Linking Entities in Tweets to Wikipedia Knowledge Base

Xianqi Zou, Chengjie Sun, Yaming Sun, Bingquan Liu, and Lei Lin

School of Computer Science and Technology
Harbin Institute of Technology, China
{xqzou,cjsun,ymsun,liubq,linl}@insun.hit.edu.cn

Abstract.: Entity linking has received much more attention. The purpose of entity linking is to link the mentions in the text to the corresponding entities in the knowledge base. Most work of entity linking is aiming at long texts, such as BBS or blog. Microblog as a new kind of social platform, entity linking in which will face many problems. In this paper, we divide the entity linking task into two parts. The first part is entity candidates' generation and feature extraction. We use Wikipedia articles information to generate enough entity candidates, and as far as possible eliminate ambiguity candidates to get higher coverage and less quantity. In terms of feature, we adopt belief propagation, which is based on the topic distribution, to get global feature. The experiment results show that our method achieves better performance than that based on common links. When combining global features with local features, the performance will be obviously improved. The second part is entity candidates ranking. Traditional learning to rank methods have been widely used in entity linking task. However, entity linking does not consider the ranking order of non-target entities. Thus, we utilize a boosting algorithm of non-ranking method to predict the target entity, which leads to 77.48% accuracy.

Keywords: entity linking, global feature, topic distribution, boosting algorithm.

1 Introduction

Entity linking, a task to make the mentions certain in tweets, is to link mentions to the unambiguous knowledge base. Entity linking usually can be divided into two major steps: entity candidates' generation and ranking.

The first step is to generate candidate sets of entities. The entity candidates' generation process often involves query expansion. Mining the text of mentions and making full use of various resources will expand mentions to the entities that are literally close. Using search engine with the HTML marks could obtain huge sets of relevant candidates. The main drawback is that this approach requires to crawl a lot of documents from the search engine. For different search engines, we need to make different rules to expand the mentions. We utilize Wikipedia article pages as extended entity reference resources and fully mine the Wikipedia article's text. To a specified mention, the entity candidates will be quite large, and include many ambiguous candidates. Therefore, we design a reasonable classifier to reduce the number of entity candidates and increase the coverage of target entities.

C. Zong et al. (Eds.): NLPCC 2014, CCIS 496, pp. 368–378, 2014.

As the language used in tweet is not formal and the text is short, only limited features could be effectively extracted. For entity linking, features are usually represented in space vector. The element's value in the vector can be binary or TF-IDF, and this vector is very sparse. The local features mainly focus on the relationships between entities and mentions. In addition to the context semantic similarity of TF-IDF, there are anchor text prior probability, string edit distance between mention and entity's Wikipedia article title, the position of ranking returned by search engine, the length and page views of the candidates' Wikipedia articles. Global features are mainly considering the relationship among candidates. For different mentions from the same tweet, their target entities should have coordinate relationship. We use topic distribution instead of common links to get global feature during belief propagation. Topic distribution provides more information than common links in the perspective of semantics.

A problem often faced in information retrieval is to rank query's related documents, in the recommendation system is primarily to recommend interested items to users, however in entity linking is to rank the entity candidates and choose the candidate of the Top1. Ranking task usually adopts scoring mechanism. However, in entity linking, learning to rank method is often used to rank the candidates. Pairwise approach will transform the order of entity candidates into pairs, which the more related entity with the less one to be labeled as +1, otherwise -1. While the listwise approach makes use of the order sequence of entity candidates. In information retrieval, the number of training set is very large due to the fact that every pair of the candidates and every different order sequence could be made into a new training example. Since only predicting the Top1 candidate as the result of candidates' ranking without considering the other candidates' relative ranking and the mentions in entity linking have the similar feature distribution, we believe the model based on single candidate will produce a better result in entity linking task. We adopt Regularized Greedy Forest model [1], which is based on Gradient Boosting Decision Tree [2], to predict the target candidate. Our experiment results show that this method which doesn't rely on relative ranking features could produce a better performance.

2 Related Work

Entity candidates' generation directly affects the performance of the entity linking. Generating entity candidates usually adopts abbreviated words expansion, domain dictionary building. Chang [3] used online abbreviation dictionary to expand relatively formal medical abbreviations. Han [4] matched the words before the phrases, such as "also called", "known as", "short for". But these methods have not considered abbreviations for informal words. Nadeau [5] used supervised machine learning methods and taken advantage of part of speech information. However, this method could only be limited to the situation that abbreviation and the expansion of words are in the same sentence. When extended to the other documents, the performance would be seriously affected by the noise data. Zhang [6] also made use of supervised machine learning methods which relied not only on HTML mark and language clues, but also on semantic information.

Features measuring mention and entity's relevancy are referred to local features. Local features often used vector dot product, cosine similarity, K-L divergence, Jaccard distance, etc. Liu [7] utilized both local features and global features. Local features included edit distance similarity between mentions and entities, the probability of mention as anchor text to be appeared in the entity's Wikipedia article, etc. Global features described the relationship among the entities of different mentions in a tweet, such as the common link of entities' Wikipedia articles. Han [8] built a collective graph model to coordinate the relationship among entities of different mentions and correlate mentions with entities in Wikipedia knowledge base. Zheng [9] taken advantage of deep neural network to represent the mentions' text and entities' Wikipedia articles respectively.

Linking mentions into knowledge base is usually seen as a way to provide semantic information. The idea has been widely applied to various kinds of media, such as text, multimedia, news, radio reports, etc. Milene and Witten [10] used traditional classifiers such as Naïve Bayes, C4.5[11]. Meij adopted the Gradient Boosting Decision Tree model to predict the target entity. Another common method to link mentions to entities in knowledge base was to use learning to rank model. Zheng [12] utilized pairwise method (Ranking Perceptron [13]) and listwise method (ListNet [14]). In this paper, we make use of belief propagation process on the topic distribution to obtain global feature and analyze the validity of this feature and other local features. Then we compare the learning to rank method with Regularized Greedy Forest, an improved Gradient Boosting Decision Tree model.

3 Candidates Generation and Ranking

3.1 Candidates Generation

Entity candidates' generating process involves query expansion. Various resources could be made full use of to expand the mentions. We mine the Wikipedia pages and extract redirect page title, disambiguation page title, bold words in first paragraph,

Table 1. Feature set for filtering ambiguous candidates

Feature	Feature Description
Cap_All	Whether mention is capitalized.
LCS_First	Edit distance between mention and first letter of entity word.
Length_3	Whether the length of mention is less than 3.
Parenthesis	Whether entity has parenthesis.
Match_All	Whether mention and entity's string match exactly.
Redirects	Whether entity is from redirect page.
Disam	Whether entity is from disambiguation page.
Anchor	Whether entity is anchor text.
Bold	Whether entity is bold.

anchor texts of the mention's candidates. The set of entity candidates is quite large and contains many ambiguous candidates. Thus we train SVM model to filter parts of the candidates to ensure the quality of the entity candidates. Table 1 shows the features we used to train our model.

3.2 Candidates Ranking

Features for Linking

In this part, we will introduce the features used for candidates ranking. The features include local features and global features. First we define the symbol used in describing features as showed in Table 2.

Table 2. Symbol used to describe feature

Symbol	Description
M	Mention.
E	Entity candidate.
E_i	Entity candidate of the ith mention.

- f_1 is the prior probability which is the quantitative proportion of M to be presented as anchor text in M s candidates' Wikipedia article.
- f_2 is edit distance similarity between E and M .
- f_3, f_4 are binary values. The feature value will be 1, if M contains E, otherwise 0.
- f_5 is a feature related with ranking position of Wikipedia search result. If the position is in the first place, the value will be 1, otherwise 0.
- f_6 is the TF-IDF similarity between M 's tweet and E 's Wikipedia article.
- f_7, f_8 are E 's Wikipedia article's length and page view counts respectively.
- f_9, f_{10} are links similarity and topic distribution similarity between E_i and E_j of different mentions in a tweet.

Wikipedia articles contain a large number of anchor texts. These text information together with the non-text information, such as text length and page view counts, can be viewed as the prior knowledge. Feature f_9, f_{10} are global features which describe the coordination degree between two entities of different mentions while the others are local features which only consider the relationship between mention and its candidate entity.

If the entity candidates of different mentions are related to some extent, their Wikipedia articles will also have common inner links. The more the same common inner links they have, the more similar their contents are. The candidates of different mentions in a tweet also have semantic relationships to some degree. The same topic distribution should be followed by these entity candidates. We take advantage of belief

propagation on the candidates' common links and topic distribution to obtain the coordination degree between mentions and their candidates. The belief propagation is showed as:

$$V_k = (1-\lambda) * B * V_{k-1} + \lambda * V_0 \tag{1}$$

In the belief propagation, we use V to represent the belief vector. The vector has n_1 elements related with mentions and n_2 elements with candidates. B is the matrix of belief with $(n_1 + n_2)^2$ elements. $B[i, j]$ is the belief that element j propagates to element i. If element j is an entity and element i is a mention, the value of $B[i, j]$ is showed in formula 2. If element j is an entity and element i is also an entity, $B[i, j]$ will use formula 3 as its value. Otherwise, $B[i, j]$ will be 0. We initial the belief vector $V_0[1 : n_1]$ with formula 2. $V_0[n_1 : n_1 + n_2]$ is set to be 0. λ is the tradeoff between original and reallocated belief. After the K times of iterations in the belief propagation, we will get the final belief vector V_k.

$$P(E \mid M) = \frac{TFIDF(M, E)}{\sum_{E \in C_M} TFIDF(M, E)} \tag{2}$$

$$P(E_j \mid E_i) = \frac{S(E_i, E_j)}{\sum_{E \in N_{E_i}} S(E_i, E)} \tag{3}$$

Where $S(E_i, E_j)$ could make use of various similarity measurement between two candidates of different mentions in a tweet. The similarity between two candidates which is based on common links could be obtained as:

$$LS(E_i, E_j) = 1 - \frac{log(max(|A|, |B|)) - log(|A \cap B|)}{log(|W|) - log(min(|A|, |B|))} \tag{4}$$

Where W is the counts of Wikipedia articles and A, B are the sets of entities which link to entity E_i, E_j respectively. In this paper, we make use of topic distribution similarity to compute $S(E_i, E_j)$.

Methods for Linking

Ranking problem in entity linking and information retrieval have the same substance. Thus the method of learning to rank is often used to rank the entity candidates from pairwise approach(RankSVM [15], RankBoost [16], RankNet [17]) to listwise approach(AdaRank [18], ListNet). We take advantage of Regularized Greedy Forest. According to Friedman, when setting the shrinkage parameter small enough, the gradient boosting could achieve good performance. The Fully-Corrective Gradient Boosting is to avoid the small step size problem as showed in Algorithm 1.

Algorithm 1. Fully-Corrective Gradient Boosting

$h_0(\mathrm{x}) \leftarrow \arg\min_{\rho} L(\rho, \mathrm{Y})$

`for k = 1 to K do`

$\tilde{Y}_k \leftarrow -\left[\frac{\partial L(h, \mathrm{Y})}{\partial h}\right]_{h=h_{k-1}(\mathrm{X})}$

$b_k \leftarrow \mathrm{A}(\mathrm{X}, \tilde{\mathrm{Y}}_k)$

`let` $H_k = \left\{\sum_{j=1}^{k} \beta_j b_j(x) : \beta_j \in \mathbb{R}\right\}$

$h_k \leftarrow \arg\min_{h \in H_k} L(h(\mathrm{X}), \mathrm{Y})$

`end`

`return` $h(x) = h_K(x)$

Regularized Greedy Forest uses the Fully-Corrective Gradient Boosting to learn a decision forest. At the same time, all the parameters of the trained decision trees would be adjusted during each model training time, such as the number of decision trees, leaf nodes. The sparse combination of decision rules is completed by the greedy search. After obtaining a large number of decision trees, RGF will regularized the model appropriately.

4 Experiments

4.1 Experiment Setting

We adopt the Yahoo scientist Meij's annotation data set. We divide 502 tweets into 5:1 as the training set and testing set. For the text information and non-text information related with entities, we make use of the Wikipedia API to get them. The metrics to measure the experiment's results are

$$\text{Accuracy} = \frac{|\{C_{i,0} \mid C_{i,0} = \varsigma_i\}|}{N} \tag{5}$$

$$Coverage = \frac{|\{\varsigma_i \mid \varsigma_i \in C_i\}|}{N} \tag{6}$$

Where C_i is the candidate set of mention i, $C_{i,0}$ is the candidate which ranks at the first position for mention i. ς_i is the gold standard annotation for mention i. N is the number of mentions in the data set.

Firstly, we make an experiment to decide the size of candidates set which to ensure a higher coverage of the gold candidate. Secondly, we verify the effectiveness of local features and global features respectively. In the end, we compare the learning to rank methods with Regularized Greedy Forest.

4.2 Results and Analysis

We train a SVM classifier with all the features described in Table 1 to make a binary classification of all entity candidate. For the positive candidates our model predicts, we utilize the edit distance between mention and the entity candidate to choose the Top K of them. Different value of *K* will effects the coverage of target candidate. As is shown in Fig. 1, if *K* is 1, the coverage of target entity will be 64%. When we set *K* to be 20, the target entity coverage no longer grows. Therefore, we make use of the Top 20 candidates of the SVM classifier result as the candidates.

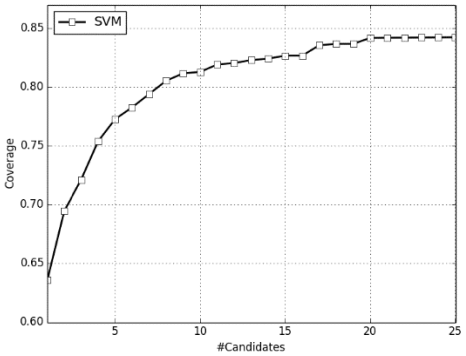

Fig. 1. Coverage of different size of candidates set

We extract local features, global features and utilize a basic model (RankSVM) to analyze the validity of them. Local features contain text information and non-text information. Table 3 shows that feature f_5 is the most effective among these local features. Since feature f_5 is associated with the Wikipedia search results, it has the effect of reducing ambiguity partly. Feature f_2, f_3, f_4 are mainly considering the literal relationship between mention and candidate. Compared with feature f_5, their accuracy is relatively lower. Feature f_8, as a prior knowledge, achieves 62.16% accuracy.

Table 3. Local Feature Analysis

Local Feature	Accuracy(%)
f_1	17.12
f_2	59.46
f_3	64.86
f_4	54.05
f_5	65.77
f_6	33.33
f_7	9.91
f_8	62.16

For global features, feature f_{10} is more effective than feature f_9 as showed in Table 4, since feature f_{10} could provide more latent semantic information than feature f_9. When local features combined with global features, the accuracy has reached to 72.97% in Table 5. The convergence rate of the belief propagation progress based on different entity similarity is also different. Fig. 2 shows that the convergence rate on feature f_{10} is faster to feature f_9. The entities that have the same inner links may be about different topics in the Wikipedia page. Since feature f_9 has more noise information, its belief propagation iterates more steps to get a balance state.

Table 4. Global Feature Analysis

Global Feature	Accuracy(%)
f_9	9.01
f_{10}	11.71

Table 5. Performance of Commbined Features

Local Feature + Global Feature	Accuracy(%)
$f_1+f_2+f_3+f_4+f_5+f_6+f_7+f_8$	70.27
f_9+f_{10}	13.51
$f_1+f_2+f_3+f_4+f_5+f_6+f_7+f_8+f_9+f_{10}$	72.97

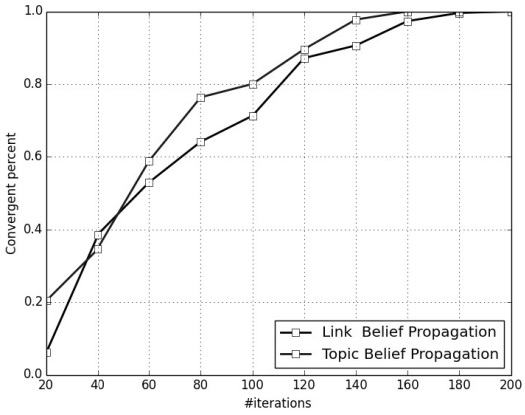

Fig. 2. Convergent Percent of Tweets during the Belief Propagation Based on Different Entity Similarity

Table 6 has shown that the performance of pairwise approach is better than listwise approach. As in entity linking task, the rank position of target entity is +1 while non-target entities are -1 equally. Therefore, the listwise approach relies more on single feature to make the probability of one ranking combination higher while others lower

and equal. The learning to ranking methods lose more information about non-target entities. Compared learning to rank methods with Regularized Greedy Forest, the later has achieved a higher accuracy with the accuracy of 77.48%. In tweets, the candidates' features distribution of different mentions is similar, while in information retrieval, the documents' feature distribution of different queries may vary a lot. Thus the RGF model, which trained on the original data, could achieve a better performance.

Table 6. Experiment results comparation of different methods

Model	Accuracy(%)
RankSVM	72.97
RankBoost	72.07
RankNet	65.77
AdaRank	67.57
ListNet	66.67
GBDT	74.77
RGF_L2	**77.48**

Fig. 3 shows the accuracy of different leaves during the training process of RGF model. Using L2 regularization with the leaves, RGF model could achieve the highest accuracy when the number of leaves is 1000.

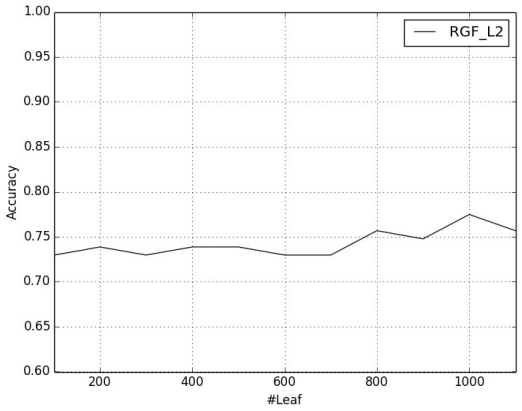

Fig. 3. The Accuracy of Different Leaves in RGF_L2

5 Conclusions

In this paper, we adopt the method of belief propagation based on the topic distribution to obtain global feature, which achieves better performance than that uses

common links. When combining global features with local features, performance will be obviously improved. We compared the traditional learning to rank methods with Regularized Greedy Forest. Experiment results show that the RGF model could achieve a better performance in the tweets' entity linking task.

Acknowledgment. This work is supported by the Key Basic Research Foundation of Shenzhen (JC201005260118A) and National Natural Science Foundation of China (61100094 & 61300114).The authors are grateful to the anonymous re-viewers for their constructive comments.

References

1. Johnson, R., Zhang, T.: Learning nonlinear functions using regularized greedy forest. arXiv preprint arXiv:1109.0887 (2011)
2. Friedman, J.H.: Greedy function approximation: A gradient boosting machine. Annals of Statistics, 1189–1232 (2001)
3. Chang, J.T., Schütze, H., Altman, R.B.: Creating an online dictionary of abbreviations from MEDLINE. Journal of the American Medical Informatics Association 9(6), 612–620 (2002)
4. Han, X., Zhao, J.: NLPR_KBP in TAC 2009 KBP Track: A Two-Stage Method to Entity Linking. In: Proceedings of Test Analysis Conference (TAC 2009) (2009)
5. Nadeau, D., Turney, P.: A supervised learning approach to acronym identification (2005)
6. Zhang, W., Sim, Y.C., Su, J., Tan, C.L.: Entity linking with effective acronym expansion, instance selection and topic modeling. In: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence, vol. 3, pp. 1909–1914 (2011)
7. Liu, X., Li, Y., Wu, H., Zhou, M., Wei, F., Lu, Y.: Entity linking for tweets. In: Proceedings of the 51th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics (2013)
8. 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, pp. 765–774 (2011)
9. He, Z., Liu, S., Li, M., Zhou, M., Zhang, L., Wang, H.: Learning entity representation for entity disambiguation. In: Proc. ACL 2013 (2013)
10. Milne, D., Witten, I.H.: Learning to link with wikipedia. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, pp. 509–518 (2008)
11. Quinlan, J.R.: C4. 5: programs for machine learning. Morgan Kaufmann (1993)
12. Zheng, Z., Li, F., Huang, M., Zhu, X.: Learning to link entities with knowledge base. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pp. 483–491 (2010)
13. Shen, L., Joshi, A.K.: Ranking and reranking with perceptron. Machine Learning 60(1-3), 73–96 (2005)
14. Cao, Z., Qin, T., Liu, T.-Y., Tsai, M.-F., Li, H.: Learning to rank: From pairwise approach to listwise approach. In: Proceedings of the 24th International Conference on Machine Learning, pp. 129–136 (2007)
15. Joachims, T.: Optimizing search engines using clickthrough data. In: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 133–142 (2002)

16. Freund, Y., Iyer, R., Schapire, R.E., Singer, Y.: An efficient boosting algorithm for combining preferences. The Journal of Machine Learning Research 4, 933–969 (2003)
17. Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., Hullender, G.: Learning to rank using gradient descent. In: Proceedings of the 22nd International Conference on Machine Learning, pp. 89–96 (2005)
18. Xu, J., Li, H.: Adarank: a boosting algorithm for information retrieval. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 391–398 (2007)