


## Article

# Chinese Medical Question Answer Matching Using End-to-End Character-Level Multi-Scale CNNs

Sheng Zhang \* , Xin Zhang, Hui Wang, Jiajun Cheng, Pei Li and Zhaoyun Ding

College of Information Systems and Management, National University of Defense Technology, Changsha 410073, China; ijunzhanggm@gmail.com (X.Z.); huiwang@nudt.edu.cn (H.W.); jiajun.cheng@nudt.edu.cn (J.C.); peili@nudt.edu.cn (P.L.); zyding@nudt.edu.cn (Z.D.)

\* Correspondence: zhangsheng@nudt.edu.cn; Tel.: +86-0731-8457-6454

Received: 4 July 2017; Accepted: 26 July 2017; Published: 28 July 2017

**Abstract:** This paper focuses mainly on the problem of Chinese medical question answer matching, which is arguably more challenging than open-domain question answer matching in English due to the combination of its domain-restricted nature and the language-specific features of Chinese. We present an end-to-end character-level multi-scale convolutional neural framework in which character embeddings instead of word embeddings are used to avoid Chinese word segmentation in text preprocessing, and multi-scale convolutional neural networks (CNNs) are then introduced to extract contextual information from either question or answer sentences over different scales. The proposed framework can be trained with minimal human supervision and does not require any handcrafted features, rule-based patterns, or external resources. To validate our framework, we create a new text corpus, named cMedQA, by harvesting questions and answers from an online Chinese health and wellness community. The experimental results on the cMedQA dataset show that our framework significantly outperforms several strong baselines, and achieves an improvement of top-1 accuracy by up to 19%.

**Keywords:** question answer matching; medical domain; question answering; answer selection; character embeddings

## 1. Introduction

Along with the rapid growth of the (mobile) Internet, more and more people in China choose to seek medical help by posting questions on some online health and wellness communities, such as DingXiangYuan (<http://dxy.com/>) and XunYiWenYao (<http://www.xywy.com/>). This provides great convenience for users to get medical advice from qualified doctors, as they need not go to the hospital and are able ask questions anywhere and anytime. However, a large number of users—many of whom often ask similar, if not identical, questions—have placed a tremendous burden on the doctor-side, and cause timely reply to be nearly impossible. Thus, to enhance the user experience, it is essential to develop techniques which can efficiently address the problem of medical question answer matching; namely, selecting automatically from some existing medical answer records (e.g., those posted previously by the registered doctors) the one that best matches a user's question.

This paper focuses mainly on the problem of *Chinese medical question answer matching*, in which the languages of questions and answers under consideration are both limited to be Chinese. Compared with open-domain question answer matching in English studied recently by Feng et al. [1] and Tan et al. [2], the problem investigated in this paper is arguably more challenging due to the combination of two main factors: (1) the domain-restricted nature; and (2) some language-specific features of Chinese. The underlying challenges are further discussed as follows.

First of all, due to the lack of delimiters between Chinese words, word segmentation is normally taken as an indispensable preprocessing step for many Chinese natural language processing

(NLP) tasks such as part-of-speech (POS) tagging and semantic parsing, and thus considerably impacts the accuracy of those downstream tasks. Though the performance of the off-the-shelf Chinese word segmentation toolkits (e.g., ICTCLAS (<http://ictclas.nlpir.org/>), Jieba (<https://github.com/fxsjy/jieba>), THULAC (<http://thulac.thunlp.org/>), LTP (<https://github.com/HIT-SCIR/ltp>)) has reached a level satisfactory to many practical applications, they still yield errors [3], which would be inevitably propagated through the framework with a pipelined architecture and hence cause overall performance degradation. Moreover, the great variety of proper terms contained would cause further and sometimes very sharp accuracy decline to those general-purpose word segmentation tools when being directly applied to medical texts. For example, the medical terms “盐酸 西替利嗪 片” (cetirizine hydrochloride tablets) and “葡萄糖酸钙 片” (calcium gluconate tablets) in Table 1 are incorrectly segmented into “盐酸 西替 利嗪 片” and “葡萄糖 酸 钙片” by the Jieba segmentation toolkit (<https://github.com/fxsjy/jieba>). Although the introduction of domain-specific lexicons can mitigate the negative effect of professional terminology on word segmentation, building such lexicons always seems prohibitive, as it involves a large amount of manual labor and requires a great deal of domain-specific knowledge and expertise. Even worse is that pre-defined lexicons tend to be inappropriate when coping with the user-composed and unedited medical questions and answers posted on online communities, as they are written in an informal style and often contain many short-hand notations, non-standard acronyms, and even typos and ungrammatical sentences. For instance, “坐 (sit)” and “不听使唤 (out of control)” are incorrectly typed as “做 (do)” and “不停使唤 (keep calling)” respectively, in the question given in Table 1.

**Table 1.** An example of a question with its ground-truth answer and an irrelevant answer from the cMedQA dataset.

我经常做(坐)的太久，站起来，右脚就会不停(听)使唤的像抽筋一样！不知道怎么回事！	
<b>Question:</b>	When I get up after sitting for too long, my right foot is out of control like getting cramps! I don't know what's the matter!
你好：这种情况考虑缺钙引起的症状，最好到医院检查微量元素看看可以结合医生服用钙片治疗看看，如钙尔奇或葡萄糖酸钙等，注意多吃蔬菜，水果。	
<b>One Good Answer:</b>	Hello: The symptom is caused by calcium deficiency and you'd better get medical tests for trace elements at the hospital. You can also take calcium tablets, such as Caltrate or calcium gluconate tablets, and eat more vegetables and fruits.
你好，根据你的情况，建议口服盐酸西替利嗪片或者葡萄糖酸钙片治疗就可以。少吃一些辛辣刺激的东西，多吃蔬菜水果等清凉的食物，另外，还要保持皮肤的清洁，每天洗澡的时候可以用具有杀菌作用的浴液。	
<b>One Irrelevant Answer:</b>	Hello, according to your situation, I suggest that you should take cetirizine hydrochloride tablets or calcium gluconate tablets. You'd better eat less irritating food, and more fruits and vegetables. Moreover, you'd better keep your skin clean, and take a shower every day with sterilized bath liquid.

To address the above issues, we present an end-to-end deep learning framework using character-level multi-scale convolutional neural networks (CNNs). Specifically, the framework adopts a character-level representation, viz. the character embeddings, instead of word embeddings conventionally used, so as to avoid word segmentation during preprocessing as well as its negative impacts on subsequent components. Such representations are pre-trained using the Chinese characters in both questions and answers, and similar to word embeddings, describe each character as a fixed-length vector.

Since characters are less semantically meaningful compared with words in Chinese, the matching task depends more upon the capability of subsequent modules to extract relevant semantic information. This motivates us to introduce CNNs into the proposed framework, which have been proven to be good at capturing the local context of characters and words (i.e., the local interaction and dependency within  $n$ -gram). As different Chinese phrases (or words) are often different in length, we employ multi-scale CNNs (multiCNNs) to capture the interaction between them more appropriately, and hence to better

encode the semantic association between the questions and answers. The proposed multiCNNs module consists of a stack of feature maps with different scales.

To validate the proposed framework, we develop a dataset by collecting questions and answers from an online Chinese medical questions answering website (<http://www.xywy.com>), on which questions posted by users mainly contain the description of symptoms, diagnosis and treatment of diseases, use of drugs, psychological counseling, etc., and only certified doctors are permitted to answer questions. The dataset, named cMedQA, consists of 54,000 questions and more than 101,000 answers. We made cMedQA publicly available for academic use only (<https://github.com/zhangsheng93/cMedQA>). Table 1 gives an example of our data. More details of cMedQA can be found in Section 4.1.

In summary, the contribution of this paper is four-fold:

- \* To our best knowledge, we are the first to investigate the challenging problem of medical question answer matching in Chinese.
- \* We propose a character-level multi-scale CNN architecture for representation learning of Chinese medical texts, which employs character embeddings to circumvent the negative influence of Chinese word segmentation, and uses multiple convolutional feature maps to extract semantic information over different scales.
- \* We create and release the cMedQA dataset, which to the best of our knowledge is the first publicly available Chinese medical questions answer matching corpus.
- \* The experimental results on the cMedQA dataset demonstrate that the proposed framework significantly outperforms a variety of strong baselines with an improvement of top-1 accuracy by up to 19%.

The rest of the paper is organized as follows: Section 2 briefly overviews the related work; Section 3 presents the end-to-end character-level multi-scale convolutional neural framework; Experimental setting and results are described in Section 4; Finally, we draw conclusions and discuss future work in Section 5.

## 2. Related Work

Two broad classes of related work pertinent to our work are briefly surveyed in this section. First, we review the previous studies on traditional question answering. Next, we overview the recent work on applying deep learning to open-domain question answer matching.

### 2.1. Traditional Question Answering

Traditional approaches to medical question answering normally involve rule-based algorithms and statistical methods with handcrafted feature sets.

Jain and Dodiya [4] present rule-based architectures for question-answering systems in the medical domain and also discuss in detail the intricacies of rule-based question processing and answers retrieval. However, as user questions are always presented in a large number of different ways, rule-based methods may not be able to cover all such linguistic variety.

Wang et al. [5] propose an alternative approach, which first splits sentences into words to train the word vector for each sentence and then evaluates the similarity of each question-answer pair by computing the similarity between individual words. Abacha and Zweigenbaum [6] translate questions to machine-readable representations. The approach is thus able to convert a wide range of natural language questions into a standard language form. Later, the authors [7] extend their previous work by applying semantic techniques at both the representation and interrogation levels so as to create a structured query to match the entries in the knowledge base. These methods depend on manually designed patterns and handcrafted features, which often involve tremendous human labor and expertise.

Li [8] employs a multi-label classification method and the BM25 values [9] to retrieve from a pre-built corpus multiple questions, together with their answers, that are similar to the input one.

The resulting answers are further ranked to form a single passage using the TextRank algorithm proposed by Mihalcea and Tarau [10]. The proposed approach is essentially an information retrieval (IR)-based one, and the basic ideas are query expansion and retrieval results summarization.

Goodwin and Harabagiu [11] present a novel framework to select and rank scientific articles containing answers for clinical decision support (CDS) [12]. However, this method is likely to suffer from the lack of available external resources and the complexity of parsing differently structured texts.

Some models about Chinese question answering have been proposed recently. Li and Croft [13] constructed a Chinese question answering system named Marsha. The system recognizes known question types and formulates queries, then searches and extracts answers. Li et al. [14] proposed a method to calculate the similarity of two sentences by computing the similarity between words. Li et al. [15] constructed a semantic pattern matching model for musical domain to automatically translate questions to SPARQL queries for building the final answers. However, these methods take word segmentation as an essential step of Chinese text processing, and they do not take the inaccuracy of word segmentation toolkits into consideration, although the accuracy of word segmentation toolkit in general domain is satisfactory. Wang et al. [16] proposed an approach integrating count-based and embedding-based features. They also point out in their work that character-based models outperform word-based models, which gives us a hint that dealing with Chinese characters may avoid the inaccuracy brought by word segmentation.

To summarize, some traditional question answering methods often rely too much on manually developed rules, well-designed patterns, matching of pre-defined labels, and various external resources, which lead them to be human-labor intensive and vulnerable to concept drift frequently occurring when being applied to new datasets. In addition, some previous studies in Chinese question answering do not consider the adverse effect of word segmentation, and the adverse effect may become particularly prominent when we process domain-specific texts. As far as we know, few works have been dedicated to the key task of Chinese medical question answer matching, which is the focus of this paper.

## 2.2. Deep Learning in Open-Domain Question Answer Matching

As deep learning techniques have the advantage of capturing both low- and high-level semantics, they begin to be applied to open-domain question answer matching in recent years.

Hu et al. [17] proposed two different convolutional models to learn representations of sentences, which is the pioneer work for solving general sentence matching problems using neural networks. Following that, Feng et al. [1] and Zhou et al. [18] employed CNNs to learn the representations of questions and answers, which are further used to compute the similarity between different questions and their candidate answers. Later, in order to extract sequence information from a sentence, Tan et al. [2,19] utilized recurrent neural networks (RNNs) and their variant, long short-term memory networks (LSTMs), to learn the sentence-level representations. It is noteworthy that the authors also exploit attention mechanisms to augment the semantic association between questions and answers.

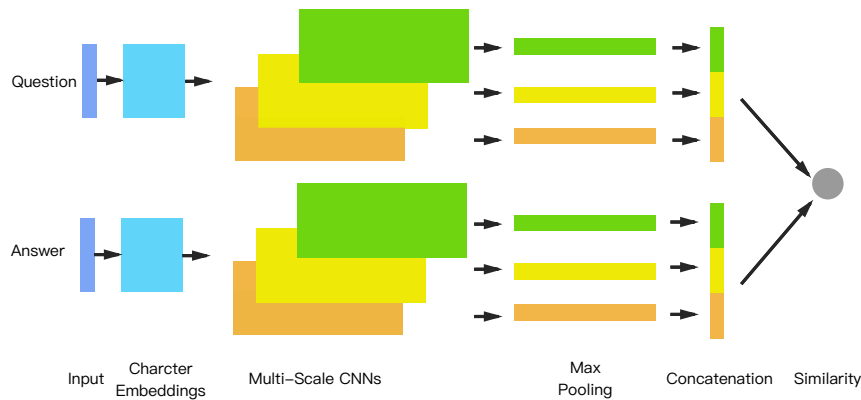
However, all studies mentioned above are pertinent to English texts. The approaches proposed in them are likely to suffer considerable performance degradation when being applied directly to process Chinese documents, as the latter language differs greatly from English.

## 3. Methodology

In this Section, we present our novel end-to-end character-level multi-scale convolutional neural framework for Chinese medical questions answering. First, we discuss the correct embedding methods for encoding Chinese medical texts. Next, we describe in detail the basic convolutional neural network architectures and our improved multi-scale convolutional neural network architectures.

Figure 1 illustrates our framework, given a question and its answer which are represented by character embeddings. The embeddings are used as inputs to our architecture. Next, feature maps of convolutional neural networks at different scales are applied to extract different local n-gram information. Max pooling is then applied to simplify the representation and to generate vectors for

the question and the answer. Finally, the cosine value of the two vectors is computed to measure the similarity of the question and the answer. It should be noted that layers with the same color share the same parameters and weights [1].



**Figure 1.** The framework of our model. The question and the answer are first split into characters, and they are represented by character embeddings. The embeddings are then fed into a multi-scale convolutional neural network architecture to extract local information. After that, max pooling layer reduces the representation by choosing the max value along the sequence. Finally, the vectors are concatenated to calculate the similarity of the question and the answer. It should be noted here that layers with the same color share parameters and weights.

### 3.1. Distributed Representation of Characters

In many natural language processing tasks, one fundamental step is to convert text sequences to machine-readable features, which are always fixed-length vectors. These features determine the upper limit of the effectiveness of the algorithms.

In recent years, embedding-based methods have been used to extract features from text, which shows their utility in semantic representation. A useful and often-mentioned type of embedding method is word embedding, also referred to as the distributed word representation. Bengio et al. [20] proposed a neural network language model (NNLM) which combined a neural network with natural language processing to train word embeddings. After that, Mikolov et al. [21] proposed Word2Vec, a very effective language model inspired from NNLM. Word2Vec has received increasing attention in recent years and has been successfully applied to many tasks, such as document classification [22,23], knowledge graph extraction [24,25], and implicit matrix factorization [26,27].

Shown in Figure 2a is an example of the Word2Vec model. Given a sentence sequence  $Sent = [w_1, w_2, \dots, w_{l_w}]$ , where  $l_w$  is the length of the sequence and  $w_i \in \mathbb{R}^{d_w}$  indicates the word vector of the  $i^{th}$  word in the sentence. Let  $Context(w_i) = [w_{i-k}, \dots, w_{i-1}, w_{i+1}, \dots, w_{i+k}]$  be the context of  $w_i$ , where  $2k$  is the size of the contextual window. Let  $p(w_i|Context(w_i))$  be the probability of the  $i^{th}$  word in the sentence being  $w_i$ . The target of the model is to optimize the log maximum likelihood estimation (logMLE):

$$\mathcal{L}_w(\text{MLE}) = \sum_{w_i \in S} \log p(w_i|Context(w_i)). \quad (1)$$

However, word-level embedding methods may suffer from the following shortcomings:

1. Word segmentation is an essential preprocessing step for most Chinese NLP tasks. The quality of word segmentation determines the quality of word embeddings, which then influence the accuracy of the down-stream model. Moreover, word segmentation methods suffer a great reduction in accuracy when directly applied to specific domains (e.g., medical domain).

2. The words that do not or rarely appear in the lexicon or dictionary are called unseen words or rare words. Unseen or rare words may influence the quality of word embeddings.
3. As the number of words is far greater than that of characters, training word embeddings or adopting word embeddings as features may consume a great deal of computing resources and storage resources.

Inspired by Zhang et al. [28], we separate the sentence into individual characters. In order to train distributed representation of characters, we have made minor changes to Word2Vec. As can be seen from Figure 2b, in the context window, each character is used to predict the middle character. After being trained, each character is mapped into a fixed-length vector  $c_i \in \mathbb{R}^{d_c}$ , where  $d_c$  is the dimensionality.

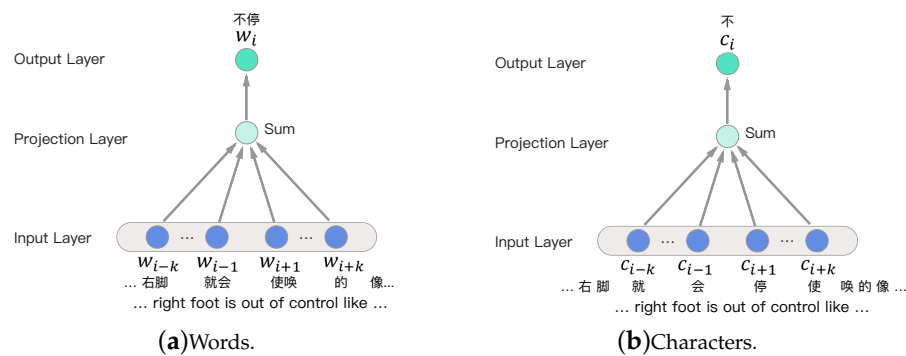


Figure 2. Distributed representation of words and characters.

More specifically, given a sentence sequence  $Sent = [c_1, c_2, \dots, c_{l_c}]$ , where  $l_c$  is the number of characters in the sequence, the context of  $c_i$  is  $Context(c_i) = [c_{i-k}, \dots, c_{i-1}, c_{i+1}, \dots, c_{i+k}]$ . Therefore, Equation (1) can be modified into:

$$\mathcal{L}_c(\text{MLE}) = \sum_{c_i \in S} \log p(c_i | Context(c_i)). \quad (2)$$

Character embeddings reduce the chances of errors brought about by word segmentation algorithms [28]. Furthermore, since the number of characters is much less than the number of words, and the number of unseen or rare characters is also much smaller than that of unseen or rare words, character-level methods reduce the size of the representation of sentences and accelerate the computation. Therefore, character-level embeddings are able to mitigate the adverse effects of word embeddings described above.

Character embeddings have been applied to many tasks, such as neural machine translation [29–32], text classification [33], English factoid question answering [34], and Chinese dependency parsing [28].

Though character embeddings are able to tackle the second and third issues listed above both for English and Chinese, it should be noted that there are still differences between Chinese and English with regard to character embeddings. Character-level methods in Chinese are mainly focused on mitigating the negative impact of word segmentation because there is no natural delimiter in Chinese texts to separate the phrases [28], while various methods in English have been proposed to solve the phenomenon that the words which share the same lexeme have different morphological forms [29]. Besides, the number of Chinese characters is far greater than English characters (only 26) and the characters in Chinese texts also contain more information than those in English.



### 3.2. A Single Convolutional Neural Network Architecture

As is already known, the meaning of a word (character) is usually affected by its surroundings, especially the context. Currently, the convolutional neural network (CNN) has been proven to be superior in a wide variety of NLP tasks [17,35–37], since the CNN is capable of explicitly capturing local contextual information. The network does not depend on other external information such as part-of-speech tags or a parse tree.

In general, a convolutional neural network architecture consists of two main steps: convolution and pooling. The convolution step extracts local context features using a fixed-size sliding window, while the pooling step selects the maximum or average value of the features extracted from the former layer to reduce the presentation.

More specifically, the question and answer are represented by character embedding sequences having a fixed length:  $\{c_1, \dots, c_{l_c}\}$ . We denote the dimensionality of the character vectors by  $d_c$ , and each element is a real-valued number, thus  $c_i \in \mathbb{R}^{d_c}$ . Each sentence is normalized to a fixed length sequence by adding zero paddings if the sentence is short or by truncating the excess otherwise. After embeddings, each question and answer can be presented by matrix  $Q_e \in \mathbb{R}^{l_c \times d_c}$  and  $A_e \in \mathbb{R}^{l_c \times d_c}$ .

Given a sequence  $Z = [z_1, z_2, \dots, z_{l_c-s+1}]$ , where the size of feature map is  $s$ , and  $z_i = [c_i, c_{i+1}, \dots, c_{i+s-1}] \in \mathbb{R}^{s \times d_c}$  is the concatenation of continuous  $s$  character vectors of the sentence, we can define the convolutional operation as follows:

$$O_j = f(W_j \circ [z_1, z_2, \dots, z_{l_c-s+1}] + b), \quad (3)$$

where  $O_j \in \mathbb{R}^{l_c-s+1}$  is the output of convolutional operation, matrix  $W_j \in \mathbb{R}^{s \times d}$  and vector  $b$  are the parameters to be trained,  $f(\cdot)$  is the activation function, and  $W \circ Z$  indicates the element-wise multiplication of  $W$  with each element in  $Z$ . If the number of filter maps is  $d_o$ , the output of the layer is  $O = [O_1, O_2, \dots, O_{d_o}] \in \mathbb{R}^{(l_c-s+1) \times d_o}$ . Thus, embedding matrices  $Q_E \in \mathbb{R}^{l_c \times d_c}$  and  $A_E \in \mathbb{R}^{l_c \times d_c}$  are converted into  $Q_O \in \mathbb{R}^{(l_c-s+1) \times d_o}$  and  $A_O \in \mathbb{R}^{(l_c-s+1) \times d_o}$  through a convolutional neural layer and by sharing the same parameters ( $W_j$  and  $b$ ).

A pooling layer is then applied after the convolutional layer. Max pooling and average pooling are commonly used in the pooling layer, which chooses the max or average value of the features extracted from the former layer to reduce the representation. In this paper, we use 1-max pooling as our method to select the max value of each filter, and max value may be relatively more sensitive to the existence of some pattern in pooled region. The max pooling operation is shown as follows:

$$p = [\max O_1, \max O_2, \dots, \max O_{d_o}], \quad (4)$$

where  $\max O_i$  selects the max value in  $O_i$ , and  $p \in \mathbb{R}^{d_o}$  is the output vector of the pooling layer.

A vector  $q \in \mathbb{R}^{d_o}$  is acquired after max pooling layer, which can be used as a representation of a question. Similarly, a vector  $a \in \mathbb{R}^{d_o}$  is also extracted from the answer. The formula that may be used to measure the similarity between the questions and the answers is given as follows:

$$\text{Sim}(q, a) = \text{Cos}(q, a) = \frac{\|q \cdot a\|}{\|q\| \cdot \|a\|}, \quad (5)$$

where  $\|\cdot\|$  is the length of the vector.

### 3.3. A Multi-Scale Convolutional Neural Networks Architecture

As we described above, convolutional neural networks are capable of capturing local contextual information. However, the single CNN architecture usually has single-sized feature maps with a fixed sliding convolutional window. Therefore, the scale of information extracted from the model is limited. Considering the expression structure of Chinese phrases and the variety of expressions available in the language, a single CNN architecture may be insufficient to extract character-level information.

The multi-scale convolutional neural networks (multiCNNs) architecture works in a similar way to the singleCNNs described above, except that they employ feature maps at different scales to extract the information. As shown in Figure 1, the question and the answer are represented using a concatenation of several vectors from the max pooling layer.

More specifically, given a set of filter map sizes  $S = \{s_1, s_2, \dots, s_t\}$ , where the  $i^{th}$  CNN filter's map size is denoted with  $s_i$ , similar to Equation (3), the output of the convolution for the  $i^{th}$  CNN is given by:

$$O_j^{s_i} = f(W_j^{s_i} \circ [z_1, z_2, \dots, z_{l_c-s_i+1}] + b^{s_i}), \quad (6)$$

where  $O_j^{s_i} \in \mathbb{R}^{l_c-s_i+1}$ .

Therefore, the output of the layer becomes  $O^{s_i} = [O_1^{s_i}, O_2^{s_i}, \dots, O_{d_o}^{s_i}]$ .

Similarly, after the pooling layer, the output vector from the  $i^{th}$  CNN is:

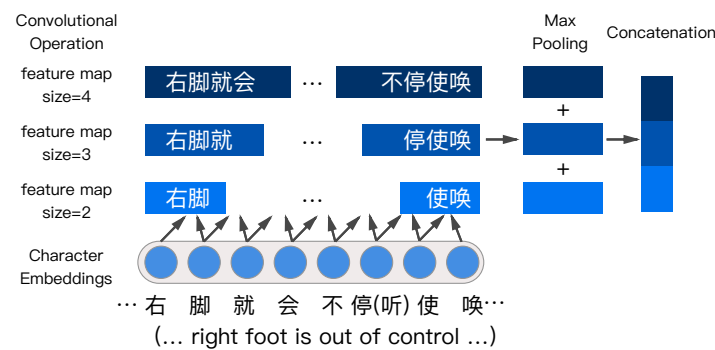
$$p^{s_i} = [\max O_1^{s_i}, \max O_2^{s_i}, \dots, \max O_{d_o}^{s_i}]. \quad (7)$$

Unlike singleCNN, we concatenate output vectors from different-scale CNNs as the final representation of the question or the answer:

$$p = [p^{s_1}, p^{s_2}, \dots, p^{s_t}]. \quad (8)$$

After that, the similarity measurement is calculated similarly using Equation (5).

The multiCNNs architecture is fully capable of extracting the relevant language features from Chinese texts. Figure 3 shows an example of the process using which the multiCNNs architecture extracts local information from a region. As we already know, different Chinese phrases usually contain different numbers of characters; therefore, a single convolutional neural network will perform convolutional operation over a fixed-length region, which is similar to combining several adjacent characters into words. Therefore, the multiCNNs architecture performs convolutional operation over different fixed-length regions and extracts a different number of adjacent character embeddings.



**Figure 3.** A multi-scale convolutional neural networks architecture. The architecture extracts local features of character embeddings using multi-scale convolutional feature maps. The features are then fed into max pooling layer to reduce the representation. After that, vectors from the pooling layer are concatenated into a single vector.

### 3.4. Objective Function

Given a question  $q_i$ , the ground truth answer is  $a_i^+$ , while  $a_i^-$  is the incorrect answer randomly sampled from the complete answer pool. A good network would be capable of maximizing  $Sim(q_i, a_i^+)$  while minimizing  $Sim(q_i, a_i^-)$ .



In order to train the neural networks, we follow an approach to [1,17,19], whereby we define the training max-margin loss function as follows:

$$\mathcal{L} = \max\{0, M - \text{Sim}(q_i, a_i^+) + \text{Sim}(q_i, a_i^-)\}, \quad (9)$$

where  $M$  is the margin value, which is a constant. If  $\text{Sim}(q_i, a_i^+) - \text{Sim}(q_i, a_i^-) > M$ , the cost is zero and no updates are required.

When training the network, we provide each question separately, the ground truth answer and a randomly-sampled negative answer to the network for up to  $K$  times. Then, we compute the loss  $\mathcal{L}$ , and use optimizers such as GradientDescentOptimizer or AdagradOptimizer to update the parameters of networks.

## 4. Experiments and Results

### 4.1. Datasets

As there are no public Chinese medical questions and answers datasets available, we have developed and collected a dataset from the Internet. The dataset was acquired from an online Chinese medical question answering forum (<http://www.xywy.com/>). Questions are proposed by users and are responded to by qualified medical doctors. Each question is often answered by several doctors, and only certified doctors are eligible to answer questions.

To the best of our knowledge, this is the first publicly available dataset based on Chinese medical questions and answers. The dataset is in version 1.0 and is available for non-commercial research (<https://github.com/zhangsheng93/cMedQA>). We will update and expand the database from time to time. In order to protect privacy, the data is anonymized and no personal information is included.

As shown in Table 2, the data which we refer to as cMedQA is split into three sets—namely, training set, development set, and test set. The second and third columns show the number of questions and answers, respectively. The following two columns illustrate the average number of words per question and answer, while the last two columns indicate the average number of characters.

**Table 2.** Statistics of the cMedQA dataset.

	#Ques	#Ans	Ave. #Words Per Question	Ave. #Words Per Answer	Ave. #Characters Per Question	Ave. #Characters Per Answer
Train	50,000	94,134	97	169	120	212
Dev	2000	3774	94	172	117	216
Test	2000	3835	96	168	119	211
Total	54,000	101,743	96	169	119	212

While training the framework, for each question  $q_i$  in training data, there is a ground truth answer  $a_i^+$  (if a question has multiple ground truth answers, we will choose one randomly each time) and a randomly sampled answer  $a_i^-$  from the complete answer pool. Using this mechanism, we generate 30 tuples of  $(q_i, a_i^+, a_i^-)$  for each question. Thus, during the training phase, a total of 1,500,000 tuples are fed into the network at each epoch.

To evaluate the framework, for each question in the development set and test set, we randomly sample 100 candidate answers, including ground truth answers. The development set is used for parameter search and optimization, while the test set is used to evaluate the models. It should be noted that the whole answer space can be regarded as the candidate pool, so each question should be computed for similarities with 101,743 candidate answers. However, this is impractical because of time-consuming computations [1]. Therefore, we set the pool size to be 100 in our study, which is practical and still a challenging task.

## 4.2. Metrics

In this study, the top-1 accuracy is used as a metric to evaluate the accuracy of the technique. Top-k accuracy is a commonly used metric in many different information retrieval tasks. The definition of top-k accuracy ( $ACC@k$ ) is given as follows:

$$ACC@k = \frac{1}{N} \sum_{i=1}^N 1[a_i \in C_i^k], \quad (10)$$

where  $a_i$  is the ground truth answer for question  $q_i$ , and  $C_i^k$  is the candidate set with top-k highest similar answers.  $1[\cdot] \rightarrow \{0, 1\}$  denotes the indicator function. When the condition in parentheses is true, the function value is 1, otherwise it is 0.

Unlike traditional information retrieval [38], answering a question requires the provision of the best possible answer rather than a ranked list of answers. Therefore, we have adopted top-1 accuracy ( $ACC@1$ ) as our metric, which is a highly demanding measurement. Specifically, in terms of a question, if the size of the answers candidate pool is 100, the  $ACC@1$  for random selection methods is only 1%. It should be noted that in our experiment, if a question has multiple ground truth answers, any correctly selected ground truth answer will contribute to the accuracy.

## 4.3. Baselines

- \* **Character Matching:** Character matching method counts the number of characters that are the same in the question and the answer.
- \* **Word Matching:** Similar to the character matching method, the word matching method counts the number of words that are the same. However, this method requires word segmentation; therefore, we use two word segmentation toolkits and present a discussion of the impact of different toolkits.
- \* **BM25:** BM25 (Best Matching) is a ranking function in Information Retrieval (IR); it has also been used in question answer matching [8,16]. The definition of BM25 is given as follows:

$$BM25(q_i, a_i) = \sum_{w_j \in q_i} IDF(w_j) \cdot \frac{f(w_j, a_i) \cdot (k+1)}{f(w_j, a_i) + k \cdot (1 - b + b \cdot \frac{|a_i|}{|a|_{avg}})}, \quad (11)$$

where  $IDF(w_j)$  is the inverse document frequency (IDF) value of word (character)  $w_j$  in question,  $|a_i|$  is the length of the answer  $a_i$ ,  $|a|_{avg}$  is the average length of answers, and  $f(w_j, a_i)$  is the frequency of  $w_j$  in  $a_i$ .  $k$  and  $b$  are free parameters. In our experiment, according to Christopher et al. [39], we set  $k = 2.0$  and  $b = 0.75$ .

- \* **Average of Embeddings:** Character (Word) embeddings contain semantic and syntactic information, and an average of these can be used to directly represent features. Thus, we compute the average of each question and answer embeddings, and calculate the cosine similarity between the two average embeddings.
- \* **Embedding Matching:** Inspired by Li et al. [14], we compute the similarity of two sentences by calculating the similarity of words (characters). We modified the algorithm by computing the similarity of every two embeddings, and find the best matching score of each word (character) of the question. The answer with the best matching score will be chosen.
- \* **biLSTM:** Tan et al. [2] used bi-directional LSTM networks to extract the semantic representations of questions and answers. LSTM networks are a variant of recurrent neural networks (RNN), which are capable of capturing sequence information [2,40,41].

#### 4.4. Experimental Settings

Models presented in our paper were implemented using TensorFlow and TensorLayer from scratch. All questions and answers in the training set are used to train the character embeddings using gensim (<https://radimrehurek.com/gensim/>). In order to make a comparison between word embeddings and character embeddings, we use two word segmentation toolkits (Jieba and ICTCLAS) for Chinese words segmentation. ICTCLAS is a popular toolkit and reported good F-scores in many segmentation tasks (<http://thulac.thunlp.org/>). Jieba provides an easy-to-use interface, and we use it as a control. The Python package gensim is used to train both word embeddings and character embeddings. After training, we get dictionaries with 35,933 (Jieba) words, 20,720 (ICTCLAS) words, and 3922 characters, respectively. The max sequence length of the question and answer are fixed at 400 for characters and 200 for words. We add paddings to the beginning if the original sequence is short and truncate the excessive parts.

For the CNN, the filter has a size of 3 with 800 feature maps, while the multiCNN architectures have filters of sizes (3,4), respectively, with 800 feature maps each. In this paper, we also use biLSTM [2] to make a comparison. The biLSTM has the output length of 200 in each direction and the hidden states are provided directly for pooling.

AdagradOptimizer is used in this paper, the learning rate is initially set to 0.01, and the margin value is set to 0.05.

#### 4.5. Results

In this section, a detailed examination of the experimental results are presented. Table 3 summarizes the results for different approaches on cMedQA.

**Table 3.** The top-1 accuracy results of the models. biLSTM: bi-directional long short-term memory network; Multi-CNN: multi-scale CNN.

	Embeddings	Model	Dev (%)	Test (%)
1		Random Selection	01.00	01.00
2		Word Matching (Jieba)	37.05	36.60
3		Word Matching (ICTCLAS)	36.15	37.30
4	None	Character Matching	33.65	34.90
5		BM25 (Jieba)	37.60	40.00
6		BM25 (ICTCLAS)	41.35	41.35
7		BM25 (Character)	44.80	45.40
8	Word (Jieba)		15.60	16.80
9	Word (ICTCLAS)	Average of Embeddings	18.05	18.75
10	Character Embeddings		24.90	24.00
11	Word (Jieba)		24.55	23.65
12	Word (ICTCLAS)	Embedding Matching	27.85	29.10
13	Character Embeddings		30.80	32.30
14		SingleCNN [1]	46.30	47.65
15	Word Embeddings (Jieba)	BiLSTM [2]	51.70	52.70
16		MultiCNNs	48.40	51.15
17		SingleCNN [1]	53.25	53.75
18	Word Embeddings (ICTCLAS)	BiLSTM [2]	56.15	57.65
19		MultiCNNs	54.05	55.40
20		SingleCNN	64.50	64.05
21	Character Embeddings	BiLSTM	61.65	63.20
22		MultiCNNs	<b>65.35</b>	<b>64.75</b>

Rows 1 to 13 provide a survey of the results of the baseline methods without using neural network architectures. More specifically, as can be seen from Rows 2 to 4, which show the results of matching based methods, the results of the two methods differ by less than 1%. Due to the abundant information contained in words, word-based methods (Rows 2 and Row 3) surpass the character-based method (Row 4). In comparison to word (character) matching methods, BM25 methods (Rows 5 to 7) utilize more statistical information and show up to 11% improvement.

Row 8 to Row 13 show the results of embedding-based shallow methods. Character embedding methods (Row 10,13) take the lead, which indicates that the semantic information of embeddings is better extracted from characters. Word embeddings preprocessed by the ICTCLAS toolkit rank second. However, the average of the embeddings methods is inferior to word matching methods, which may be due to their incomplete semantic information extraction capability.

The following three rows (Rows 14 to 16) illustrate the results of three deep neural models when they are fed with word embeddings preprocessed by the Jieba toolkit. It is observed that the biLSTMs architecture stands out amongst these three models. The potential reason may be that recurrent neural networks include the variant LSTM which is capable of extracting sequence information. They are able to extract the semantic information of the whole sentence, which can effectively reduce the semantic gap between the questions and answers. Our results are consistent with experimental results reported by Tan et al. [19]. Moreover, the multiCNN model shows more than 2 to 3% improvement on the development and test data-sets, respectively.

The next three rows (Rows 17 to 19) show the similarity trends: biLSTM is ranked first, followed by multiCNNs. It is shown that the ICTCLAS method is much more accurate for words segmentation as compared to Jieba in most cases, although it runs slower than Jieba (<http://thulac.thunlp.org/>). It is also shown in the table that the word embedding methods based on ICTCLAS (Rows 17 to 19) outperform the methods based on Jieba (Rows 14 to 16).

The following three rows (Rows 20 to 22) summarize the results of these models on character embeddings. The multiCNNs architecture achieves the highest performance due to the benefits of multi-scale local information extraction. The singleCNN model ranks second. The performance indicates the effectiveness of the multi-scale convolution operations. Compared to biLSTM models, CNN-based models are shown to be superior to the LSTM-based model with regard to character embeddings. The reason for this is that the CNN models take advantage of local information, which is similar to combining several adjacent characters into words. The biLSTM architecture draws greater attention to sequence information and less attention to local information, as the effectiveness of character-level information extraction is inferior to CNN models.

To summarize, neural network-based methods are superior to word (character) matching methods, which demonstrates that highly diverse and non-formal expressions may confuse the matching methods. Character embedding methods (Rows 20 to 22) significantly outperform the word-level methods (Rows 14 to 19) by up to 19%. The improvements for both CNN-based models and the RNN-based model from character embeddings are remarkable, which demonstrates the effectiveness of our character-level embeddings model for medical-domain questions answer matching.

#### 4.6. Discussion

##### 4.6.1. Random Embeddings vs. Pre-Trained Embeddings

A character embedding matrix is a concatenation of character vectors. The number of rows in the matrix is the total number of characters appearing in texts, while the number of columns is the dimensionality of the character vector.

Normally, there are three approaches to initializing a character (word) embeddings matrix. First, an embedding matrix is randomly initialized with uniformly distributed random real values when constructing a deep neural network. The embeddings matrix is regarded as a parameter and will be updated while training the neural network. This method is used by [35,37,42]. Another initialization

method is to use publicly available pre-trained embeddings. These embeddings are pre-trained with some large corpora, such as Wikipedia and Google News. Because of the large number of texts in the corpus, the trained vectors are rich in semantic information and are suitable to be directly used to other models, especially when the dataset is small. This strategy is widely used in general-domain tasks [2,3,43]. Finally, we can also train the embedding matrix with texts according to the specific task. Vectors trained in this way correspond to the dataset. Fundamentally, our proposed architectures belong to the last category.

Since we are focusing on medical care, no pre-trained embedding matrix from a large corpus existed for this application. Hence, we decide to make a comparison between randomly initialized embeddings and pre-trained embeddings from the cMedQA dataset. We also changed the embedding initialization method for the multiCNNs architecture and fixed other parameters.

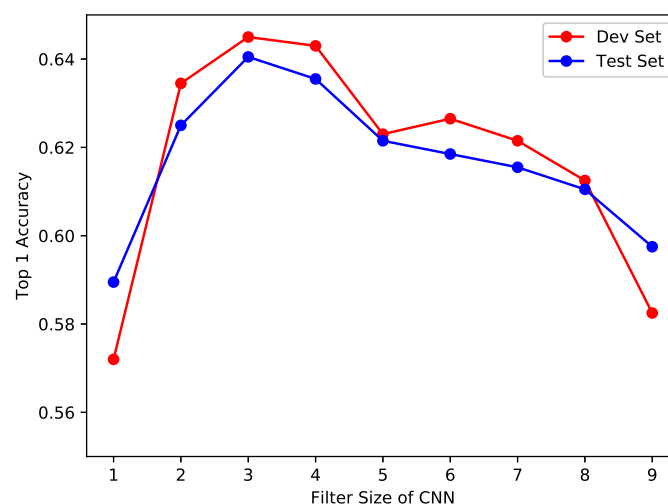
The multiCNNs architecture with randomly initialized embeddings acquired 62.25% and 63.45% accuracy on development set and test set, respectively. Compared to the results in Table 3, it shows an over 3% decrease on development sets and 1% on test sets, which illustrates that pre-training embeddings from the task-related corpus contributes to the accuracies.

#### 4.6.2. Relationship between SingleCNN and MultiCNNs

From the description in Section 3.3, the multi-scale CNNs architectures can be regarded as an ensemble of different types of CNNs. Therefore, we discuss the relationship between singleCNN and MultiCNNs.

Figure 4 illustrates the top-1 accuracy results for different CNNs with different feature map sizes. As can be seen from the figure, the optimal size of a single feature map is three for both the development set and test set, followed by four and two. It also shows a sharp decrease when the filter size goes up. It is easily understood that words are made up of characters, and two, three, or four are the common number of characters that compose a word.

Table 4 provides a survey of the effect of combining different filter sizes. As can be seen from the table, when we set the filter sizes of multiCNNs to (3,4), it achieves the highest performance, which is in accordance with the results in Table 4. Similarly, when we combine different top sizes (Rows 1 to 4), the performance surpasses that of singleCNN.



**Figure 4.** Top-1 Accuracy according to the filter size of CNN.

**Table 4.** Results of different types of feature maps.

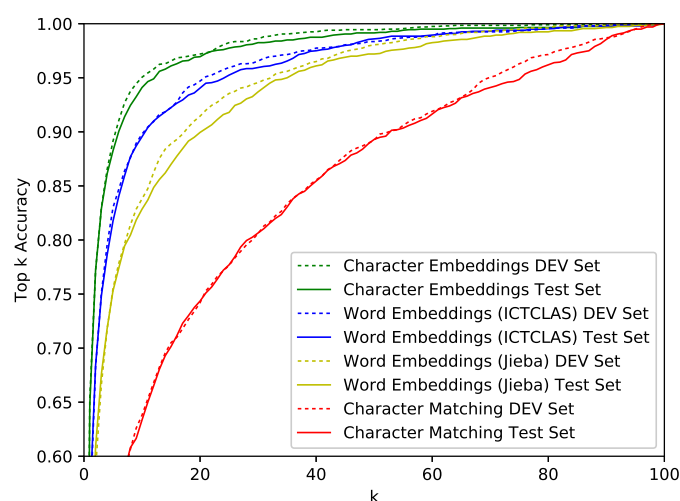
MultiCNNs	Dev (%)	Test (%)
(3,4)	<b>65.35</b>	<b>64.75</b>
(2,3)	65.00	64.30
(2,4)	64.95	64.15
(2,3,4)	64.90	64.50
(2,4,6)	64.35	64.05
(2,3,5)	64.00	64.25
(3,5,7)	63.40	62.85

#### 4.6.3. Accuracy of Multiple Selection

Traditional information retrieval methods provide several candidates for users to use to select relevant information, while the question answer matching requires the most credible answer rather than a ranked list of answers. However, the lexical gap between questions and answers usually confuses the matching-based methods. Therefore, selecting the correct answer is a crucial step in question answering and requires non-trivial techniques to fully capture the complex and diverse semantic relationship between the questions and answers.

We have proved the effectiveness of our framework in mitigating the lexical gap problem. However, in order to demonstrate that our approaches are superior to existing methods (even when it is allowed to acquire multiple candidate answers for users), we expand top-1 accuracy to top-k accuracy and examine scores for different k values.

Figure 5 shows top-k accuracy results for different k values. Embedding-based methods are applied to multiCNNs architecture. As is shown in the figure, embedding-based multiCNNs architectures outperform matching-based models, regardless of the k value. In addition, these embedding-based methods show rapid growth to 0.95 when k is small. When the accuracy is fixed to 0.90, the k values of green lines are about twice as low as blue lines and four times lower than yellow lines. The same phenomenon occurs when the accuracy is fixed at 0.95, which demonstrates that our character-level embedding-based multiCNNs architecture retrieves relevant information effectively.

**Figure 5.** Top-k Accuracy according to k.

## 5. Conclusions

In this paper, we presented an approach to address the Chinese medical question answer matching task using an end-to-end character-level multi-scale convolutional neural network framework. The framework does not require any extra rule-based patterns, syntactic information, or any



other external resources. According to the experimental results, our framework outperforms word-embedding methods, and our multi-scale strategy is capable of capturing character-level information well.

In the future, we would extend our model for disease diagnosis and treatment recommendations according to the description of the symptoms given by users. We aim to construct a hybrid framework that integrates medical domain knowledge into our medical question answering technique. In addition, Chung et al. [41] propose a recurrent neural network model to infer the scale of data by themselves. If the network can discover the latent structure of the sequence, then it can efficiently adjust multi-scale representation of the network to achieve better performance.

**Acknowledgments:** The work in this paper is financially supported by National Natural Science Foundation of China under grant (No. 71331008) and (No. 61105124).

**Author Contributions:** Sheng Zhang and Hui Wang conceived and designed the experiments; Sheng Zhang performed the experiments; Sheng Zhang, Jiajun Cheng and Pei Li analyzed the data; Zhaoyun Ding contributed analysis tools; Sheng Zhang, Xin Zhang and Jiajun Cheng wrote the paper.

**Conflicts of Interest:** The authors declare no conflict of interest. The founding sponsors had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, and in the decision to publish the results.

## References

1. Feng, M.; Xiang, B.; Glass, M.R.; Wang, L.; Zhou, B. Applying deep learning to answer selection: A study and an open task. In Proceedings of the 2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU), Scottsdale, AZ, USA, 13–17 December 2015; pp. 813–820.
2. Tan, M.; dos Santos, C.; Xiang, B.; Zhou, B. Improved representation learning for question answer matching. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany, 7–12 August 2016.
3. Qiu, X.; Huang, X. Convolutional Neural Tensor Network Architecture for Community-Based Question Answering. In Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI), Buenos Aires, Argentina, 25–31 July 2015; pp. 1305–1311.
4. Jain, S.; Dodiya, T. Rule Based Architecture for Medical Question Answering System. In Proceedings of the Second International Conference on Soft Computing for Problem Solving (SocProS 2012), Rajasthan, India, 28–30 December 2012; Springer: New Delhi, India, 2014; pp. 1225–1233.
5. Wang, J.; Man, C.; Zhao, Y.; Wang, F. An answer recommendation algorithm for medical community question answering systems. In Proceedings of the 2016 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI), Beijing, China, 10–12 July 2016; pp. 139–144.
6. Ben Abacha, A.; Zweigenbaum, P. Medical question answering: Translating medical questions into sparql queries. In Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium, Miami, FL, USA, 28–30 January 2012; ACM: New York, NY, USA, 2012; pp. 41–50.
7. Abacha, A.B.; Zweigenbaum, P. MEANS: A medical question-answering system combining NLP techniques and semantic Web technologies. *Inf. Process. Manag.* **2015**, *51*, 570–594.
8. Li, C. Research and Application on Intelligent Disease Guidance and Medical Question Answering Method. Master's Thesis, Dalian University of Technology, Dalian, China, 2016.
9. Robertson, S.; Zaragoza, H. The probabilistic relevance framework: BM25 and beyond. *Found. Trends® Inf. Retr.* **2009**, *3*, 333–389.
10. Mihalcea, R.; Tarau, P. *TextRank: Bringing Order into Texts*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2004.
11. Goodwin, T.R.; Harabagiu, S.M. Medical Question Answering for Clinical Decision Support. In Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, Indianapolis, IN, USA, 24–28 October 2016; ACM: New York, NY, USA, 2016; pp. 297–306.
12. Roberts, K.; Simpson, M.; Demner-Fushman, D.; Voorhees, E.; Hersh, W. State-of-the-art in biomedical literature retrieval for clinical cases: A survey of the TREC 2014 CDS track. *Inf. Retr. J.* **2016**, *19*, 113–148.

13. Li, X.; Croft, W.B. Evaluating question-answering techniques in Chinese. In Proceedings of the First International Conference on Human language Technology Research, San Diego, CA, USA, 18–21 March 2001; Association for Computational Linguistics: Stroudsburg, PA, USA, 2001; pp. 1–6.
14. Li, S.; Zhang, J.; Huang, X.; Bai, S.; Liu, Q. Semantic computation in a Chinese question-answering system. *J. Comput. Sci. Technol.* **2002**, *17*, 933–939.
15. Li, T.; Hao, Y.; Zhu, X.; Zhang, X. A Chinese question answering system for specific domain. In *WAIM 2014: Web-Age Information Management, Proceedings of the International Conference on Web-Age Information Management, Macau, China, 3–5 June 2014*; Springer: New York, NY, USA, 2014; pp. 590–601.
16. Wang, B.; Niu, J.; Ma, L.; Zhang, Y.; Zhang, L.; Li, J.; Zhang, P.; Song, D. A Chinese Question Answering Approach Integrating Count-Based and Embedding-Based Features. In *Natural Language Understanding and Intelligent Applications, Proceedings of the International Conference on Computer Processing of Oriental Languages, Kunming, China, 2–6 December 2016*; Springer: New York, NY, USA, 2016; pp. 934–941.
17. Hu, B.; Lu, Z.; Li, H.; Chen, Q. Convolutional neural network architectures for matching natural language sentences. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; pp. 2042–2050.
18. Zhou, X.; Hu, B.; Chen, Q.; Tang, B.; Wang, X. Answer sequence learning with neural networks for answer selection in community question answering. *arXiv* **2015**, arXiv:1506.06490.
19. Tan, M.; Xiang, B.; Zhou, B. LSTM-based Deep Learning Models for non-factoid answer selection. *arXiv* **2015**, arXiv:1511.04108.
20. Bengio, Y.; Ducharme, R.; Vincent, P.; Jauvin, C. A neural probabilistic language model. *J. Mach. Learn. Res.* **2003**, *3*, 1137–1155.
21. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.; Dean, J. Distributed Representations of Words and Phrases and their Compositionality. *arXiv* **2013**, arXiv:1310.4546v1.
22. Taddy, M. Document Classification by Inversion of Distributed Language Representations. *arXiv* **2015**, arXiv:1504.07295.
23. Huang, C.; Qiu, X.; Huang, X. Text classification with document embeddings. In *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*; Springer: New York, NY, USA, 2014; pp. 131–140.
24. Wang, Z.; Zhang, J.; Feng, J.; Chen, Z. Knowledge Graph Embedding by Translating on Hyperplanes. In Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, Québec City, QC, Canada, 27–31 July 2014; pp. 1112–1119.
25. Lin, Y.; Liu, Z.; Sun, M.; Liu, Y.; Zhu, X. Learning Entity and Relation Embeddings for Knowledge Graph Completion. In Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI), Austin, TX, USA, 25–30 January 2015; pp. 2181–2187.
26. Levy, O.; Goldberg, Y. Neural word embedding as implicit matrix factorization. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; pp. 2177–2185.
27. Yang, C.; Liu, Z. Comprehend deepwalk as matrix factorization. *arXiv* **2015**, arXiv:1501.00358.
28. Zhang, M.; Zhang, Y.; Che, W.; Liu, T. Character-level chinese dependency parsing. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, MD, USA, 23–25 June 2014.
29. Ling, W.; Trancoso, I.; Dyer, C.; Black, A.W. Character-based neural machine translation. *arXiv* **2015**, arXiv:1511.04586.
30. Chung, J.; Cho, K.; Bengio, Y. A character-level decoder without explicit segmentation for neural machine translation. *arXiv* **2016**, arXiv:1603.06147.
31. Luong, M.T.; Manning, C.D. Achieving open vocabulary neural machine translation with hybrid word-character models. *arXiv* **2016**, arXiv:1604.00788.
32. Costa-Jussa, M.R.; Fonollosa, J.A. Character-based neural machine translation. *arXiv* **2016**, arXiv:1603.00810.
33. Zhang, X.; Zhao, J.; LeCun, Y. Character-level convolutional networks for text classification. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; pp. 649–657.
34. Golub, D.; He, X. Character-level question answering with attention. *arXiv* **2016**, arXiv:1604.00727.

35. Kalchbrenner, N.; Grefenstette, E.; Blunsom, P.; Kartsaklis, D.; Sadrzadeh, M. A Convolutional Neural Network for Modelling Sentences. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, MD, USA, 22–27 June 2014; pp. 212–217.
36. Kim, Y. Convolutional Neural Networks for Sentence Classification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1746–1751.
37. Yin, W.; Schütze, H.; Xiang, B.; Zhou, B. ABCNN: Attention-Based Convolutional Neural Network for Modeling Sentence Pairs. *Trans. Assoc. Comput. Linguist.* **2016**, *4*, 259–272.
38. Goeuriot, L.; Jones, G.J.; Kelly, L.; Müller, H.; Zobel, J. Medical information retrieval: Introduction to the special issue. *Inf. Retr. J.* **2016**, *19*, 1–5.
39. Christopher, D.M.; Prabhakar, R.; Hinrich, S. Introduction to information retrieval. *Introd. Inf. Retr.* **2008**, *151*, 177.
40. Hsu, W.N.; Zhang, Y.; Glass, J. Recurrent Neural Network Encoder with Attention for Community Question Answering. *arXiv* **2016**, arXiv:1603.07044v1.
41. Chung, J.; Ahn, S.; Bengio, Y. Hierarchical multiscale recurrent neural networks. *arXiv* **2016**, arXiv:1609.01704.
42. Cui, Y.; Liu, T.; Chen, Z.; Wang, S.; Hu, G. Consensus attention-based neural networks for chinese reading comprehension. *arXiv* **2016**, arXiv:1607.02250.
43. Yu, L.; Hermann, K.M.; Blunsom, P.; Pulman, S. Deep learning for answer sentence selection. *arXiv* **2014**, arXiv:1412.1632.



© 2017 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).