A Retrieval-Based Matching Approach to Open Domain Knowledge-Based Question Answering

Han Zhang[1], Muhua Zhu[2], Huizhen Wang[1]

[1]Natural Language Processing Laboratory at Northeast University, Shenyang, China
kobe1992724@outlook.com wanghuizhen@mail.neu.edu.cn
[2]Tencent AI Lab, Shenzhen, China
muhuazhu@tencent.com

Abstract. In this paper, we propose a retrieval and knowledge-based question answering system for the competition task in NLPCC 2017. Regarding the question side, our system uses a ranking model to score candidate entities to detect a topic entity from questions. Then similarities between the question and candidate relation chains are computed, based on which candidate answer entities are ranked. By returning the highest scored answer entity, our system finally achieves the F1-score of 41.96% on test set of NLPCC 2017. Our current system focuses on solving single-relation questions, but it can be extended to answering multiple-relation questions.

Keywords: Question Answering, Knowledge Base, Entity Linking, Relation Chain Inference

1 Introduction

Automatic open-domain question answering is a challenging problem in the fields of information retrieval and natural language processing. In the direction of question answering, recent years have seen a surge of research interests in knowledge-based question answering (KBQA) in both academia and industry, which is defined to be retrieving a specific entity from knowledge base as the answer to a given question. In this paper, we introduce our system which is designed specifically for the KBQA competition in NLPCC 2017.

The challenge of retrieval-based KBQA is how to match unstructured natural language questions with structured data in knowledge base. To understand a question, it is necessary to figure out the topic entity and relation chain inside the question. Thus, **topic entity linking** and **relation chain inference** are the most important modules in our system.

To detect the topic entity in a question, n-gram words that appear both in the question and mention list of knowledge base are selected as candidate mentions. Then we extract their lexical features, syntactic features and features building on a sequence labeling model. These features are used as inputs to a ranking model: RankNet [1]. The RankNet model outputs the scores of candidate mentions. At last, candidate mentions are linked to candidate entities.

In the relation chain inference module, we use as features the similarity scores computed at character-level, word-level and semantic-level respectively. BLEU is an evaluation method proposed for machine translation. We reform it into character-based F1 BLEU which acts as a character-level similarity feature. Word-based cosine similarity is used as a word-level similarity feature. Regarding semantic-level features, we use convolution neural network (CNN) [2] to represent both questions and corresponding relation chains as semantic vectors. The cosine similarity between semantic vectors of questions and relation chains is used as a semantic-level feature. We also use RankNet to score candidate relation chains in this module.

Our system finally achieve the F1-score of 41.96% on the test set of NLPCC 2017 and ranks 2nd among all the participants. The rest of this paper is structured as follows: we review related work in Section 2, describe the system architecture and detailed modules of our system in Section 3, and present the experimental results in Section 4. Finally, Section 5 presents our conclusion and future work.

2 Related Work

Knowledge-based question answering is a challenging task in the field of NLP. The mainstream approaches can be divided into three categories: semantic parsing based [3-7], information extraction based [8-10] and retrieval based [11-13].

The semantic parsing based approaches translate natural language questions into a series of semantic representations in logic forms. They query the answer in knowledge base through the corresponding query statement. Yih et al. [14] present a semantic parsing method via staged query graph generation. Convolution neural network is used to calculate the similarities between question and relation chains.

The information extraction based approaches extract topic entities from questions and generate a knowledge base subgraph with the topic entity node as the center. Each node in the subgraph can be used as a candidate answer. By examining the questions and extracted information according to some rules or templates, they obtain the feature vectors of the questions. A classifier is then constructed to filter candidate answers based on input feature vectors. Yao et al. [15] associate question features with answer patterns described by Freebase. They also exploit ClueWeb, mined mappings between knowledge base relations and natural language text, and show that it helps both relation prediction and answer extraction.

The idea of retrieval-based method is similar to that of information extraction based methods. The question and candidate answers are mapped to distributed representation. The distributed representations are trained on labeled data, aiming to optimize the matching function between the question and the correct answer. Zhang et al. [16] combine bi-directional LSTM with an attention mechanism to represent the questions dynamically according to diverse focuses of various candidate answers.

These approaches work well on the English open dataset **WebQuestion**. However, their performances on a Chinese KBQA dataset have not been presented before.

3 Our Approach

Figure 1 illustrates the system architecture of our approach. The first step of our system is to conduct word segmentation, POS tagging, named entity recognition and dependency parsing on an input question. On the base of the preprocessing results, the entity linking module is used to detect a topic mention in the question. Mentions appearing in a question are selected as candidates. We use a pairwise ranking model, RankNet to score candidate mentions. In the relation chain inference module, features designed at three different levels are used to measure the similarity between a question and candidate relation chains. Finally, we rank candidate answer entities by calculating the weighted sum of scores in the entity linking module and relation chain inference module. The candidate answer entity with the highest score is outputted as the final answer.

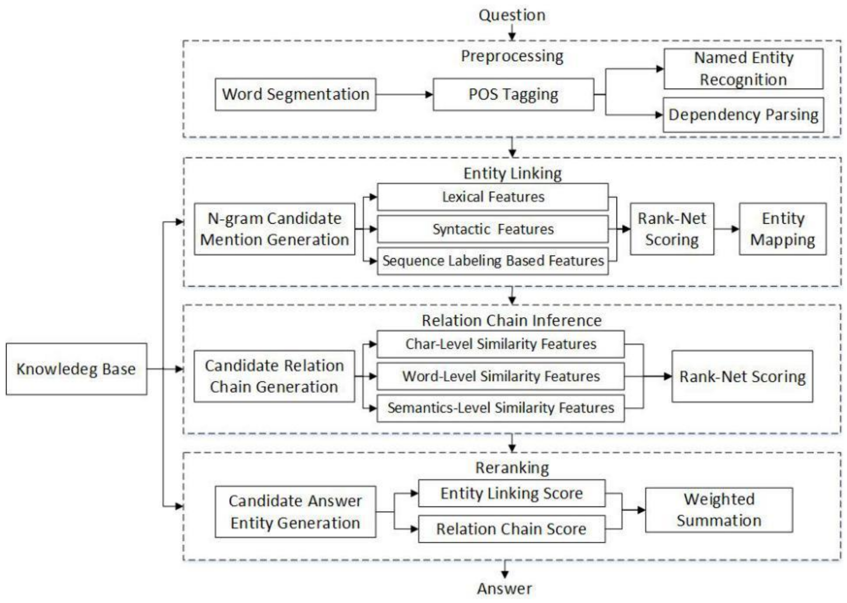

Fig. 1. System architecture of our approach

3.1 Entity Linking

Our entity linking module is used to detect a topic mention in the question. We first combine N-gram words to generate candidate phrases by ranging N from 1 till to the length of the input question.[1] Then we check the mention list of our knowledge base to retain candidate phrases that appear in the mention list. Such candidate phrases are used

[1] The length of a sentence refers to the number of word counts.

as **candidate mentions**. Because of the enormous amount of mentions in the knowledge base, a question generally has more than one candidate mention. In order to rank these candidate mentions, we extract their lexical features, syntactic features and features that build on a sequence labeling model.

- **Lexical Features**

Mentions in the knowledge base have different probabilities of being a topic mention from the perspective of lexicology. The lexical features used in our system are defined as follows:

F1: string length of the mention. A mention with a bigger string length (such as "海带/炒/猪肝 || stir fried liver with kelp") is more likely to be a topic mention than shorter ones (such as "海带 || kelp"). So we use the number of word tokens in a candidate mention as a feature.

F2: whether enclosed in a book title mark. The mention in a book title mark (such as "《红楼梦》|| A dream of Red Mansions") has high possibility to be a topic mention. So we design a binary feature to indicate whether a candidate mention is enclosed by a boot title mark.

F3: whether the mention is a stop word. Stop words such as ("什么 || what") have relatively low possibility to be a topic mention. So we collect a list of 800 stop words and design a binary feature to indicate whether a candidate mention appears in the stop-word list.

F4: average IDF value. A mention with a low Inverse Document Frequency (IDF) value (such as "在/哪里 || where") tends to have low probabilities to be a topic mention. So we compute IDF values of the words in the mention and then use the average of the IDF values as a feature for the candidate mention.

F5: whether the mention is a named entity. Named entities (such as "李军 || Jun Li" and "龙泉镇 || Longquan Town") have higher possibility to be a topic mention. So, according to the results of automatic named entity recognition, we design a binary feature to indicate whether a mention is a named entity.

- **Syntactic Features**

Syntactic features are important to judging whether a mention is a topic mention. The syntactic features used in our system are defined as follows:

F6: whether the mention is a noun or noun phrase. We design the binary feature to indicate whether a mention is a noun phrase. A mention is regarded as a noun phrase if 1) the words in the mention have the "ATT" dependency relations only, and 2) the mention ends with a noun. Fig.2 shows an example for this feature where "国际/贸易/实务 || practice of international trade" is recognized as a noun phrase.

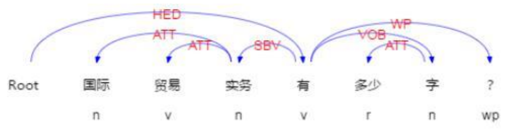

Fig. 2. An illustrating example for F6.

F7: whether the mention is the subject (object) when the object (subject) is a pronoun. This feature encodes the syntactic relation between the subject and object in the dependency tree of a question. As the example in Figure 3 shows, the object "谁 || who" is a pronoun and the corresponding subject "李军 || Jun Li" tends to be a topic mention.

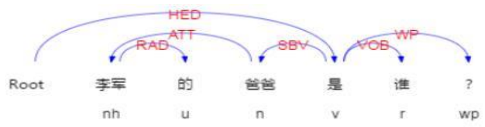

Fig. 3. An illustrating example for F7.

F8: whether the mention precedes the POS pattern "r + q + n". We find that topic entities tend to appear before the POS pattern "r+q+n", as the example in Figure 4 shows, where the topic mention "红楼梦 || A dream of Red Mansions" is followed by the pattern "r + q + n".

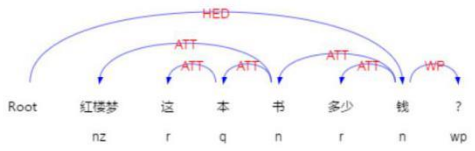

Fig. 4. An illustrating example of F8.

- **Sequence Labeling Model-Based Feature**

F9: Sequence Labeling Score. We utilize a neural sequence labeling model to assign scores to candidate mentions, which is demonstrated in Figure 5. We use Bi-LSTM [17] to encode words in a question, which is capable of learning long-term dependencies between words and taking into consideration both the previous and future context. Candidate mentions are represented with the tags of **BEG** and **END**, which are output of the softmax layer.

Specifically, the output of softmax layer is $S_i(tag)$, where i is the position in the question. c refers to a candidate mention. b and e are the first position and last position of c in the question. The probabilistic score of c is calculated as:

$$P(c) = S_{b-1}(\text{"BEG"}) * S_{e+1}(\text{"END"}) \tag{1}$$

To train the sequence labeling model, we use the cross entropy loss function:

$$loss = -\sum y_c * log\big(P(c)\big) + (1 - y_c) * log(1 - P(c)) \tag{2}$$

where $y_c = 1$ if c is a positive candidate, and otherwise $y_c = 0$.

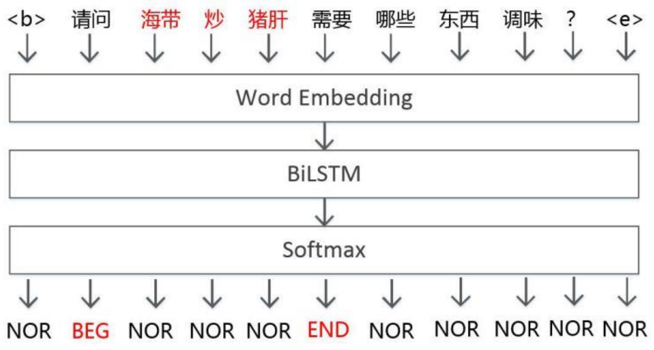

Fig. 5. The sequence labeling model for scoring candidate mentions

The above features are fed to RankNet, a single hidden layer neural network with a pairwise loss function. RankNet is in charge of ranking candidate mentions.

Finally, candidate mentions are linked to candidate entities. Due to the limited size of training data, our system does not perform entity disambiguation.

3.2 Relation Chain Inference

Relation chain is defined to be the path from a topic entity to its corresponding answer entity in the knowledge base. We can generate candidate relation chains according to candidate entities from the question. Because all the questions in NLPCC 2017 dataset are single-relation ones, we take the relation chains of length 1 as candidates. The resulting candidate relation chains are in a large number, so selection of relation chains cannot be formalized as a multi-class classification problem. In this work, we generalize questions into question patterns by replacing the candidate mention with the tag "<entity>". Then we use character-level, word-level and semantic-level features to measure the similarities between the question pattern and candidate relation chains.

- **Character-Level F1 BLEU as Similarity**

BiLingual Evaluation Understudy (BLEU) [18] is an evaluation metric proposed for the task of machine translation to measure the quality of a machine generated translation against translation references. The basic BLEU (without Brevity Penalty) formula is defined as follows when there is only one reference translation:

$$P_n = \frac{\sum_{w_n \in c} Min(Count_c(w_n), Count_r(w_n))}{\sum_{w_n \in c} Count_c(w_n)} \tag{3}$$

$$BLEU_N = \sqrt[N]{\prod_{n=1}^{N} P_n} \tag{4}$$

where c is a candidate translation, r is a reference translation, w_n is an n-gram that appears in candidate translation and $Count_c(w_n)$ is the counts of w_n appearing in c. As shown in the formula, basic BLEU only measures the accuracy of a candidate translation, the recall, however, is not considered.

We adapt the basic BLEU measure to compute character-level similarity. We first change the word-based n-grams into character-based n-grams. Second, in order to consider both accuracy and recall, we change the formula into an F1 measure. To this end, we use the mean value instead of square root of the product, which aims to get a smooth value when one of the F_n is equal to 0. The improved character-based F1 BLEU is defined as follows:

$$overlap_n = \sum_{w_n \in c} Min(Count_c(w_n),\ Count_r(w_n)) \quad (5)$$

$$P_n = \frac{overlap_n}{\sum_{w_n \in c} Count_c(w_n)} \quad (6)$$

$$R_n = \frac{overlap_n}{\sum_{w_n \in r} Count_r(w_n)} \quad (7)$$

$$F_n = \frac{P_n * R_n}{2*(P_n + R_n)} \quad (8)$$

$$Char\ Based\ F1\ BLEU_N = \frac{1}{N}\sum_{n=1}^{N} F_n \quad (9)$$

where c is a candidate relation chain and r is a question pattern. By contrast to character-based and word-based cosine similarities, this F1 BLEU metric manages to combine both fine-grained and coarse-grained n-grams and can reduce the negative impact caused by word segmentation errors.

- **Word-Based Cosine Similarity**

This is a traditional method to calculate the similarity between two texts. We remove stop-words in a question pattern and corresponding candidate relation chains. Then we represent them as vectors using the approach of bag-of-words, based on which similarity is calculated with the cosine function.

- **CNN-Based Semantics Similarity**

As shown in Figure 6, we utilize CNN models to represent a question pattern and its candidate relation chains as semantic vectors respectively, one model for the question pattern and the other model for relation chains. The first layer of the networks is an embedding layer. We use word2vec toolkit to train word and character embeddings.[2] The final representation of a word is the concatenation of its word embedding and the embeddings of its characters inside. The second layer is convolution layer. We use 200 filters with context windows of 2 words, and 200 filters with context windows of 3

[2] https://code.google.com/archive/p/word2vec/

words. A max-pooling operation is applied for feature mapping. Finally, a fully-connected layer outputs semantic vectors of the question pattern and candidate relation chains.

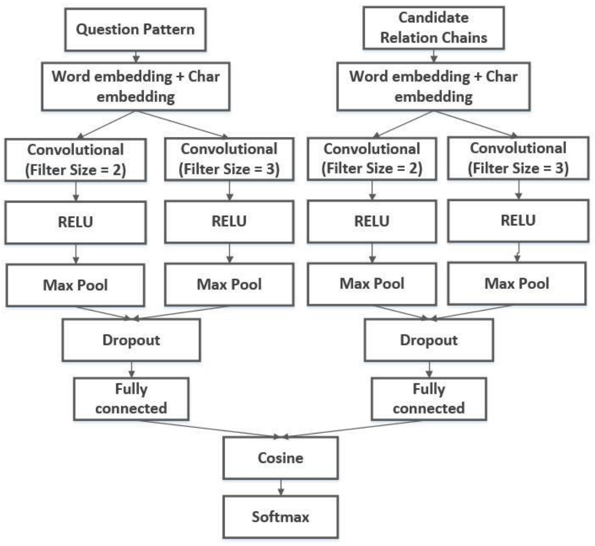
Question Pattern
Candidate Relation Chains
Word embedding + Char embedding
Word embedding + Char embedding
Convolutional (Filter Size = 2)
Convolutional (Filter Size = 3)
Convolutional (Filter Size = 2)
Convolutional (Filter Size = 3)
RELU
Max Pool
Dropout
Fully connected
Cosine
Softmax

Fig. 6. CNN-based similarity module in our system

Based on the semantic vectors outputted by CNN, we compute the cosine similarity between a question pattern and candidate relations. The final layer is a softmax layer. The output of softmax is defined as follows:

$$P(r|q) = \frac{e^{Cosine(q,r)}}{\sum_{r' \in C} e^{Cosine(q,r')}} \tag{10}$$

where q is a question pattern, r is a candidate relation chain, and C is the set of candidate relation chains. The cross entropy loss function, as the objective of optimization, is defined as follows:

$$loss = -\sum_q log(P(r_q^+|q)) \tag{11}$$

where r_q^+ is the positive candidate relation chain of q.

3.3 Ranking Answer Entities

Candidate topic entities and candidate relation chains are obtained in aforementioned two modules. Then we generate the candidate answer entities. The scores returned by entity linking and relation chain inference are used together to rank candidate answer entities. The scoring function for ranking answer entities is defined as follows:

$$score_{answer\_entity} = \gamma * score_{topic\_entity} + (1-\gamma) * score_{relation\_chain} \tag{12}$$

The entity which ranks highest is returned as the final answer to an input question.

4 Experiments

4.1 Dataset

In this paper, we used the dataset provided by the NLPCC 2017 open domain KBQA competition. The dataset includes 24,479 single-relation question-answer pairs for training, a Chinese knowledge base with 43M SPO triples, and 7M mapping data from mentions to entities. The test set contains 48,850 questions. Most of questions are noise data that are excluded for system performance evaluation. After removing the noise data, there are 7,631 questions remaining to be answered.

Because the questions in the dataset are all single-relation questions, what we really need for building the system are question-triple pairs rather than question-answers pairs. To this end, we used gold-standard answers to extract answer triples backward from the knowledge base. We randomly sampled 200 automatically generated question-triple pairs and manually evaluated the accuracy. The accuracy of the sample data is 96%. We use 20,479 of question-triple pairs as training data and 4,000 for system development.

4.2 Setup

The word embeddings used in our sequence labeling model and CNN models are pre-trained by using word2vec. We used the skip-gram model [19] and the dimension was set to 256. In order to prevent overfitting, we utilized both dropout and batch normalization techniques. SGD was adopted as the optimization method. Regarding the two DNN models, the initial learning rate and decay rate were set to 0.05 and 0.7 respectively. Learning rate decay comes to play after 20 epochs for sequence labeling model training and 30 epochs for CNN model training. Mini-batching is also used and the batch sizes were set to 10 and 20 respectively.

Our KBQA system is multitasking. As shown in Equation (12), we use linear combination to add the scores returned by entity linking and relation chain inference. We tuned the combination parameter γ on the validation set and finally set it to 0.76.

4.3 Results

The organizer of NLPCC 2017 provided a CNN-based KBQA system. We used it as our baseline system and compared the results of the baseline system and our system on the NLPCC 2017 KBQA test set. We also evaluated the accuracy of our system when using character-based F1 BLEU, word-based cosine similarity and CNN-based similarity separately in the relation chain inference module. The results are presented in Table 1. From the results we can see that the performance of our system is superior to the performance of the baseline system.

Table 1 shows that character-based F1 BLEU outperforms word-based cosine similarity because it considers both coarse-grained and fine-grained literal similarities between the question and relation chains. The CNN-based similarity method captures semantic information, thus it is better than other two similarity measures. We get the best results when combining all the three similarity-based features. Our system achieves the F1-score of 41.96% on the test set and obtains the 2nd place in the final leaderboard.

	Accuracy	Recall	F1 Score
Baseline system	16.4	16.4	16.4
Char-based F1 BLEU	36.7	48.4	38.5
Word-based cosine similarity	28.8	52.7	31.2
CNN-based similarity	38.5	41.2	38.7
All features	41.3	43.6	42.0

Table 1. The results of the baseline and our system

4.4 Error Analysis

We randomly sampled 200 error cases and analyzed the causes of these errors. We find that 69.5% of errors are attributed to annotation errors occurring in the dataset. Such annotation errors include “annotated answer entity not in knowledge base” (32.5%), “wrong answer entity” (13%), “no relation chain connected to topic entity can match with the question” (9%), “topic entity not in knowledge base (8.5%)” and “no enough information exits in the question for entity disambiguation” (6.5%). 23% of them are caused by relation chain errors and the other 7.5% are caused by entity linking. Our system still have room for further improvement.

5 Conclusion

In this paper, we described our KBQA system for NLPCC 2017 KBQA competition. Our system adopts a multitask framework, which uses an entity linking model to detect topic entities and a model combining three similarity methods to find out the relation chain that is asked in a question. Our system performs well on test set, though there is room left for our system to improve. Due to the limited size of training data, we do not process multi-relation questions and entity disambiguation. It will be what we plan to work on in the future.

Reference

1. Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to Rank using Gradient Descent. ICML. 2005.
2. Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. IEEE. 2015.

3. Luke S Zettlemoyer and Michael Collins. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. UAI. 2005.
4. Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. Induc-ing probabilistic CCG grammars from logical form with higherorder unification. EMNLP. 2010.
5. Percy Liang, Michael I. Jordan, and Dan Klein. Learning Dependency-Based Compositional Semantics. ACL. 2011.
6. Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic Parsing on Freebase from Question-Answer Pairs. EMNLP. 2013.
7. Jonathan Berant and Percy Liang. Semantic parsing via paraphrasing. ACL. 2014.
8. Hannah Bast and Elmar Haussmann. More accurate question answering on freebase. Information and Knowledge Management. 2015.
9. Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. Open question answering over curated and extracted knowledge bases. Knowledge Discovery and Data Mining. 2014.
10. Xuchen Yao. Lean question answering over freebase from scratch. ACL. 2015.
11. Antoine Bordes, Sumit Chopra, and Jason Weston. Question answering with subgraph embeddings. EMNLP. 2014.
12. Antoine Bordes, Jason Weston, and Nicolas Usunier. Open question answering with weakly supervised embedding models. ECML. 2014.
13. Li Dong, Furu Wei, Ming Zhou, and Ke Xu. Question answering over Freebase with multicolum convolutional neural networks. ACL. 2015.
14. Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. Semantic parsing via staged query graph generation: question answering with knowledge base. ACL. 2015.
15. Xuchen Yao and Benjamin Van Durme. Information extraction over structured data: Question answering with freebase. ACL. 2014.
16. Yuanzhe Zhang, , Kang Liu, Shizhu He, Guoliang Ji, Zhanyi Liu, Hua Wu, and Jun Zhao. Question Answering over Knowledge Base with Neural Attention Combining Global Knowledge Information. arXiv. 2016.
17. Xuezhe Ma, and Eduard Hovy. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. arXiv. 2016.
18. Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. ACL. 2002.
19. Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. Computer Science. 2013.