Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/mbe

MBE, 20(3): 4912–4939.
DOI: 10.3934/mbe.2023228
Received: 31 October 2022
Revised: 03 December 2022
Accepted: 13 December 2022
Published: 05 January 2023

Research article

Dual-process system based on mixed semantic fusion for Chinese medical knowledge-based question answering

Meiling Wang[1], **Xiaohai He**[1,*], **Zhao Zhang**[2], **Luping Liu**[3], **Linbo Qing**[1] **and Yan Liu**[4]

[1] College of Electronics and Information Engineering, Sichuan University, Chengdu 610064, China

[2] Sichuan Rongke Huaxin Technology Co., LTD, Chengdu, China

[3] Bytedance, Shenzhen, China

[4] Department of Neurology, The Affiliated Hospital of Southwest Jiaotong University & The Third People's Hospital of Chengdu, Sichuan, China

* **Correspondence:** Email: hxh@scu.edu.cn.

Abstract: Chinese medical knowledge-based question answering (cMed-KBQA) is a vital component of the intelligence question-answering assignment. Its purpose is to enable the model to comprehend questions and then deduce the proper answer from the knowledge base. Previous methods solely considered how questions and knowledge base paths were represented, disregarding their significance. Due to entity and path sparsity, the performance of question and answer cannot be effectively enhanced. To address this challenge, this paper presents a structured methodology for the cMed-KBQA based on the cognitive science dual systems theory by synchronizing an observation stage (System 1) and an expressive reasoning stage (System 2). System 1 learns the question's representation and queries the associated simple path. Then System 2 retrieves complicated paths for the question from the knowledge base by using the simple path provided by System 1. Specifically, System 1 is implemented by the entity extraction module, entity linking module, simple path retrieval module, and simple path-matching model. Meanwhile, System 2 is performed by using the complex path retrieval module and complex path-matching model. The public CKBQA2019 and CKBQA2020 datasets were extensively studied to evaluate the suggested technique. Using the metric average F1-score, our model achieved 78.12% on CKBQA2019 and 86.60% on CKBQA2020.

Keywords: entity extraction; entity linking; path retrieval; path matching; semantic self-attention fusion

1. Introduction

Chinese medical knowledge-based question answering (cMed-KBQA) is a challenging natural language processing task that requires matching the satisfied nodes of the structured knowledge base as factual answers to natural language questions. The knowledge base is the source of information for cMed-KBQA. It is a directed graph with nodes representing entities and edges representing their connections. Formally, it is composed of a variety of triples (S, R, O), where S is the subject (S), R is the relation (R) and O is the object (O). The popular knowledge bases include the English-language knowledge base (Freebase [1], DBpedia [2], Wikidata [3]) and the Chinese-language knowledge base (Zhishi. me [4], CNDBpedia [5]). There are increasingly more studies on knowledge-based question answering (KBQA) as domain knowledge bases grow in quantity and quality.

Since cMed-KBQA is a part of KBQA, its implementation methods are mainly based on the most common KBQA methods. Currently, there are three ways to do KBQA: methods based on semantic parsing, methods based on retrieving information, and methods based on deep learning. Semantic parsing-based methods [6–10] take the question into logical forms of entities and relations. Then, they use the logical forms to build query statements that get the final answer. Most methods use particular query languages, such as SPARQL [11] and Cypher [12], to obtain rich logical structure forms that can be used to make query statements of the same form. Information retrieval-based methods [13,14] try to get the candidate paths that match the questions from the knowledge base, figure out their semantic similarity to the questions and then output the best paths to get the final answers. This method makes it easy to create training data and ask for answers, so it gets much attention and has significant performance advantages. With the high growth of deep learning, many scholars are integrating deep neural networks into the first and second approaches, calling them deep learning-based methods. These approaches turn questions and answers into vector spaces through representation learning methods. The method makes complicated KBQA tasks more straightforward by breaking them up into tasks like figuring out the degree of similarity of two things, classifying them, creating a sequence, etc. Early strategies to get question-and-answer feature vectors were mostly word embedding models like Word2Vec [15] and GloVe [16]. With the rise of pre-trained models in the past few years, BERT [17], XLNet [18] and GPT [19] have been able to get good results. Multiple studies have shown that combining deep learning with KBQA tasks not only makes the task easier but it also makes it work much better.

Natural language questions are constructed in a variety of ways. In KBQA tasks, simple questions can be answered accurately based on a single triple, as shown on the left in Figure 1. Current research has achieved a relatively high level of accuracy for simple questions. In contrast, complex questions require multiple triples to bridge queries to get the correct answer, as shown on the right in Figure 1. It is hard to distinguish and respond to complicated questions due to their complexity and irregular forms. Also, existing deep learning-based methods only looked at how questions and knowledge base paths were represented and extracted the subject entity [20,21] as accurately as possible, ignoring their importance. Because of this, the question's topic entities and the candidate paths that go with them are not excavated in depth. In addition, no one has tried to use stage-based reasoning to answer questions based on knowledge like a human.

For a natural language question, the manual way to answer it is in two steps: (i) figure out the question's structure and get what it says; and (ii) use the question's content to analyze it in depth and capture the answer by querying your knowledge store. In cognitive science [22–24], this operation is seen as a dual-process system (DPS), which says that human reasoning is based on two different cognitive systems. System 1 is a set of implicit, unconscious and intuitive ways that our brain gets information after we pay attention to it. Then, System 2, an explicit, conscious and controllable way of thinking, uses the working memory to do sequential thinking, which is a slower but a uniquely human trait [25]. From this perspective, the cMed-KBQA task can be processed based on DPS: System 1 is in charge of quickly getting information from the question and simple paths, and System 2 uses the knowledge base to do deep reasoning to find complex paths.

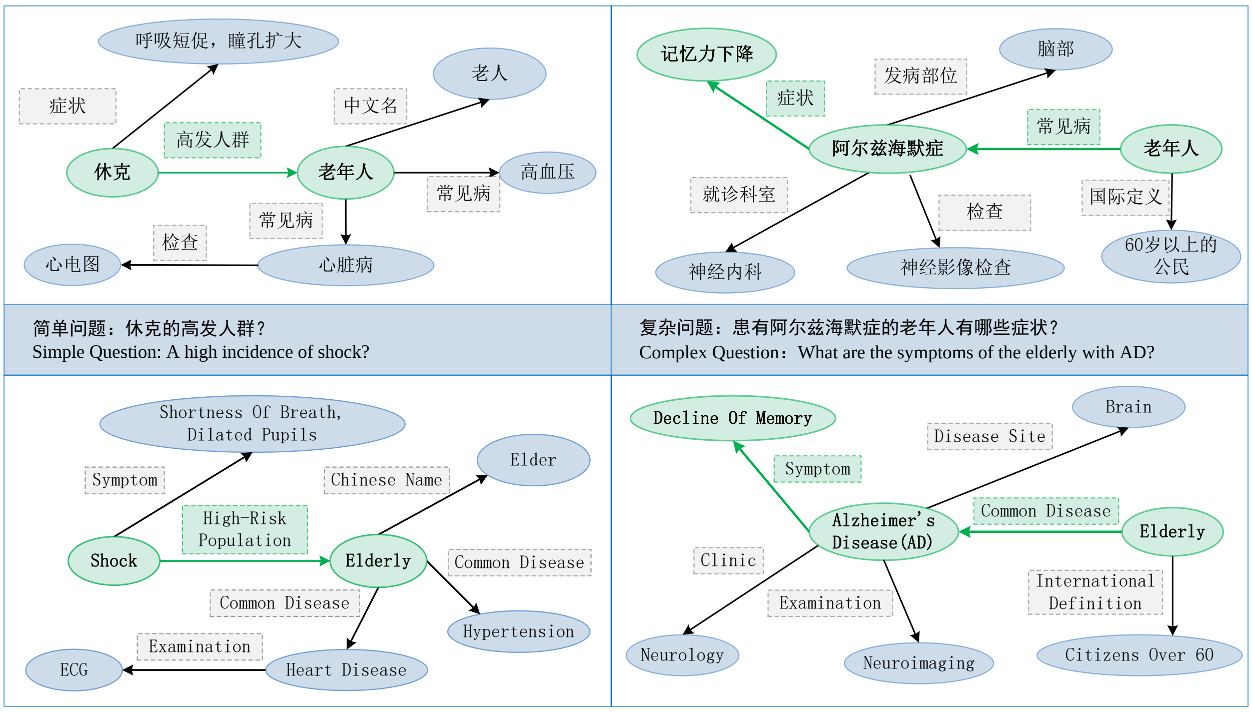


Figure 1. Examples of cMed-KBQA task. The case on the left is based on a simple question and the case on the right is based on a complex question. The English-language version was obtained through Google Translate and is for reference only.

Motivated by the above theory, we propose a novel DPS architecture for the cMed-KBQA task, as demonstrated in Figure 2. Concretely, System 1 in our DPS model consists of an entity extraction module, entity linking module, simple path retrieval module and simple path-matching model, which utilizes a simple path-matching model to capture the candidate simple paths in question. For System 2, we use the complex path retrieval module and the complex path-matching model to reason about the knowledge base (PKUBASE) and find the complex paths. System 2 first executes System 1 to find simple candidate paths; then, a complex path retrieval module is conducted to obtain enriched complex paths from the knowledge base. During the path-matching process, we propose a mixed semantic fusion mechanism to get helpful information from both the question and the candidate paths.

The following are the significant contributions of this paper:

1) A unique DPS structure for the cMed-KBQA task is presented, influenced by how the human brain works. System 1 is developed to acquire simple paths, whereas System 2 retrieves complicated path-related information.
2) The path-matching module has been developed through the use of a mixed semantic fusion mechanism to capture valuable information from questions and candidate paths.
3) The experimental results on CKBQA2019 and CKBQA2020 revealed that our DPS model significantly outperforms published methods.

The remainder of this paper is organized as follows. Section 2 summarizes the related work of KBQA and cMed-KBQA. Section 3 presents the detailed description of our DPS model. The experimental results and discussion are depicted in Section 4. Section 5 presents the conclusion.

2. Literature review

2.1. KBQA

KBQA is a novel field of study in natural language processing that has attracted the interest of academic researchers. There are two broad types of KBQA tasks: those that involve answering simple questions and those that involve more complicated questions. The former can be answered by a single triple in a given knowledge base, while the latter requires reasoning by bridging multiple triples in a given knowledge base to obtain an answer. Apparently, the complex question-answering task is more challenging. As a result, there is a rush of studies on KBQA tasks. Current mainstream methods for building a knowledge base can be broken down into three types: methods based on semantic parsing, methods based on information retrieval, and methods based on deep learning.

Semantic parsing-based methods convert the question into logical forms consisting of entities and relations and then build query statements based on the logical forms to acquire the final answer. The core task of semantic parsing-based methods is to comprehend natural language questions. Traditional semantic parsing methods [8, 10, 26–29] are efficient at analyzing simple questions, while there is still great potential for improvement on complex questions. To better comprehend complex questions, existing methods extract the composition and logical form of the question based on dependencies [30] and topic representations [31]. However, the resolution results are still unsatisfactory for the long-range dependency questions. To mitigate the propagation of parsing errors, Zhu et al. [32] attempted to enhance the parsing of the question by adding structure-aware feature encoders, and Chen et al. [33] improved the matching between the logical form and the question by adding constraints on the query structure to filter noisy queries. In conclusion, these techniques aim to use linguistic analysis of human language to convert the question into a logically sound form that can be used to interpret the knowledge base.

Information retrieval-based methods attempt to retrieve the knowledge base to acquire the candidate paths corresponding to the answers, calculate their semantic similarity to the questions, and then output the optimal paths to achieve the final answers. Retrieving information relies on a high-quality knowledge base. However, missing issues [34] are unavoidable in public knowledge. Sun et al. [35,36] proposed forming a heterogeneous graph out of the subgraphs derived from the imperfect knowledge base and then reasoning about this network. Xiong et al. [37] and Han et al. [38] offered to fuse additional textual information into the entity representation to complement the knowledge. To improve the learned entity presentations and address insufficient knowledge base issues, Saxena et al. [39] used pre-trained comprehension word embedding. Compared with previous methods, this group of methods makes it easier to put end-to-end models into action and needs less language knowledge. However, the correctness of these methods still needs further improvement.

Deep learning-based methods convert questions and answers into vector space by means of representation learning methods, which in turn simplify sophisticated KBQA tasks into similarity calculation tasks, classification tasks, or sequence creation tasks, etc. Bordes et al. [40] trained an embedding model to evaluate the similarity of questions and answers by computing the distance between their representation vectors. In order to generate potential action scenes, Guo et al. [41] created a preprocessor decoder and transformed the KBQA job into a similarity calculation task. Huang et al. [42] designed a hybrid model to simultaneously acquire the representation of head and tail entities in vector space. Wang et al. [43] used knowledge base encoding methods to create graph-structured question terms in order to address the matching question for entities and relations. In addition, there are many convolutional neural network (CNN)-integrated methods [44, 45] and long short-term memory-combined methods [46, 47] to improve the performance of KBQA. Although these methods have sped up the development of KBQA, their performance still requires improvement.

2.2. *cMed-KBQA*

Chinese KBQA (CKBQA). Recently, with the rapid development of KBQA technology, there have been more and more CKBQA jobs. Cao et al. [48] offered a pipe approach that is referred to as DUTIR. This method consists of four pieces, and it restricts the maximum number of hops for relational paths to two. Using custom-built features, Zhang et al. [49] and Wang et al. [50] categorized the question so that the appropriate approach can be taken. By combining phrase and statement semantics, Luo et al. [51] proposed an approach dubbed FSM for the calculation of question-and-answer consistency. Wu et al. [52] developed a training method for semantic similarity models by using the dynamic sampling of negative examples to enrich the diversity of relations in the training set. The above methods have achieved some results on CKBQA tasks. However, the research used to solve the cMed-KBQA problem is relatively limited.

cMed-KBQA. With the publication of the Chinese medical dataset CKBQA2020, the study of cMed-KBQA tasks have emerged progressively. Tang et al. [53] and Xiong et al. [54] have proposed a comprehensive knowledge-based approach for answering complex questions and designing four dedicated similarity calculation models for handling complex cases of complex problems. Dai et al. [55] has given a unified strategy, MIQA, that takes a holistic view of the various activities involved in option path development and uses established rules. Focusing on entity extraction, Zhang et al. [56] constructed a SEE model based on text consistency strategies and annotation models for different dimensional information. In contrast to the previous work, we pay more attention to the details of hard problems and the different paths that semantic information can be expressed and shown.

3. Proposed DPS model

This section presents the proposed DPS framework. Section 3.1 defines the cMed-KBQA problem. Section 3.2 summarizes the main architecture. Section 3.3 presents the detailed System 1—simple path matching. Section 3.4 depicts the comprehensive System 2—complex path matching.

3.1. Problem statement

On the cMed-KBQA task, given a question $\mathbf{Q} = \{\mathbf{q}_1, \mathbf{q}_2, ..., \mathbf{q}_m\}$ and a knowledge base $\mathcal{G} = \{< e_s, r, r_o > | e_s, e_o \in \mathcal{E}, r \in \mathcal{R}\}$, where e_s and e_o respectively represent the entity subjects and objects, r symbolizes the relationship describing their connection, and $\mathcal{E}$ and $\mathcal{R}$ are the sets of all entities and relations, respectively. System 1 seeks to identify entities of interest and connect them to the knowledge base to obtain the theme entities and simple path candidates $\mathbf{P} = \{\mathbf{p}_1^S, \mathbf{p}_2^S, ..., \mathbf{p}_n^S\}$, (where n is the number of simple path candidates) in the knowledge base. System 2 is responsible for locating complex candidate paths in a question, i.e., the sequence of paths that lead from the question's theme entity to the knowledge base answer. System 2's task is defined as a complex path-matching issue. The algorithm simulates the matching score between the vector layer and the presentation of the query $\mathbf{q}$ for each path $\mathbf{p}_i^C$ in the path candidate set $\mathbf{P}^C$, and the path with the most excellent score is chosen as the ultimate premise path. Specifically,

$$\mathbf{P}^+ = \text{argmax } S(\mathbf{Q}; \mathbf{P}^C). \tag{3.1}$$

Algorithm 1 DPS algorithm based on mixed semantic fusion.

```
Input: A question $Q_i$, a SPARQL sentence $S_i$ and an answer $a_i$.
Output: System 1 parameter $\alpha$, System 2 parameter $\beta$
 1: for number of training interactions of each epoch do
 2:     Produce the question $Q_i$ based on the entity extraction and entity linking to get the topic entity
        set $E_q$.
 3:     Produce the topic entity set $E_q$ based on the simple path retrieval.
 4:     Train System 1 based on the candidate's simple path and SPARQL sentence in the dataset.
 5:     Produce the BCE loss to constrain the model for training.
 6: end for
 7: return System 1 parameter $\alpha$
 8: for number of training interactions of each epoch do
 9:     Produce the question $Q_i$ based on System 1 (with parameter $\alpha$) to get the simple path set $P_s$.
10:     Produce the simple path set $P_s$ based on the complex path retrieval.
11:     Train System 2 based on the candidate's simple path and SPARQL sentence in the dataset.
12:     Produce the BCE loss to constrain the model for training.
13: end for
14: return System 2 parameter $\beta$
```

3.2. Overall architecture

To mitigate entity and path sparsity in existing approaches, the proposed DPS framework via a mixed semantic fusion mechanism acquires as much valuable information about candidate paths as possible. As illustrated on the left of Figure 2, the DPS model includes System 1 (simple path matching) and System 2 (complex path matching). System 1 includes an entity extraction module, an entity linking module, a simple path retrieval module and a simple path-matching model, i.e., four essential components. For System 1, we obtain its optimal network parameters through training, laying the foundation for the prediction of simple paths in the next step. For System 2, we first employ the trained System 1 to capture simple candidate paths. Second, we apply the complex path-retrieval module and complex path-matching model to reason on a knowledge base (PKUBASE) to retrieve the complex paths. Lastly, we use the trained Systems 1 and 2 to determine the answer to a question. The DPS algorithm based on mixed semantic fusion is summarized in Algorithm 1.

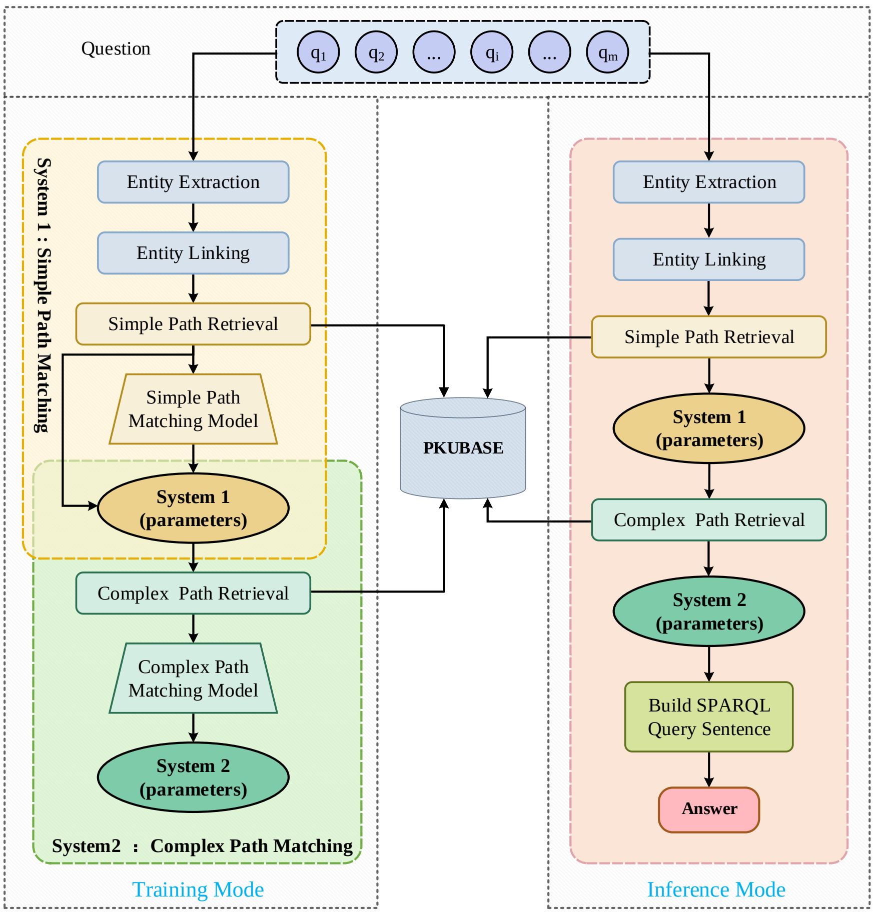


Figure 2. Architecture of the proposed DPS framework for the cMed-KBQA task. The left is a training mode, while the right is the inference mode.

3.3. *Simple path matching (System 1)*

System 1 is an observation module that aims to extract the topic entities of the question and query the corresponding simple paths from the knowledge base. It is made up of four important parts: an entity extraction module, an entity linking module, a simple path retrieval module, and a simple path-matching model.

3.3.1. Entity extraction

Entity extraction is an essential component of cMed-KBQA tasks. To capture the subject set E_q that corresponds to the question Q, the initial phase is questioning utterance segmentation. The goal of this step is to detect all entities as much as possible. The entity extraction task is implemented via two methods. In the first method, questions are segmented by using Jieba [*], which appends the dictionary of dataset mentions. PkuSeg [†] is applied in the second way to separate questions. Finally, we use the combined set of the two types of subscripts described previously as the mentioned set of questions.

3.3.2. Entity linking

The subject is the entity most concerned about in the question. The entity linking enumerates all the candidate entities in the knowledge base according to the mentioned set; then, it performs entity disambiguation and links them back to the knowledge base, which outputs topic entities to the downstream tasks. After obtaining the mentioned set of questions, the most challenging problem of entity linking is connecting the mention to the relevant entity in the knowledge base in an effective manner. Here, candidate topic entities are obtained by using exact matching [54]. We use an entity candidate set to identify mentions accurately, and then we add the entity that has been precisely identified to the subject set E_q. Finally, the satisfied entities are appended to the topic entity set E_q.

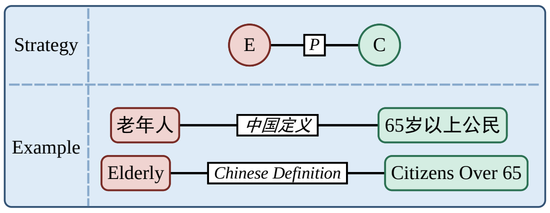


Figure 3. Simple-path query strategy. The red circle represents the input entity, while the green circle represents the output entity. The English-language version was obtained through Google Translate and is for reference only.

3.3.3. Simple path retrieval

We consider the production of candidate paths to be an item tree method of searching because it is based on the findings of subject extraction. Given the identified topic entity set, the candidate paths are recalled by drawing circles around the topic entities. The paths from the topic entities to the answer entities are generated as the basis for ranking the answers. The particular rules for retrieving candidate paths can be seen in Figure 3.

[*]https://github.com/fxsjy/jieba
[†]https://github.com/lancopku/PKUSeg-python

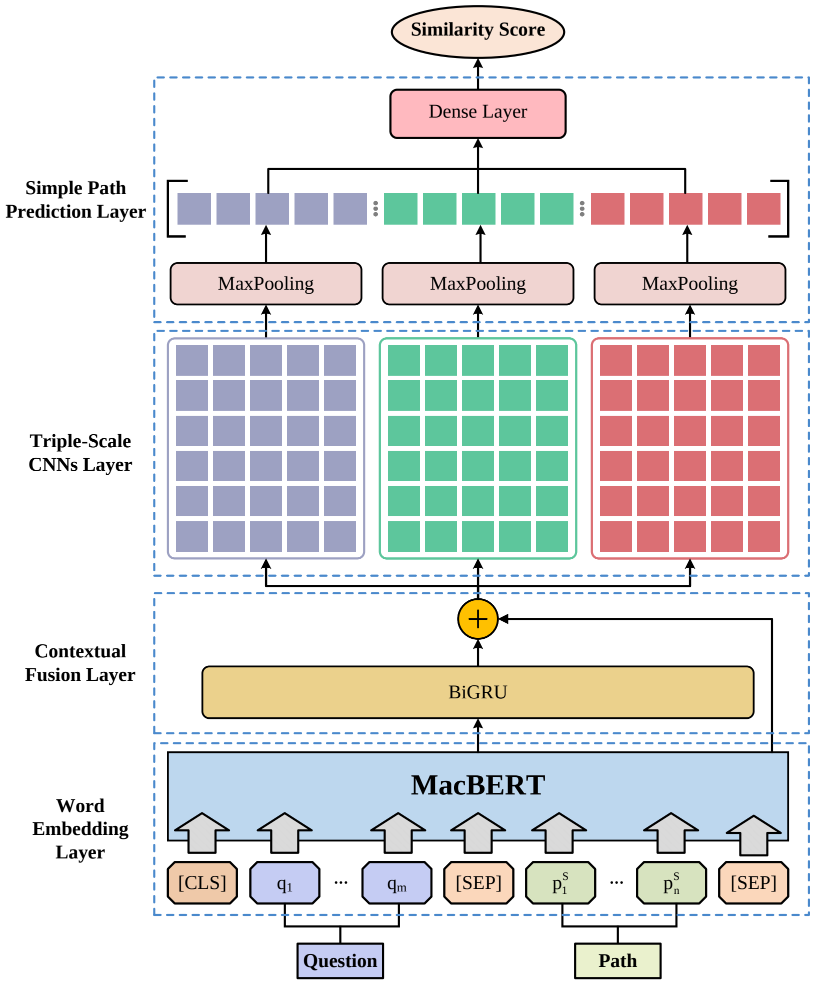


Figure 4. Architecture of the simple path-matching model.

3.3.4. simple path-matching model

As we have already said, we can get the possible paths by using a simple path query method. It is noticed that the many ways people use language that makes it hard for a trained model to understand the high-level differences between a question and a path. We made a simple path-matching model based on mixed semantic fusion from multiple foci to combine the features of question-and-path pairs and then figure out their degree of similarity. The architecture of the simple path-matching model is shown in Figure 4, and it consists of a word embedding layer, a contextual fusion layer, a triple-scale CNN layer and a simple path prediction layer. Below, we describe each part in detail.

Word embedding layer. A suitable keyword representation is critical to the cMed-KBQA task. As shown in Figure 4, in light of the peculiarities of the Chinese language, the embedding layer that we use is a modified version of MacBERT [57] with comparable settings. When encoding question and path sequences with MacBERT, it is necessary for us to insert two special characters ([CLS] and [SEP]) at the start and finish of the sequence. Formally, the connection between the question $\mathbf{Q} = \{\mathbf{q}_1, \ldots, \mathbf{q}_m\}$ and the simple path $\mathbf{P}^S = \{\mathbf{p}^S_1, \ldots, \mathbf{p}^S_n\}$ is represented as $\mathbf{QP}^S = \{\mathbf{q}_0, \mathbf{q}_1, \ldots, \mathbf{q}_m, \mathbf{q}_{m+1}\, \mathbf{p}^S_1, \ldots, \mathbf{p}^S_n, \mathbf{p}^S_{n+1}\}$, where $\mathbf{q}_0$ denotes [CLS] and $\mathbf{q}_{m+1}$ and $\mathbf{p}^S_{n+1}$ indicate [SEP]. The tokenized question and simple path are fed into the MacBERT encoder to produce $\mathbf{QP}^S_{emb}$:

$$\mathbf{QP}^S_{emb} = \text{MacBERT}(\mathbf{QP}^S) = [\mathbf{E}^S_0, \mathbf{E}^S_1, \ldots, \mathbf{E}^S_m, \ldots, \mathbf{E}^S_n, \mathbf{E}^S_{m+n+1}], \tag{3.2}$$

where $\mathbf{E}^S_i \in \mathbb{R}^d$ is the context-aware representation for the i-th token and d is the dimension of the vector.

Contextual fusion layer. It is common practice to utilize recurrent neural networks (RNNs) to model data that contain sequential information, despite the fact that standard RNN models are vulnerable to vanishing and bursting gradients. To overcome this obstacle, we utilized the prevalent moderately effective gated recurrent unit (GRU) model to further encode question and path features. To obtain the joint feature, the join embedding sequence $\mathbf{QP}^S_{emb} \in \mathbb{R}^{b\times(m+n)\times d}$ is fed into a GRU:

$$\mathbf{QP}^S_{feat} = \text{GRU}([\mathbf{E}^S_0, \mathbf{E}^S_1, \ldots, \mathbf{E}^S_m, \ldots, \mathbf{E}^S_n, \mathbf{E}^S_{m+n+1}]) = [\mathbf{G}^S_0, \mathbf{G}^S_1, \ldots, \mathbf{G}^S_m, \ldots, \mathbf{G}^S_n, \mathbf{G}^S_{m+n+1}], \tag{3.3}$$

where $\mathbf{QP}^S_{feat} \in \mathbb{R}^{b\times(m+n)\times d}$, b represents the batch-size, $m + n$ denotes the length of a sequence and d is the dimension of the vector. To get the contextual information, we add the embeddings and features of the sequence, as follows:

$$\mathbf{QP}^S_{context} = \text{Linear}(\mathbf{QP}^S_{emb}) + \mathbf{QP}^S_{feat} = [\mathbf{C}^S_0, \mathbf{C}^S_1, \ldots, \mathbf{C}^S_m, \ldots, \mathbf{C}^S_n, \mathbf{C}^S_{m+n+1}]. \tag{3.4}$$

Triple-scale CNN layer. CNNs are often used in the natural language processing field to learn the unique parts of phrases by rolling the convolutional kernel over the text. However, the length of many Chinese words and sentences varies. So, it might not be possible to get the characteristics of a variable-length phrase by using a convolutional kernel of a specific size. A single-scale CNN is insufficient, particularly for encoding at the word level. Therefore, to describe the sequence samples, we use different scales of convolutional kernels to pull out characteristics of phrases with different lengths. Triple-scale CNNs have the potential to pick up a wide range of details at the character, word or phrase level. Given the kernel size of a set of convolutional filters $\mathbf{S} = \{\mathbf{S}_1, \mathbf{S}_2, ..., \mathbf{S}_t\}$, ($t \in \{1, 2, 3\}$), $\mathbf{S}_i$ indicates a convolutional filter. The outputted feature maps are produced through convolution, which can be expressed as follows:

$$\mathbf{M}^{S_i}_S = \text{GELU}(\mathbf{W}^{S_i}_S(\mathbf{QP}^S_{context}) + b^{S_i}_S), \tag{3.5}$$

where $\mathbf{W}^{S_i}_S$ denotes the training parameters, GELU is an activation function, and $b^{S_i}_S$ indicates the bias of each convolutional filter. $\mathbf{M}^{S_i}_S$ is the output feature of each convolutional filter.

Simple path prediction layer. To map $\mathbf{M}^{S_i}_S$ to the sentence space of the same dimension, we use the max pooling operation. This method not only extracts useful features from each dimension of $\mathbf{M}^{S_i}_S$, but it also reduces computational costs. The dimension mapping operation corresponding to each dimension is as follows:

$$\mathbf{P}^{S_i}_S = \text{MaxPooling}(\mathbf{M}^{S_i}_S), \tag{3.6}$$

where $\mathbf{P}_S^{S_i}$ represents sequence features. Lastly, we concatenate them from t scales to produce the final semantic representations:

$$\mathbf{D}_{QP}^{S} = \text{Concat}(\mathbf{P}_S^{S_1}, \mathbf{P}_S^{S_2}, \ldots, \mathbf{P}_S^{S_t}), \tag{3.7}$$

where Concat() is a concatenation operation.

Finally, we employ a dense layer to calculate the question-and-path pair similarity scores and then choose the path with the highest score as the best option. In addition, to optimize the model's performance during training, we use binary cross entropy as the loss function.

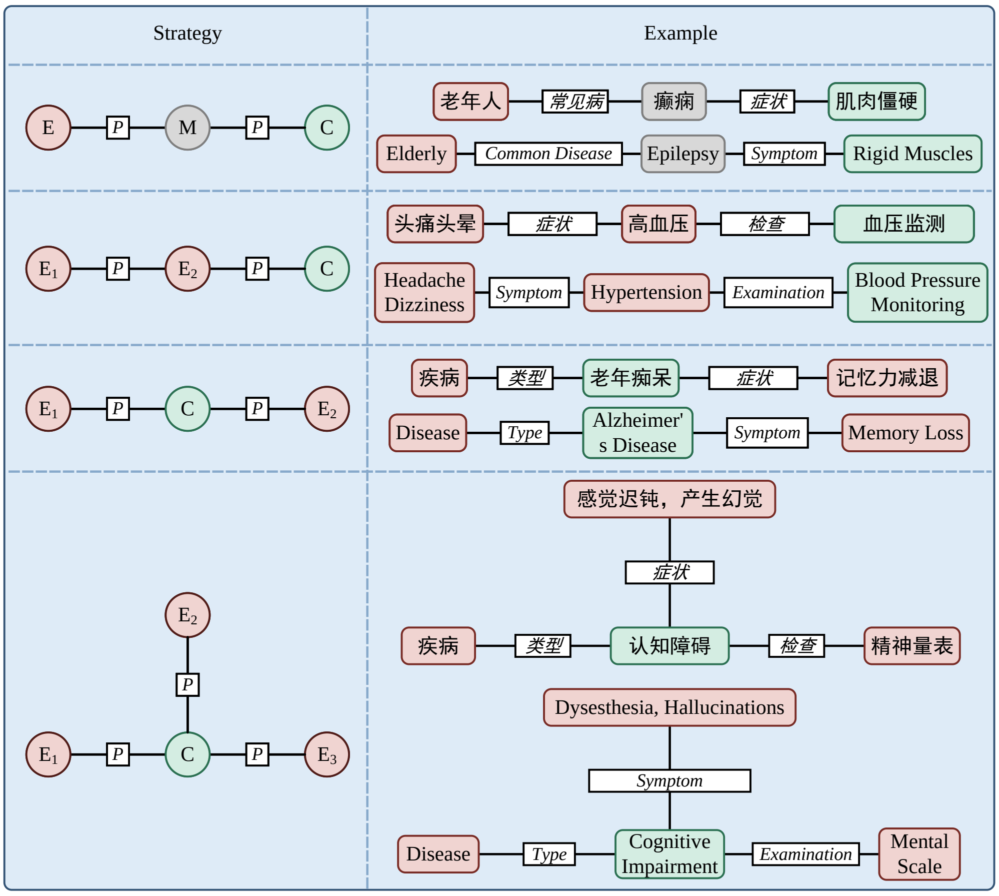


Figure 5. A complicated path query strategy: (1) The red circle represents the input entity. (2) The green circle donates the output entity. (3) The grey circle symbolizes the middle entity. The English-language version was obtained through Google Translate and is for reference only.

3.4. Complex path matching (System 2)

System 2 is a module for expressive reasoning. It looks at all of the simple paths of the entities in the topic and finds the ones that match the utterance. It contains a complex path retrieval module and a complex path-matching model.

3.4.1. Complex path retrieval

Based on the Section 3.3.2 processing, we obtained the mentioned set of questions. First, System 1 is used to predict the simple candidate paths and select the top K as the simple paths, as described in Section 3.3.4. Next, based on the predicted paths, we use the complex path query strategy to make the SPARQL query (e.g., select distinct ?r from PKUBASE where {{{} {} ?v . ?v ?r ?x .}} limit 100) that goes with them. The search is then extended to find the multi-hop paths that best match the utterance. The specific query strategy is shown in Figure 5.

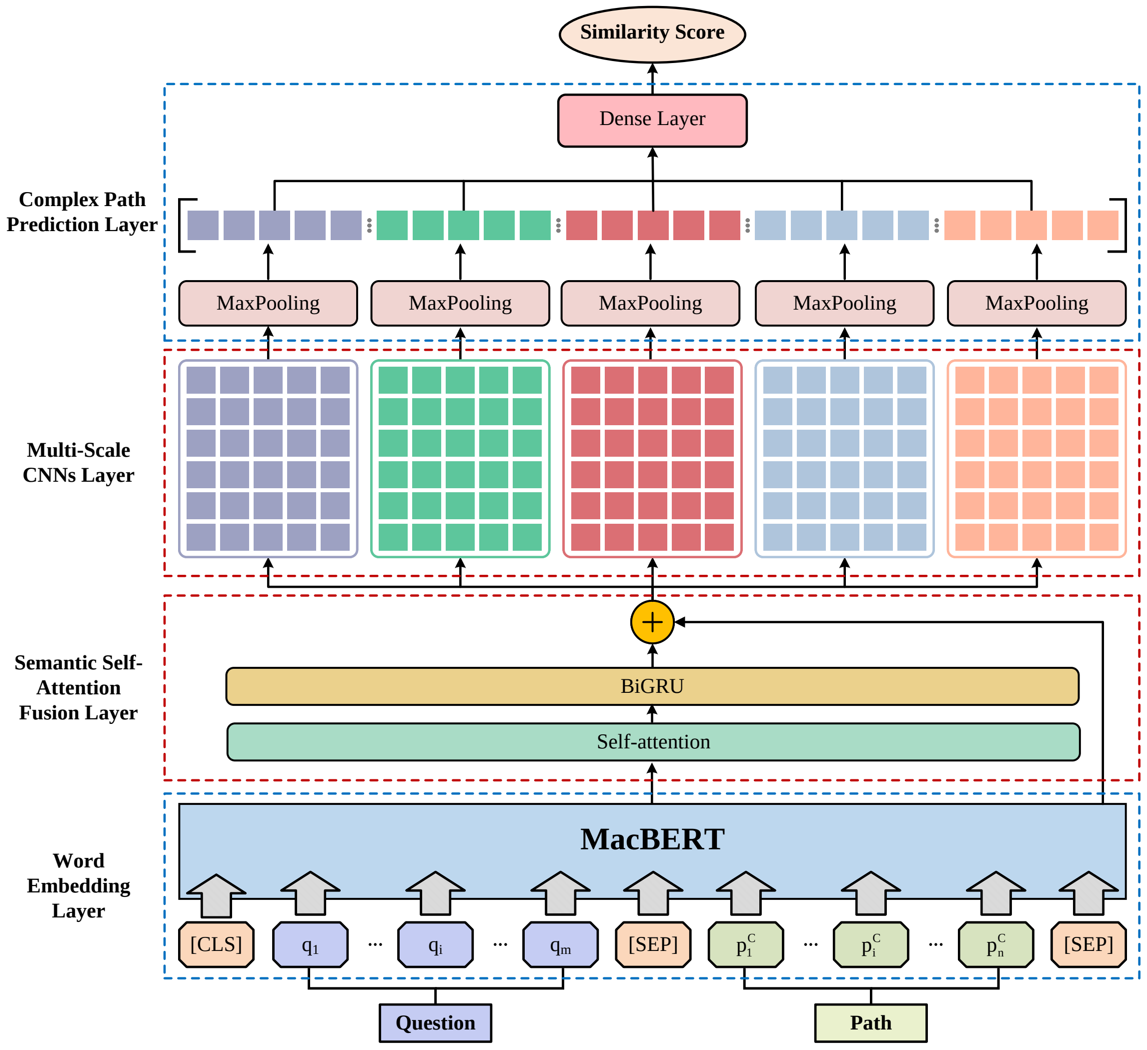


Figure 6. Architecture of the complex path-matching model.

3.4.2. Complex path-matching model

To capture the best path more precisely, we have designed the complex path-matching model, as shown in Figure 6, which consists of a word embedding layer, a semantic self-attention fusion layer, a multi-scale CNN layer and a complex path prediction layer. System 2 aims to obtain the optimal path from the candidate complex paths selected in Section 3.4.1.

Word embedding layer. Similar to System 1, We employ MacBERT [57] with shared parameters as the embedding layer. We need to add two special characters ([CLS] and [SEP]) at the beginning and end of the sequence when encoding the question sequence and complex path sequence with MacBERT. Formally, the connection between the question $\mathbf{Q} = \{\mathbf{q}_1, \ldots, \mathbf{q}_m\}$ and the complex path $\mathbf{P}^C = \{\mathbf{p}^C_1, \ldots, \mathbf{p}^C_n\}$ can be represented as $\mathbf{QP}^C = \{\mathbf{q}_0, \mathbf{q}_1, \ldots, \mathbf{q}_m, \mathbf{q}_{m+1}, \mathbf{p}^C_1, \ldots, \mathbf{p}^C_n, \mathbf{p}^C_{n+1}\}$, where $\mathbf{q}_0$ denotes [CLS] and $\mathbf{q}_{m+1} and \mathbf{p}^C_{n+1}$ represent [SEP]. The tokenized question and complex path are fed into the MacBERT encoder to get $\mathbf{QP}^C_{emb}$:

$$\mathbf{QP}^C_{emb} = \text{MacBERT}(\mathbf{QP}^C) = [\mathbf{E}^C_0, \mathbf{E}^C_1, \ldots, \mathbf{E}^C_m, \ldots, \mathbf{E}^C_n, \mathbf{E}^C_{m+n+1}]. \tag{3.8}$$

Semantic self-attention fusion layer. Unlike System 1, System 2 must analyze longer sequences of information and more sophisticated semantic levels. In order to obtain a more valuable text representation, System 2 first introduces a self-attention mechanism to calculate a correlation with the text sequence itself and then uses the internal representation of the text sequence to obtain a more meaningful text representation. First, we use different linear transformations to encode the joint sequence embedding $\mathbf{QP}^C_{emb}$ to get the $\mathbf{Q}^C$, $\mathbf{K}^C$, and $\mathbf{V}^C$ values that the attention mechanism needs. Next, the self-attention $\mathbf{ATT}^C$ is obtained through a form of self-attention calculation. Finally, The joint sequence embedding $\mathbf{QP}^C_{emb}$ and self-attention $\mathbf{ATT}^C$ are calculated by using a dot product to obtain a new sequence representation. The specific calculation is as follows:

$$\mathbf{Q}^C = \text{Linear1}(\mathbf{QP}^C_{emb}), \tag{3.9}$$

$$\mathbf{K}^C = \text{Linear2}(\mathbf{QP}^C_{emb}), \tag{3.10}$$

$$\mathbf{V}^C = \text{Linear3}(\mathbf{QP}^C_{emb}), \tag{3.11}$$

$$\mathbf{ATT}^C = \text{softmax}\left(\frac{\mathbf{Q}^C(\mathbf{K}^C)^T}{\sqrt{d_k}}\right)\mathbf{V}^C, \tag{3.12}$$

$$\mathbf{QP}^C_{att} = \mathbf{QP}^C_{emb} \odot \mathbf{ATT}^C. \tag{3.13}$$

In addition, we used the current GRU model, which is pretty good at encoding questions and path properties. The self-attention embedding sequence $\mathbf{QP}^C_{att}$ is fed into a GRU to obtain the joint features $\mathbf{QP}^C_{feat}$:

$$\mathbf{QP}^C_{feat} = \text{GRU}(\mathbf{QP}^C_{att}) = [\mathbf{G}^C_0, \mathbf{G}^C_1, \ldots, \mathbf{G}^C_m, \ldots, \mathbf{G}^C_n, \mathbf{G}^C_{m+n+1}], \tag{3.14}$$

where $\mathbf{QP}^C_{feat} \in \mathbb{R}^{b\times(m+n)\times d}$, b is the batch size, $m+n$ is the sequence length, and d is the vector dimension. To get contextual information on the sequence, we add the following embeddings and features to the sequence joint features:

$$\mathbf{QP}^C_{context} = \text{Linear}(\mathbf{QP}^C_{emb}) + \mathbf{QP}^C_{feat} = [\mathbf{C}^C_0, \mathbf{C}^C_1, \ldots, \mathbf{C}^C_m, \ldots, \mathbf{C}^C_n, \mathbf{C}^C_{m+n+1}]. \tag{3.15}$$

Multi-scale CNN layer. Because complex paths can go in many different directions, we use finer combinations of convolutions to get more useful semantic information. As shown in Figure 6, we use five different scales of convolutional kernels to capture the semantic information of complex paths. Given the kernel size of a set of convolutional filters $\mathbf{S} = \{\mathbf{S}_1, \mathbf{S}_2, ..., \mathbf{S}_t\}, (t \in \{1, 2, 3, 4, 5\})$, $\mathbf{S}_i$ indicates a convolutional filter. The outputted feature maps are produced through convolution, which can be expressed as follows:

$$\mathbf{M}_C^{S_i} = \text{GELU}(\mathbf{W}_C^{S_i}(\mathbf{QP}_{context}^C) + b_C^{S_i}), (\mathbf{S}_i \in \{\mathbf{S}_1, \mathbf{S}_2, ..., \mathbf{S}_t\}), \tag{3.16}$$

where $\mathbf{W}_C^{S_i}$ denotes the training parameters, GELU is an activation function, $b_C^{S_i}$ indicates the bias of each convolutional filter, t represents the number of convolutional filters and $\mathbf{M}_C^{S_i}$ is the output feature of each convolutional filter.

Complex path prediction layer. We use the max pooling operation to map $\mathbf{M}_C^{S_i}$ to the sentence space of the same dimension. This method extracts not only valuable features from each dimension of $\mathbf{M}_C^{S_i}$, but it also reduces computational costs. The dimension mapping operation corresponding to each dimension is as follows:

$$\mathbf{P}_C^{S_i} = \text{MaxPooling}(\mathbf{M}_C^{S_i}), (\mathbf{S}_i \in \{\mathbf{S}_1, \mathbf{S}_2, ..., \mathbf{S}_t\}), \tag{3.17}$$

where $\mathbf{P}_C^{S_i}$ represents sequence features. Finally, we concatenate them from t scales to generate the final semantic representations:

$$\mathbf{D}_{QP}^C = \text{Concat}(\mathbf{P}_C^{S_1}, \mathbf{P}_C^{S_2}, \ldots, \mathbf{P}_C^{S_t}), (\mathbf{S}_i \in \{\mathbf{S}_1, \mathbf{S}_2, ..., \mathbf{S}_t\}), \tag{3.18}$$

where Concat() is a concatenation operation.

Finally, we use a dense layer to calculate the similarity score between the question and the path and select the path with the highest score. We use binary-cross-entropy as a loss function to optimize the parameters of the model during the learning phase.

4. Experimental results

In this section, we first discuss the settings of the experiments we used to test our method. Then, we present the experimental results and analysis in detail. Experimental settings are introduced in Section 4.1, and then extensive comparisons with state-of-the-art methods are presented in Section 4.2. Finally, Section 4.3 further discusses the proposed DPS model.

4.1. Experimental settings

To provide the visualization of the proposed method details, we describe the datasets used in this paper, evaluation metrics and configuration of the experiments in this section.

4.1.1. Datasets

To demonstrate the performance of the DPS model, several studies were carried out on a cMed-KBQA dataset—CKBQA2020 [‡]. To validate the generalization performance of the DPS model, we further performed experimental comparative analysis on the publicly available dataset CKBQA2019 [§]. The two hand-labeled datasets are available publicly thanks to the China Conference on Knowledge Graph and Semantic Computing KBQA tasks. Every data case consists of a question, a query language expression (SPARQL) and the responses to that inquiry. In addition, we utilize the Chinese knowledge base PKUBASE [¶], which has about 66,000,000 triplets, 25,000,000 entities and 410,000 relations in the trials. The PKUBASE provides three files: pkubase-triples containing the main triples of the knowledge base; pkubase-types containing the category triples of each entity; and pkubase-mention2ent, which can be used to assist in entity linking by containing the priority of each entity to a certain alias. The details of CKBQA2019 and CKBQA2020 are presented below.

- **CKBQA2019**: The dataset was published by the CCKS2019 CKBQA task, and it consists of a knowledge base, a mention-entity file and three question-and-answer pair files. Among them, Hundsun Technologies Inc. offers about 1000 question-answer pairs in the finance industry, while the Computer Technology Research Institute of Peking University offers about 3000 open-domain question-answer pairs. The distribution of question-answer pairs is as follows: the training set has 2298, and the development and test sets have 766.
- **CKBQA2020**: The dataset was released by the CCKS2020 COVID-19 question-answering Task. It contains nearly 50% complex questions with multi-hop relations. Unlike the CCKS2019 CKBQA task, this task introduces data about COVID-19. The medical data and the composition of the provided files are the same as in CCKS2019, but its question-answer pair distribution differs, e.g., the training set has 4000, and the development and test sets have 1529.

4.1.2. Evaluation metrics

In our experiments, we employ precision, recall and an average F1 score to evaluate the model. Let Q represent the question set, A_i represents the model's projected response set for the ith question and G_i represents the ground-truth set for the ith question. The essential formulas for computation are as follows:

$$Precision\,(\mathbf{P}) = \frac{1}{|Q|}\sum_{i=1}^{|Q|} P_i,\ P_i = \frac{A_i \cap G_i}{|A_i|}, \tag{4.1}$$

$$Recall\,(\mathbf{R}) = \frac{1}{|Q|}\sum_{i=1}^{|Q|} R_i,\ R_i = \frac{A_i \cap G_i}{|G_i|}, \tag{4.2}$$

$$F_1\,(\mathbf{F}_1) = \frac{1}{|Q|}\sum_{i=1}^{|Q|} \frac{2P_iR_i}{P_i + R_i}. \tag{4.3}$$

[‡]http://www.sigkg.cn/ccks2020
[§]http://www.sigkg.cn/ccks2019
[¶]https://github.com/pkumod/gAnswer/tree/pkubase

$$Precision\,(\mathbf{P}) = \frac{1}{|Q|}\sum_{i=1}^{|Q|} P_i,\; P_i = \frac{A_i \cap G_i}{|A_i|}, \tag{4.4}$$

$$Recall\,(\mathbf{R}) = \frac{1}{|Q|}\sum_{i=1}^{|Q|} R_i,\; R_i = \frac{A_i \cap G_i}{|G_i|}, \tag{4.5}$$

$$F_1\,(\mathbf{F}_1) = \frac{1}{|Q|}\sum_{i=1}^{|Q|} \frac{2P_iR_i}{P_i + R_i} \tag{4.6}$$

4.1.3. Implementation details

In our work, all experiments were conducted by using the Pytorch library [58] on a local workstation with an Ubuntu 18.04 operating system. The proposed framework was implemented by using Python 3.7 with an NVIDIA GeForce GTX 3090 GPU with 24 GB of memory. Adam was our optimizer in the Systems 1 and 2 training steps to reduce training loss. The batch size was set to 4, the learning rate was set to $2e - 5$ and the maximum length of model input strings was set to 64. The training process included 100 epochs for the DPS model. In addition, we terminated it early when the certification set's F1 score exceeded its maximum value. In System 1, the multi-scale CNN architecture was equipped with size (1, 2, 3) filters with 300 feature maps; in System 2, there was size (1, 2, 3, 4, 5) filters with 300 feature maps. The parameter K was set to 10.

4.2. Comparisons with state-of-the-art approaches

In this section, we focus on the performance demonstration of the proposed method compared with the currently published methods. It contains a detailed introduction of the compared methods and a comparative analysis of the performance metrics of different methods.

4.2.1. Baseline methods

To validate the effectiveness of the DPS model, we compared our method to various baselines in the literature.

- **DUTIR** [48] offers a pipelined model for recognizing entities and their attributes, linking and filtering entities, matching and bridging text for questions and validating the performance of language models that have already been trained to answer knowledge-based questions.
- **CNNWR** [49] builds an integration system with a path similarity model, a relationship similarity model, a rule similarity model, and some strategic rules.
- **FSM** [51] makes a CKBQA system that combines multiple semantic similarities, and all of the system's parts fully integrate the semantic features of questions and answers.
- **Pathselection** [52] designs a technique based on the dynamic filtering of negative cases to enrich the training set's relations. In order to combat the growth of candidate paths, classification and beam search strategies are contrasted.

- **MIQA** [55] demonstrates a combined methodology that considers all tasks in candidate path generation and uses linguistic resemblance and re-ranking algorithms to determine the optimal route.
- **SEE** [56] constructs a framework based on a pre-trained model, which includes denotation recognition, entity linking, candidate answer generation and answer ranking.
- **ARTEMIS** [50] suggests a pipeline method based on feature fusion, which consists of six parts: mention recognition, entity linking, question classification, path generation, path ranking, and answer retrieval, and it uses multiple feature fusion strategies to recall answers more accurately from the knowledge base.

4.2.2. Performance comparison

To comprehensively investigate the proposed DPS model, it is contrasted with contemporary procedures under the settings defined in Section 4.2.1. Table 1 summarizes the achievements of several solutions on CKBQA2019 and CKBQA2020.

Results on the CKBQA2019 dataset. The dataset used in our experiments, CKBQA, is a competition dataset. To show the effectiveness of the proposed method, we compare the experimental results with the winning method in the competition. For the fairness of the comparison, we used the average F1 value used in the competition as the evaluation indicator. The trial conclusions are depicted in the upper part of Table 1. Like our DPS model, the top three teams (DUTIR, CNNWR, and FSM) also used the pre-trained model BERT. Their method utilized the most prevalent BERT model, but we employed the MacBERT model, which was reflected in the performance of the reading comprehension task. In addition, their method requires executing the named entity recognition model before recognizing topic entities, whereas our model focuses on path matching. It can be seen in Table 1 that ESI-L and ESI-D-B approaches are also superior because they employ model fusion methodologies and recognize topic entities more effectively. In contrast, our method focuses on obtaining the paths of the questions and is more detailed and efficient at capturing the semantic information of the questions. The results show that our DPS model achieved a 78.12% average F1 score on CKBQA2019, outperforming the best model ESI-D-B (76.11%). And its inference speed was 1.9 s. The inference speed is measured by the number of samples processed per second during inference. It implies that our model enhances its performance level effectively.

Results on the CKBQA2020 dataset. The CKBQA2020 dataset is also a competition dataset. Like with CKBQA2019, we employed the average F1 value as the evaluation metric and compared the experimental results with the winning method in the competition. Among the baselines, MIQA and SEE use human-created patterns to construct correlations, and subsequently, sorting algorithms are used to estimate the meaningful rating. ARTEMIS first categorizes issues as easy or difficult and then designs various complicated-question-solving strategies. As shown in the bottom half of the results in Table 1, the performance of the aforementioned baselines is highly dependent on the scope of the manually generated rules or the reliability of question categorization, resulting in mediocre outcomes in question answering. Furthermore, the data indicate that the DPS obtained an average F1 score of 86.60% on CKBQA2020, which is 0.53% better than the best model. And its inference speed was 2 s. The inference speed is measured by the number of samples processed per second during inference. It indicates that the DPS model has some enhancement effects on answering the questions.

Table 1. Evaluation results for the public dataset. The benchmark results were extracted from the original journals. It highlights the top performance in bold.

	Index	Models	Average F1 (%)
CKBQA2019	1	DUTIR *(2019)* [48]	67.68
	2	CNNWR *(2019)* [49]	73.08
	3	FSM *(2019)* [51]	73.54
	4	Pathselection *(2021)* [52]	73.09
	5	Fusion-model (ESI-L) *(2020)* [53]	74.68
	6	Fusion-model2 (ESI-D-B) *(2021)* [54]	76.11
	7	**DPS (ours)**	**78.12**
CKBQA2020	8	MIQA *(2020)* [55]	85.08
	9	SEE *(2020)* [56]	85.47
	10	ARTEMIS *(2020)* [50]	86.07
	11	**DPS (ours)**	**86.60**

4.3. Discussion of the proposed DPS model

In this section, we discuss the experimental results of the proposed approach, mainly from three perspectives: ablation study, parameter analysis and case study.

4.3.1. Ablation study

In the DPS model shown in Figure 2, Systems 1 and 2 have been designed to learn a semantic representation of the candidate path. To investigate the effectiveness of each module, we further conducted a group of experiments on CKBQA2019 and CKBQA2020 datasets. The experimental results are given in Table 2. For consistency, when the experimental conditions and instrumentation matched those in Section 4.1.3, we replicated the MacBERT model.

To test the effects of various variables on the DPS model, we introduced Systems 1 and 2 for the experiment separately. Quantitative precision, recall and average F1 score comparisons on CKBQA2019 and CKBQA2020 datasets are presented in Table 2. We observe that the fusion System 1 achieved better performance than the MacBERT-only model on two public datasets. On the CKBQA2019 dataset, the fusion System 1 obtained favourable results with a precision of 76.72%, a recall of 77.47% and an average F1 score of 76.58%. Compared to the CKBQA2019 dataset, the fusion System 1 performed better on the CKBQA2020 dataset. It is apparent that the fusion System 1 obtained better results with a precision of 85.29%, a recall of 85.77% and an average F1 score of 84.96%, which were 1.64, 2.01 and 1.89% higher than the MacBERT-only model on the CKBQA2020 dataset. System 2 yielded significantly superior results over System 1 alone, with a 0.64% increase in precision, a 0.47% increase in recall and a 0.58% increase in average F1 score. The distribution of performance improvements on the CKBQA2019 dataset remained consistent with the CKBQA2020 dataset. The results showed that: 1) System 1 enhanced the efficiency of the DPS model, verifying that the vital meaning of questions and simple candidate paths can be effectively learned by using a hybrid semantic fusion mechanism; 2) System 2 considerably enhanced the model's performance, suggesting that the rich, complicated paths enable the model to gather more valuable question-related information.

Table 2. Evaluation of each module on the CKBQA2019 and CKBQA2020 datasets with average F1 score.

Indicator	Important Materials			CKBQA2019			CKBQA2020		
	MacBERT	System 1	System 2	Precision (%)	Recall (%)	Average F1 (%)	Precision (%)	Recall (%)	Average F1 (%)
1	✓			76.69	77.23	76.25	83.65	83.76	83.07
2	✓	✓		76.72	77.47	76.58	85.29	85.77	84.96
3	✓		✓	77.28	78.98	77.88	85.93	86.24	85.54
4	✓	✓	✓	**78.27**	**79.74**	**78.12**	**86.98**	**87.10**	**86.60**

Finally, we incorporated Systems 1 and 2 into the benchmark MacBERT model for the best results. As shown in Table 2, the DPS model obtained competing results on the cMed-KBQA task. On the CKBQA2019 dataset, the DPS model achieved good performance with a precision of 78.27%, a recall of 79.74% and an average F1 score of 78.12%. Compared to the CKBQA2019 dataset, the DPS model achieved better performance with a precision of 86.98%, a recall of 87.10%, and an average F1 score of 86.60%. Notably, the CKBQA task improved when System 1 and System 2 were used together. The considerable performance improvement indicates that our approach can be practical enough to learn semantic information about questions and simple paths. In addition, it can acquire more complex paths and learn deeper semantic information. Thus, our proposed method provides more valuable semantic information for improving the performance of CKBQA models.

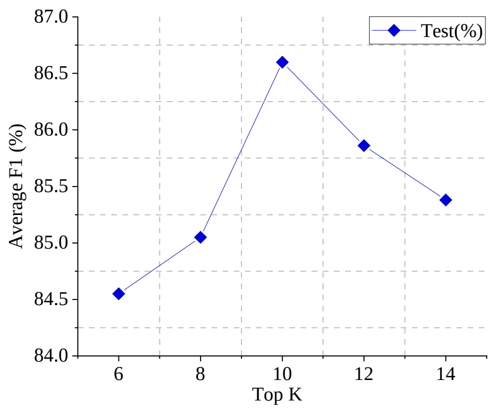


Figure 7. Performance of the DPS model with different top-k predicted paths on the CKBQA2020 test dataset.

4.3.2. Parameters analysis

As the proposed method is about candidate paths and stops entity and path sparsity in pipeline methods, the number of candidate paths needs to be set to an appropriate value. We conducted extensive experiments on CCKS2020 to verify the impact of taking values of K in the first K candidate paths. In our research, K is a critical indicator for selection of the predicted path, as it determines the similarity of questions and candidate paths. As K decreases, the DPS network becomes more selective, reducing the number of possible routes to only a handful of the most important ones. Instead, many random connections are made when K is large. Consequently, the DPS framework requires a fair value for the parameter K. Thus, we used an integer of 2 to define K between the numbers 6 and 14. Figure

7 illustrates that the overall performance tendency changes with the parameter K. Our DPS model's efficiency increaseed with growing K, reaching a peak at $K = 10$. When the parameter K was greater than 10, the accuracy of the proposed methodology dropped significantly due to unnecessary data. According to the findings, setting K to 10 is the best parameter choice for cMed-KBQA tasks. The average F1 score of the proposed model was 86.60%. A value of 10 was chosen to serve as the default for the parameter K in each experiment.

4.3.3. Error analysis

The effective and comprehensive simple paths are critical to improving the performance of the DPS model. In this section, we provide a brief analysis of the performance of the simple path-matching model (System 1). During the training, we set the number of epochs to 100 and stopped training when the accuracy did not beat the best metric five times in a row. As shown in Figure 8, the training loss rapidly decreased to a plateau as the training iterations increased. It proves that the convergence in System 1 is relatively good. As the loss values slowly stopped going up, System 1's accuracy reached 80 and 90% on the CKBQA2019 and CKBQA2020 datasets, respectively. Also, on the CKBQA2020 dataset, System 1 performed better than on the CKBQA2019 dataset. We guess that this is because the CKBQA2020 dataset has more data and the model can learn more from it. Overall, the ability of System 1 to predict gives some clues about how well the cMed-KBQA task will be done.

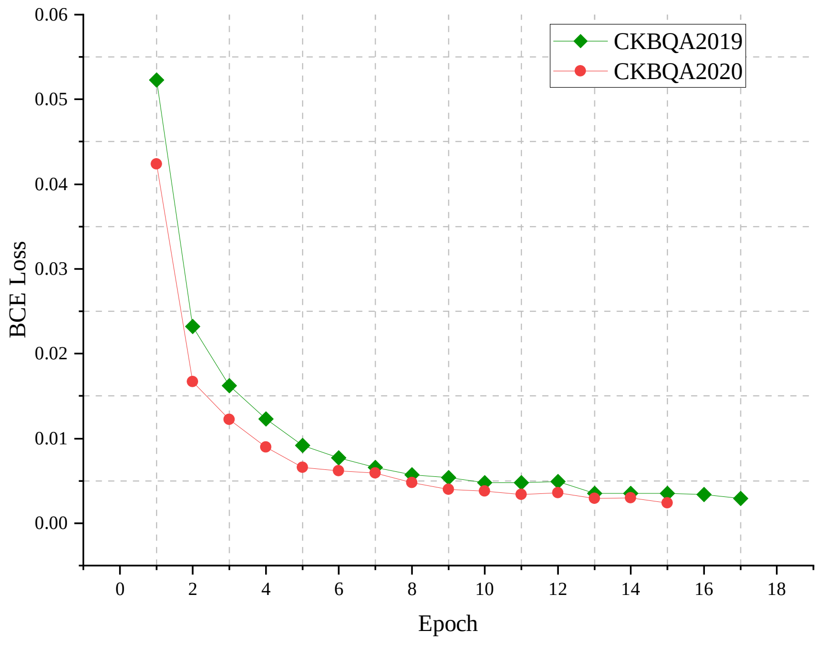
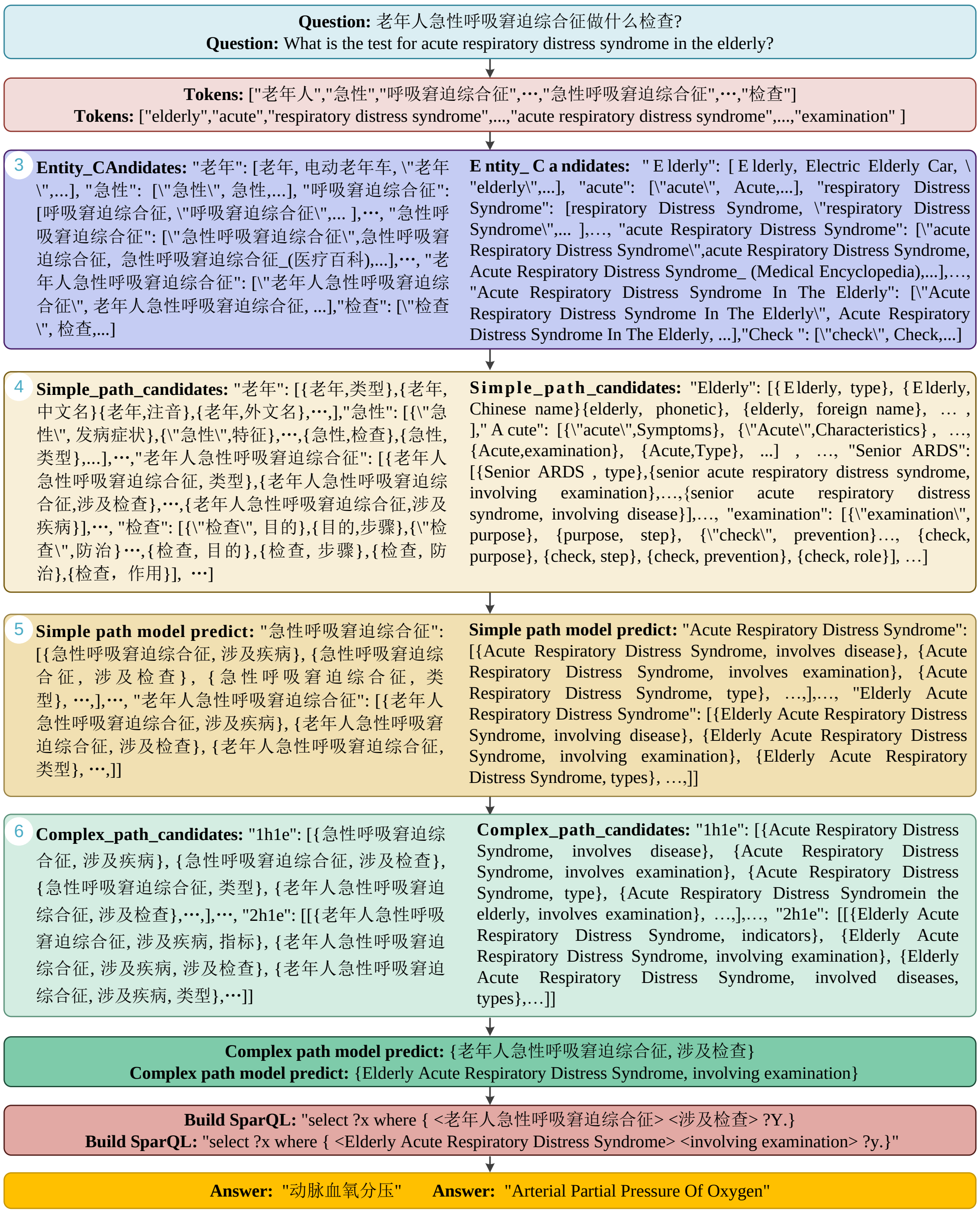


Figure 8. BCE loss of the simple path-matching model (System 1) with different numbers of epochs on the CKBQA2019 and CKBQA2020 dataset.

4.3.4. Case study of cMed-KBQA task



Question: 老年人急性呼吸窘迫综合征做什么检查?
Question: What is the test for acute respiratory distress syndrome in the elderly?

Tokens: ["老年人","急性","呼吸窘迫综合征",⋯,"急性呼吸窘迫综合征",⋯,"检查"]
Tokens: ["elderly","acute","respiratory distress syndrome",...,"acute respiratory distress syndrome",...,"examination"]

3 **Entity_CAndidates:** "老年": [老年, 电动老年车, \"老年\",...], "急性": [\"急性\", 急性,...], "呼吸窘迫综合征": [呼吸窘迫综合征, \"呼吸窘迫综合征\",...],⋯, "急性呼吸窘迫综合征": [\"急性呼吸窘迫综合征\",急性呼吸窘迫综合征, 急性呼吸窘迫综合征_(医疗百科),...],⋯, "老年人急性呼吸窘迫综合征": [\"老年人急性呼吸窘迫综合征\", 老年人急性呼吸窘迫综合征, ...],"检查": [\"检查\", 检查,...]

E ntity_ C a ndidates: " E lderly": [E lderly, Electric Elderly Car, \"elderly\",...], "acute": [\"acute\", Acute,...], "respiratory Distress Syndrome": [respiratory Distress Syndrome, \"respiratory Distress Syndrome\",...],…, "acute Respiratory Distress Syndrome": [\"acute Respiratory Distress Syndrome\",acute Respiratory Distress Syndrome, Acute Respiratory Distress Syndrome_ (Medical Encyclopedia),...],…, "Acute Respiratory Distress Syndrome In The Elderly": [\"Acute Respiratory Distress Syndrome In The Elderly\", Acute Respiratory Distress Syndrome In The Elderly, ...],"Check ": [\"check\", Check,...]

4 **Simple_path_candidates:** "老年": [{老年,类型},{老年,中文名}{老年,注音},{老年,外文名},⋯,],"急性": [{\"急性\", 发病症状},{\"急性\",特征},⋯,{急性,检查},{急性,类型},...],⋯,"老年人急性呼吸窘迫综合征": [{老年人急性呼吸窘迫综合征, 类型},{老年人急性呼吸窘迫综合征,涉及检查},⋯,{老年人急性呼吸窘迫综合征,涉及疾病}],⋯, "检查": [{\"检查\", 目的},{目的,步骤},{\"检查\",防治}⋯,{检查, 目的},{检查, 步骤},{检查, 防治},{检查， 作用}], ⋯]

Simple_path_candidates: "Elderly": [{ E lderly, type}, { E lderly, Chinese name}{elderly, phonetic}, {elderly, foreign name}, … ,]," A cute": [{\"acute\",Symptoms}, {\"Acute\",Characteristics} , …, {Acute,examination}, {Acute,Type}, ...] , …, "Senior ARDS": [{Senior ARDS , type},{senior acute respiratory distress syndrome, involving examination},…,{senior acute respiratory distress syndrome, involving disease}],…, "examination": [{\"examination\", purpose}, {purpose, step}, {\"check\", prevention}…, {check, purpose}, {check, step}, {check, prevention}, {check, role}], …]

5 **Simple path model predict:** "急性呼吸窘迫综合征": [{急性呼吸窘迫综合征, 涉及疾病}, {急性呼吸窘迫综合征， 涉 及 检 查}，{ 急 性 呼 吸 窘 迫 综 合 征， 类型}, ⋯,],⋯, "老年人急性呼吸窘迫综合征": [{老年人急性呼吸窘迫综合征, 涉及疾病}, {老年人急性呼吸窘迫综合征, 涉及检查}, {老年人急性呼吸窘迫综合征, 类型}, ⋯,]]

Simple path model predict: "Acute Respiratory Distress Syndrome": [{Acute Respiratory Distress Syndrome, involves disease}, {Acute Respiratory Distress Syndrome, involves examination}, {Acute Respiratory Distress Syndrome, type}, …,],…, "Elderly Acute Respiratory Distress Syndrome": [{Elderly Acute Respiratory Distress Syndrome, involving disease}, {Elderly Acute Respiratory Distress Syndrome, involving examination}, {Elderly Acute Respiratory Distress Syndrome, types}, …,]]

6 **Complex_path_candidates:** "1h1e": [{急性呼吸窘迫综合征, 涉及疾病}, {急性呼吸窘迫综合征, 涉及检查}, {急性呼吸窘迫综合征, 类型}, {老年人急性呼吸窘迫综合征, 涉及检查},⋯,],⋯, "2h1e": [[{老年人急性呼吸窘迫综合征, 涉及疾病, 指标}, {老年人急性呼吸窘迫综合征, 涉及疾病, 涉及检查}, {老年人急性呼吸窘迫综合征, 涉及疾病, 类型},⋯]]

Complex_path_candidates: "1h1e": [{Acute Respiratory Distress Syndrome, involves disease}, {Acute Respiratory Distress Syndrome, involves examination}, {Acute Respiratory Distress Syndrome, type}, {Acute Respiratory Distress Syndromein the elderly, involves examination}, …,],…, "2h1e": [[{Elderly Acute Respiratory Distress Syndrome, indicators}, {Elderly Acute Respiratory Distress Syndrome, involving examination}, {Elderly Acute Respiratory Distress Syndrome, involved diseases, types},…]]

Complex path model predict: {老年人急性呼吸窘迫综合征, 涉及检查}
Complex path model predict: {Elderly Acute Respiratory Distress Syndrome, involving examination}

Build SparQL: "select ?x where { <老年人急性呼吸窘迫综合征> <涉及检查> ?Y.}
Build SparQL: "select ?x where { <Elderly Acute Respiratory Distress Syndrome> <involving examination> ?y.}"

Answer: "动脉血氧分压" **Answer:** "Arterial Partial Pressure Of Oxygen"

Figure 9. A case study of cMed-KBQA via our proposed DPS. The English-language version was obtained through Google Translate and is for reference only.

To demonstrate the working mechanism of our suggested DPS model, a case study of the cMed-KBQA task is presented in Figure 9. When asked the question “What is the test for acute respiratory distress syndrome in the elderly?”, the correct reasoning path is the {Elderly Acute Respiratory Distress Syndrome, involving examination}. To accurately answer the above questions, a dual-processing system is required for matching reasoning. A case study of answer reasoning using our proposed DPS is shown in Figure 9.

For the question “What is the test for acute respiratory distress syndrome in the elderly?”. In the first stage, our model obtained the tokens (”elderly”, ”acute”, ”respiratory distress syndrome”,..., ”acute respiratory distress syndrome”, ..., and ”examination”) of the question by both Jieba and PkuSeg. We then connected the knowledge base to query the corresponding candidate entities and candidate paths (as shown in Parts 3 and 4 of Figure 9). Next, we employed System 1 for simple path prediction (as shown in part 5 of Figure 9). In the second stage, based on the paths predicted by System 1, we connected the knowledge base to query its associated candidate complex paths (as shown in Part 6 of Figure 9). Subsequently, we utilized System 2 to predict our best path {Elderly Acute Respiratory Distress Syndrome, involving examination}. Finally, we built a SPARQL query statement based on the best path to obtain the answer ”Arterial Partial Pressure Of Oxygen” through the query knowledge base.

During the training stages of Systems 1 and 2, our model applied a multi-scale CNN mechanism to acquire more valuable information. The DPS model incorporating the DPS can obtain more accurate answers.

5. Conclusions, limitations, and future research

In this paper, we have proposed a practical framework with a DPS for the cMed-KBQA task and achieved significant improvement on the CKBQA2019 and CKBQA2020 datasets when compared with previous methods. Specifically, System 1 of the DPS model consists of an entity extraction module, an entity linking module, a simple path retrieval module and a simple path-matching model; it utilizes a simple path-matching model to capture the candidate simple paths in question. For System 2, we applied the complex path retrieval module and complex path-matching model to reason on a knowledge base (PKUBASE) to retrieve the complex paths. System 2 first executes System 1 to find candidate simple paths; then, a complex path retrieval module is conducted to obtain enriched complex paths from the knowledge base. For the path matching, we proposed a mixed semantic fusion mechanism to capture valuable information from both the question and the candidate path. Experiments showed that our methodology outperforms the other methods presented in previous papers when implemented on DPSs.

As an important branch of intelligent question answering, research related to cMed-KBQA not only improves the utilization of medical information resources, but it also provides medical practitioners with a huge space and many choices. In addition, the development of a medical intelligent question-answering system has also led to changes in the traditional disease-centric service concept of medical information search, with the concept of human-centered services becoming more and more practical. Although the proposed approach improved the performance on cMed-KBQA tasks, there are some limitations in its practical applications, such as how to realize the end-to-end question-answering system and achieve an accuracy rate that can be practically applied.

For future work, we will first explore end-to-end ways to improve the performance of models and the effectiveness of real-world applications, avoiding the improper propagation of subtasks in pipeline approaches. Second, we would like to explore diverse paths through the use of multiplex relational attention networks [59,60] and graph neural networks [61] combined with semantic attention.

Acknowledgment

This work was supported by the Chengdu Major Technology Application Demonstration Project (No. 2019-YF09-00120-SN), the National Natural Science Foundation of China (Nos. 61871278 & 61871279), the Sichuan Science and Technology Program (No. 2021YFS0239), and the Fundamental Research Funds for the Central Universities (No. 2021SCU12061).

Conflict of interest

The authors declare that there are no conflict of interest.

References

1. K. Bollacker, R. Cook, P. Tufts, Freebase: a shared database of structured general human knowledge, in *Proceedings of the 22nd national conference on Artificial intelligence*, **2** (2007), 1962–1963.

2. C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, et al., Dbpedia-A crystallization point for the web of data, *J. Web Semant.*, **7** (2009), 154–165. https://doi.org/10.1016/j.websem.2009.07.002

3. V. Denny, M. Krötzsch, Wikidata: a free collaborative knowledgebase, *Commun. ACM*, **57** (2014), 78–85. https://doi.org/10.1145/2629489

4. X. Niu, X. Sun, H. Wang, S. Rong, G. Qi, Y. Yu, Zhishi: me-weaving Chinese linking open data, in *International Semantic Web Conference*, Springer, Berlin, Heidelberg, (2011), 205–220. https://doi.org/10.1007/978-3-642-25093-4_14

5. B. Xu, Y. Xu, J. Liang, C. Xie, B. Liang, W. Cui, et al., CN-DBpedia: A never-ending Chinese knowledge extraction system, in *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, Springer, Cham, (2017), 428–438.

6. Q. Cai, A. Yates, Large-scale semantic parsing via schema matching and lexicon extension, in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, **1** (2013), 423–433.

7. J. Berant, A. Chou, R. Frostig, P. Liang, Semantic parsing on freebase from question-answer pairs, in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, (2013), 1533–1544.

8. S. Reddy, M. Lapata, M. Steedman, Large-scale semantic parsing without question-answer pairs, *Trans. Assoc. Comput. Ling.*, **2** (2014), 377–392. https://doi.org/10.1162/tacl_a_00190

9. W. Yih, M. Chang, X. He, J. Gao, Semantic parsing via staged query graph generation: Question answering with knowledge base, in *Proceedings of the Joint Conference of the 53rd Annual Meeting of the ACL and the 7th International Joint Conference on Natural Language Processing of the AFNLP*, (2015), 1–11.
10. S. Reddy, O. Täckström, M. Collins, T. Kwiatkowski, D. Das, M. Steedman, et al., Transforming dependency structures to logical forms for semantic parsing, *Trans. Assoc. Comput. Ling.*, **4** (2016), 127–140. https://doi.org/10.1162/tacl_a_00088
11. E. Hoffer, N. Ailon, Deep metric learning using triplet network, *International Workshop on Similarity-based Pattern Recognition*, (2015), 84–92.
12. N. Francis, A. Green, P. Guagliardo, L. Libkin, T. Lindaaker, V. Marsault, et al., Cypher: An evolving query language for property graphs, in *Proceedings of the 2018 International Conference on Management of Data*, (2018), 1433–1445. https://doi.org/10.1145/3183713.3190657
13. X. Yao, B. Durme, Information extraction over structured data: Question answering with freebase, in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, **1** (2014), 956–966.
14. M. Petrochuk, L. Zettlemoyer, Simplequestions nearly solved: A new upperbound and baseline approach, preprint, arXiv:1804.08798. https://doi.org/10.48550/arXiv.1804.08798
15. T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, preprint, arXiv:1301.3781. https://doi.org/10.48550/arXiv.1301.3781
16. J. Pennington, R. Socher, C. D. Manning, Glove: Global vectors for word representation, in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, (2014), 1532–1543.
17. J. Devlin, M. W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, (2019), 4171–4186.
18. Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, Q. Le, XLNet: generalized autoregressive pretraining for language understanding, in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, (2019), 5753–5763.
19. A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, Language models are unsupervised multitask learners, *OpenAI blog*, **1** (2019), 9–24.
20. L. Li, Y. Zhai, J. Gao, L. Wang, L. Hou, J. Zhao, Stacking-BERT model for Chinese medical procedure entity normalization, *Math. Biosci. Eng.*, **20** (2023), 1018–1036. https://doi.org/10.3934/mbe.2023047
21. C. Li, K. Ma, Entity recognition of Chinese medical text based on multi-head self-attention combined with BILSTM-CRF, *Math. Biosci. Eng.*, **19** (2022), 2206–2218. https://doi.org/10.3934/mbe.2022103
22. S. A. Sloman, The empirical case for two systems of reasoning, *Psychol. Bull.*, **119** (1996), 3–22. https://doi.org/10.1037/0033-2909.119.1.3
23. J. St. B. T. Evans, In two minds: dual-process accounts of reasoning, *Trends Cognit. Sci.*, **7** (2003), 454–459. https://doi.org/10.1016/j.tics.2003.08.012

24. J. St. B. T. Evans, Dual-processing accounts of reasoning, judgment, and social cognition, *Annu. Rev. Psychol.*, **59** (2008), 255–278. https://doi.org/10.1146/annurev.psych.59.103006.093629

25. B. Alan, Working memory, *Science*, **255** (1992), 556–559. https://doi.org/10.1126/science.1736359

26. R. M. Terol, P. M. Barco, M. Palomar, A knowledge based method for the medical question answering problem, *Comput. Biol. Med.*, **37** (2007), 1511–1521. https://doi.org/10.1016/j.compbiomed.2007.01.013

27. Q. Cai, A. Yates, Large-scale semantic parsing via schema matching and lexicon extension, in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, **1** (2013), 423–433.

28. J. Berant, A. Chou, R. Frostig, P. Liang, Semantic parsing on freebase from question-answer pairs, in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, (2013), 1533–1544.

29. T. Kwiatkowski, E. Choi, Y. Artzi, L. Zettlemoyer, Scaling semantic parsers with on-the-fly ontology matching, in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, (2013), 1545–1556.

30. K. Luo, F. Lin, X. Luo, K. Q. Zhu, Knowledge base question answering via encoding of complex query graphs, in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, (2018), 2185–2194. https://doi.org/10.18653/v1/D18-1242

31. P. Kapanipathi, I. Abdelaziz, S. Ravishankar, S. Roukos, A. G. Gray, R. F. Astudillo, et al., Question answering over knowledge bases by leveraging semantic parsing and neuro-symbolic reasoning, in *Proceedings of the AAAI Conference on Artificial Intelligence*, (2020), 1–10.

32. S. Zhu, X. Cheng, S. Su, Knowledge-based question answering by tree-to-sequence learning, *Neurocomputing*, **372** (2020), 64–72. https://doi.org/10.1016/j.neucom.2019.09.003

33. Y. Chen, H. Li, Y. Hua, G. Qi, Formal query building with query structure prediction for complex question answering over knowledge base, in *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, (2021), 3751–3758. https://doi.org/10.24963/ijcai.2020/519

34. B. Min, R. Grishman, L. Wan, C. Wang, D. Gondek Distant supervision for relation extraction with an incomplete knowledge base, in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics*, (2013), 777–782.

35. H. Sun, B. Dhingra, M. Zaheer, K. Mazaitis, R. Salakhutdinov, W. W. Cohen, Open domain question answering using early fusion of knowledge bases and text, in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, (2018), 4231–4242. https://doi.org/10.18653/v1/D18-1455

36. H. Sun, T. B. Weiss, W. W. Cohen, Pullnet: Open domain question answering with iterative retrieval on knowledge bases and text, in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, (2019), 2380–2390. https://doi.org/10.18653/v1/D19-1242

37. W. Xiong, M. Yu, S. Chang, X. Guo, W. Wang, Improving question answering over incomplete kbs with knowledge-aware reader, in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, (2019), 4258–4264. https://doi.org/10.18653/v1/P19-1417

38. J. Han, B. Cheng, X. Wang, Open domain question answering based on text enhanced knowledge graph with hyperedge infusion, in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, (2020), 1475–1481. https://doi.org/10.18653/v1/2020.findings-emnlp.133

39. A. Saxena, A. Tripathi, P. Talukdar, Improving multi-hop question answering over knowledge graphs using knowledge base embeddings, in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, (2020), 4498–4507. https://doi.org/10.18653/v1/2020.acl-main.412

40. A. Bordes, J. Weston, N. Usunier, Open question answering with weakly supervised embedding models, in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, (2014), 165–180.

41. D. Guo, D. Tang, N. Duan, M. Zhou, J. Yin, Dialog-to-action: conversational question answering over a large-scale knowledge base, in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, **7** (2018), 2946–2955.

42. X. Huang, J. Zhang, D. Li, P. Li, Knowledge graph embedding based question answering, in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, (2019), 105–113. https://doi.org/10.1145/3289600.3290956

43. R. Wang, M. Wang, J. Liu, W. Chen, M. Cochez, S. Decker, Leveraging knowledge graph embeddings for natural language question answering, in *International Conference on Database Systems for Advanced Applications*, **11446** (2019), 659–675. https://doi.org/10.1007/978-3-030-18576-3_39

44. L. Dong, F. Wei, M. Zhou, K. Xu, Question answering over freebase with multi-column convolutional neural networks, in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, **1** (2015), 260–269.

45. Y. Lai, Y. Feng, X. Yu, Z. Wang, K. Xu, D. Zhao, Lattice cnns for matching based chinese question answering, in *Proceedings of the AAAI Conference on Artificial Intelligence*, **33** (2019), 6634–6641. https://doi.org/10.1609/aaai.v33i01.33016634

46. Y. Hao, Y. Zhang, K. Liu, S. He, Z. Liu, H. Wu, et al., An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge, in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, **1** (2017), 221–231. https://doi.org/10.18653/v1/P17-1021

47. K. Tai, R. Socher, C. D. Manning, Improved semantic representations from tree-structured long short-term memory networks, in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, **1** (2015), 1556–1566.

48. M. Cao, S. Li, X. Wang, Z. Yang, H. Lin,, DUTIR: Chinese open domain knowledge base question answering system, in *Proceedings of the Evaluation Tasks at the China Conference on Knowledge Graph and Semantic Computing*, **1** (2019), 1–6.

49. P. Zhang, K. Wu, Z. Zhu, Y. Jia, X. Zhou, W. Chen, et al., Combining neural network models with rules for Chinese knowledge base question answering, in *Proceedings of the Evaluation Tasks at the China Conference on Knowledge Graph and Semantic Computing*, **1** (2019), 1–12.

50. Z. Wang, Y. Hou, M. Wang, C. Li, Chinese knowledge base question answering method based on fusion feature, in *Proceedings of the evaluation tasks at the china conference on knowledge graph and semantic computing*, **1** (2020), 1–7.

51. J. Luo, C. Yin, X. Wu, L. Zhou, H. Zhong, Chinese knowledge base question answering system based on mixed semantic similarity, in *Proceedings of the Evaluation Tasks at the China Conference on Knowledge Graph and Semantic Computing*, **1** (2019), 1–12.

52. K. Wu, X. Zhou, Z. Li, X. Liang, W. Chen, Path selection for Chinese knowledge base question answering, *J. Chin. Inf. Process.*, **35** (2021), 113–122.

53. M. Tang, H. Xiong, L. Wang, X. Lin, A dynamic answering path based fusion model for KGQA, in *International Conference on Knowledge Science, Engineering and Management*, **12274** (2020), 235–246. https://doi.org/10.1007/978-3-030-55130-8_21

54. H. Xiong, S. Wang, M. Tang, L. Wang, X. Lin, Knowledge graph question answering with semantic oriented fusion model, *Knowl.-Based Syst.*, **221** (2021), 106954–106964. https://doi.org/10.1016/j.knosys.2021.106954

55. W. Dai, H. Liu, Y. Liu, R. Lv, S. Chen, An integrated path formulation method for open domain question answering over knowledge base, in *Proceedings of the Evaluation Tasks at the China Conference on Knowledge Graph and Semantic Computing*, **1** (2020), 1–10.

56. H. Zhang, R. Li, S. Wang, J. Huang, Retrieval-matching knowledge base question answering system based on pre-trained language model, in *Proceedings of the Evaluation Tasks at the China Conference on Knowledge Graph and Semantic Computing*, **1** (2020), 1–10.

57. Y. Cui, W. Che, T. Liu, B. Qin, S. Wang, G. Hu, Revisiting pre-trained models for Chinese natural language processing, in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, **1** (2020), 657–668. https://doi.org/10.18653/v1/2020.findings-emnlp.58

58. B. Steiner, Z. DeVito, S. Chintala, S. Gross, A. Paske, F. Massa, et al., Pytorch: An imperative style, high-performance deep learning library, in *Proceedings of the Advances in Neural Information Processing Systems*, (2019), 8026–8037.

59. Y. Zhao, H. Zhou, A. Zhang, R. Xie, Q. Li, F. Zhuang, Connecting embeddings based on multiplex relational graph attention networks for knowledge graph entity typing, *IEEE Trans. Knowl. Data Eng.*, (2022), 1–12. https://doi.org/10.1109/TKDE.2022.3142056

60. H. Zhu, X. He, M. Wang, M. Zhang, L. Qing, Medical visual question answering via corresponding feature fusion combined with semantic attention, *Math. Biosci. Eng.*, **19** (2022), 10192–10212. https://doi.org/10.3934/mbe.2022478

61. D. Prakash, L. Tuan, L. Thomas, B. Yoshua, B. Xavier, Graph neural networks with learnable structural and positional representations, preprint, arXiv:2110.07875. https://doi.org/10.48550/arXiv.2110.07875