Entity Linking in Web Tables with Multiple Linked Knowledge Bases

Tianxing Wu, Shengjia Yan, Zhixin Piao, Liang Xu,
Ruiming Wang, Guilin Qi

School of Computer Science and Engineering, Southeast University, China
{wutianxing, sjyan, piaozhx, liang.xu, wangruiming, gqi}@seu.edu.cn

Abstract. The World-Wide Web contains a large scale of valuable relational data, which are embedded in HTML tables (i.e. Web tables). To extract machine-readable knowledge from Web tables, some work tries to annotate the contents of Web tables as RDF triples. One critical step of the annotation is entity linking (EL), which aims to map the string mentions in table cells to their referent entities in a knowledge base (KB). In this paper, we present a new approach for EL in Web tables. Different from previous work, the proposed approach replaces a single KB with multiple linked KBs as the sources of entities to improve the quality of EL. In our approach, we first apply a general graph-based algorithm to EL in Web tables with each single KB. Then, we leverage the existing and newly learned "`sameAs`" relations between the entities from different KBs to help improve the results of EL in the first step. We conduct experiments on the sampled Web tables with Zhishi.me, which consists of three linked encyclopedic KBs. The experimental results show that our approach outperforms the state-of-the-art table's EL methods in different evaluation metrics.

Keywords: Entity Linking, Web Tables, Linked Knowledge Bases

1 Introduction

The current World-Wide Web contains a large scale of relational data in the form of HTML tables (i.e. Web tables), which have already been viewed as an important kind of sources for knowledge extraction on the Web. To realize the vision of Semantic Web, various efforts [9,10,11,12,19,20] have been made to interpret the implicit semantics of Web tables by annotating their contents as RDF triples. One critical step of such annotation is entity linking (EL), which refers to map the string mentions in table cells to their referent entities in a given knowledge base (KB). For example, in the third column of the Web table in Figure 1, EL aims to link the string mention "`Michael Jordan`" to the entity "`Michael Jordan (American basketball player)`" in a given KB. Without correct identified entities, the annotation on Web tables is hard to get accurate RDF triples. Thus, in this paper, we focus on studying the problem of EL in Web tables.

Team	City	Owner	Arena
Los Angeles Clippers	Los Angeles	Steve Ballmer	Staples Center
Dallas Mavericks	Dallas	Mark Cuban	American Airlines Center
...	...	...	...
Charlotte Hornets	Charlotte	Michael Jordan	Time Warner Cable Arena
Chicago Bulls	Chicago	Jerry Reinsdorf	United Center

Fig. 1: An Example of Web Table Describing the Information of NBA Teams

There exist two main problems in previous work for EL in Web tables as follows. **1)** Many work [9,10,11,19,21,22] strongly relies on the features based on specific information, such as column headers (e.g. "`Team`", "`City`", etc. in the first row of Figure 1) in Web tables, entity types in the target KB, and so on. Therefore, it is obvious that these approaches can not work well when the given Web table or KB contains no or few such information. **2)** Most of the existing approaches [2,9,10,16,19,21,22] only consider linking string mentions in table cells to a single KB, which can not ensure good coverage of EL in Web tables. This problem is also presented in [15] when performing EL in natural language text.

To overcome the above problems, we propose a new general approach for EL in Web tables with multiple linked KBs. The proposed approach contains two steps. We first apply a graph-based algorithm without using any specific information to EL in Web tables with each single KB. Then, we present three heuristic rules leveraging the existing and newly learned "`sameAs`" relations between the entities from different KBs to improve the results of EL in the first step. The second step of our approach can not only reduce the errors generated by EL with each single KB, but also improve the coverage of the EL results. In experiments, we map the string mentions in the cells of sampled Web tables to the entities in Zhishi.me [14], which is the largest Chinese linked open data and composed of three Chinese linked online encyclopedic KBs: Chinese Wikipedia[1], Baidu Baike[2] and Hudong Baike[3]. The evaluation results show that our approach outperforms two state-of-the-art systems (i.e. TabEL [2] and LIEGE [16]) in terms of MRR (i.e. Mean Reciprocal Rank[4]), precision, recall and F1-score.

The rest of this paper is organized as follows. Section 2 outlines some related work. Section 3 introduces the proposed approach in detail. Section 4 presents the experimental results and finally Section 5 concludes this work and describes the future work.

[1] `https://zh.wikipedia.org`
[2] `http://baike.baidu.com`
[3] `http://www.baike.com`
[4] `https://en.wikipedia.org/wiki/Mean_reciprocal_rank`

2 Related Work

In this section, we review some related work regarding semantic annotation on Web tables, which usually tackles three tasks: entity linking (EL), column type inference and relation extraction between the entities in the same row but different columns. After Cafarella et al. [6] reported that there are more than 150 million Web tables embedded with high-quality relational data, lots of researchers realized that Web tables are important sources that can be used for many applications, such as information extraction and structured data search. Hence, there emerged various work about semantic annotation on Web tables.

Hignette et al. [9] proposed an aggregation approach to annotate the contents of Web tables using vocabularies in the given ontology. It first annotates cells, then columns, finally relations between those columns. Similarly, Syed et al. [19] also presented a pipeline approach, which first infers the types of columns, then links cell values to entities in the given KB, finally selects appropriate relations between columns. Zhang [22] designed a tool called TableMiner for annotating Web tables. TableMiner only focuses on column type inference and EL, and can not extract relations from Web tables. Afterwards, Zhang [21] also proposed some strategies to improve TableMiner. Limaye et al. [10] and Mulwad et al. [11] described two approaches which can respectively jointly model the EL, column type inference and relation extraction tasks for Web tables. The main difference between our approach and these work is that we do not use any specific information for the task of EL, such as column headers and captions of Web tables, entity types in KBs, semantic markups in Web pages, and so on.

There also exists some work in specific scenarios about semantic annotation on Web tables without the step of EL. In the work of Venetis et al. [20], their approach weakens the impacts of EL, and directly infers the types of columns and determines the relationships by the frequency of different patterns in large scale isA and relation databases, which are both built from Web pages but usually unavailable to most of the researchers. Besides, Muñoz et al. [12] proposed an approach to mine RDF triples from Wikipedia tables. In this work, they can directly identify the entities in Wikipedia with internal links and article titles.

The closest work to our approach is done by Shen et al. [16] and Bhagavatula et al. [2]. Shen et al. [16] tried to link the string mentions in list-like Web tables (multiple rows with one column) to the entities in a given KB. Bhagavatula et al. [2] presented TabEL, a table entity linking system, which uses a collective classification technique to collectively disambiguate all mentions in a given Web table. Both of these two work do not use any specific information for EL, and can be applied to any KB. Here, we focus on EL with multiple linked KBs instead of a single KB, in order to improve the quality of EL in Web tables.

3 Approach

In this section, we introduce our proposed approach for entity linking (EL) in Web tables, which consists of two main steps: EL with any single KB and improving EL using "`sameAs`" links between multiple linked KBs.

Team	City	Owner	Arena
Los Angeles Clippers	Los Angeles	Steve Ballmer	Staples Center
Dallas Mavericks	Dallas	Mark Cuban	American Airlines Center
...	...	...	...
Charlotte Hornets	Charlotte	Michael Jordan	Time Warner Cable Arena
Chicago Bulls	Chicago	Jerry Reinsdorf	United Center

related mentions for "Michael Jordan"

Fig. 2: An Example of Related Mentions in the Same Row or Column

3.1 Entity Linking with a Single KB

Candidate Generation For each string mention in table cells, we first need to identify its candidate referent entities in the given KB. Here, we segment each mention in word level, so each mention can be represented by a set of words. If an entity e in the given KB or one of e's synonyms in BabelNet [13] (a Web-scale multilingual synonym thesaurus) contains at least one word of some mention m, then e is taken as one candidate referent entity of the mention m. For example, the mention "`Charlotte`" has candidate referent entities such as "`Charlotte, North Carolina`", "`Charlotte, Illinois`" and "`Charlotte Hornets`". The results of candidate generation is that each mention may correspond to a set of candidate entities.

Entity Disambiguation In entity disambiguation, we aim to choose an entity from the candidate set as each mention's referent entity in the given KB. As shown in Figure 2, we can easily find that mentions in the same row or column tend to be related. In other words, there exists some potential association between any two mentions appearing in the same Web table. Therefore, we choose to jointly disambiguate all the mentions in one table using a graph-based algorithm:

a) Firstly, for each given table, we build an *Entity Disambiguation Graph* only using mentions and their candidate referent entities as the graph nodes.
b) Secondly, in each constructed *Entity Disambiguation Graph*, we compute the initial importance of each mention for joint disambiguation and the semantic relatedness between different nodes as the **EL impact factors** to decide whether an entity is the referent entity of a given mention.
c) Finally, with iterative probability propagation using the **EL impact factors** until convergence, each entity gets its probability to be the referent entity of the given mention and our algorithm makes the EL decisions based on these probabilities.

In the following part of this section, we describe the above three steps in detail.

a) Building *Entity Disambiguation Graph*. For each given table, we build an *Entity Disambiguation Graph*, which consists of two kinds of nodes and two kinds of edges introduced as follows.

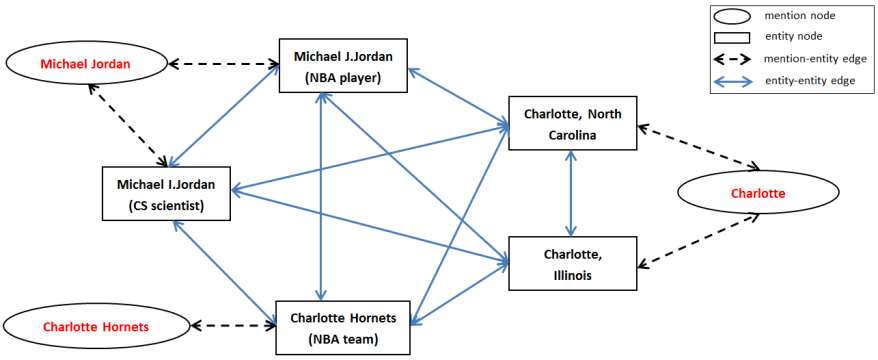

Fig. 3: An Example of Constructed *Entity Disambiguation Graph*

- **Mention Node**: These nodes refer to the mentions in Web tables.
- **Entity Node**: These nodes represent mentions' candidate referent entities in the given KB.
- **Mention-Entity Edge**: A mention-entity edge is an undirected edge between a mention and one of its candidate referent entities.
- **Entity-Entity Edge**: An entity-entity edge is an undirected edge between entities.

An example of the constructed *Entity Disambiguation Graph* is given in Figure 3. Due to the limited space, it is only the part of the constructed *Entity Disambiguation Graph* for the Web table in Figure 2, and lots of nodes and edges are not shown in Figure 3. Note that each mention such as "`Michael Jordan`" should has a mention-entity edge linking to any of its candidate entities "`Michael J.Jordan (NBA player)`" and "`Michael I.Jordan (CS scientist)`". Entity-entity edges should also be created between all the entity nodes in the graph.

b) Computing the EL Impact Factors. After constructing the *Entity Disambiguation Graph* for the given Web table, each node or edge is assigned with a probability. For the entity nodes, their probabilities refer to the possibilities of they being the referent entities of mentions, and are initialized as 0 before affected by the **EL impact factors**, which are actually 1) the probabilities of mention nodes, and they can be viewed as the importance of mentions for joint disambiguation; 2) the probabilities assigned to edges, and they are semantic relatedness between nodes. In this paper, we equally treat each mention, so when there exist k mentions in the Web table, the importance of each mention is initialized to $1/k$. Since there are two kinds of edges in each constructed *Entity Disambiguation Graph*, entity-entity edges and mention-entity edges should be respectively associated with the semantic relatedness between entities and that between mentions and entities.

For the semantic relatedness between mentions and entities, we use two features to measure it as follows.

- **String Similarity Feature**. If a mention m and an entity e are of similar strings, it is possible that e is m's the referent entity in the given KB. Hence, we define the string similarity feature $strSim(m, e)$ as

$$strSim(m, e) = 1 - \frac{EditDistance(m, e)}{\max\{|m|, |e|\}} \tag{1}$$

  where $|m|$ and $|e|$ are the string lengths of the mention m and entity e, respectively. $EditDistance(m, e)$ means the edit distance[5] between m and e, and it is a way to quantify how dissimilar two strings are. In other words, the more similar in string level mention m and entity e are, the higher the value of $strSim(m, e)$ is.
- **Mention-Entity Context Similarity Feature**. Given a mention m and one of its candidate referent entities e, if they are semantic related, they tend to share similar context. Here, for obtaining the context of the given mention m, we first collect other mentions in the row or column where m locates. Then, we segment each collected mention into a set of words. Finally, we take all the words as the context of m and it is denoted by $menContext(m)$. For the context of the entity e, we first collect all the RDF triples which e exists in, and then segment each object (when e is the subject) or each subject (when e is the object) into a set of words. These words are also treated as e's context $entContext(e)$. To calculate the *mention-entity context similarity feature* $contSim_{me}(m, e)$ between the mention m and the entity e, we apply the Jaccard Similarity[6] as follows:

$$contSim_{me}(m, e) = \frac{|menContext(m) \cap entContext(e)|}{|menContext(m) \cup entContext(e)|} \tag{2}$$

Given a mention m and an entity e, to integrate the *string similarity feature* $strSim(m, e)$ with the *mention-entity context similarity feature* $contSim_{me}(m, e)$, we define the **Mention-Entity Semantic Relatedness** $SR_{me}(m, e)$ as follows:

$$SR_{me}(m, e) = 0.99 \times (\alpha_1 \cdot strSim(m, e) + \beta_1 \cdot contSim_{me}(m, e)) + 0.01 \tag{3}$$

where both α_1 and β_1 are set to 0.5 in this work. $SR_{me}(m, e)$ at least equals 0.01, in order to keep the connectivity of the *Entity Disambiguation Graph* during the subsequent process of probability propagation.

For the semantic relatedness between entities, we also define following two features to measure it.

- **Triple Relation Feature**. If two entities are in the same RDF triple, they are obviously semantic related. Thus, we compute the *triple relation feature* $IsRDF(e_1, e_2)$ between the entity e_1 and the entity e_2 as

$$IsRDF(e_1, e_2) = \begin{cases} 1, & e_1 \textit{ and } e_2 \textit{ are in the same RDF triple} \\ 0, & \textit{otherwise} \end{cases} \tag{4}$$

[5] https://en.wikipedia.org/wiki/Edit_distance
[6] https://en.wikipedia.org/wiki/Jaccard_index

– **Entity-Entity Context Similarity Feature**. Similar to the idea introduced in the *mention-entity context similarity feature*, i.e. semantic related entities may be of similar context, and we use the same process for extracting the context of each entity. Given an entity e_1 and an entity e_2, we also use Jaccard Similarity to compute the *entity-entity context similarity feature* $contSim_{ee}(e_1, e_2)$ between their respective context $entContext(e_1)$ and $entContext(e_2)$ as

$$contSim_{ee}(e_1, e_2) = \frac{|entContext(e_1) \cap entContext(e_2)|}{|entContext(e_1) \cup entContext(e_2)|} \tag{5}$$

To acquire the semantic relatedness between an entity e_1 and an entity e_2, we compute the **Entity-Entity Semantic Relatedness** $SR_{ee}(e_1, e_2)$ integrating *triple relation feature* $IsRDF(e_1, e_2)$ with the *entity-entity context similarity feature* $contSim_{ee}(e_1, e_2)$ as follows:

$$SR_{ee}(e_1, e_2) = 0.99 \times (\alpha_2 \cdot IsRDF(e_1, e_2) + \beta_2 \cdot contSim_{ee}(e_1, e_2)) + 0.01 \tag{6}$$

where both α_2 and β_2 are also set to 0.5.

c) Iterative Probability Propagation. To combine different EL impact factors for the EL decisions, we utilize iterative probability propagation to compute the probabilities associated with entity nodes (i.e. the probabilities for entities to be the referent entities of mentions) until convergence. The detailed process of our proposed iterative probability propagation on each *Entity Disambiguation Graph* is described as follows.

Given an *Entity Disambiguation Graph* $\boldsymbol{G = (V, E)}$ containing $\boldsymbol{n}$ nodes (with $\boldsymbol{k}$ mention nodes and $\boldsymbol{l}$ entity nodes), each node is assigned to an integer index from $\boldsymbol{1}$ (0 ~ n-1) to $\boldsymbol{n}$. We use these indexes to represent the nodes, and an $n \times n$ adjacency matrix of the *Entity Disambiguation Graph* $\boldsymbol{G}$ is denoted as $\boldsymbol{A}$, where $\boldsymbol{A}_{ij}$ refers to the transition probability from the node i to the node j and $\boldsymbol{A}_{ij} = \boldsymbol{A}_{ji}$. Since the edge between the node i to the node j has been associated with a probability, which is the semantic relatedness (defined in Equation 3 and Equation 6) between different nodes, we define $\boldsymbol{A}_{ij}$ as

$$\boldsymbol{A}_{ij} = \begin{cases} \frac{SR_{me}(i,j)}{SR_{me}(i,*)}, & \text{if } i \neq j \text{ and } i \text{ represents a mention node} \\ \gamma \times \frac{SR_{ee}(i,j)}{SR_{ee}(i,*)}, & \text{if } i \neq j \text{ and } i, j \text{ represent two entity nodes} \\ (1-\gamma) \times \frac{SR_{me}(i,j)}{SR_{me}(i,*)}, & \text{if } i \neq j,\ i \text{ is an entity node and } j \text{ is a mention node} \\ 0, & \text{if } i = j \end{cases} \tag{7}$$

where $SR_{me}(i, j)$ is the *mention-entity semantic relatedness* between a mention node and an entity node (defined in Equation 3), $SR_{me}(i, j) = SR_{me}(j, i)$,

$SR_{ee}(i,j)$ is the *entity-entity semantic relatedness* between entity nodes (defined in Equation 6), $SR_{ee}(i,*)$ means the total *entity-entity semantic relatedness* between i and its adjacent entity nodes and γ is set to 0.5. If i represents a mention node, $SR_{me}(i,*)$ is the total *mention-entity semantic relatedness* between i and all of its adjacent nodes. If i denotes an entity node, $SR_{me}(i,*)$ is the total *mention-entity semantic relatedness* between i and its adjacent mention nodes.

Here, we finally define a notation, i.e. an $n \times 1$ vector $\boldsymbol{r}$ for all the nodes, where $\boldsymbol{r}(i)$ means the probability for the node i to be the referent entity of some mention (if i is an entity node). To compute $\boldsymbol{r}$ with iterative probability propagation, we first set its initial value $\boldsymbol{r}^0$. As introduced before, if the node i is a mention node, $\boldsymbol{r}^0(i)$ is set to the initial importance of i, i.e. $1/k$. If i is an entity node, $\boldsymbol{r}^0(i) = 0$. Then, we update $\boldsymbol{r}$ in the process of iterative probability propagation using other EL impact factors, i.e. *mention-entity semantic relatedness* and *entity-entity semantic relatedness* encoded in the matrix $\boldsymbol{A}$. In this way, the recursive form of $\boldsymbol{r}$ is given as follows:

$$\boldsymbol{r}^{t+1} = ((1-d) \times \frac{\boldsymbol{E}}{n} + d \times \boldsymbol{A}) \times \boldsymbol{r}^t \tag{8}$$

where t is the number of iterations and $\boldsymbol{E}$ is an $n \times n$ square matrix of all 1's. In this formula, to ensure the matrix $\boldsymbol{A}$ is aperiodic to converge, we add a special kind of undirected edges from each node to all the other nodes and give each edge a small transition probability controlled by the damping factor 衰减 d. In other words, during the process of iterative probability propagation, there exists a probability that the EL impact factors are propagated by neither the defined *mention-entity edges* nor *entity-entity edges*, but the above special kind of edges associated with small transition probabilities. Since the process of iterative probability propagation is similar to the PageRank algorithm [5], we apply the same setting that $d = 0.85$. After the iterative probability propagation, given a mention m and its corresponding set of candidate referent entities $ESet(m) = \{e_1, e_2, ..., e_s\}$, we pick the entity which is of the highest probability in $ESet(m)$ as the referent entity of m.

Different from other methods, our approach for EL in Web tables with a single KB does not rely on any specific information but only general RDF triples. Thus, it can be applied to any KB containing RDF triples in Linking Open Data[7], including DBpedia [1,3], Yago [17,18], Freebase [4], Zhishi.me [14] and etc..

3.2 Improving Entity Linking with Multiple Linked KBs

Entity Linking (EL) in Web tables only with a single KB can not always ensure a good coverage. One solution is to respectively perform the task of EL with different KBs so that we can improve the coverage of the EL results. However, the problem is that there may exist conflicts among the results of EL with different KBs. In this paper, we have done a test on Zhishi.me consisting of three largest Chinese linked online encyclopedic KBs, i.e. Chinese Wikipedia, Baidu Baike and

[7] `http://linkeddata.org/`

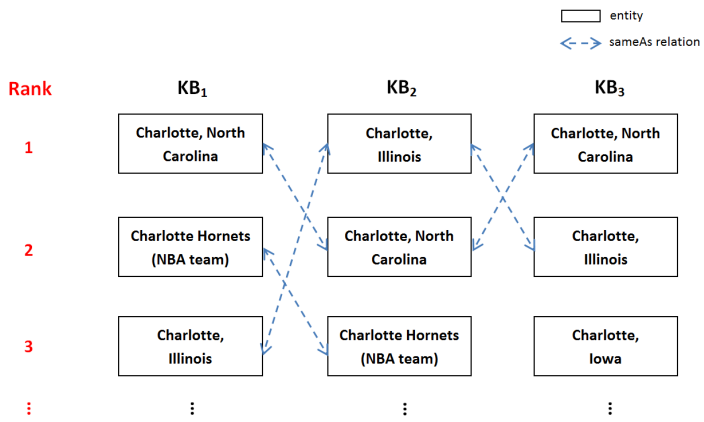

Fig. 4: An Example of EL Results: the Ranking Lists of Entities in Different KBs for the Mention "`Charlotte`" in Figure 2

Hudong Baike. We first apply our proposed approach for EL with a single KB to our extracted Web tables (more than 70 thousand) with Chinese Wikipedia, Baidu Baike and Hudong Baike, respectively. Then, given a mention in a Web table and its identified entities in three KBs, if two identified entities have the "`sameAs`" relation, they can be considered as the same individual, otherwise they are different, i.e. there exists a conflict. According to the statistics, the conflicts exist in totally 38.94% EL results (one result refers to a mention with its identified entities from different KBs). After that, we observe the EL results of the above test and analyse the reasons for such conflicts among the results of EL with different KBs as follows:

- **Reason 1**: For some KBs, the EL results are really incorrect, that is to say, some potential correct referent entities do not rank the highest.
- **Reason 2**: The "`sameAs`" relations are incomplete between KBs, i.e. there does not exist the "`sameAs`" relations between some equivalent entities from different KBs.

Based on these two reasons, we present our approach in detail to solve the conflicts of EL with different KBs in the following part of this section.

Suppose that there are n different linked KBs. For each given KB, we first apply our proposed approach to EL with a single KB to the given Web table. Then, for each mention, we can get its n ranking lists of referent entities. Afterwards, with the "`sameAs`" relations between entities in different KBs, we group the entities representing the same individual into different sets. For example, in Figure 4, we can get 4 sets as follows:
1) Set_1 = {"`Charlotte, North Carolina`" (KB_1), "`Charlotte, North Carolina`" (KB_2), "`Charlotte, North Carolina`" (KB_3)};
2) Set_2 = {"`Charlotte, Illinois`" (KB_1), "`Charlotte, Illinois`" (KB_2), "`Charlotte, Illinois`" (KB_3)};

3) $Set_3 = \{$"`Charlotte Hornets`" (KB_1), "`Charlotte Hornets`" (KB_2) $\}$;
4) $Set_4 = \{$"`Charlotte, Iowa`" $(KB_3)\}$.
After that, we compute the average ranking, the highest ranking and the number of the entities in each set. For example, for the entities in set_1, the average ranking is computed as $(1 + 2 + 1)/3 = 1.33$, the highest ranking is 1 and the number is 3. Finally, we propose three rules as follows to solve the conflicts by choosing one set as the final EL results for the given mention.

- **Rule 1**: If both the average ranking and the highest ranking of the entities in a set rank the highest, and the number of the entities in this set is not less than half of the number of KBs, then we choose this set as the final EL results for the given mention.
- **Rule 2**: If there exist two or more sets that the average ranking and the highest ranking of the sets' corresponding entities are the same and rank the highest, also the number of the entities in each of these sets is not less than half of the number of KBs, then we choose one set at random as the final EL results for the given mention.
- **Rule 3**: If the number of the entities in each set is less than half of the number of KBs, the original EL results of the given mention remain unchanged.

To obtain the global and local optimal EL results at the same time, we consider not only the average ranking and the highest ranking of the entities in each set, but also the number of times that each individual (represented by a set of entities) occurs in different KBs. If the number of the entities in a set is less than half of the number of KBs, it means that the individual represented by these entities is covered by few KBs, so the average ranking is not convincing and it is not reasonable to take this set of entities to solve the conflicts among the results of EL with all the KBs.

According to *Reason 2*, if there exist more "`sameAs`" relations between the entities from different KBs, we may better solve the conflicts with our proposed rules. Here, in order to learn new "`sameAs`" relations, we define three features and train a supervised learning classifier Support Vector Machine (SVM), which is of the best performance in most situations [8]. The proposed features are introduced as follows:

- **Synonym Feature**. This feature tries to detect that whether the strings of two entities may represent synonyms. We input the strings of two entities e_1 and e_2 into BabelNet [13], if these two strings may represent synonyms in BabelNet, the synonym feature $isSyn = 1$, otherwise $isSyn = 0$.
- **String Similarity Feature**. This feature captures the linguistic relatedness between entities. It is denoted by $strSim(e_1, e_2)$, where e_1 and e_2 are entities. We use Equation 1 to compute this feature using edit distance.
- **Entity-Entity Context Similarity Feature**. For two given entities in different KBs, this feature measures the similarity between the extracted context of entities and is already defined in Equation 5.

After completing the "`sameAs`" relations with this SVM classifier, we also utilize our proposed rules to decide the final EL results. It is to verify whether the

performance is improved compared with that of the rules only using the existing "`sameAs`" relations.

4 Experiments

In this section, we evaluated our approach on the sampled Web tables with three linked KBs (i.e. Chinese Wikipedia, Baidu Baike and Hudong Baike) in Zhishi.me, and compared our approach with two state-of-the-art systems for EL in Web tables and two degenerate versions of our approach.

4.1 Data Set and Evaluation Metrics

Since entity linking in Web tables with multiple linked KBs is a new task, we do not have any existing benchmark. Therefore, we need to generate ground truths by ourselves. We have extracted more than 70 thousand Web tables containing relational data from the Web. We randomly sampled 200 Web tables and invited five graduate students to manually map each string mention in table cells to the entities in each KB of Zhishi.me. The labeled results is based on majority voting and are publicly available[8]. Besides, in order to train the SVM classifier to learn new "`sameAs`" relations between different KBs, we also need to manually generate the labeled data. We first randomly selected 500 existing "`sameAs`" relations as the positive labeled data. Then, we random selected 3,000 entity pairs, each of which consists of the entities from different KBs. Finally, we also asked the five graduate students to label them and 3,000 entity pairs were all labeled as negative.

For each sampled Web table, we performed EL with our approach and the designed comparison methods. We evaluated the results with four metrics, which are Precision, Recall, F1-score and MRR (Mean Reciprocal Rank [7]). F1-score is the harmonic mean of precision and recall. Mean Reciprocal Rank is used for evaluating the quality of the ranking lists. For a mention m, the reciprocal rank in EL is the multiplicative inverse of the rank for m's referent entity. For example, if the correct referent entity of m is in the second place in the ranking lists generated by some EL algorithm, the reciprocal rank is 1/2.

4.2 Comparsion Methods

We compared our approach with the following methods.

- ***TabEL***: TabEL [2] is the current state-of-the-art system for EL in Web tables, and it uses a collective classification technique with several general features to collectively disambiguate all mentions in a given Web table. Besides, any KB can be used for EL in Web tables with TabEL.

[8] `https://github.com/jxls080511/MK-EL`

Table 1: the Overall EL Results Evaluated with Each Single KB

Knowledge Base	Approach	Precision	Recall	F1-score	MRR
Chinese Wikipeida	TabEL	0.823	0.809	0.816	0.858
	LIEGE	0.778	0.747	0.762	0.813
	Our-s	0.830	0.797	0.813	0.860
	Our-m-e	0.861	0.821	0.841	0.881
	Our-m-(e+n)	**0.873**	**0.828**	**0.850**	**0.887**
Baidu Baike	TabEL	0.659	0.628	0.643	0.707
	LIEGE	0.629	0.576	0.601	0.670
	Our-s	0.696	0.652	0.673	0.725
	Our-m-e	0.758	0.705	0.731	0.746
	Our-m-(e+n)	**0.774**	**0.727**	**0.750**	**0.776**
Hudong Baike	TabEL	0.681	0.649	0.665	0.780
	LIEGE	0.661	0.632	0.646	0.751
	Our-s	0.708	0.642	0.673	0.768
	Our-m-e	0.729	0.700	0.714	0.787
	Our-m-(e+n)	**0.744**	**0.708**	**0.726**	**0.796**

- ***LIEGE***: LIEGE [16] is a general approach to link the string mentions in list-like Web tables (multiple rows with one column) to the entities in a given KB. It proposes an iterative substitution algorithm with three features to EL in Web lists. This approach can also be applied to EL in Web tables with any KB.
- ***Our-s***: It is a degenerate version of our approach. It only uses our proposed approach for EL with a single KB and does not utilize the rules with "`sameAs`" relations to improve the EL results.
- ***Our-m-e***: It is also a degenerate version of our approach. After performing EL with each single KB, it only uses existing "`sameAs`" relations (without newly learned "`sameAs`" relations) to improve the EL results.

4.3 Result Analysis

In the whole version of our proposed approach (denoted as ***Our-m-(e+n)***), we first apply a graph-based algorithm without using any specific information to EL in Web tables with each single KB, and then we leverage the existing and newly learned "`sameAs`" relations between the entities from different KBs to help improve the results of EL. Table 1 gives the overall results of our approach and the designed comparison methods evaluated with each single KB, and we can see that:

- Our approach for EL with a single KB, i.e. *Our-s*, is comparable to the state-of-the-art system TabEL and outperforms LIEGE, which reflects the effectiveness of our proposed graph-based algorithm.
- *Our-m-e* is always better than *Our-s* in precision, recall, F1-score and MRR. It shows the value of our proposed heuristic rules for improving the EL results of *Our-s*.

– The whole version of our approach, i.e. *Our-m-(e+n)* outperforms all the other comparison methods, which verifies the superiority of our approach for EL in Web tables with multiple linked KBs. Compared with *Our-m-e*, the better performance of *Our-m-(e+n)* demonstrates that the newly learned "`sameAs`" relations are beneficial to solve the conflicts among the EL results of *Our-s* with different KBs.

Besides, we also calculated the precision, recall and F1-score as the evaluation results of our approach (i.e. *Our-m-(e+n)*) on the whole Zhishi.me. The precision is 0.831, recall is **0.903** and F1-score is 0.866. The most important thing is that the recall is significantly improved, which shows EL in Web tables with multiple linked KBs can really ensure a good coverage.

5 Conclusions and Future Work

In this paper, we presented a new approach for EL in Web tables with multiple linked KBs. We first proposed an algorithm based on graph-based iterative probability propagation to perform EL with each single KB. In order to improve the EL results generated by the first step, we then applied three heuristic rules leveraging the existing and newly learned "`sameAs`" relations between the entities from different KBs. The experimental results showed that our approach not only outperforms the designed state-of-the-art comparison methods in different evaluation metrics, but also can use any single KB or linked KBs for EL in the Web tables.

As for the future work, we first will build more benchmarks of other languages for the new task of EL in Web tables with multiple linked KBs and further verify the effectiveness of our approach in other languages, especially English. We then plan to provide APIs or tools as the programming interface of our proposed approach. Finally, we also consider to extend our approach to cross-lingual entity linking in Web tables with multiple linked KBs.

6 Acknowledgements

This work is supported in part by the National Natural Science Foundation of China (NSFC) under Grant No. 61272378, the 863 Program under Grant No. 2015AA015406 and the Research Innovation Program for College Graduates of Jiangsu Province under Grant No. KYLX16_0295.

References

1. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: Dbpedia: A nucleus for a web of open data. In: ISWC, pp. 722–735 (2007)
2. Bhagavatula, C.S., Noraset, T., Downey, D.: Tabel: Entity linking in web tables. In: ISWC, pp. 425–441 (2015)

3. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: Dbpedia-a crystallization point for the web of data. Web Semantics: science, services and agents on the world wide web **7**(3), 154–165 (2009)
4. Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J.: Freebase: a collaboratively created graph database for structuring human knowledge. In: SIGMOD, pp. 1247–1250 (2008)
5. Brin, S., Page, L.: Reprint of: The anatomy of a large-scale hypertextual web search engine. Computer networks **56**(18), 3825–3833 (2012)
6. Cafarella, M.J., Halevy, A., Wang, D.Z., Wu, E., Zhang, Y.: Webtables: exploring the power of tables on the web. PVLDB **1**(1), 538–549 (2008)
7. Craswell, N.: Mean reciprocal rank. In: Encyclopedia of Database Systems, pp. 1703–1703. Springer (2009)
8. Fernández-Delgado, M., Cernadas, E., Barro, S., Amorim, D.: Do we need hundreds of classifiers to solve real world classification problems? The Journal of Machine Learning Research **15**(1), 3133–3181 (2014)
9. Hignette, G., Buche, P., Dibie-Barthélemy, J., Haemmerlé, O.: Fuzzy annotation of web data tables driven by a domain ontology. In: ESWC, pp. 638–653 (2009)
10. Limaye, G., Sarawagi, S., Chakrabarti, S.: Annotating and searching web tables using entities, types and relationships. PVLDB **3**(1-2), 1338–1347 (2010)
11. Mulwad, V., Finin, T., Joshi, A.: Semantic message passing for generating linked data from tables. In: ISWC, pp. 363–378 (2013)
12. Muñoz, E., Hogan, A., Mileo, A.: Using linked data to mine rdf from wikipedia's tables. In: WSDM, pp. 533–542 (2014)
13. Navigli, R., Ponzetto, S.P.: Babelnet: Building a very large multilingual semantic network. In: ACL, pp. 216–225 (2010)
14. Niu, X., Sun, X., Wang, H., Rong, S., Qi, G., Yu, Y.: Zhishi. me-weaving chinese linking open data. In: ISWC, pp. 205–220 (2011)
15. Pereira, B.: Entity linking with multiple knowledge bases: An ontology modularization approach. In: ISWC, pp. 513–520 (2014)
16. Shen, W., Wang, J., Luo, P., Wang, M.: Liege:: link entities in web lists with knowledge base. In: SIGKDD, pp. 1424–1432 (2012)
17. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: a core of semantic knowledge. In: WWW, pp. 697–706 (2007)
18. 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)
19. Syed, Z., Finin, T., Mulwad, V., Joshi, A.: Exploiting a web of semantic data for interpreting tables. In: WebSci, vol. 5 (2010)
20. Venetis, P., Halevy, A., Madhavan, J., Paşca, M., Shen, W., Wu, F., Miao, G., Wu, C.: Recovering semantics of tables on the web. PVLDB **4**(9), 528–538 (2011)
21. Zhang, Z.: Learning with partial data for semantic table interpretation. In: EKAW, pp. 607–618 (2014)
22. Zhang, Z.: Towards efficient and effective semantic table interpretation. In: ISWC, pp. 487–502 (2014)