Mining RDF from Tables in Chinese Encyclopedias

Weiming Lu(✉), Zhenyu Zhang, Renjie Lou, Hao Dai, Shansong Yang, and Baogang Wei

College of Computer Science and Technology, Zhejiang University, Hangzhou, China
luwm@zju.edu.cn

Abstract. Web tables understanding has recently attracted a number of studies. However, many works focus on the tables in English, because they usually need the help of knowledge bases, while the existing knowledge bases such as DBpedia, YAGO, Freebase and Probase mainly contain knowledge in English.

In this paper, we focus on the RDF triples extraction from tables in Chinese encyclopedias. Firstly, we constructed a Chinese knowledge base through taxonomy mining and class attribute mining. Then, with the help of our knowledge base, we extracted triples from tables through column scoring, table classification and RDF extraction. In our experiments, we practically implemented our approach in 6,618,544 articles from *Hudong Baike* with 764,292 tables, and extracted about 1,053,407 unique and new RDF triples with an estimated accuracy of 90.2%, which outperforms other similar works.

1 Introduction

Nowadays, large-scale knowledge bases (KBs) are playing an increasing role in many intelligent applications. These knowledge bases contain millions of facts, such as information about people, locations, organizations, which are represented as RDF triples (subject-predicate-object triples).

Most of the knowledge bases are primarily builded by integrating the existing structured knowledge (e.g. Wikipedia's infoboxes for DBPedia [3]), or extracting knowledge from unstructured text such as NELL [5]. However, unstructured text can be very noisy, and the existing structured knowledge is quite limited. Web tables are content-rich, and relatively easier for knowledge extraction than the unstructured text. Therefore, many approaches have been tried to populate KBs by using Web tables.

The main challenge for RDF mining from Chinese tables is that the existing knowledge bases such as DBPedia, YAGO [18] and Freebase [4] contain very limited knowledge in Chinese, which makes the understanding of Chinese text very difficult. Fortunately, there are two large-scale collaboratively Chinese encyclopedias named Baidu Baike[1] and Hudong Baike[2], and they claimed they contain

[1] http://www.baike.baidu.com
[2] http://www.baike.com

J. Li et al. (Eds.): NLPCC 2015, LNAI 9362, pp. 285–298, 2015.
DOI: 10.1007/978-3-319-25207-0_24

more than 11.2 and 12.2 million articles respectively[3]. Therefore, in this paper, we will build a Chinese knowledge base from these encyclopedias, and then mine RDF from tables in encyclopedias for knowledge population.

Our paper has the following differences against other works. Firstly, we have to construct a Chinese knowledge base from scratch to help RDF mining. In addition to the *subclass-of*, *instance-of* and *class-attribute* relations, the probabilities for these relations should also be provided. Secondly, we directly classify tables into *Genuine Table with Header* and *Genuine Table without Header*, and detect object columns besides subject column in tables, which can improve the performance of the table understanding. This is because the attributes of an entity extracted from *infobox* could be incomplete or even missing, and information about different entities could be mixed in one table. For example, Figure 2a mixes the information about persons and organizations.

The rest of the paper is organized as follows. Section 2 provides the overview of our work. Then, we describe the Chinese knowledge base construction in Section 3, and the table understanding in Section 4. Experiments, related work and conclusions appear in Section 5, 6 and 7.

2 Approach Overview

The procedure of our work contains three major components: table extraction, table classification and RDF extraction, as shown in Figure 1. In order to help the RDF mining, we construct a Chinese knowledge base from the encyclopedias, which includes *Taxonomy Mining* and *Class Attribute Mining*.

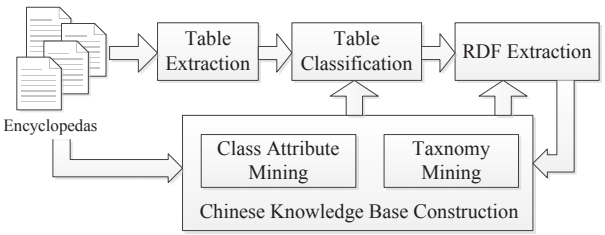


Fig. 1. The architecture of RDF mining from Chinese encyclopedias

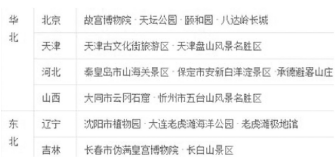
Table Extraction. We mainly focus on the *wikitables* in encyclopedias as in [14]. At first, we locate the tables by looking for table-related HTML tags (e.g. table, tr and td), and then parse the table into a matrix by using the normalization technologies in [14]. In addition, many *wikitables* in encyclopedias always put table's abstract in the top row, which has only one column, so we delete this row in the tables.

[3] At Jan 10, 2015.

Table Classification. Although well-formed tables have been obtained, there are still several categories in the tables, which will lead to different subsequent operations. The categories include *Genuine Table with Header*, *Genuine Table without Header* and *Non-genuine Table* (entities in the same class are distributed in different columns), are shown in Figure 2. The aim of table classification is to classify the tables into categories, with the help of the Chinese knowledge base.

排名	财富(亿元)	姓名	公司	总部	行业	年龄	出生地
1	430	黄光裕	鹏润投资	北京	家电零售、房地产、投资	39	广东
2	360	杜双华	日照钢铁	山东	钢铁	43	河北
3	330	杨惠妍	碧桂园	广东	房地产	27	广东

(a) Genuine Table with Header



华北	北京	故宫博物院 · 天坛公园 · 颐和园 · 八达岭长城
	天津	天津古文化街旅游区 · 天津盘山风景名胜区
	河北	秦皇岛市山海关景区 · 保定市安新白洋淀景区 · 承德避暑山庄
	山西	大同市云冈石窟 · 忻州市五台山风景名胜区
东北	辽宁	沈阳市植物园 · 大连老虎滩海洋公园 · 老虎滩极地馆
	吉林	长春市伪满皇宫博物院 · 长白山景区

(b) Genuine Table without Header

中国旅游景点（一）

中国地域辽阔，美丽的自然风光与丰富的人文古迹比比皆是，美不胜收。

梨园剧场	和平寺	北京九龙游乐园	新文化运动纪念馆
颐和园	大栅山森林公园	青龙湖水上乐园	劳动人民文化宫
平谷老象峰景区	虎峪自然风景区	繁竹院公园	十渡景区
密云南山滑雪场	皇姑沁风景区	李大钊烈士陵园	玉渊潭
房山十渡	平西抗日烈士陵园	中央电视塔	银山塔林

(c) Non-genuine Table

Fig. 2. Three categories of tables

RDF Extraction. When *Genuine Tables* are obtained, we extract RDF triples from these tables. The header could be used as the predicate in triples for *Genuine Table with Header*. For tables without header, we estimate the predicate with the help of our KB.

Class Attribute Mining and **Taxonomy Mining** are two components for building the Chinese knowledge base from encyclopedias. In **Class Attribute Mining**, we mine the proper attributes for each class in the knowledge base, and provide the probabilities of $p(attribute|class)$ and $p(class|attribute)$. **Taxonomy Mining** mines the probability of $p(class|entity)$ from encyclopedias.

We crawled 6,618,544 unique articles from *Hudong Baike*, where each article represents an entity, and it may include infobox, catalog, tags and innerlinks, which can be used to build the Chinese knowledge base. Finally, we extracted 4,897,722 *toc*s (table of content), 728,039 *infobox*s and 764,292 *wikitable*s, where the *wikitable*s are our focus in this paper for RDF mining.

3 Chinese Knowledge Base from Scratch

In this section, we mainly focus on the Chinese knowledge base construction from scratch, including *taxonomy mining* and *class attribute mining*.

3.1 Taxonomy Mining

Hudong Baike provides a category system for article navigation, but it does not form a real subsumption hierarchy, which includes *isa* and *notisa* relations between categories. Therefore, we have to distinguish *isa* and *notisa* relations, and use *isa* relation to form *subclass-of* relations. We treat this problem as a binary classification problem: given two categories c_i and c_j, we train a classifier to predicate whether c_i and c_j have the *isa* relation, where linguistic features and structural features [23] are used for the classifier.

Similarly, *instance-of* relation induction problem is also treated as a binary classification problem: given an article a and its category c, we train a classifier to predicate whether a is an instance of c. Here, we also use linguistic features and structural features as in [23] for the classifier.

In practice, we trained SVM classifiers and used precision, recall, and F1 to evaluate the performance of *subclass-of* and *instance-of* relations based on a manually labeled data set including randomly selected 1000 pairs of categories, and 1000 pairs of article and its category. We got precision=87.27%, recall=86.75%, F1=87.01% for *subclass-of* relation, and precision=80.33%, recall=67.58%, F1=73.41% for *instance-of* relation.

Finally, the induced taxonomy could be represented as $T = \{E, C, R_i, R_c, P_i, P_c\}$, where E and C is the set of entities and classes. When $r_i(e, c) \in R_i \subset E \times C$, it means entity e and class c has *instance-of* relation. Similarly, $r_c(c_i, c_j) \in R_c \subset C \times C$ indicates c_i is a subclass of c_j. In addition, we also consider the probability output of classifiers as the probability of relations. $p_i(e, c) \in P_i$ is the probability of *instance-of* relation for e and c, and $p_c(c_i, c_j) \in P_c$ is the probability of *subclass-of* relation for c_i and c_j.

If entity e can reach a class c along with *instance-of* and *subclass-of* relations, there is a path $path(e, c) = \langle e, c_1, c_2, ..., c_k \rangle$, where $c_k = c$, $r_i(e, c_1) \in R_i$, and $r_c(c_i, c_{i+1}) \in R_c$. Then, for each entity $e \in E$, we can get its classes $C(e) = \{c | \exists path(e, c), c \in C\}$. For each class c, we can get its entities $E(c) = \{e | \exists path(e, c), e \in E\}$.

Entity e and its classes $C(e)$ can form an graph $G = (V_G, E_G)$ as shown in Figure 3, where $V_G = \{e, c \in C(e)\}$ and $E_G = \{r_i(e, c \in C(e)) \in R_i, r_c(c_i \in C(e), c_j \in C(e)) \in R_c\}$.

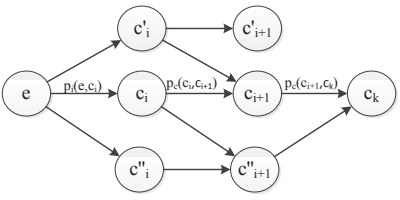


Fig. 3. Graph for an entity

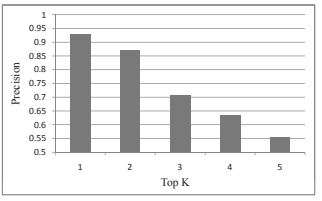


Fig. 4. Performance of RWR

In this graph, we use RWR (random walk with restart)[19] to calculate the relevance score $p(e, c \in C(e))$ for each pair of $(e, c \in C(e))$. Formally, the walker starts from the node e, and then follows an edge to another node at each step. Additionally, at every step, the walker would return to the node e with a non-zero probability c. Let v_e be a vector of zeros with the element corresponding to node e set to 1 ($v_e(e) = 1$), and then the steady state probability vector $u_e = (u_e(e), u_e(c_1), u_e(c_2), ..., u_e(c_N))$ could be estimated by matrix multiplication. Let A be the adjacency matrix of the graph G, where $A(e, c \in C(e)) = p_i(e, c)$ *or* $A(c_i \in C(e), c_j \in C(e)) = p_c(c_i, c_j)$, then u could be calculated by $u_e = (1 - c)Au_e + v_e$. Finally, $p'(e, c \in C(e)) = u_e(c)$.

We randomly selected 85 entities with its corresponding ranked class list, and asked students to evaluate them. The precision of top K is shown in Figure 4, which shows the RWR can find proper classes for entities.

3.2 Class Attribute Mining

After taxonomy mining, we could use *infobox* to mine the attributes for each class with the help of the taxonomy.

Let E_b be the set of entities whose corresponding articles have *infobox*, and then the attribute set of each entity $e \in E_b$ would be represented as $A(e)$, which are extracted from its *infobox*. Then, given a class c, we can obtain its attribute set $A(c) = \cup_{e \in E(c)} A(e)$. Meanwhile, we can also obtain the class set $C'(a) = \{c | \exists e \in E, a \in A(e), c \in C(e)\}$ for attribute a. The co-occurrent frequency of attribute a and class c can be calculated by $f(a, c) = |\{e | a \in A(e) \cap e \in E(c)\}|$, and the occurrence of class c and attribute a are $f_{cls}(c) = |\{e | e \in E(c)\}|$ and $f_{attr}(a) = |\{e | a \in A(e), e \in E\}|$ respectively. However, even attribute a and class c occur frequently, they may not be the right attribute for class or right class for attribute. Figure 5 ranks the attribute set $A(Movie)$ and the class set $C'(Release\ time)$ respectively according to the occurrence frequency.

Attributes	Frequency
导演(Director)	58361
上映时间(Release time)	45688
主演(Actors)	44093
类型(Type)	41733
制片地区(Region)	34429
外文名(English name)	33411
编剧(Writer)	32787
对白语言(Language)	28776
片长(Length)	23502
色彩(Color)	20335
.....	

(a) The attribute rank list for the class *Movie*

Categories	Frequency
电影(Movie)	45688
影视(Film and television)	18514
艺术(Art)	17493
电视剧(Drama)	9463
剧情片(Feature Film)	6274
娱乐(Entertainment)	5737
导演(Director)	4204
年(Year)	4154
喜剧(Comedy)	3523
影片(Film)	3481
.....	

(b) The class rank list for the attribute *Release time*

Fig. 5. Attribute and class rank list according to the frequency

But we find that although *Art* and *Entertainment* are ranked in the front, they are not the proper classes for attribute *Release time*. So, we should filter some improper classes for a given attribute.

Here, we train a classifier to filter them, and the features for attribute-class pair (a, c) are list as follows.

1. $|C'(a)|$.
2. $R(C'(a), c)$.The rank of c in $C'(a)$ ordered by $f(a, c)$.
3. (2)/(1).
4. $f(a, c)/f(c)$.
5. $\frac{1}{|C(a)|}\sum_{c' \in C'(a)} \frac{f(a,c')}{f_{cls}(c')}$.
6. (4)/(5)
7. $|A(c)|$.
8. $R(A(c), a)$. The rank of a in $A(c)$ ordered by $f(a, c)$.
9. (8)/(7)
10. $\frac{1}{|A(c)|}\sum_{a' \in A(c)} \frac{f(a',c)}{f_{attr}(a')}$.

In order to train classifiers, we randomly select 408 attribute-class pairs from $(a, c \in C(a))$ or $(a \in A(c), c)$, and the pair (a, c) must satisfy $|C(a)| \geq 10, |A(c)| \geq 10$. Then, students are asked to label the pairs to form training data and testing data. Four classifiers in Weka: *Naive Bayes*, *Decision Tree J48*, *Logistic Classifier*, and *Random Forest* are trained, and the test results are shown in Table 1.

Table 1. The performance of (a, c) classification with different classifiers

Method	Prec	Rec	F1	Method	Prec	Rec	F1
Naive Bayes	0.828	0.811	0.819	Decision Tree J48	0.805	0.805	0.805
Logistic Classifier	0.811	0.811	0.811	Random Forest	0.812	0.811	0.811

We use Naive Bayes as our classifier to clean the $C(a)$ and $A(c)$, and then score $p''(a, c \in C(a)) = f(a, c)/f_{attr}(a)$.

4 Table Classification and Understanding

In this section, we extract RDF from *Genuine Tables* with the following steps: *Column Scoring*, *Table Classification*, and *RDF Extraction*.

4.1 Column Scoring

Wikitable is parsed into a matrix $T(m, n)$ with m rows and n columns in Section 2, so we can combine the information of row and column to compute the score for each column. Here, we don't detect the header at first as other related works, while compute the row score for first row directly, and then use the score with other features to classify tables, which will be described in Section 4.2.

For the i^{th} column, we firstly obtain its candidate class set $C_i = \cup_{1 \leq j \neq i \leq n} C'(T(1, j))$. Then, the row score for i^{th} column and class $c \in C_i$ can be calculated

by $s_{row}(i,c) = \sum_{1 \le j \ne i \le n} p''(T(1,j),c)$. The column score for i^{th} column and class $c \in C_i$ can be calculated by $s_{col}(i,c) = \sum_{1 \le j \le m} p'(T(j,i),c)$. Finally, the overall score for i^{th} column is calculated by $score(i) = \max_{c \in C_i} s_{row}(i,c) \cdot s_{col}(i,c)$.

Obviously, the column with larger score is more likely to be the subject column. Therefore, subject column can be determined by $sub_{col} = \arg\max_i score(i)$, and the corresponding class is $sub_{cls} = \arg\max_{c \in C_{sub_{col}}} [s_{row}(sub_{col},c) \cdot s_{col}(sub_{col},c)]$. The object columns are much related to the header column, so they can be filtered by comparing $p''(T(1,j), sub_{cls})$ with a threshold λ. That is, the columns with $p''(T(1,j), sub_{cls}) \ge \lambda, 1 \le j \ne sub_{cls} \le n$ can be considered as object columns, which are denoted as $objs_{col}$.

4.2 Table Classification

Subject column and object columns could be detected in *wikitables* in Section 4.1, but it may be not correct for some tables, especially for *Non-genuine Tables*. So in this section, we directly classify *wikitables* into three categories *Genuine Table with Header*, *Genuine Table without Header* and *Non-genuine Table* by combining the column scores and other features.

We classify features into five groups: statistics features (S), cell features (C), layout features (L), predicate features (P) and score features (Sc).

Statistics Features

- Average number of cells in rows *acr*. The number of cells in every rows is counted before table normalization.
- Average number of cells in columns *acc*. The number of cells in every columns is counted before table normalization.
- ratio of *acr*/*acc*.
- Deviation of the number of cells in rows, and the deviation of the number of cells in columns.
- Average cell length, and deviation of cell length.
- Within-row length consistency, and Within-column length consistency [22].

Cell Features

- number of cells containing HTML tags (*th*, *img*, *b*, *a* respectively).
- Percentage of numeric cells.
- Percentage of alphabetical cells, date cells, string cells and empty cells.
- Within-row type consistency and within-column type consistency [22].

Layout Features

- row number of first row in un-normalized table.
- Ratio of cells containing HTML tag *th* in the first row to the all cells containing HTML tag *th*.
- Ratio of cells containing HTML tag *b* in the first row to the all cells containing HTML tag *b*.

– the number of columns with different cell type in the first row with other cells in the same column.

Predicate Features

– Weighted average position of predicate in row *wapr*. if $T(i,j) \in \cup_{e \in E} A(e)$, we call $T(i,j)$ is a predicate, which will form a set $S = \{T(i,j) | T(i,j) \in \cup_{e \in E} A(e)\}$. Then $wapr = \frac{1}{|S|} \cdot \sum_{T(i,j) \in S} i \cdot f_{attr}(T(i,j))$
– Weighted average position of predicate in column *wapc*. $wapc = \frac{1}{|S|} \cdot \sum_{T(i,j) \in S} j \cdot f_{attr}(T(i,j))$.
– Weighted deviation position of predicate in row, i.e. the deviation of $i \cdot f_{attr}(T(i,j))$ for all $T(i,j) \in S$
– Weighted deviation position of predicate in column, i.e. the deviation of $j \cdot f_{attr}(T(i,j))$ for all $T(i,j) \in S$

Score Features

– maximal and average row score for the first row, i.e. $\max_{1 \le i \le n}(\max_{c \in C_i} s_{row}(i,c))$, $\frac{1}{n} \sum_{i=1}^{n} \max_{c \in C_i} s_{row}(i,c)$.
– maximal and average column score for columns, i.e. $\max_{1 \le i \le n}(\max_{c \in C_i} s_{col}(i,c))$, $\frac{1}{n} \sum_{i=1}^{n} \max_{c \in C_i} s_{col}(i,c)$.
– maximal and average overall score for columns, i.e. $\max_{1 \le i \le n} score(i)$, $\frac{1}{n} \sum_{i=1}^{n} score(i)$.
– position of the subject column p_s, i.e. $\arg\max_{1 \le i \le n} score(i)$.
– position of the column with largest column score p_{lc}, i.e. $\arg\max_{1 \le i \le n}(\max_{c \in C_i} s_{col}(i,c))$.
– position of the object column with smallest column score p_{sc}, i.e. $\arg\min_{i \in objs_{col}}(\max_{c \in C_i} s_{col}(i,c))$.
– $|p_s - p_{lc}|$
– $p_s - p_{sc}$

With all these features, we train classifiers to classify *wikitables* into three categories *Genuine Table with Header*, *Genuine Table without Header* and *Non-genuine Table*.

4.3 RDF Extraction

In this section, we extract RDF from two type of tables *Genuine Table with Header* and *Genuine Table without Header*.

For *Genuine Table with Header*, table headers could be used as predicates in RDF triples. So with the detected subject column sub_{col} and object columns $objs_{col}$, we can easily extract RDF from tables: $\{\langle s_{ik}, p_j, o_{ij} \rangle | 1 \le i \le m, k = sub_{col}, j \in objs_{col}, p_j = \text{ header of the } j^{th} \text{ column}\}$.

For *Genuine Table without Header*, we mine the predicate between subject column and object columns for RDF triples. Given subject column and object columns, we extract all pairs on the same row i: $\{(e_{ij}, e_{ik}) | j = sub_{col}, k \in objs_{col}, 1 \le i \le m\}$. Since *infobox* in articles have been parsed into triples

in knowledge base, we can query the relation for each pair in the knowledge base. Obviously, some pairs would not be found in the knowledge base for the knowledge base is incomplete. But we can still select the proper predicate for subject column and each object column by major voting. Then, the triples could be extracted from the tables.

5 Experiments

We crawled 6,618,544 unique articles from *Hudong Baike* at Jan 10, 2015, and then extracted about 764,292 distinct *wikitables* from the articles according to the HTML tags. After ill-formed and small table ($< 2 \times 2$) filtering, we finally obtain 757,282 tables for our RDF extraction.

5.1 Column Scoring Evaluation

In order to evaluate the performance of column scoring, we randomly selected 76 tables, and labeled 76 subject columns (denoted as L_s), and 156 object columns (denoted by L_o). Our approach labeled the subject column and object columns in each tables by column's overall score, which are denoted by M_s and M_o respectively. Then, the precision, recall, and the F measure for subject column could be calculated by $precision = \frac{|L_s \cap M_s|}{|M_s|} = 94.44\%$, $recall = \frac{|L_s \cap M_s|}{|L_s|} = 94.44\%$, and $F_1 = \frac{2 \cdot precision \cdot recall}{precison + recall} = 94.44\%$. Similarly, the precision, recall, and the F measure for object column is $precision = 93.3\%$, $recall = 71.8\%$, and $F_1 = 81.2\%$. The recall for object column is relatively low, since many object column headers occur very infrequently as attributes of entities. But we still can extract sufficient triples from the tables.

We also compared the performance for subject column detection by using overall score, row score and column score, the results are shown in Table 2.

Table 2. The performance of subject column detection by using different scoring method

Method	Prec	Rec	F1
row score	88.41%	84.72%	86.51%
column score	88.73%	87.50%	88.11%
overall score	94.44%	94.44%	94.44%

Table 3. The F_1 of table classification with different classifiers

	S	S+C	S+C+L	S+C+L+P	ALL
NB	0.612	0.590	0.606	0.603	0.704
J48	0.702	0.713	0.720	0.737	0.790
LC	0.691	0.699	0.740	0.707	0.736
RF	0.702	0.762	0.756	0.786	**0.825**

Obviously, the overall score which combines row score and column score reaches the best performance. Column score can obtain a better performance than row score. This is because in *Hudong Baike*, only 11% articles have *infobox*, which makes the attribute quite sparse.

5.2 Table Classification Evaluation

In order to evaluate table classification, we randomly selected 60 entities in 12 different fields (e.g. Nature, Culture, Art, Economy, Science, etc.), and extracted 229 *wikitables* from the corresponding articles. In these tables, we found 86 of them are *Non-genuine Table*s, 93 of them are *Genuine Table*s, including 50 *Genuine Table with Header*. All these tables are used as training data, and then we also randomly picked up 300 tables in random fields as the test data, where each category has about 1/3 tables.

We trained four classifiers, including *Naive Bayes*(NB), *Decision Tree J48*(J48), *Logistic Classifier*(LC), *Random Forest*(RF), with different feature composition, and the results are shown in Table 3.

From the table, we can find that (1) all features are useful for table classification, since the classifiers can reach the best in F-measure when using all features. (2) Random Forest performs the best, so it is selected as our table classifier.

Finally, we classified all the 757,282 tables, and obtained 441,500 *Genuine Tables* and 315,782 *Non-genuine Table*s. The *Genuine Tables* consist of 71,981 *Genuine Table with Header* and 369,519 *Genuine Table without Header*.

5.3 RDF Extraction Evaluation

When *Genuine Tables* are obtained, RDF triples could be extracted from them. However, there are no similar works in extracting RDF triples from tables in Chinese encyclopedias. We compared our approach to the following methods with some modification to fit the Chinese text.

- **Table Classification.** [22] mainly classifies tables into *Genuine Table* and *Non-genuine Table*, which didn't distinguish the tables with or without header row. So we firstly classify the tables into these two categories as in [22], and then used header detector in [21] to locate headers in tables. However, [21] used Probase [25] for table understanding, which has little knowledge in Chinese. So we replace the Probase with our knowledge base. We denote this method as tc_c, and our table classification as tc_f.
- **RDF Mining.** [14] used DBPedia to mine RDF from Wikipedia' tables directly, and it didn't need to detect subject column and object columns in advance. However, DBPedia also has limited knowledge in Chinese, so it is not suitable for RDF mining from tables in Chinese encyclopedias. We tried to mine the predicate between two cells from different columns by using our knowledge base, and then extracted same features as in [14] for classifiers to predict whether the triples are correct or incorrect. We denote this method as tm_c, and our RDF mining with subject column and object column detection as tm_f.

In addition, we also built a raw knowledge base for comparison from the category system and *infobox* without any incorrect relation removal as in Section 3. This knowledge base is denoted by kb_c, and our knowledge base with refinement is denote by kb_f. Therefore, we can have different combinations for these

methods. In Table 4, we show the total number of RDF triples extracted from tables (denoted as *tNo.* in the table) and the number of new triples (denoted as *nNo.*) which don't appear in the *infobox*s with different methods. That is to say, theses new triples could be used to populate the knowledge base. In addition, we randomly selected 500 triples from the extracted triples for human evaluation, and calculated the accuracy of triples for each method.

Table 4. The performance of different combination of methods, where tc_c, tm_c and kb_c are the table classification, triple mining and knowledge base for comparison, while tc_f, tm_f and kb_f are our approaches for table classification (with ternary classification), triple mining (with subject column and object column detection) and knowledge base (with taxonomy and class-attribute refinement)

Method	*tNo.*	*nNo.*	*Acc*	Method	*tNo.*	*nNo.*	*Acc*
$tc_c + tm_f + kb_c$	3,907,243	1,499,608	67.8%	$tc_c + tm_c + kb_f$	3,395,423	1,314,791	74.3%
$tc_c + tm_f + kb_f$	3,002,028	1,215,170	80.0%	$tc_f + tm_f + kb_c$	3,617,509	1,332,534	76.4%
$tc_f + tm_c + kb_f$	3,056,287	1,140,470	81.8%	$tc_f + tm_f + kb_f$	2,787,027	1,053,407	**90.2%**

From the table, we find that although the number of triples extracted by our approach is the smallest, our approach reaches the best performance with the largest accuracy. Moreover, the improvement in table classification, triple mining and knowledge base construction are all helpful to promote the final performance.

6 Related Work

Our work is most related to knowledge base construction and web table understanding.

6.1 Knowledge Base Construction

Several knowledge bases have been built from Web sources, such as DBpedia [3], YAGO [18], Freebase [4], NELL [5], Probase [25], and Knowledge Vault [8].

DBpedia [3] mainly utilized the Wikipedia for automatic construction of large knowledge bases. YAGO [18] links WordNet [17] and Wikipedia to form a large and extendable knowledge base by using the taxonomy from WordNet and facts from Wikipedia. Probase [25] can automatically inference an open-domain, probabilistic taxonomy from the entire web with an iterative learning algorithm. NELL [5] aims to build a never-ending language learner to iteratively promote beliefs in knowledge base through a semi-supervised learning method. Knowledge Vault [8] created a much bigger knowledge base by fusing together multiple extraction sources with prior knowledge derived from an existing KB.

However, all these existing knowledge bases contain very limited knowledge in Chinese, so we have to create a Chinese knowledge base from scratch. Recently, several works about the construction of Chinese knowledge base are proposed, such as

Zhishi.me [16] and XLore [24], but they can not be used for our problem directly. Zhishi.me focused on the infobox information extraction and Chinese LOD construction. XLore can utilize a classification-based method to correctly semantify the wikis'category systems, but it doesn't mine the probabilities of the relations.

6.2 Web Table Understanding

Many works have been proposed for web table understanding, including table classification [6,9,10], table annotation [7,11,12,20,29], knowledge population from tables [13,14,21,26,28], and more advanced applications involving web tables, such as table search, fact search engine [27], search join [1,2] and table summarization [15].

Our work is mainly related to table annotation and knowledge population from tables. Table Miner [29] annotated Web tables by using an incremental, bootstrapping learning approach seeded by automatically selected partial content from tables. [20] leveraged a database of class labels and relationships which are automatically extracted from the Web to recover semantics of tables. [12] annotated table cells with entities, table columns with types, and relations that pairs of table columns simultaneously through a graphical model. InfoGather [26] tries to promote information gathering tasks by considering both indirectly and directly matching tables. Furthermore, InfoGather+ [28] was proposed with a probabilistic graphical model to discover the semantic labels of columns and semantic matches between columns over all web tables collectively.

Our work is similar to [21] and [14]. However, [21] used Probase [25] for table understanding, which has little knowledge in Chinese, and [14] used DBPedia to mine RDF from Wikipedia' tables directly, without detecting subject column and object columns.

7 Conclusion

In this paper, we proposed an approach to mine RDF triples from tables in Chinese encyclopedias for knowledge base population. Since there is no knowledge base like DBpedia and Probase to help table understanding, we constructed a Chinese knowledge base from scratch through taxonomy mining and class attribute mining at first. Then, we extracted RDF triples from tables with the following steps: column scoring, table classification and RDF extraction.

In future, we would like to evaluate the performance improvement with the iteration between knowledge population and table understanding. In addition, we would also like to introduce crowdsourcing technologies [11] to promote the table understanding.

Acknowledgments. This work is supported by the Fundamental Research Funds for the Central Universities(2014QNA5008), Chinese Knowledge Center of Engineering Science and Technology(CKCEST), and the National Natural Science Foundation of China (No.61103099).

References

1. Bhagavatula, C.S., Noraset, T., Downey, D.: Methods for exploring and mining tables on wikipedia. In: Proceedings of the ACM SIGKDD Workshop on Interactive Data Exploration and Analytics, pp. 18–26. ACM (2013)
2. Bizer, C.: Search joins with the web. In: ICDT, p. 3 (2014)
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. ACM (2008)
5. Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka Jr., E.R., Mitchell, T.M.: Toward an architecture for never-ending language learning. In: AAAI, vol. 5, p. 3 (2010)
6. Crestan, E., Pantel, P.: Web-scale table census and classification. In: WSDM, pp. 545–554. ACM (2011)
7. Deng, D., Jiang, Y., Li, G., Li, J., Yu, C.: Scalable column concept determination for web tables using large knowledge bases. Proceedings of the VLDB Endowment **6**(13), 1606–1617 (2013)
8. Dong, X., Gabrilovich, E., Heitz, G., Horn, W., Lao, N., Murphy, K., Strohmann, T., Sun, S., Zhang, W.: Knowledge vault: a web-scale approach to probabilistic knowledge fusion. In: SIGKDD, pp. 601–610. ACM (2014)
9. Fang, J., Mitra, P., Tang, Z., Giles, C.L.: Table header detection and classification. In: AAAI (2012)
10. Lautert, L.R., Scheidt, M.M., Dorneles, C.F.: Web table taxonomy and formalization. ACM SIGMOD Record **42**(3), 28–33 (2013)
11. Li, G.: A human-machine method for web table understanding. In: Wang, J., Xiong, H., Ishikawa, Y., Xu, J., Zhou, J. (eds.) WAIM 2013. LNCS, vol. 7923, pp. 179–189. Springer, Heidelberg (2013)
12. Limaye, G., Sarawagi, S., Chakrabarti, S.: Annotating and searching web tables using entities, types and relationships. Proceedings of the VLDB Endowment **3**(1–2), 1338–1347 (2010)
13. Mulwad, V., Finin, T., Joshi, A.: Automatically generating government linked data from tables. In: Working Notes of AAAI Fall Symposium on Open Government Knowledge: AI Opportunities and Challenges, vol. 4 (2011)
14. Muñoz, E., Hogan, A., Mileo, A.: Using linked data to mine rdf from wikipedia's tables. In: WSDM, pp. 533–542. ACM (2014)
15. Nguyen, T.T., Nguyen, Q.V.H., Weidlich, M., Aberer, K.: Result selection and summarization for web table search. In: ICDE (2015)
16. Niu, X., Sun, X., Wang, H., Rong, S., Qi, G., Yu, Y.: Zhishi.me - weaving chinese linking open data. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part II. LNCS, vol. 7032, pp. 205–220. Springer, Heidelberg (2011)
17. Stark, M.M., Riesenfeld, R.F.: Wordnet: an electronic lexical database. In: Proceedings of 11th Eurographics Workshop on Rendering, vol. 37. MIT Press (1998)
18. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: a core of semantic knowledge. In: WWW, pp. 697–706. ACM (2007)
19. Tong, H., Faloutsos, C., Pan, J.Y.: Fast random walk with restart and its applications. In: ICDM, pp. 613–622. IEEE Computer Society (2006)

20. Venetis, P., Halevy, A., Madhavan, J., Paşca, M., Shen, W., Wu, F., Miao, G., Wu, C.: Recovering semantics of tables on the web. Proceedings of the VLDB Endowment **4**(9), 528–538 (2011)
21. Wang, J., Wang, H., Wang, Z., Zhu, K.Q.: Understanding tables on the web. In: Atzeni, P., Cheung, D., Ram, S. (eds.) ER 2012 Main Conference 2012. LNCS, vol. 7532, pp. 141–155. Springer, Heidelberg (2012)
22. Wang, Y., Hu, J.: A machine learning based approach for table detection on the web. In: WWW, pp. 242–250. ACM (2002)
23. Wang, Z., Li, J., Li, S., Li, M., Tang, J., Zhang, K., Zhang, K.: Cross-lingual knowledge validation based taxonomy derivation from heterogeneous online wikis. In: AAAI (2014)
24. Wang, Z., Li, J., Wang, Z., Li, S., Li, M., Zhang, D., Shi, Y., Liu, Y., Zhang, P., Tang, J.: Xlore: a large-scale english-chinese bilingual knowledge graph. In: International Semantic Web Conference (Posters & Demos), pp. 121–124 (2013)
25. Wu, W., Li, H., Wang, H., Zhu, K.Q.: Probase: a probabilistic taxonomy for text understanding. In: SIGMOD, pp. 481–492. ACM (2012)
26. Yakout, M., Ganjam, K., Chakrabarti, K., Chaudhuri, S.: Infogather: entity augmentation and attribute discovery by holistic matching with web tables. In: SIGMOD, pp. 97–108. ACM (2012)
27. Yin, X., Tan, W., Liu, C.: Facto: a fact lookup engine based on web tables. In: WWW, pp. 507–516. ACM (2011)
28. Zhang, M., Chakrabarti, K.: Infogather+: semantic matching and annotation of numeric and time-varying attributes in web tables. In: SIGMOD, pp. 145–156. ACM (2013)
29. Zhang, Z.: Start small, build complete: Effective and efficient semantic table interpretation using tableminer. Under Transparent Review: The Semantic Web Journal (2014)