Automatic answer ranking based on sememe vector in KBQA[1]

Yadi Li, Lingling Mu, Hao Li, Hongying Zan
School of Information Engineering, Zhengzhou University, Zhengzhou, P.R.China
1837186472@qq.com; iellmu@zzu.edu.cn

***Abstract*—This paper proposes an answer ranking method used in Knowledge Base Question Answering (KBQA) system. This method first extracts the features of predicate sequence similarity based on sememe vector, predicates' edit distances, predicates' word co-occurrences and classification. Then the above features are used as inputs of the ranking learning algorithm Ranking SVM to rank the candidate answers. In this paper, the experimental results on the data set of KBQA system evaluation task in the 2016 Natural Language Processing & Chinese Computing (NLPCC 2016) show that, the method of word similarity calculation based on sememe vector has better results than the method based on word2vec. Its accuracy, recall rate and average F1 value respectively are 73.88%, 82.29% and 75.88%. The above results show that the word representation with knowledge has import effect on natural language processing.**

Keywords-KBQA; word similarity; Ranking SVM; sememe vector

I. INTRODUCTION

The automatic Question Answering (QA) system is the basic form of the next generation search engine[1], and its main features are as follows: (1)Users' inputs are natural language questions. (2)The returned answer is no longer the form of sorted documents, but a direct answer.

The basement of the automatic QA is to deeply understand the natural language questions of the users and extract the meanings from them, and then the corresponding answers are obtained according to the users' questions. KBQA is one of the forms of automatic QA, which generates answers from knowledge base. An example of question mapping and candidate answers extraction is shown in Table I.

TABLE I. The example of question mapping and candidate answers extraction

The "entity-predicate" mapping of the question	轴承(bearing)--直径(diameter) 多少(how many)
The set of corresponding candidate answers	轴承 ||| OD (外径 outside diameter) ||| 170mm 轴承 ||| d (内径 inside diameter) ||| 80mm 轴承 ||| 外文名(English name) ||| bearing

At present, there are some available large-scale knowledge bases, such as DBpedia[2], Freebase[3] and YAGO[4], etc. Most of these knowledge bases are graph structures composed of "*entity-relationship-entity*" triples as the basic units. Therefore, transforming natural language questions into structured queries becomes the basic work of searching answers in the knowledge bases.

With the development of natural language processing technology, there are mainly two forms of solutions to the KBQA. One is the method based on semantic analysis, which uses the method of semantic analysis to analyze questions and convert natural language questions into more advanced expressions or standard question statements. The other is based on the method of information extraction, which firstly extracts the key topic entities and relationships in the questions. Then it searches the candidate answer entities in the knowledge bases that are related to the key entities. Finally, it chooses the most possible answer by calculating the similarities or correlations between the candidate answers and the questions.

Word similarity is often used in KBQA. When the predicates sequences in the questions are different from that in the knowledge base, it is necessary to sort the candidate answers in the knowledge base by calculating the similarities between questions' predicates and candidate answers' predicates. The word similarity can be simply calculated by the word vectors that can be trained by Word2vec[5] and Glove[6] and so on. Word2vec and Glove use unsupervised learning method to train word vectors based on a huge amount of unlabeled data. However, they ignore the knowledge information and sometimes cannot correctly distinguish the semantic difference.

The method proposed by this paper is based on the method of information extraction. Firstly, we extract the entities and relationships in the questions. Then, we extract all candidate answers that contain the questions' entities in the knowledge base. Finally, we use the Ranking SVM algorithm with four features to select the most possible answer. This paper uses sememe vector to calculate the word similarity and get a better result on limited training data.

II. RELATED WORK

The research of KBQA has a long history in natural language processing. The common techniques for KBQA are the methods based on semantic analysis, feature-driven and representation learning[7].

The method based on semantic analysis maps questions in natural language forms to semantically equivalent logical expressions through certain grammars[8]. Cai et al.[9] used the purely supervised learning method to train a semantic analyzer. They developed a matching algorithm to find the word-related labels in the knowledge base and established a lexicon extender. The lexicon extender linked the words to the relevant labels in the knowledge base to complete the semantic analyzer learning. However, this method still had some problems. It cannot get rid of the dependence on manual annotation, and it requires higher

[1] The authors were supported financially by the National Social Science Fund of China (18ZDA315), Programs for Science and Technology Development in Henan province (No.192102210260) and the Key Scientific Research Program of Higher Education of Henan (No.20A520038).

accuracy of word-related labels in the lexicon extender. Kwiat-kowski et al.[10] proposed a solution based on ontology matching, which is independent of artificial word triggers. Yao et al.[11] used question words, question intentions and entity types to construct the questions' feature graphs. For each edge e(s, t) in the graphs, "e, s, s|t, s|e|t" are extracted as the features of questions to search for answers. Lai et al.[12] combined the predicate sequence similarity with the entities' lengths and the frequency features of the answers' templates to rank the candidate answers and select the most possible answer. Kun Xu et al.[13] selected three types of features in the QA system based on the freebase: the entities' scores based on entities linking technology, the predicates' scores based on Convolutional Neural Network (CNN), the co-occurrences of the answer types and the question words. They used the machine learning algorithm to find answers with three types of features.

This paper proposes an answer ranking method used in KBQA system, which combines the features of predicates' word similarity, edit distances, word co-occurrences and classification with the ranking SVM algorithm to rank the candidate answers.

III. WORD SIMILARITY BASED ON SEMEME VECTOR

In HowNet[2], there are various relationships among sememes. The upper and lower relationships organize all sememes in a hierarchical graph. We can use the upper and lower position of sememes and combine the PageRank[14] algorithm to represent the sememes as vectors. Then the sememe similarity is calculated by the sememe vector, the concept similarity is calculated by the sememe similarity. Finally, the word similarity is calculated by concept similarity.

A. Sememe Information Content (SIC)

The SIC refers to the size of information contained in a sememe itself. It is an important feature to distinguish the degree of difference between sememes. If SIC is very close, the similarity between sememes is higher. In the sememe structure graphs, the structural information content of the sememe nodes is mainly considered. The deep structure information of the sememe nodes is an important feature of SIC. In order to further refine the differences among sememes, the layers in which the descendants' nodes are located should be considered. Therefore, we can calculate the SIC by using the sememe structures according to the method of reference[15]. The calculation formula is the following formula (1).

$$SIC_s = \frac{\log(deep(s)+1)}{\log(deep_{max}+1)} \times \left(1 - \frac{\log(\sum_{\alpha \in hypo(s)} \frac{1}{deep(\alpha)} + 1)}{\log(node_{max})}\right) \quad (1)$$

In the sememe structure graph, *deep(s)* represents the layers of the sememe node *s*. The root node *ROOT* is the first layer, $deep_{max}$ is the depth of the layer where the last layer is located, α is the descendant node of the sememe *s*, *hypo(s)* is the set of all the descendants of sememe *s*, $node_{max}$ is the total number of sememe nodes.

2http://www.keenage.com/

B. The Representation of Sememe Vector

In the structure graph of sememes, the transfer probability matrix M is constructed by SIC.

M is defined as a matrix of N×N. N is the total number of sememe nodes. All nodes is traversed in the sememe structure graph, and each element in M is assigned a value according to the formula (2).

$$M_{ji} = \begin{cases} \frac{SIC_j}{\sum_{k \in Out(i)} SIC_k}, & \text{If i and j have a connection} \\ 0, & \text{else} \end{cases} \quad (2)$$

Where *Out(i)* is a set of all nodes connected to the sememe node *i*, SIC represents the information content of the sememe nodes.

Each sememe is represented as a vector based on the transition probability matrix M and PageRank algorithm. The sememe vectors are calculated by the following formulas：

$$\overrightarrow{p_s} = cM \cdot \overrightarrow{p_s} + (1-c)\overrightarrow{v_s} \quad (3)$$

$$\overrightarrow{v_s}(j) = \begin{cases} \frac{SIC_j}{\sum_{k \in Out(s)} SIC_k}, & \text{If s and j have a connection} \\ 0, & \text{else} \end{cases} \quad (4)$$

Where *c* is the damping coefficient between 0 and 1, *M* is the transition probability matrix that represents the SIC of each sememes. The vector $\overrightarrow{p_s}$ is the vector representation of the sememe *s*, its dimension is *N* and initial value is $\{\frac{1}{N}, \frac{1}{N} \ldots, \frac{1}{N}\}$. The value of $\overrightarrow{p_s}$ is changed during the iterations until it converges. The vector $\overrightarrow{v_s}$ has N dimensions.

C. The Calculation of Word Similarity

After representing the sememes as vectors, the similarities between the sememes are calculated by the cosine similarity of the vectors. According to the description structures of word concept based on sememes in HowNet, the four parts of the sememe set similarity are calculated by the sememe similarity, and then the concept similarity is calculated. We take the maximum similarity among two concepts as the word similarity[15]. The specific calculation is described as follows:

Formulas (5)-(6) are used to calculate the similarity of sememe *i* and *j*.

$$dis(i,j) = \cos(\overrightarrow{P_i}, \overrightarrow{P_j}) \quad (5)$$

$$sim_s(i,j) = \frac{dis(i,j)}{\alpha + dis(i,j)} \quad (6)$$

Where α is an adjustable parameter.

The different semantics of words are described by various concepts, so sometimes there are many similarities between the concepts. It is necessary to calculate the similarities among two concepts separately.

The concept similarity of concepts C1 and C2 consists of four parts: the first basic sememe similarity $sim_1(s_1,s_2)$,

the other basic sememe similarity $sim_2(s_1,s_2)$, the relational sememe similarity $sim_3(s_1,s_2)$, the symbolic sememe similarity $sim_4(s_1,s_2)$. Therefore, the concept similarity between the two concepts is calculated by the formula (7):

$$sim_c(c_1,c_2)=\sum_{i=1}^{4}\beta_i\prod_{j=1}^{i}sim_j(s_1,s_2) \quad (7)$$

Where β_i is an adjustable parameter.

Suppose the word w_1 has m concept descriptions: $c_{11},c_{12},...c_{1m}$, and the word w_2 has n concept descriptions: $c_{21},c_{22},...c_{2n}$. Then the maximum value is taken as the word similarity by the reference[16]

$$sim_w(w_1,w_2)=\max_{i=1....m,\ j=1...n}(sim_c c_{1i},c_{2j}) \quad (8)$$

IV. ANSWER RANKING METHODS IN KBQA

The answers of KBQA are from a structured knowledge base, which usually exist in the form of triples. The structures of the knowledge base have various triple forms: "*entity-relationship-entity*" and "*entity-attribute-attribute value*", etc. In the knowledge base, this paper takes the "*relationship*" and "*attribute*" as "*predicate*", and each triple is called "*assertion*".

The process of the KBQA system is divided into three steps: questions analysis, candidate answers extraction, and answers sorting. The questions analysis phase identifies the entities in the questions and maps the questions to the structured forms of "*entities-predicates*". The candidate answers extraction phase searches the triple of "*entity-predicate-entity*" in the knowledge base according to the entities in the questions, and extracts the structural assertions of the corresponding entities as the set of candidate answers. After completing the questions analysis and candidate answers extraction, the candidate answers are sorted according to the predicate features of "*entity-predicate*" and "*entity-predicate-entity*", and the highest score answer is returned.

The process of KBQA system is shown in Figure 1.

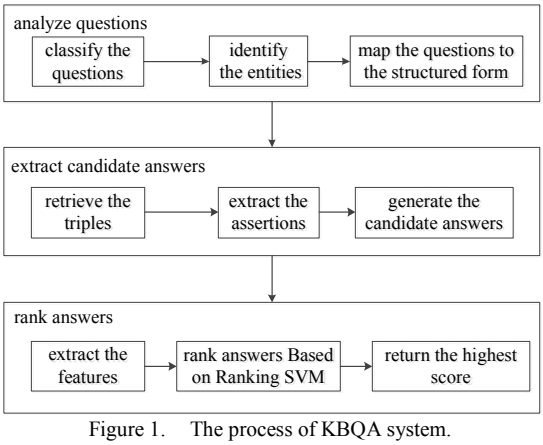

Figure 1. The process of KBQA system.

A. Features Extraction

In this paper，the four features that extracted to rank the candidate answers are the predicate sequence similarity based on sememe vector, predicates' edit distances, predicates' word co-occurrences and classification.

1) Predicate sequence similarity based on sememe vector.

The words in the candidate answers' predicates mostly have fixed collocations, such as "出品 公司(Produce Company)", "效力 球队(Serve Team)", etc. When these words are used in sentences, their positions are usually adjacent, such as "《卧虎藏龙》是由哪个公司出品的? (Which company produced 'Crouching Tiger, Hidden Dragon'?)", "李明曾经效力于什么球队? (What team did Li Ming ever serve for?)". Therefore, after removing the stop words in the questions, these words are usually in adjacent positions. When calculating the similarities between the questions' predicates and the candidate answers' predicates, this paper uses the local contiguous subsequences of questions' predicates and the sequences of candidate answers' predicates to calculate the similarity, The word similarity calculation is based on the method of sememe vector described in section III.

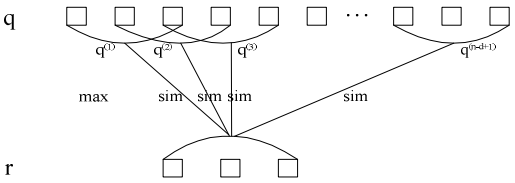

Figure 2. The model of predicate similarity calculation.

In this paper, the set of predicates' word sequences in questions is called *q*. The number of words in *q* is *n*. The set of predicates' word sequences in the candidate answers is called *r*. The number of words in *r* is *d*. $q^{(i)}$ is the local word sequence in *q*, it has a continuous length *d and* starts with the i-th word. The model of predicate similarity calculation is shown in Figure 2. We can calculate the similarity between all local word sequences and *r*, and then we take the maximum value as the predicate sequence similarity of *q* and *r*, such as the formula (9).

$$sim(q,r)=\max_{i=1,...,n-d+1}(sim(q^{(i)},r)) \quad (9)$$

When calculating the similarity between the local word sequences $q^{(i)}$ and *r*, the similarity of each word in $q^{(i)}$ and all words in *r* is calculated separately, then the maximum values are taken. The sum of the maximum values is averaged as the similarity of $q^{(i)}$ and *r*, such as the formula(10).

$$sim(q^{(i)},r)=\frac{\sum_{j=1}^{d}\max_{m}sim(q_j^{(i)},r_m)}{d} \quad (10)$$

Where $q_j^{(i)}$ is the j-th word in the subsequence $q^{(i)}$, r_m is the m-th word in *r*, $\max_{m}sim(q_j^{(i)},r_m)$ is the maximum value of all words' similarities in $q_j^{(i)}$ and *r*, and *d* is the number of words in $q^{(i)}$.

When the similarity is calculated based on the local subsequences, the length of the questions' predicate *q* may be smaller than the length of the candidate answers' predicate *r*. At this time, the similarity of the entire *q* and *r* is directly calculated, such as the formula (11).

$$\mathrm{sim}(q,r)=\frac{\sum_{k=1}^{n}\max_{m}\mathrm{sim}(q_k,r_m)}{n} \quad (11)$$

Where q_k is the k-th word in q, r_m is the m-th word in r, $\max_{m}\mathrm{sim}(q_k,r_m)$ is the maximum value of all words' similarities in q_k and r, and n is the number of words in q.

2) Predicates' edit distances.

By analyzing the questions' predicates and the candidate answers' predicates, the smaller the edit distances between them, the smaller the differences between the predicates. Therefore, the edit distances can also be considered as a feature of the candidate answers ranking. The edit distances represent the minimum number of times that a string needs to be edited to another string. This process includes insertion, deletion, replacement and other operations. The edit distance is calculated as follows:

For the two strings of "发行公司(Publishing Company)" and "发行商(Publisher)", we can align the string "发行(Publish)". Then the string "公(Gong)" is replaced with the string "商(Shang)" and the string "司(Si)" is deleted. After the above operations, the "发行公司(Publishing Company)" is converted into "发行商(Publisher)". Therefore, the edit distance between them is 2.

In this paper, the reciprocal of the editing distance d is selected as a ranking feature, such as the formula (12).

$$d'=\begin{cases}\frac{1}{d}, & \text{若 } d\neq 0\\ 1, & \text{若 } d=0\end{cases} \quad (12)$$

3) Predicates' word co-occurrences.

Based on statistical thoughts, the co-occurrences of words in questions and answers are also an important factor in the answers matching. If the words in the questions and the candidate answers often co-occur in the question-answer pairs, the probability that the candidate answers are the correct answers is higher.

In the training data, we need to count the number of co-occurrences of each word in the questions' predicates and the candidate answers' predicates. If the co-occurrences are higher, it means that when some words appear in the questions, the corresponding words tend to appear in the candidate answers' predicates. For example: "时候(moment)" and "时间(time)", "多少(how many)" and "数(number)".

Suppose the words sequence of the questions' predicates is q and the words sequence of the candidate answers' predicates is r, l_1 and l_2 are the number of words they contained respectively, then the co-occurrences feature of the two predicates is shown as formula (13).

$$f=\frac{\sum_{i=1}^{l_1}\sum_{j=1}^{l_2}co(q_i,r_j)}{l_1\cdot co_{\max}} \quad (13)$$

Where $co(q_i,r_j)$ is the number of co-occurrences of the words q_i and r_j in the questions and candidate answers of the training data, and $co_{\max}$ is the maximum number of co-occurrences of all words.

4) Classification feature.

The feature of classification refers to the probability of consistency between the category of each candidate answer and question. Classifying the questions can narrow the search range of the candidate answers. At the same time, it can improve the efficiency and accuracy of returning answers. If the types of candidate answers and the questions are consistent, the probability of being the correct answers is higher.

This paper respectively trains two Maximum Entropy (ME) models for questions and candidate answers to obtain the types of questions and candidate answers. The classification categories include "description, person, place, number, institution name, entity, time and others". If a question description is "where is XXX?", the categories of this question and answer are all "place category". Similarly, if a question description is "when is XXX?", the categories of this question and answer are all "time category". The process of obtaining candidate answers' classification features is shown as Figure 3.

Suppose there are m questions and answers. Where a_i is the i-th candidate answer; q_j is the j-th question; C is the most probable category for the j-th question. $P(a_i|C_k)$ is the probability of the candidate answer a_i under each category C_k (k=1,2,...m). $P(a_i|C)$ is the probability of the candidate answer a_i under the category C.

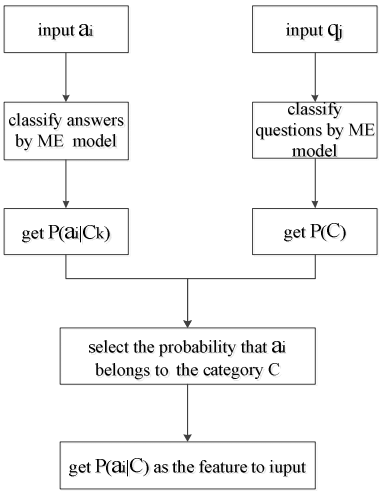

Figure 3. The acquisition process of candidate answers' classification features.

As is shown in Figure 3, the ME model of questions' classification is used to get the category C. The ME model of answers' classification is used to get the probability $P(a_i|C_k)$. Finally, we select the set of $P(a_i|C)$ as the classification features of the candidate answers.

B. Ranking Answers Based on Ranking SVM

The Ranking SVM[3] algorithm is a ranking algorithm proposed by Herbrich et al. The four types of features extracted above in this paper are used as inputs of the Ranking SVM. The training data is used to train the model parameters, and the ranking learning model is used to give

[3]http://www.cs.cornell.edu/people/tj/svm_light/svm_rank.html

each candidate answers a score, then we can rank answers according to the scores.

V. Experiments and Results Analysis

A. Dataset

This paper selects the KBQA's evaluation data in NLPCC 2016 as the experimental data. It contains 14,609 QA pairs as training data and 9,870 questions as test data. It also contains a structured knowledge base with a total of 47,943,429 "*entity-predicate-entity*" triples, and there are 8,721,640 entities.

B. Experimental Data Preprocessing

By observing the training data, phrases such as "我想知道(I want to know)", "我很好奇(I'm curious)", "谁能告诉我(Who can tell me)", "有谁知道(Who knows)", "什么是(What is)" appear in most questions, the original meaning of the questions is not affected after these phrases are removed. Therefore, data preprocessing needs to remove these phrases from questions.

Entity recognition is the primary task of question analysis, so we need to find out the entities contained in the questions. This paper uses pattern matching to identify the entities in the questions according to the structural features of knowledge base and the syntactic features of simple questions. After identifying the entities, the questions' remaining words are used to perform words segmentation and stop words removal. Then the questions can be mapped to the form of "*entity-predicate*".

Here is an example, for the question "龙泉镇在中国的哪个地方?(Where is Longquan Town in China?)", the "龙泉镇(Longquan Town)" and "中国(China)" are entities that successfully match the entities sets. After identifying "龙泉镇(Longquan Town)", the remaining words "在中国的哪个地方? (Where is it in China?)" are used to perform words segmentation and stop words removal. Therefore, the question can be mapped to the structured form of "龙泉镇(Longquan Town)— 中国(China)　地方(Place)". In the same way, after identifying "中国(China)", the question can be mapped to the form of "中国(China)— 龙泉镇(Longquan Town)　地方(Place)".

After mapping the questions to the form of "entity-predicate", the entities are searched in the knowledge base. Then the corresponding assertions of the entities are extracted as the candidate answers of the questions.

This paper uses the pattern matching to identify the entities in the questions. The candidate answer triples are extracted from the knowledge base according to the entities.

C. Evaluation Indicators

In this paper, the average accuracy rate (Precision), average recall rate (Recall), average MRR and average F1 value are used as evaluation indicators (formulas14-18).

$$\text{Precision} = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{\#(C_i, A_i)}{|Ci|}\times 100\% \quad (14)$$

$$\text{Recall} = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{\#(C_i, A_i)}{|\text{A}i|}\times 100\% \quad (15)$$

$$\text{MRR} = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{\text{rank}_i} \quad (16)$$

$$\text{AveragedF1} = \frac{1}{|Q|}\sum_{i=1}^{|Q|}F_i\times 100\% \quad (17)$$

For the i-th question, C_i represents the set of answers generated by the experimental system, $|C_i|$ is the number of answers in C_i, A_i is the standard answer set, $|A_i|$ is the number of answers in A_i, $\#(C_i, A_i)$ is the number of the same answers in C_i and A_i, $|Q|$ is the total number of test questions, $rank_i$ is the position of the correct answer in the candidate answers set C_i of the i-th question. If C_i does not contain the correct answer, the value of MRR is 0.

$$\text{F}_i = \frac{2\cdot\frac{\#(C_i, A_i)}{|Ci|}\cdot\frac{\#(C_i, A_i)}{|Ai|}}{\frac{\#(C_i, A_i)}{|Ci|}+\frac{\#(C_i, A_i)}{|Ai|}} \quad (18)$$

F_i is the F1 value of the i-th question.

D. Experimental Design

HowNet is a finite semantic dictionary and there are OOV words. We used the following methods to calculate the word similarity.

1) The method based on sememe vector.

For words that do not exist in HowNet, the word similarity is defined as 0; otherwise, the calculation of word similarity is described in section III.

2) The method based on word2vec.

The word2vec[4] is used as word vector model. The corpus is the People's Daily standard corpus in 1998. According to the experiment，the context window is 5, the word vector dimension is 200, and the word similarity is calculated by the cosine distances. For words do not exist in the corpus, the similarity is defined as 0.

3) The method based on the combination of sememe vector and word2vec.

For the words that exist in HowNet, we use the sememe vector to calculate the word similarity. Otherwise, we use the word2vec to calculate. For words not exist in HowNet and the corpus, the similarity is defined as 0.

This paper conducts two groups of experiments to verify the effect of word similarity. The experiments in the first group use the feature of predicate sequence similarity to rank the candidate answers, which is called the KBQA based on single feature. The experiments in the second group combine the Ranking SVM algorithm with the features of predicate sequence similarity, predicates' edit distances, predicates' word co-occurrences and classification to rank the candidate answers, which is called the KBQA based on multi-features.

E. Experimental Results and Analysis

1) Results and Analysis of KBQA Based on Single Feature.

In this group of experiments, only the feature of predicate sequence similarity is used to rank the candidate answers. The candidate answers with the highest ranking

[4] http://word2vec.googlecode.com/svn/trunk/

are returned. This paper uses three methods to calculate the predicate sequence similarity. The process is described in section V.D. The results of these three methods applied in the KBQA are shown in TABLE II.

TABLE II. The experimental results based on the feature of predicate sequence similarity

Method	AveF1	Precision	Recall	MRR
sememe vector	71.77%	69.64%	78.80%	0.7311
word2vec	70.60%	68.70%	76.55%	0.7175
sememe vector +word2vec	71.81%	69.70%	78.83%	0.7314

As can be seen from Table II, only the predicate similarity is used to sort the candidate answers, the F1 values of the obtained results are all over 70%, and the recall rate is also more than 76%. The above results show that the word similarity calculation has a better effect on the answer retrieval of the KBQA. In addition, the word similarity calculated by the combination of the sememe vector and word2vec is the most effective, the method based on sememe vector is the middle, and the method based on word2vec is the lowest. Therefore, the KBQA experiment with the combination of the two methods is the best; the result is higher than the other methods in four evaluation indicators.

2) Results and Analysis of KBQA Based on Multi-features.

In this group of experiments, when ranking the candidate answers, we use the ranking learning algorithm Ranking SVM combined with predicate similarity, edit distances, word co-occurrences, and classification features to select the top answers. The three calculation methods of word similarity are applied to the KBQA experiment, and the results are shown in Table III.

TABLE III. The experimental results based on multi-features

Method	AveF1	Precision	Recall	MRR
sememe vector	75.88%	73.88%	82.29%	0.7749
word2vec	74.12%	72.35%	79.52%	0.7518
sememe vector +word2vec	75.91%	73.93%	82.32%	0.7723

As can be seen from Table III, the method based on the sememe vector is higher than the method based on the word2vec in the four evaluation indicators. The features of edit distances, word co-occurrences, and classification are the same. Only the feature of predicate sequence similarity has impact on evaluation indicators. The differences of the predicate sequence similarity feature are mainly due to the different methods to calculate the word similarity. It can be seen from the experimental results that the method based on the sememe vector has achieved good results in the application of the KBQA system.

In the QA system experiments, this paper only selects the candidate answers with the highest score to return. The recall rate based on the method of sememe vector is 82.29%, which shows that 8,123 questions get the correct answers among the 9,870 questions.

VI. CONCLUSION

This paper proposes a method of ranking candidate answers based on sememe vector. This method combines the features of predicates' word similarity in the the questions and the candidate answers, edit distances, word co-occurrences and classification with the ranking SVM algorithm to rank the candidate answers of the questions and select the answers with the highest score.

The experimental results show that the features of the candidate answers ranking obtained in this paper have good results. The recall rate of word representation based on sememe vector is 82% in the KBQA system. This shows that the language knowledge base plays an important role in the word representation.

REFERENCES

[1] Etzioni O. Search needs a shake-up. Nature, 2011, 476(7358): 25−26.

[2] Lehmann J, Isele R, Jakob M, et al. DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia [J]. Semantic Web, 2015, 6(2): 167-195.

[3] Bollacker K, Evans C, Paritosh P, Sturge T, Taylor J. Freebase:a collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACMSIGMOD International Conference on Management of Data.Vancouver, Canada: ACM, 2008. 1247−1250.

[4] Suchanek F M, Kasneci G, Weikum G. Yago: a core of semantic knowledge. In: Proceedings of the 16th International Conference on World Wide Web. Alberta, Canada: ACM, 2007. 697−706.

[5] Church K W. Word2Vec[J]. Natural Language Engineering, 2017, 23(1): 155-162.

[6] Pennington J, Socher R, Manning C. Glove: Global vectors for word representation[C]//Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014: 1532-1543.

[7] Liu K, Zhang Y Z, Guoliang J I, et al. Representation Learning for Question Answering over Knowledge Base: An Overview[J]. Acta Automatica Sinica, 2016.

[8] Liang P, Jordan M I, Klein D. Learning dependency-based compositional semantics[J]. Computational Linguistics, 2013, 39(2): 389-446.

[9] Berant J, Chou A, Frostig R, et al. Semantic parsing on freebase from question-answer pairs[C]//Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. 2013: 1533-1544.

[10] Kwiatkowski T, Choi E, Artzi Y, et al. Scaling semantic parsers with on-the-fly ontology matching[C]//Proceedings of the 2013 conference on empirical methods in natural language processing. 2013: 1545-1556.

[11] Yao X, Van Durme B. Information extraction over structured data: Question answering with freebase[C]//Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2014: 956-966.

[12] Lai Y, Lin Y, Chen J, et al. Open domain question answering system based on knowledge base[M]//Natural Language Understanding and Intelligent Applications. Springer, Cham, 2016: 722-733.

[13] Xu K, Reddy S, Feng Y, et al. Question answering on freebase via relation extraction and textual evidence[J]. arXiv preprint arXiv:1603.00957, 2016.

[14] Page L, Brin S, Motwani R, et al. The PageRank Citation Ranking: Bringing Order to the Web[J]. Technical Report, Stanford Digital Libraries, 1998.

[15] Li H, Mu L, Zan H. Computation of Word Similarity Based on the Information Content of Sememes and PageRank Algorithm[M]// Chinese Lexical Semantics. Springer International Publishing, 2016.

[16] Liu Q, Li S. Vocabulary semantic similarity calculation based on HowNet[C]. Taipei: The 3rd Chinese Lexical Semantics Seminar. 2002: 59-76.