JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 37, 959-974 (2021)
DOI: 10.6688/JISE.202107_37(4).0013

A Domain-Specific Non-Factoid Question Answering System based on Terminology Mining and Siamese Neural Network*

MING-QI LV [1], HAO ZHANG [1], KANG-JUN ZHU [1], CHAO HUANG [1] AND TIE-MING CHEN [1,+]
[1]*College of Computer Science*
Zhejiang University of Technology
Hangzhou, 310023 P.R. China
E-mail: {mingqilv; 2111712326; tmchen}@zjut.edu.cn; zkjofworld@gmail.com; mela1121@163.com

The non-factoid question answering system (QAS) responds to an input question by fetching an answer from a question answering (QA) database. The existing non-factoid QASs still cannot well adapt to specific professional domains due to the lack of domain knowledge. Aiming at this problem, this paper proposes a domain-specific non-factoid QAS by combining information retrieval technique and deep neural network. First, it extracts professional terms from the domain-specific documents. The professional terms can be used as an important source of domain knowledge. Second, it trains a deep Siamese neural network for semantically matching the questions. Finally, it queries and ranks the candidate answers based on the professional terms and the deep Siamese neural network. We conducted experiments based on two real domain-specific QA databases, and the experiment results have demonstrated the effectiveness of the proposed QAS.

Keywords: question answering system, terminology mining, Siamese neural network, domain knowledge, natural language processing

1. INTRODUCTION

The purpose of question answering system (QAS) is similar to that of information retrieval system (IRS), which attempts to return documents satisfying the information need of a user based on an entered natural language query. However, in IRS, the user enters several keywords and IRS returns a large collection of documents. In QAS, the user enters a question and QAS comes up with a concise answer. For a QAS, the returned answers can be factoid or non-factoid. A factoid answer is usually a simple fact represented as a keyword or a concept [1], while a non-factoid answer is typically a longer readable document [2].

Since non-factoid answers contain richer information, non-factoid QAS is more suitable for professional domains, which often need more detailed information to clarify the professional questions. Most existing non-factoid QASs are implemented based on search techniques [2, 3]. Typically, they firstly analyze and understand the entered question, and then query similar candidate questions from a question answering (QA) database, *e.g.*, frequent asked question (FAQ) and community question answering (CQA) [2]. Finally, they rank the candidate questions and return the answers corresponding to the top-*N* questions.

Received July 25, 2019; revised October 12, 2019; accepted November 27, 2019.
Communicated by Berlin Chen.
* This work was supported in part by the Zhejiang Provincial Natural Science Foundation of China (No. LY18 F020033) and the National Natural Science Foundation of China (No. 61772026).
+ Corresponding author.

Although the existing non-factoid QASs have achieved satisfied performance on several evaluation tasks (*e.g.*, TREC [4], CLEF [5], NTCIR [6], *etc.*), they still cannot well adapt to the specific professional domains due to the following reasons. First, most domain-specific questions contain professional terms, which are essential for understanding the questions. For example, there is a real question in our dataset: "*What types of projects need to conduct the environment impact assessment*?" Here, *environment impact assessment* is a professional term, which is the key to understand the question. However, since a professional term is usually composed of several words, it is difficult to be discovered (*e.g.*, in English) or segmented (*e.g.*, in Chinese). In addition, even if the profession terms were correctly discovered or segmented, they are treated as normal words in the existing QASs, so their importance cannot be well identified. Recent researches on word embedding (*e.g.*, word2vec, ELMo, BERT, *etc.*) could learn latent semantic meanings and correlations between words. However, they require a huge amount of documents to train a reasonable word model. This requirement is almost impossible to be satisfied in domain-specific professional domains. Second, the entered questions and the questions in QA database might usually express the same semantic meaning in different ways. For example, "*How's the weather tomorrow*?" and "*What will the weather be like tomorrow*?" intend to ask the same question.

Aiming at these problems, this paper proposes a domain-specific non-factoid QAS by combining terminology mining and deep neural network. For the first problem, it extracts professional terms from the domain-specific documents by exploiting text mining technique and Web knowledge. These professional terms are then used as keywords to query candidate questions. For the second problem, it ranks the candidate questions based on a deep Siamese neural network, which uses shared parameters to learn the semantic similarity between questions. The primary contributions of this paper are summarized as follows.

- We propose a terminology mining method. It firstly extracts frequent phrases from the domain-specific documents using a frequent substring mining algorithm. Then, it generates professional terms from the frequent phrases through a filtering step based on lexical analysis and a verification step based on Web knowledge.
- We design a domain-specific non-factoid QAS by combining information retrieval (IR) technique and deep neural network. It firstly queries candidate questions from the QA database by using the professional terms as keywords. Then, it ranks the candidate questions by considering both the matching score of the professional terms and the semantic similarity computed by a deep Siamese neural network. The integration of IR technique and deep neural network is adopted to make a balance between keyword-level matching and sentence-level matching.
- We evaluate the proposed method based on two real QA databases in the domain of environment protection and childcare. The results show the effectiveness of the terminology mining and the strategy of QAS design.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 details the domain-specific non-factoid QAS, including the architecture of QAS, the terminology mining method, and the Siamese neural network. Section 4 reports the experiment results. Section 5 concludes the paper.

2. RELATED WORK

2.1 The Non-Factoid QAS

Most of the existing QAS researches focus on factoid QA task due to the following reasons. First, the factoid QASs are more suitable for open domain questions [8, 34], which could be properly answered by a simple fact represented as an entity or a concept, while open domain QA databases could be more easily obtained for experiments. Second, most advanced QAS techniques could only be applied for the factoid QA task, *e.g.*, knowledge graph [7, 8], machine comprehension [9, 35]. However, the factoid QASs are not suitable for professional domain QA task.

For non-factoid QA task, early researches apply template techniques to design the QASs [10, 11], which gets answers based on template matching or rewriting pattern matching. However, template technique based QASs need to manually define a large number of templates or patterns, leading to scalability problems. IR techniques are mostly applied for designing non-factoid QASs. For example, Lu *et al.* [12] and Liu *et al.* [13] applied IR technique to query the Web search engine and rank the relevant results to get the answer. Feng *et al.* [14] designed a QAS by using keyword searching, which is implemented based on vector space model and Lucene search engine. Komiya *et al.* [15] proposed two methods to improve the non-factoid QASs. One is the modification of measurement of mutual information for query expansion, and the other is an answer evaluation method that utilizes the Web relevance score and the translation probability. Savenkov [16] designed a non-factoid QAS by combining candidates extracted from answers in the QA database that are relevant to the questions and Web documents retrieved from Web search engine. However, IR techniques cannot well address the lexico-syntactic gap problem, *i.e.*, two questions may have the same meaning but differ lexically and syntactically.

With the development of deep learning and embedding technology, recent researches tried to apply deep neural networks to design non-factoid QASs. For example, Zhou *et al.* [17] firstly applied auto-encoders to learn the semantic representation of queries and answers, and then trained the question-answer matching model by maximizing the similarity between the queries and the corresponding answers on top of their semantic representative vectors. Qiu and Huang [18] proposed a convolutional neural tensor network, which integrates the sentence modeling and semantic matching into a single model. Das *et al.* [19] proposed a deep Siamese neural network called SCQA for similar question retrieval. SCQA consists of twin convolutional neural networks with shared parameters and a contrastive loss function joining them. The experiment results showed that deep Siamese neural network can well address the lexico-syntactic gap problem and outperforms the existing translation models, topic models and other deep neural networks. However, it is very difficult for the deep learning techniques to identify the professional terms and their importance to understand the questions, especially when the training corpus is not large enough. It is precisely the situation of professional domains.

2.2 The Key-Phrase Extraction

The existing key-phrase extraction methods could be divided into two categories,

i.e., supervised methods and unsupervised methods. The supervised methods apply machine learning technique for key-phrase extraction. For example, Abulaish and Anwar [20] extracted key-phrases based on a classification model trained based on a variety of lexical and grammatical features in the documents. Turney [21] developed a key-phrase mining algorithm called GenEx, which classifies key-phrases by combining the parametrized heuristic rules and the genetic algorithm. However, supervised methods require a large number of documents, where the true key-phrases are correctly tagged. It is almost impossible for specific professional domains. The unsupervised methods extract key-phrases by exploiting features such as word frequency, word positions and so on. For example, Kumar *et al.* [22] proposed a key-phrase extraction algorithm for scientific documents using N-gram filtration technique. Gao *et al.* [23] proposed a two-step key-phrase extraction method, where the position-weighted TF*PDF algorithm is proposed to obtain and weight the candidate hot terms. Xie *et al.* [24] regarded a document as a sequential dataset and proposed a sequential pattern mining algorithm to extract key-phrases.

However, these methods cannot well extract the professional terms. First, a professional term is usually composed of several words. Thus, they cannot be discovered before they are properly identified (*e.g.*, in English) or segmented (*e.g.*, in Chinese). Second, the existing keyword extraction algorithms cannot distinguish between professional terms and normal keywords. Thus, they would treat the professional terms and normal keywords equally.

3. METHOD

3.1 Preliminary

We formally define the non-factoid question answering problem in this paper as follows. Given a real-time question Q and a QA database $D = \{D_1, D_2, \ldots, D_n\}$, where $D_k = (D_k.Q, D_k.A)$, $D_k.Q$ is a question, and $D_k.A$ is the corresponding answer to $D_k.Q$, the goal is to find $D_k.A$, whose corresponding question $D_k.Q$ is the most relevant to Q. Thus, the non-factoid question answering task could actually be viewed as a semantic query problem.

Fig. 1 shows the architecture of the proposed domain-specific non-factoid QAS, which consists of three modules, *i.e.*, the terminology mining module, the semantic matching module, and the answer retrieval module. The terminology mining module extracts professional terms from the domain-specific documents offline through a frequent phrase mining step and a professional term generation step. The semantic matching module trains a semantic sentence matching model offline based on a deep Siamese neural network. The answer retrieval module processes the user's online question through a query step and a ranking step by leveraging the extracted professional terms and the semantic sentence matching model.

3.2 The Terminology Mining Module

The professional terms are essential for understanding the domain-specific questions, while a professional term is usually composed of several words. For example,

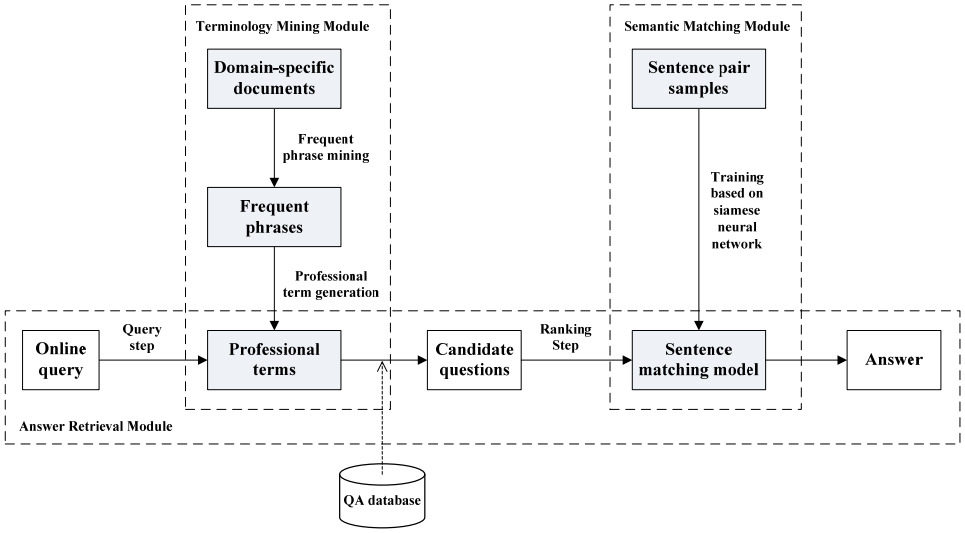


Fig. 1. The architecture of the proposed domain-specific non-factoid QAS.

convolutional neural network is a professional term in the computer science domain and *environment impact assessment* is a professional term in the environment protection domain. To effectively extract the professional terms, we propose a terminology mining method, which works in two phases as follows.

In the first phase, we extract frequent phrases from the domain-specific documents. The simplest strategy for extracting frequent phrases is to use the N-Gram technique. Specifically, given a minimum support threshold *min_sup*, we could extract all the substrings with length larger than 1, where the occurrence count of each substring is larger than *min_sup*. However, this strategy would result in too many redundant phrases. For example, if *convolutional neural network* were a frequent phrase, all its substrings would be frequent phrases (*i.e.*, *convolutional neural* and *neural network*). Aiming at this problem, we propose a frequent phrase mining algorithm as shown in Fig. 2. The algorithm firstly creates a tree and inserts each extracted frequent N-Gram phrase into the tree according to its prefix (lines 1-14). Then, the algorithm outputs all the non-redundant frequent phrases by traversing the tree (lines 15-20). Here, we define a frequent phrase P_j is a super-phrase of *Pi*, if *Pi* is a substring of P_j and the occurrence count of P_j equals that of P_i. Then, a frequent phrase P_i is non-redundant, if P_i does not have any super-phrases. Fig. 3 gives a toy example to illustrate the frequent phrase mining process, given *DS* = {"ABCBC", "ABCB", "ACAC"} and *min_sup* = 2, where each letter in *DS* stands for a word.

Frequent Phrase Mining Algorithm	
Input:	A set of domain-specific document, *DS*
	The minimum support of frequent phrases, *min_sup*
Output:	A set of frequent phrases, *PS*

```
1. Create a tree Tr with a root node Root
2. for n = 1 to ∞ do
3.     Scan each document in DS to find all the n-Gram phrases nPS, the support of
         each phrase in which ≥ min_sup
4.     if nPS ≠ Ø then
5.         if n = 1 then
6.             for each phrase P_i in nPS do
7.                 Append P_i as a child node to Root
8.         else
9.             for each phrase P_i in nPS do
10.                Conduct a deep first search in Tr to find a branch Br that
                     matches P_i[0:n−1]
11.                Append P_i[n-1] as a child node to the last node Ln in Br
12.                Set the weight of the edge between P_i[n−1] and Ln as the
                     occurrence count of P_i
13.    else
14.        Break the loop
15. Conduct a deep first traversal in Tr and for each branch Br_i do
16.    if length(Br_i) > 1 then
17.        if the last node in Br_i is a leaf node then
18.            Treat Br_i as a frequent phrase and append it to PS
19.        else if length(Br_i) > 2 && the weight of the last edge < the weight of
               the previous edge then
20.            Treat Br_i[0:length(Br_i)−1] as a frequent phrase and append it to PS
```

Fig. 2. The pseudo code of the frequent phrase mining algorithm.

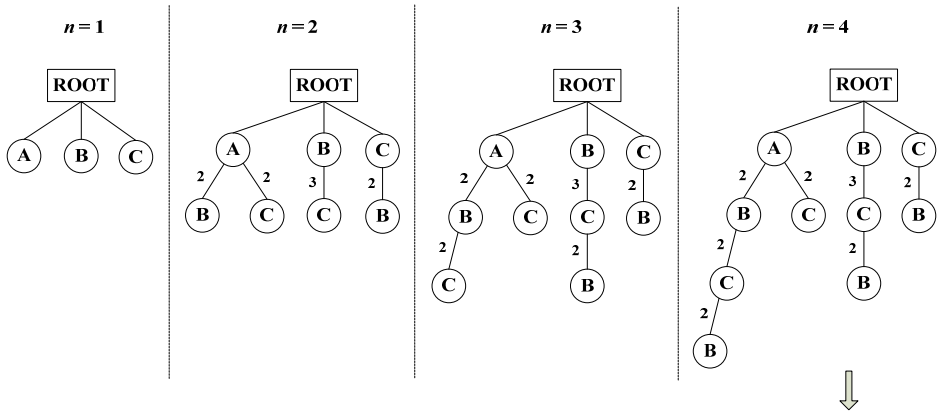


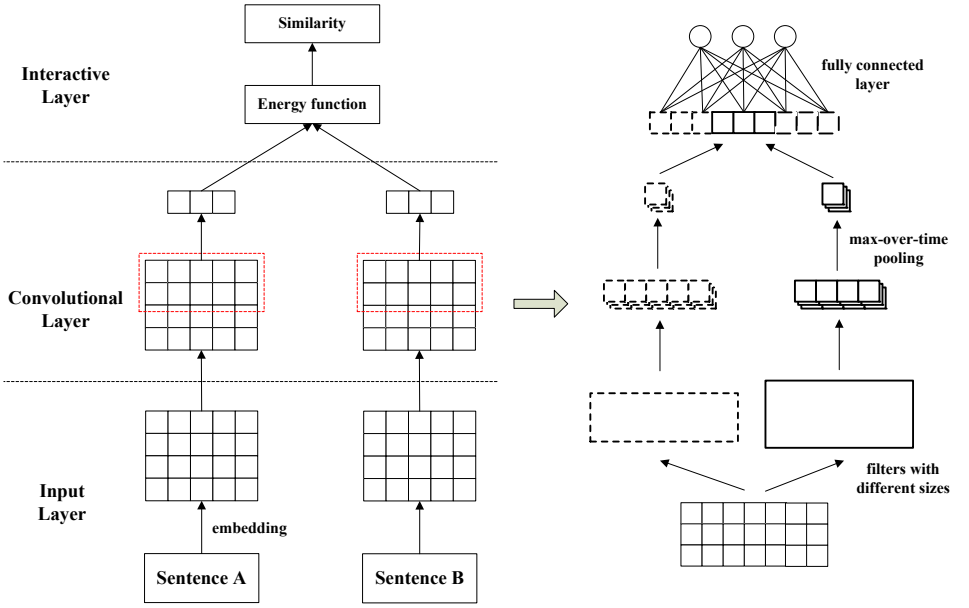
Fig. 3. A toy example for illustrating the frequent phrase mining algorithm.

In the second phrase, we generate professional terms from the extracted frequent phrases. Since the extracted frequent phrases are likely to contain many common phrases, we design a filtering step and a verification step to filter out common phrases as many as possible. The filtering step works based on a word frequency filter and a POS (Part Of Speech) filter. Given the set of extracted frequent phrases *PS*, the word frequency filter works as follows: For each frequent phrase P_i in *PS*, we submit P_i to a search engine (*e.g.*, Google, Baidu, *etc.*) and obtain the number of retrieved results nr_i. If nr_i is larger than a predefined threshold δ, we will filter out P_i from *PS*. The intuition is that the common phrases are likely to be used more frequently than professional terms. The POS filter works as follows: We firstly perform POS tagging for each domain-specific document in *DS*. Then, for each frequent phrase P_i in *PS*, if P_i has never been tagged as a noun phrase in the POS tagging results, it will be filtered out from *PS*. We use the regular expression based chunking method in NLTK tool to get the noun phrases on the basis of POS tagging. The intuition is that most professional terms are noun phrases. On the other hand, the verification step is designed by leveraging Web knowledge and works as follows: For each frequent phrase P_i in *PS*, we query P_i in several existing Web knowledge bases (*e.g.*, Wikipedia, Probase, *etc.*). If P_i exists in at least one of the Web knowledge bases (*e.g.*, as an entry in Wikipedia, as an entity in Probase, *etc.*), it will be kept as a professional term. Otherwise, it will be filtered out from *PS*. After performing the filtering and verification steps, the final set of frequent phrases will be treated as the set of professional terms *TS*.

3.3 The Semantic Matching Module

The semantic matching module trains a semantic sentence matching model, which measures the semantic similarity between different sentences. The semantic sentence matching model is used to rank the candidate questions by measuring the semantic similarity between the real-time question and each candidate question.

We apply a Siamese neural network to train the semantic sentence matching model. Siamese neural networks have been proposed for a number of metric learning tasks [25, 26]. A Siamese neural network is composed of twin networks, which share the same parameters and accept different inputs. The twin networks are joined by an energy function, which computes the semantic relatedness of the two inputs. As shown in Fig. 4 (a), the proposed Siamese neural network consists of three layers, *i.e.*, the input layer, the convolutional layer, and the interactive layer. First, the input layer uses pre-trained embedding word vectors to transform an input question into a fixed-sized matrix. Note that the input question would be padded or truncated when necessary. Second, the convolutional layer consists of a pair of convolutional neural networks (CNN), which conducts a non-linear projection of the input matrix into the semantic space. As shown in Fig. 4 (b), each CNN is designed based on the structure of TextCNN [27], which is composed of a convolutional layer using multiple filters with different sizes, a max-over-time-pooling layer, and a fully connected layer. The final output of each CNN is a fixed-sized semantic vector. Finally, the interactive layer adopts cosine similarity as the energy function to measure the similarity between the semantic vectors.



(a) The overall architecture. (b) The architecture of the convolutional layer.

Fig. 4. The architecture of the proposed Siamese neural network.

Based on a training sample set SS with each training sample $ts_i = (\{qa_i, qb_i\}, y_i)$, where $\{qa_i, qb_i\}$ is a pair of questions and $y_i \in \{1, 0\}$ is the label (when qa_i is relevant with qb_i, $y_i = 1$, and otherwise, $y_i = 0$), the Siamese neural network is trained with the aim to maximize the similarity between relevant questions and minimize the similarity between irrelevant questions. We use the contrastive loss function as Eq. (1) to train the Siamese neural network, where y_i' is the predicted similarity between qa_i and qb_i by the interactive layer. After the Siamese neural network is trained, it could output a score of similarity for each pair of questions.

$$L = \sum_{i=1}^{|SS|} loss(qa_i, qb_i) \tag{1}$$

$$loss(qa_i, qb_i) = y_i e_i^2 + (1 - y_i)\max(1 - e_i, 0)^2 \tag{2}$$

$$e_i = |y_i' - y_i| \tag{3}$$

3.4 The Answer Retrieval Module

The answer retrieval module processes the user's online question through a query step and a ranking step, where the query step is used to search the candidate questions and the ranking step is used to rank all candidate questions to get the most appropriate answer.

Given an online input question Q, the query step works as follows: First, we find all professional terms in Q. It can be implemented by string matching algorithm (*e.g.*, for English) or word segmentation algorithm (*e.g.*, for Chinese). Second, we compute the

importance of each found professional term and keep the most important k professional terms, denoted as $KT(Q)$. The importance computation can be implemented by using the TextRank algorithm [28]. Note that if the number of found professional terms is less than k, we would supplement $KT(Q)$ with the most important $k-|KT(Q)|$ normal words, denoted as $KW(Q)$. Then, the query keywords $K(Q)=KT(Q)\cup KW(Q)$. Finally, we use $K(Q)$ to query the candidate questions based on Lucene search engine. The fetched set of candidate questions is denoted as $CQ(Q)$.

The ranking step works as follows: For each candidate question Q_i in $CQ(Q)$, we firstly compute the score of semantic similarity between Q and Q_i based on the trained Siamese neural network, denoted as $ss(Q, Qi)$. Second, we compute the score of keyword matching between Q and Q_i based on Eq. (4), denoted as $sm(Q, Qi)$, where α is used to adjust the importance of professional terms and normal words ($0.5 < \alpha < 1$). Finally, the final score of matching Q and Q_i is calculated based on Eq. (5), where λ is used to adjust the importance of semantic similarity score and keyword matching score. Then, the answer corresponding to the candidate question with the highest final matching score is returned.

$$sm(Q,Q_i)=\frac{\alpha\times|Q_i\cap KT(Q)|+(1-\alpha)\times|Q_i\cap KW(Q)|}{k} \tag{4}$$

$$s(Q,Q_i)=\lambda ss(Q,Q_i)+(1-\lambda)sm(Q,Q_i) \tag{5}$$

4. EXPERIMENT

4.1 Dataset and Experiment Setup

We used two real QA datasets (*i.e.*, EPD and CCD) to conduct the experiments. Both of datasets are in Chinese and collected from the FAQ list from real websites. The EPD dataset is in the domain of environment protection and the CCD dataset is in the domain of childcare. The EPD dataset and CCD dataset contain 699 and 1926 QA pairs, respectively.

We clarify the experiment settings as follows. First, since the two datasets are both in Chinese, we use Jieba [29], an NLP toolkit for Chinese, to pre-process the datasets, including word segmentation and stopword removal. In order to ensure that the professional terms could be accurately segmented, we add the extracted professional terms to the Jieba's word segmentation dictionary. Second, the training of the Siamese neural network requires a large number of relevant sentence pairs, while it is infeasible to generate enough relevant question pairs from the two datasets. Aiming at this problem, we adopt a transfer learning mechanism. Specifically, we use external datasets (*i.e.*, ATEC [30] and ChineseSTS [31]) that contain a large number of relevant sentence pairs to train the initial model, and then manually generate a small number of relevant question pairs from the two datasets to fine-tune the model (251 question pairs from the EPD dataset and 967 question pairs from the CCD dataset).

To evaluate the QAS, we generate a set of testing questions that are semantically similar with the candidate questions in the two datasets based on two ways as follows. First, we ask several domain experts to manually rewrite each candidate question in dif-

ferent expression. Second, we follow the approach in [32] by firstly translating each candidate question into English by using a Chinese-to-English translator and then translating it back to Chinese by using an English-to-Chinese translator. In our experiment, the Baidu translation API [33] is used. Finally, 249 testing questions are generated for the EPD dataset and 957 are generated for the CCD dataset.

The default values of the parameters are set as follows. For the terminology mining module, we set the minimum support threshold $min\_sup = 2$ and the predefined threshold of the word frequency filter $\delta = 50000000$. For the answer retrieval module, we set the number of query words $k = 4$, and the importance of professional terms $\alpha = 0.7$.

4.2 The Evaluation of Terminology Mining

In this experiment, we evaluate the performance of the terminology mining module using the EPD dataset. We ask the domain experts to tag all the professional terms in the candidate questions, which are used as the ground-truth. Then, we use three metrics to quantify the terminology mining performance, *i.e.*, Precision, Recall, and F1-score. We compare the proposed terminology mining method (denoted as OUR) with the following six methods.

- N-Gram: It refers to the N-Gram technique based method as mentioned in Section 3.2, where $min\_sup = 2$.
- N-Gram+: It refers to the N-Gram technique augmented by the filtering step and the verification step.
- PAT: It refers to the PAT-tree based frequent phrase mining algorithm proposed in [37]. PAT works without the filtering step and the verification step.
- OUR−: It refers to the proposed terminology mining method without the filtering step and the verification step.
- OUR (−F): It refers to the proposed terminology mining method without the filtering step.
- OUR (−V): It refers to the proposed terminology mining method without the verification step.

The experiment results are shown in Table 1. First, OUR has the best overall performance. Its advantage over OUR (−F) and OUR (−V) demonstrates the effectiveness of the filtering step and the verification step. Second, N-Gram has the highest Recall and the lowest Precision. It is because that N-Gram extracts all the frequent phrases, most of which are common and redundant phrases. Third, OUR (−F) outperforms OUR (−V). It indicates that the verification step that leverages Web knowledge and open knowledge graph could achieve a better performance at filtering out non-professional terms. Fourth, N-Gram+ has a worse performance than that of OUR (especially Precision). It demonstrates that the proposed frequent phrase mining algorithm could discover frequent phrases with higher quality by filtering out redundant and short ones. Fifth, OUR- and PAT outperform N-Gram+. It shows that the frequent phrase mining algorithm plays a more important role than the filtering and verification steps do. OUR- outperforms PAT. The proposed frequent phrase mining algorithm adopts a stricter strategy for allowing a substring to become a frequent phrase of a longer one, which is more suitable for extracting professional terms.

Table 1. The experiment results of terminology mining evaluation.

	Recall	Precision	F1-score
N-Gram	0.779	0.061	0.110
N-Gram+	0.604	0.078	0.138
PAT	0.565	0.103	0.174
OUR–	0.546	0.147	0.232
OUR (–F)	0.639	0.114	0.198
OUR (–V)	0.770	0.061	0.112
OUR	0.569	0.210	0.305

4.3 The Parameter Tuning

In this experiment, we investigate the importance of the terminology mining module and the semantic matching module by adjusting the parameter λ (Section 3.4) using the EPD dataset. We adopt MRR (Mean Reciprocal Rank) and P@K (Precision at K) as the evaluation metrics. The experiment results are shown in Fig. 5. All the three-evaluation metrics show a similar trend (*i.e.*, an increasing phase and then a decreasing phase is followed) by increasing λ from 0.1 to 0.9. When $\lambda = 0.4$, all the three-evaluation metrics achieve the best score. It means that the proposed QAS tends to be in favor of keyword matching rather than semantic matching. This is contrast to many existing researches on open domain QASs. It is because that most state-of-the-art open domain non-factoid QASs are designed by leveraging the word embedding techniques (*e.g.*, word2vec, BERT, *etc.*). However, word embedding techniques require a huge number of documents to train reasonable word vectors for words [38], not to mention word vectors for phrases. In our experiment, the number of domain specific documents is extremely limited, and the professional terms can also not be learnt from open domain documents using pre-training scheme. This result also demonstrates that the professional terms are essential for understanding the domain-specific questions.

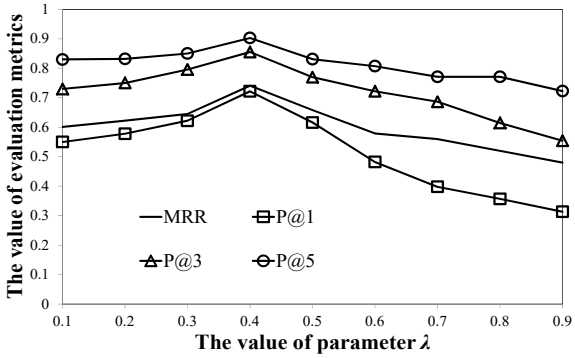


Fig. 5. The effect of parameter λ.

4.4 The Evaluation of QAS

In this experiment, we compare the proposed QAS with the following baselines.

- N_Query: It uses only the most important *k* normal keywords to query and rank the candidate questions.
- N_Semantic_Query: It uses the most important *k* normal keywords to query the candidate questions, and then rank the candidate questions based on the Siamese neural network.
- T_Query: It uses the most important *k* professional terms to query and rank the candidate questions.
- SCQA: It refers to the QAS proposed in [19], where a Siamese neural network is employed to rank the candidate questions.
- BERT_QA: It refers to the query-document matching method based on BERT in [39]. We use the pre-trained Chinese BERT model in [36] and fine-tune it using the EPD and CCD datasets.

Table 2. The comparison of different QASs.

		MRR	P@1	P@3	P@5
EPD	N_Query	0.316	0.265	0.313	0.386
	N_Semantic_Query	0.350	0.301	0.373	0.392
	T_Query	0.653	0.59	0.759	0.855
	OUR	0.742	0.722	0.855	0.903
	SCQA	0.314	0.228	0.542	0.638
	BERT_QA	0.422	0.298	0.479	0.622
CCD	N_Query	0.441	0.351	0.505	0.583
	N_Semantic_Query	0.507	0.448	0.533	0.605
	T_Query	0.531	0.448	0.583	0.652
	OUR	0.595	0.530	0.649	0.687
	SCQA	0.426	0.358	0.521	0.624
	BERT_QA	0.412	0.351	0.527	0.631

The experiment results are shown in Table 2. First, T_Query consistently outperforms N_Query. It shows that the professional terms could capture the key meaning of the questions more effectively than the normal keywords do. Especially, the performance improvement from N_Query to T_Query is far more significantly on EPD (improved by 57.9%~142.5%) than that on CCD (improved by 20.4%~27.6%). It is because that environment protection is a more professional domain than childcare. For example, we found that most keywords in EPD are professional terms (*e.g.*, volatile organic compounds, hazardous waste, factory boundary noise, *etc.*) that are almost impossible to identify by the keyword extraction algorithm without a professional term mining step, while most keywords in CCD are commonly used words (*e.g.*, have a fever, cough, pneumonia, *etc.*). Second, N_Semantic_Query consistently outperforms N_Query. It indicates that the Siamese neural network could more effectively measure the semantic relevance between questions than the simple word matching method does. In addition, SCQA implemented purely by Siamese neural network outperforms N_Semantic_Query in several evaluation metrics. However, the performance improvement is trivial as compared to the computational complexity of the deep learning model. It shows that combining the IR technique and deep learning model is a better strategy of designing QAS to balance the accuracy

and efficiency. Third, BERT_QA does not have a significant advantage over SCQA. It even performs worse on some metrics. The reason might be as follows. Although BERT is powerful at word embedding pre-training based on a large enough corpus, it cannot well learn the word vectors for the professional terms from a specific domain based on a limited amount of documents. In summary, OUR has a significant performance improvement as compared with the baselines, especially on the EPD dataset. It demonstrates the effectiveness of the proposed QAS for domain-specific professional domains.

4.5 Case Study

In this section, we study the proposed QAS by investigating two real QA cases. The input query of the first case is "*When did the environment impact assessment law start to be launched*?". The matching question found by SCQA is "*When did the environment impact assessment start to be launched*?". These two questions are nearly the same, except that the input query contains a word "law" that does not exist in the matching question. However, *environment impact assessment law* and *environment impact assessment* refer to totally different concepts in the environment protection domain. By mining the professional term *environment impact assessment law*, our method could find the correct matching question "*What is the time the environment impact assessment law was introduced*?". Similarly, for the input query of the second case "*What kind of companies need the emergency plan for environment emergencies*?", SCQA finds an incorrect matching question "*Could a company without the emergency plan go through the acceptance test procedure*?", and our method finds the correct matching question "*Who needs to formulate the emergency plan for environment emergencies*?", by discovering the professional term *emergency plan for environment emergencies*.

5. CONCLUSIONS

In this paper, we design a domain-specific non-factoid QAS by combining IR technique and deep neural network. First, it extracts professional terms from the domain-specific documents based on a frequent phrase mining algorithm and a common phrase filtering method. Second, it trains a deep Siamese neural network to match the question pairs in the semantic space. Finally, it queries and ranks the candidate answers for an online input question by considering the professional term matching and the semantic relevance simultaneously. We have conducted extensive experiments based on real QA databases in specific professional domains. The experiment results have shown that the professional terms could bring significant performance improvement. It is because that the professional terms are domain-specific knowledge that could not be easily learnt by the deep neural network, especially when the training corpus is not large enough.

In the future, we will extend our work from the following aspects. First, we will enhance the non-factoid QAS by Web knowledge. When the QA database on hand cannot support the online question, the QAS could use a Web search engine to query the candidate answers. Second, we will design a unified QAS that could answer both the factoid and non-factoid questions by integrating the professional terms into a knowledge graph.

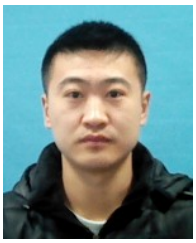
REFERENCES

1. P. Ranjan and R. C. Balabantaray, "Question answering system for factoid based question," in *Proceedings of International Conference on Contemporary Computing and Informatics*, 2016, pp. 221-224.
2. L. Yang, Q. Ai, D. Spina, *et al.*, "Beyond factoid QA: Effective methods for non-factoid answer sentence retrieval," in *Proceedings of Advances in Information Retrieval*, 2016, pp. 115-128.
3. Q. Chen, Y. Liu, C. Bruce, *et al.*, "Answer interaction in non-factoid question answering systems," in *Proceedings of Conference on Human Information Interaction and Retrieval*, 2019, pp. 249-253.
4. TREC, https://trec.nist.gov/tracks.html.
5. CLEF, http://www.clef-campaign.org/2002.html.
6. NTCIR, http://research.nii.ac.jp/ntcir/ntcir-14/tasks.html.
7. G. Ercan, S. Elbassuoni, and K. Hose, "Retrieving textual evidence for knowledge graph facts," in *Proceedings of European Semantic Web Conference*, 2019, pp. 52-67.
8. A. Aghaebrahimian and F. Jurčíček, "Open-domain factoid question answering via knowledge graph search," in *Proceedings of Workshop on Human-Computer Question Answering*, 2016, pp. 22-28.
9. H. Wang, W. Lu, and Z. Tang, "Incorporating external knowledge to boost machine comprehension based question answering," in *Proceedings of European Conference on Information Retrieval*, 2014, pp. 819-827.
10. E. Brill, J. Lin, M. Banko, S. Dumais, and A. Ng, "Data-intensive question answering," in *Proceedings of the TREC-10 Conference*, 2001, pp. 183-189.
11. J. Zhou, "A SOA based domain-specific Chinese QA system," in *Proceedings of International Conference on Grid and Cloud Computing*, 2010, pp. 467-47.
12. W. Lu, J. Cheng, and Q. B. Yang, "Question answering system based on web," in *Proceedings of International Conference on Intelligent Computation Technology and Automation*, 2012, pp. 573-576.
13. Z. Liu, X. Wang, Q. C. Chen, *et al.*, "A Chinese question answering system based on web search," in *Proceedings of International Conference on Machine Learning and Cybernetics*, 2014, pp. 816-820.
14. X. Feng, Q. Liu, C. Lao, *et al.*, "Design and implementation of automatic question answering system in information retrieval," in *Proceedings of International Conference on Informatics*, *Environment*, *Energy and Applications*, 2018, pp. 207-211.
15. K. Komiya, Y. Abe, H. Morita, *et al.*, "Question answering system using Q & A site corpus query expansion and answer candidate evaluation," *SpringerPlus*, Vol. 2, 2013, pp. 1-11.
16. D. Savenkov, "Ranking answers and web passages for non-factoid question answering: Emory university at TREC LiveQA," in *Proceedings of Text Retrieval Conference*, 2015, pp. 1-8.
17. G. Zhou, Y. Zhou, T. He, *et al.*, "Learning semantic representation with neural networks for community question answering retrieval," *Knowledge-Based Systems*, Vol. 93, 2016, pp. 75-83.
18. X. Qiu and X. Huang, "Convolutional neural tensor network architecture for community-based question answering," in *Proceedings of International Conference on Artificial Intelligence*, 2015, pp. 1305-1311.

19. A. Das, H. Yenala, M. Chinnakotla, *et al.*, "Together we stand: Siamese networks for similar question retrieval," in *Proceedings of Annual Meeting of the Association for Computational Linguistics*, 2016, pp. 378-387.
20. M. Abulaish and T. Anwar, "A supervised learning approach for automatic keyphrase extraction," *International Journal of Innovative Computing*, *Information and Control*, Vol. 8, 2012, pp. 7579-7601.
21. P. D. Turney, "Learning algorithms for keyphrase extraction," *Information Retrieval*, Vol. 2, 2000, pp. 303-336.
22. N. Kumar and K. Srinathan, "Automatic keyphrase extraction from scientific documents using *N*-gram filtration technique," in *Proceedings of ACM Symposium on Document Engineering*, 2008, pp. 199-208.
23. Y. Gao, J. Liu, and P. X. Ma, "The hot keyphrase extraction based on TF*PDF," in *Proceedings of IEEE International Conference on Trust*, 2011, pp. 1524-1528.
24. F. Xie, X. Wu, and X. Zhu, "Document-specific keyphrase extraction using sequential patterns with wildcards," in *Proceedings of IEEE International Conference on Data Mining*, 2014, pp. 1055-1060.
25. S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 539-546.
26. J. Mueller and A. Thyagarajan, "Siamese recurrent architectures for learning sentence similarity," in *Proceedings of AAAI Conference on Artificial Intelligence*, 2016, pp. 2786-2792.
27. Y. Kim, "Convolutional neural networks for sentence classification," in *Proceedings of Conference on Empirical Methods in Natural Language Processing*, 2014, pp. 1746-1751.
28. R. Mihalcea and P. Tarau, "TextRank: Bringing order into text," in *Proceedings of Conference on Empirical Methods in Natural Language Processing*, 2004, pp. 404-411.
29. Jieba, https://github.com/fxsjy/jieba.
30. ATEC, https://dc.cloud.alipay.com/index#/topic/data?id=8.
31. ChineseSTS, https://github.com/IAdmireu/ChineseSTS.
32. D. He, Y. Xia, T. Qin, *et al.*, "Dual learning for machine translation," in *Proceedings of Advances in Neural Information Processing Systems*, 2016, pp. 820-828.
33. Baidu Translation API, http://api.fanyi.baidu.com/api/trans/product/index.
34. S. Ameen, H. Chung, S. C. Han, and B. H. Kang, "Open-domain question answering framework using Wikipedia," in *Proceedings of Australasian Joint Conference on Artificial Intelligence*, 2016, pp. 623-635.
35. M. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi, "Bidirectional attention flow for machine comprehension," arXiv, 2016, No. 1611.01603.
36. Y. Cui, W. Che, T. Liu, *et al.*, "Pre-training with whole word masking for Chinese BERT," *arXiv Preprint*, 2019, No. 1906.08101.
37. L. F. Chien, "PAT-tree-based keyword extraction for Chinese information retrieval," in *Proceedings of ACM SIGIR International Conference on Information Retrieval*, 1997, pp. 50-58.
38. Z. Dai, C. Xiong, J. Callan, *et al.*, "Convolutional neural networks for soft-matching N-Grams in ad-hoc search," in *Proceedings of ACM International Conference on Web Search and Data Mining*, 2018, pp. 126-134.

39. Z. Dai and J. Callan, "Deeper text understanding for IR with contextual neural language modeling," in *Proceedings of ACM SIGIR Conference on Research and Development in Information Retrieval*, 2019, pp. 985-988.

Ming-Qi Lv (吕明琪) received the Ph.D. degree in Computer Science and Technology from Zhejiang University, China. He is currently an Associate Professor with the College of Computer Science and Technology at Zhejiang University of Technology, China. His research interests include ubiquitous computing and data mining.



Hao Zhang (张浩) received the Bachelor degree Information and Computer Science from Shenyang University of Chemical Technology, China. He is currently pursuing his master degree in the College of Computer Science and Technology, Zhejiang University of Technology, China. His research interests include data mining and machine learning.

Kang-Jun Zhu (朱康钧) received the Bachelor degree in College of Metrology and Measurement Engineering from China Jiliang University, China. He is currently pursuing his master degree in the College of Computer Science and Technology, Zhejiang University of Technology, China. His research interests include natural language processing and machine learning.

Chao Huang (黄超) received the Bachelor degree in Food Science and Engineering from Zhejiang University of Technology, China. He is currently pursuing his master degree in the College of Computer Science and Technology, Zhejiang University of Technology, China. His research interests include data mining and machine learning.

Tie-Ming Chen (陈铁明) received the Ph.D. degree in Software Engineering from Beihang University, China. He is currently a Professor with the College of Computer Science and Technology, Zhejiang University of Technology, China. His research interests include data mining and cyberspace security.