, including the output of machine translation engines, traditional dictionaries, translation candidates mined from web. All these results are collected as translation candidates and then ranked by our model to determine the best one.

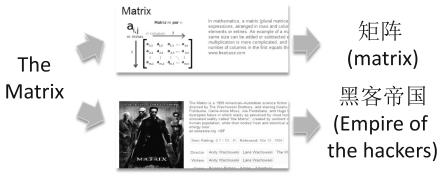

Fig. 1. Translation disambiguation for *The Matrix*. The mathematical entity *The Matrix* should be translated as 矩阵 (ju'zhen, meaning *matrix*), while the movie name *The Matrix* should be translated as 黑客帝国 (hei'ke'di'guo, meaning *Empire of the hackers*).

Given the translation candidates, to get the correct translation, we should do translation disambiguation. For example in Figure 1, the entity *The Matrix* can be translated into 矩阵 (ju'zhen, matrix) if it is a mathematical term, and 黑客帝国 (hei'ke'di'guo, Empire of the hackers) if it is a movie name. When the phrase *The Matrix* is surrounded by mathematical words in a sentence, conventional SMT models can select the correct translation 矩阵 (ju'zhen, matrix) by taking the surrounding words into consideration, or 黑客帝国 (hei'ke'di'guo, Empire of the hackers) if the sentence is about movies. Unfortunately, the entities are only words/phrases, which do not have such contextual information. In this paper, we propose taking the surrounding entities into consideration. The phrase *The Matrix* should be translated into 矩阵 (ju'zhen, matrix) if its neighbor nodes are mathematical entities, and 黑客帝国 (hei'ke'di'guo, Empire of the hackers) if its neighbors are movie names, movie actors, or directors, as shown in Figure 2.

Based on the above discussion, we propose a graph-based collective inference method for knowledge base translation. We first collect all the translation candidates with as good converage as possible from various sources, and calculate all the features for each translation candidate to run collective inference method to get the best configuration of the graph. We conducted experiments on a knowledge base translation task in the movie domain from English to Chinese, and the results indicate that our proposed method can significantly improve performance compared over baseline systems.

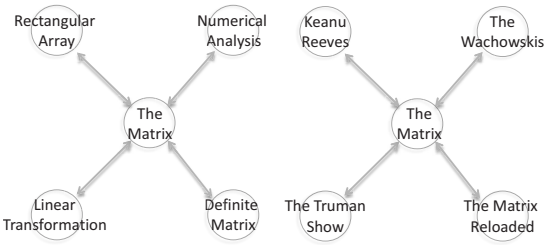

Fig. 2. Illustration of two *The Matrix* entities in the knowledge graph.

We will first introduce related work in section 2, and then we will illustrate our model in section 3, including the multi-source candidate generation, collective inference, model features, and model training. The experiments are explained in detail in section 4, followed by the conclusion and future work.

2 Related Work

[7] proposed an integrated method approach to extract an entity translation dictionary from a bilingual corpus. They first extract entities from the bilingual corpus independently for each language, and then a statistical model is used to align the entities. An iterative process is applied to extract entity pairs with high alignment probability and the entity pairs are used to improve the entity annotation quality for both languages. [8] proposed a method to improve entity translation by combining a transliteration approach with web mining, using web information as a source to complement transliteration and using transliteration to guide and enhance web mining. [1] proposed a two-step method to translate entities: first they generate a ranked list of translation candidates using bilingual and monolingual resources, and then all the candidates are rescored using monolingual clues. All the above methods treat the entities independently and try to build an entity translation table, instead of translate a knowledge base. The entity translation table can be used as a source of translation candidates collection of our method. [5] proposed a graph-based method to create a multilingual knowledge base by linking wordnet to wikipedia. Different from our method which translate a knowledge base from one language to another, they try to build a knowledge base based on a semi-structured data base, and merge entities expressed in different languages.

There are also many approaches that apply a graph-based collective method to different tasks. [6] apply a graph-based collective method to entity linking tasks, in which global interdependence between different entity linking decisions can be modeled and entities that are name mentions can be inferred jointly. [11] apply a graph-based collective method to POS tasks. They use a similarity graph to encourage similar n-grams to have similar POS tags. [10] apply a graph-based collective method to learn sentence translation consensus, in which each node is a phrase and similar phrases are connected with each other.

3 Our Method

In this section, we illustrate our model in detail. We first use a multi-source candidate generation method to generate all the translation candidates (shown in section 3.1), and then we conduct collective inference on a knowledge graph (in section 3.2 and 3.3) with confidence and consistency features (in section 3.4). We use a three-step method to train the parameters (shown in section 3.5).

3.1 Multi-Source Candidates Generation

Since the entity names in the knowledge base often contain new and rare words that are hard to find in a translation dictionary, we generate our translation candidates from multiple sources.

- Translation dictionary from **semi-structured web sites** containing movie information. Certain web sites collect movie information including movie names (some of them are in both languages), actors, and directors. We mined translation pairs from *BaiduBaike*[2], *Wikipedia*[3], *VeryCD*[4], *Douban*[5], and *IMDBCN*[6].
- **Traditional Dictionaries**. We collect translation candidates from the two dictionaries: Oxford dictionaries[7] and Longman contemporary English dicitonaries[8].
- We mined **parenthetical translation pairs** as another source, following [9].
- Translation results of **MT engine**. We fed the entity name into a state-of-the-art machine translation engine to get the translation result. It may not contain certain new words, but it's still useful for the coverage of our system.

3.2 Collective Inference with Neighbors

We find the correct translation results from the translation candidates of each entity by collective inference from any neighbors.

The basic principle of our model is that the translation of entities should be consistent with each other, which means, if there is a relationship between entity e_1 and e_2 in the knowledge base, there should also be the same kind of relationship between the translations of e_1 and e_2 . For example, the director of *The Matrix* in the knowledge base should be translated into the director of 黑客帝国 (hei'ke'di'guo, Empire of the hackers). And the entity of the mathematical term *The Matrix* should be translated into 矩阵 (ju'zhen, matrix), instead of 黑客帝国 (hei'ke'di'guo, Empire of the hackers), since all its neighbors are mathematics terms.

[2] http://baike.baidu.com/
[3] http://baike.baidu.com/
[4] http://www.verycd.com/
[5] http://www.douban.com/
[6] http://www.imdb.cn/
[7] http://oxforddictionaries.com/
[8] http://www.ldoceonline.com/

Let E be the set of entities in a knowledge base. For $\forall e \in E$, let $T(e)$ be the translation candidates of e. Based on the translation consistency principle, we propose a collective inference model, as shown in Equation 1.

$$\max_{t_i \in T(e_i)} \left(\sum_{e_i \in E} g(t_i, e_i) + \sum_{e_i, e_j \in E} f(t_i, t_j, e_i, e_j) \right) \tag{1}$$

There are two parts in our model. The first part is $g(t_i, e_i)$, which is called translation confidence only based on the entity itself, without considering its neighbors. The translation confidence $g(t_i, e_i)$ is a linear combination of several confidence features:

$$g(t_i, e_i) = \sum_k \theta_k g_k(t_i, e_i) \tag{2}$$

where $g_i(t_i, e_i)$ is the confidence feature and θ_i is the corresponding feature weight.

The second part is $f(t_i, t_j, e_i, e_j)$, called translation consistency, which is the measure of translation consistency between two entities connected with a link in the knowledge graph. Translation consistency $f(t_i, t_j, e_i, e_j)$ is defined as:

$$f(t_i, t_j, e_i, e_j) = \sum_{k'} \theta_{k'} f_{k'}(t_i, t_j, e_i, e_j) \tag{3}$$

where $f_{k'}(t_i, t_j, e_i, e_j)$ is the consistency feature and $\theta_{k'}$ is the corresponding feature weight.

The first part plays a role in the translation model while the second part plays the role the same as translation model for conventional SMT. In the the second part, translation consistency features can be treated as a special bi-gram language model in a graph, playing a similar role to n-gram language models used for sentence translation.

3.3 Translation Graph

To solve the collective inference problem, we have built a translation graph based on the English knowledge base. Each node in this graph corresponds to an entity in the knowledge base. And we also create an edge between a pair of nodes in the graph if there is a relationship between the corresponding entities in the knowledge base.

For each node in the graph, we use our multi-source candidate generation method to mine the translation candidates. We add a factor node g to each node to model the translation confidence, and a factor node f to each edge to model the translation consistency. An example of a translation graph is illustrated by Figure 3.

In Figure 3, the movie entity *The Matrix* is linked with four entities, including the actor *Keanu Reeves*, the directors *The Wachowskis*, and another two films, *The Truman Show* and *The Matrix Reloaded*. For each entity node, we collect

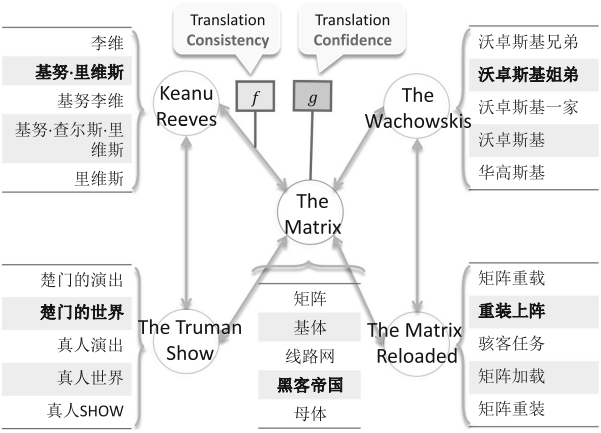

Fig. 3. An example of a translation graph.

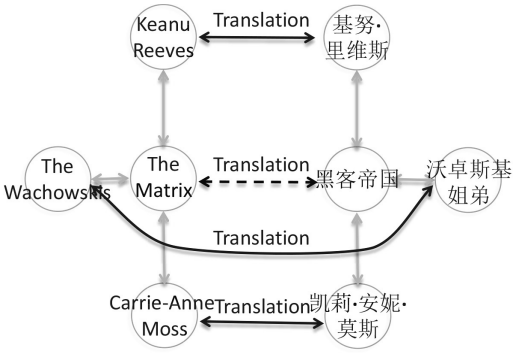

Fig. 4. An example of the confidence propagation in the translation graph.

the translation candidates, which are listed in the frame attached to the node, and the candidate in bold is the correct translation of the entity. To rank the translation candidates of the node *The Matrix* , we calculate the translation confidence features g, which are not related with the neighbors, and also the translation consistency features f, which are based on the best translations of the neighbor nodes. Then we use the trained feature weights to rank the list and get the best one as the result.

We use label propagation for inference. When feature weights are trained, starting from anchor nodes (whose translations are fixed and correct), label propagation can propagate the translation confidence from the anchor nodes to its neighbors and to the neighbors of the neighbors. For example, in Figure 4, the translation confidence of the directors *The Wachowskis* can be propagated to the movie *The Matrix*. And also, the translation confidence of the directors can be propagated to the actor nodes *Carrie-Anne Moss* and *Keanu Reeves*,

and then propagated to the node *The Matrix*. Our model will consider all the confidences received to rank the candidate list and get the final translation result. The propagation is an iteration process, and will be terminated when a pre-set limit is reached.

3.4 Features

Translation confidence features, which are defined in the g node, are listed as following:

- **Source** refers to where the candidate is from, e.g. a dictionary or SMT.
- **Bilingual Concurrence** is the concurrence ratios ($p_{bicol}(e|t)$ and $p_{bicol}(t|e)$) of the entity name and the translation candidates, which is calculated from a large corpus. $p_{bicol}(e|t)$ is defined as:

$$p_{bicol}(e|t) = \frac{\#(e,t)}{\sum_{e'} \#(e',t)} \tag{4}$$

  where e is the English entity name and t is the translation candidate, and $\#(e,t)$ is the count of the pair (e,t) shown in the corpus. $p_{bicol}(t|e)$ is calculated in a similar way. The corpus we used to calculate the pair numbers is a corpus mined from the web using the indicator of brackets [9], which contains sentences like:
  The Matrix *母体为什么不能用核能发 电?*
  *有关电影《**黑客帝国**》(**The Matrix**) 资深人士进*
  From these two sentences, we can find the move entity name and the translation.
- **Translation Score** is the word translation probabilities: $p_{smt}(e|t)$ and $p_{smt}(t|e)$. We use IBM-Model1[4] to calculate these scores. $p_{smt}(e|t)$ is defined as:

$$p_{smt}(e|t) = \frac{1}{(l_t+1)^{l_e}} \prod_{j=1}^{l_e} \sum_{i=0}^{l_t} t(e_j|t_i) \tag{5}$$

  l_e and l_t are the length of e and t. $t(e_j|t_i)$ is the word translation proability of word e_j given t_i, and this proability can be calculated with a big bilingual corpus. $p_{smt}(t|e)$ can be computed in a similiar way.

Relation consistency is the first translation consistency feature we used for f. This means $\forall e_i, e_j$ with relationship $\mathcal{R}$, whether the translation t_i, t_j has the same relationship $\mathcal{R}$. Figure 5 shows the consistency of node *The Matrix* and its neighbors. If we know that the three neighbors *The Wachowskis*, *Keanu Reeves* and *Carrie-Anne Moss* have corresponding translations 沃卓斯基姐弟, 基努?里维斯 and 凯莉?安妮?莫斯 as shown in Figure 5, and also, from semi-structured data, we can have the actor-movie and director-movie relationships similiar as in the English knowledge base, we will have more confidence that the movie name in the same position, which is 黑客帝国 (hei'ke'di'guo,

Empire of the hackers) , could be the translation of the movie node *The Matrix*. The relation consistency feature is defined as:

$$f_{kb}(t_i, t_j, e_i, e_j) = \begin{cases} 1, & \text{if} \quad t_i \in \mathcal{R}(t_j) \\ & \text{and} \quad e_i \in \mathcal{R}(e_j) \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

where $\mathcal{R}(t_j)$ is the set of neighbors for node t_j with relation $\mathcal{R}$. If t_i is the translation of e_i, and e_j is the neighbor of e_i with relation $\mathcal{R}$, while t_j is the neibhor of t_i with relation $\mathcal{R}$, then the feature for the translation t_j of the node e_j is set to 1, otherwise, the feature value is 0.

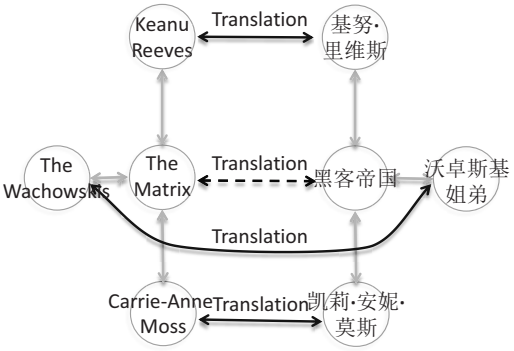

Fig. 5. Relation consistency of *The Matrix*.

We can also determine consistency features such as phrase concurrence using web search. For example, if we feed the query 凯莉 基努 黑客帝国 (黑客帝国 is the correct translation for the movie entity *The Matrix*) into a search engine, we will get about 108,000 results, but if the query is 凯莉 基努 矩阵 (矩阵 is not the correct translation), we will get only about 4,630 results. Which means, the correct translation of an entity is likely to be shown with the correct translation of its neighbors. One problem with using the search engine to calculate this feature is that the time cost is very high, so instead of using the search engine, we calculate this feature with a large monolingua corpus and the feature is called **cooccurence consistency**. Cooccurence consistency features are calculated in a smilar way to the bilingual cooccurence features.

3.5 Training

To train our model, we adopt a three-step training method, which is comprised of the pre-training for confidence feature weights, pre-training for consistency feature weights, and joint training of total feature weights.

In the pre-training phase for confidence feature weights, our model only uses the confidence features with the consistency features removed. In this case, each

Algorithm 1. Modified SampleRank

Input: q: $y \rightarrow y$: MCMC transition kernel;
w:$y \rightarrow R$: performance metric;
D_a: the anchor set;
D_t: the training set;
Output: $\frac{1}{T}\sum_{t=1}^{T}\theta_t$

Initialization: $\theta_0 \leftarrow \theta_{init}$; $t = 0$;
while *pre-set limitation not reached* **do**
  t++; $D_q \leftarrow D_a$; $\theta_t = \theta_{t-1}$;
  y_t: initial configuration in $y(x)$ with θ_t;
  while D_q *is not empty* **do**
    $e \leftarrow \text{pop}(D_q)$;
    for $e' \in \mathcal{N}(e)$ *and* $e' \notin D_q$ **do**
      Attemp an update for e' : $y_{e'} \leftarrow q(\cdot|y_{e'})$
      if $e' \in D_t$ **then**
        $y^+ = argmax_{y \in y_t, y_{t+1}} w(y)$
        $y^- = argmin_{y \in y_t, y_{t+1}} w(y)$
        $\nabla = \phi(y^+) - \phi(y^-)$
        if $\theta_t \nabla < w(y^+) - w(y^-)$ *and* $w(y_t) \neq w(y_{t+1})$ **then**
          $\theta_t = \theta_t + \eta_t \nabla$
        end
      end
      push(D_q, e');
    end
  end
end

entity is independent from others and the feature weights can be trained using linear model methods, such as perception, logistic regression, or SVM. We take the pairs of entities with the correct translations as positive samples and the pairs of entities with other translation candidates as negative samples. The trained confidence feature weights will be used to pre-train the consistency feature weights.

During the pre-training of consistency feature weights, we fix the confidence feature weights and use a graph-based training method to tune the consistency feature weights. Since the confidence feature weights are fixed, it is faster and easier to tune the model. Based on the pre-training of translation confidence and consistency feature weights, we use the same graph-based training method to tune the confidence and consistency feature weights jointly.

To train our model, we adopt sample rank [13] for the pre-training of consistency features and the joint training of all the features. Sample rank is a rank-based learning method that uses MCMC inference to train factor graphs with an atomic gradient. The following is the adopted SampleRank alogrithm shown in Algorithm1. As shown, we first initialize the feature weight vector θ as θ_{init}. For the pre-training of consistency feature weights, the confidence feature

weights of θ_{init} are set to be the results of the pre-training of the confidence feature weights, and they are fixed during the SampleRank training. For all feature weight training, confidence and consistency feature weights are set as the results of the two pre-training phases.

In addition to the initialization, another difference with traditional SampleRank is the order of the samples for training. Since we have a large number of *Anchor Nodes* (whose translation are given, will be introduced in Section 4), we modify the traditional SampleRank algorithm to leverage them. From the *Anchor Nodes* (in line 6), the translation confidence is propagated to the neighbors, and the translation of neighbor nodes are updated. If the neighbor node is a training node, it is used to update the model. The translation confidence of *Anchor Nodes* will be propagated to other nodes in the graph via the neighbors, until we update the whole graph. The training iteration will continue until a pre-set limitation is reached.

4 Experiments

4.1 Dataset

We construct a movie domain dataset by first gathering $3.2M$ English entities from the movie domain of an inhouse knowledge base. Each entity consists of a list of names in English, and their relationships with other entities. Some of the entities have IMDB IDs, making it possible to match the English entities with Chinese translations. This creat $124,908$ *golden-match* pairs of English and Chinese translation pairs, which will be used as **Anchor Nodes** We build our translation graph using these 3.2M entities and the relationships between them. We sample 2,000 english movie entities and manually search their Chinese translations using web search engines. These 2,000 pairs are split into two groups: **Training**, and **Test**. The **Training** nodes are used to run SampleRank for feature weights tuning. Model performance is evaluated with the **Test** nodes. The data statistics is shown in Table 1.

Table 1. Data statistics.

#Nodes	#Anchors	#Training	#Test
3.2M	125K	1K	1K

Our monolingual cooccurence data is collected from movie web sites, not only the pages for movies, but also the forums. Such web pages contain information about movies and actors. We extract sentences from these pages and indexed them to speed up the feature calculations. Our monolingual cooccurence data contains 7.6M sentences and 114M words.

Our bilingual cooccurence data is actually the byproduct of our implementation of [9]. We mine the sentences containing brackets from movie web sites.

Table 2. Performance comparison.

	Features	SMT	Perceptron	LR	LK_SVM	Our Method
	All	48.1%	87.1%	87.8%	87.9%	92.8%
g	-Source	-	83.0%	84.8%	84.7%	88.1%
	-TransProb	-	83.8%	85.9%	85.9%	90.3%
	-Category	-	84.4%	86.6%	85.7%	91.5%
	-BilColoc	-	83.2%	84.7%	84.1%	89.2%
f	-Consist$_{Col}$	-	-	-	-	88.7%
	-Consist$_{Rel}$	-	-	-	-	88.6%

The subsequent process of [9] may introduce errors and discard useful translation pairs. We use the unprocessed data to calculate the bilingual cooccurence features. The sentences are also indexed to speed up the feature calculation. Our bilingual cooccurence data contains 456K sentences and 5M words.

4.2 Baselines

We compare our method with several baselines. The first is the output of machine translation. The machine translation engine we used is an inhouse implemented state-of-the-art SMT system. We also use three classification methods as baselines: Perceptron, Linear Regression (LR) and Support Vector Machine (SVM). To train these three baselines, we generate the candidates and calculate the translation confidence features. From anchor and training nodes, the pairs of correct translations and the entity names are used as positive samples and the pairs of incorrect translation candidates and the entity names as negative samples to train the classifiers.

4.3 Translation Result

The performance of our method and the baseline systems are shown in Table 2. It is not surprising that the performance of machine translation was only 48.1%, since many movie names are not translated literally. For example, the Chinese name for the movie *The Matrix* is 黑客帝国 (hei'ke'di'guo, Empire of the hackers). And also, without the context information, SMT cannot do translation disambiguation. The performance of the linear classifiers are similar, with a variance of less than 1.0%. Among them, Perceptron is the worst and the Linear_Kernel SVM (LK_SVM) was the best. We also try the radial basis function (RBF) kernel and the performance is even worse than LK_SVM.

Our method can outperform the LK_SVM (the best of the baselines) by over 4.9% on accuracy. For our SampleRank training, we use LK_SVM as the training method for the pre-training of confidence feature weights, which is the initial feature weights of pre-training of consistency feature weights. Our model can

then leverage the results of LK_SVM and can also leverage the links of entities and learn the translation consensus.

All the baseline systems, SMT, Perceptron, LR and LK_SVM cannot use the information of the neighbors, since the translation of the neighbors are not given, so that, the baseline systems have no information to learn the tanslation consistency of the entities in the graph. Our proposed method can model the translation consistency using two consistency features, relation consistency and cooccurence consistency, and try to find the most consistent translation for a entity by taking its neighbors' translation into consideration.

Table 3. Performance for the popular movies.

	Top500	**Top5000**
Precision	97.0%	95.2%
Recall	98.5%	96.3%

Our method can generate about 483K movie translation for 3.2M movie names, in another word, the translation rate of our method is about 15.1%. We check the entities which cannot find the translation, most of them are very old movies, which are not well known for Chinese people, and donnot have the Chinese translation, for example, a movie *The Legacy* taken in 1978. Some of them are not very famous, for example, a movie named *Quantized Love* taken in 2014. It is not possible to chech whether there are translations for all of the 3.2M movies, so we randomly sample 400 movie names and manually search the web to find the Chinese translation. Out of these 400 movies, at most 85 Chinese translations can be found by human labeler, which means by using existing resources, the upper bound for translation rate is about 21%, so that the translation recall of our method is about $\frac{15.1\%}{21\%} = 72\%$. We also check the coverage for the most popular 500 and 5,000 movies listed in the web *http://www.imdb.com/*, and the result is shown in Table 3. As shown in Table 3, our method can achieve high precision and recall for the popular movies.

4.4 Training Phase Gain

To test the three step training method, we evaluate the performance after each of them and the results are shown in Table 6. inited by the results of LK_SVM (pre_confidence), the pre-training of consistency feature weights (pre_consistency) improves the performance about 3.7%, and the joint training of all the features (joint training) can further improve the performance about 1.2% to achieve a total improvement of over 4.9%.

4.5 Feature Gain

We remove the features one by one from the feature list to test the feature contribution to the performance, and the results are shown in Figure 2. For the confi-

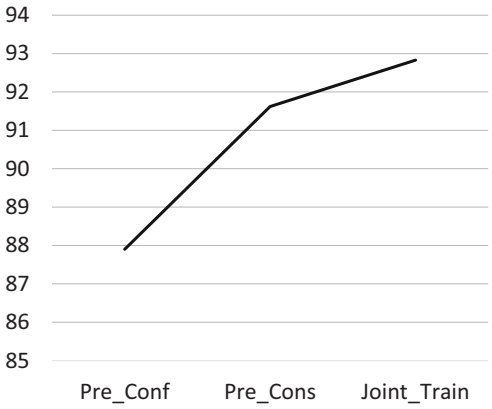

Fig. 6. Performance gain for training phases.

dence features, **Source** is the most important. If we remove the **Source** feature, all of the methods drop more than 3 points. Bilingual Cooccurence(BilColoc) is the second most important confidence feature. Compared with the confidence features, for our method, consistency features are also important. Both the cooccurence consistency feature (**Consist**$_{Col}$) and relation consistency feature (**Consist**$_{Rel}$) are important. Relation consistency feature is even stronger than the **Source** feature, which is the most important one of the confidence features.

4.6 Source Gain

We also remove the sources one by one from the candidate generation sources to test how the source contribution affects the performance. The results are shown in Table 4. From Table 4, we find that the semi-structured web sites (**-Website**) are the most important source. When they are removed from the source list, the performance drops significantly. The second most important one is parenthetical translation pairs (**-Parenth**) mined from web pages. Even the the translation results from SMT engines (**-MT**) contribute the least compared with the three other sources, when they are removed the performance drops more than 1.5 points.

Table 4. Translation candidates source contribution. **-Website** is the performance after removing the translation pairs of semi-structured web sites. **-Dict** is the results after removing the traditional dictionary. **-Parenth** was the results with the parenthetical translation pairs removed, and **-MT** shows the results after the SMT engine has been removed.

-Website	-Dict	-Parenth	-MT
81.6%	90.1%	89.7%	91.2%

5 Conclusion

In contrast to conventional sentence translation, knowledge base translation suffers from two main problems. The first is data sparsity. To handle this problem, we mine translation candidates from several different sources. The second problem is that for traditional sentence translation, we can use the surrounding words/phrases to select the best translation and do word/phrase disambiguation. For entities in a knowledge base, we donot have this information. In order to perform translation disambiguation, we have proposed a graph-based collective inference method to take the translation of the surrounding entities into consideration, and learn the translation consensus for all the entities. We conducte experiments on a movie domain knowledge base, and the results show that our method can improve the translation performance significantly, compared with several strong baselines.

References

1. Al-onaizan, Y., Knight, K.: Translating named entities using monolingual and bilingual resources. In: Proceedings of the 40th Meeting of the Association for Computational Linguistics (ACL02), pp. 400–408 (2002)
2. Bao, J., Duan, N., Zhou, M., Zhao, T.: Knowledge-based question answering as machine translation. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics. Long Papers, vol. 1, pp. 967–976. Association for Computational Linguistics, Baltimore, June 2014
3. Berant, J., Chou, A., Frostig, R., Liang, P.: Semantic parsing on Freebase from questionanswer pairs. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 1533–1544. Association for Computational Linguistics, Seattle, Washington, October 2013
4. Brown, P.F., Pietra, S.D., Pietra, V.J.D., Mercer, R.L.: The mathematics of statistical machine translation: Parameter estimation. Computational Linguistics **19**(2), 263–311 (1993)
5. de Melo, G., Weikum, G.: UWN: a large multilingual lexical knowledge base. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pp. 151–156. Association for Computational Linguistics, Stroudsburg (2012)
6. Han, X., Sun, L., Zhao, J.: Collective entity linking in web text: a graph-based method. In: Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '11, pp. 765–774. ACM, New York (2011)
7. Huang, F., Vogel, S.: Improved named entity translation and bilingual named entity extraction. In: Proceedings of Fourth IEEE International Conference on Multimodal Interfaces. Institute of Electrical & Electronics Engineers (IEEE) (2002)
8. Jiang, L., Zhou, M., Chien, L.F., Niu, C.: Named entity translation with Web mining and transliteration. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence, pp. 1629–1634 (2007)
9. Lin, D., Zhao, S., Van Durme, B., Paşca, M.: Mining parenthetical translations from the Web by word alignment. In: Proceedings of ACL-08: HLT, pp. 994–1002. Association for Computational Linguistics, Columbus, June 2008

10. Liu, S., Li, C.-H., Li, M., Zhou, M.: Learning translation consensus with structured label propagation. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics. Long Papers, vol. 1, pp. 302–310. Association for Computational Linguistics, Jeju Island, July 2012
11. Subramanya, A., Petrov, S., Pereira, F.: Efficient graph-based semi-supervised learning of structured tagging models. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pp. 167–176. Association for Computational Linguistics, Cambridge, October 2010
12. Talukdar, P.P., Pereira, F.: Experiments in graph-based semi-supervised learning methods for class-instance acquisition. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pp. 1473–1481. Association for Computational Linguistics, Uppsala, July 2010
13. Wick, M., Rohanimanesh, K., Culotta, A., Mccallum, A.: Samplerank: learning preference from atomic gradients. In: NIPS WS on Advances in Ranking (2009)
14. Yao, X., Van Durme, B.: Information extraction over structured data: question answering with Freebase. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics. Long Papers, vol. 1, pp. 956–966. Association for Computational Linguistics, Baltimore, June 2014