information

MDPI

Article

Chinese Knowledge Base Question Answering by Attention-Based Multi-Granularity Model

Cun Shen [1,2,3], Tinglei Huang [1,2,*], Xiao Liang [1,2], Feng Li [1,2] and Kun Fu [1,2,3]

1 Key Laboratory of Technology in Geo-Spatial Information Processing and Application System, Institute of Electronics, Chinese Academy of Sciences, Beijing 100190, China; shencun15@mails.ucas.ac.cn (C.S.); xliang@mail.ie.ac.cn (X.L.); lifeng@mail.ie.ac.cn (F.L.); kunfuiecas@gmail.com (K.F.)
2 Institute of Electronics, Chinese Academy of Sciences, Beijing 100190, China
3 School of Electronic, Electrical and Communication Engineering, University of Chinese Academy of Sciences, Beijing 100190, China
* Correspondence: tlhuang@mail.ie.ac.cn; Tel.: +86-10-5888-7208

Received: 22 January 2018; Accepted: 16 April 2018; Published: 19 April 2018

Abstract: Chinese knowledge base question answering (KBQA) is designed to answer the questions with the facts contained in a knowledge base. This task can be divided into two subtasks: topic entity extraction and relation selection. During the topic entity extraction stage, an entity extraction model is built to locate topic entities in questions. The Levenshtein Ratio entity linker is proposed to conduct effective entity linking. All the relevant subject-predicate-object (SPO) triples to topic entity are searched from the knowledge base as candidates. In relation selection, an attention-based multi-granularity interaction model (ABMGIM) is proposed. Two main contributions are as follows. First, a multi-granularity approach for text embedding is proposed. A nested character-level and word-level approach is used to concatenate the pre-trained embedding of a character with corresponding embedding on word-level. Second, we apply a hierarchical matching model for question representation in relation selection tasks, and attention mechanisms are imported for a fine-grained alignment between characters for relation selection. Experimental results show that our model achieves a competitive performance on the public dataset, which demonstrates its effectiveness.

Keywords: knowledge base question answering; topic entity extraction; relation selection; multi-granularity embeddings; attention mechanism

1. Introduction

Open-domain question answering is a challenging task that aims at providing corresponding answers to natural language questions. In recent years, large-scale knowledge bases of high quality are developing rapidly and have been widely applied in many fields. Typical examples include knowledge bases in English such as Freebase [1], DBpedia [2], and Chinese knowledge bases like zhishi.me [3], XLore [4], and CN-DBpedia (http://kw.fudan.edu.cn/cndbpedia/). Due to their structured form of knowledge, knowledge bases have become a significant resource of open-domain question answering. An increasing amount of research work focuses on knowledge base question answering (KBQA) [5,6]. KBQA enables people to query the knowledge base with natural language, which bridges the natural language and structured knowledge base. For KBQA, the answer to the target question is definitely extracted from knowledge bases, so the major challenge is to understand the query and pick up the best subject-predicate-object (SPO) triple from knowledge bases. For instance, given a question "特朗普是什么时候出生的？ || When was Trump born?" the task is first to locate an entity from the knowledge base that contains an entity like "唐纳德·特朗普 || Donald Trump" that describes the

mention "特朗普 || Trump", and then select a predicate like "出生日期 || date_of_birth" that is highly correlated with the description "是什么时候出生的 || When was ... born". This procedure resembles topic entity extraction and relation selection [7]. In this work, we conduct effective topic entity extraction by entity recognition and entity linking and put emphasis on relation selection task in order to find the golden answer to a question.

For topic entity extraction, the most widely used approach is to perform entity detection and entity linking over knowledge base obtaining a small subset of candidates from an overwhelming number of facts. If this subtask cannot be handled well, it tends to introduce more noise entities. Some previous studies achieve entity extraction by searching every n-grams word of a question in knowledge base [8,9], which needs to handle a large searching space. Berant et al. [5] use linguistic tools which deeply rely on logic forms of questions and some predefined rules. Other work [10] do not put emphasis on entity extraction and only use knowledge base API (e.g., Freebase API). In this paper, our first contribution is to present an effective entity linker to deal with this situation. Our entity linker first trains a Bi-LSTM-CRF model to do the entity mention detection. Based on this detected mention, we search it in the entity vocabulary. If it cannot match the knowledge base, then we introduce Levenshtein Ratio Entity Linker to improve linking accuracy.

Based on the results of entity linking, each predicate of the target entity is regarded as a relation candidate of next subtask. After that, the model works on relation selection, namely identify the relation which best matches the description of the given question. In previous work, deep learning methods are widely applied in the relation selection of KBQA. Yih et al. [11] model both questions and relations as tri-grams of characters with CNN. Golub et al. [9] take character-level information into account and import attention-based LSTM neural network. Yin et al. [12] propose an attentive pooling approach, which can obtain more accurate representations of relation. Yu et al. [13] combine word-level and relation-level representations and use hierarchical residual bi-directional LSTM to obtain question representations. These relation selection methods are all in accordance with the pattern of encoding and comparison, in which the neural network learns the vector representations of questions and relations, respectively, and then computes the similarity between the vectors as its semantic similarity. These only use word-level embeddings in the experiments, which do not fully utilize the semantic information. Different from English, Chinese characters usually contain specific meaning, thus we consider a multi-granularity approach combining character-level, word-level and relation-level for text embeddings, which is able to handle out-of-vocabulary (OOV) problems while still has the ability to exploit text semantics. Furthermore, attention mechanisms are incorporated to emphasize important units. Overall, we process a method to learn attention-based interactions between question and relation, and multi-granularity embeddings are also introduced to further improve the performance of relation selection. Firstly, the question is represented as a sequence of vectors with a two-layer bidirectional Gated Recurrent Unit (GRU) hierarchical matching networks and relation is represented with a Bi-GRU respectively, where the question is embedded as a sequence of characters with word information, while relation is modeled the same with relation-level as complementary. Then representation results are merged to vectors with attention mechanism. Finally, a logistic layer scores the semantic similarity based on the extracted features. In general, this paper contributes in two aspects:

- Propose a Chinese entity linker which is based on Levenshtein Ratio. The entity linker can effectively handle abbreviation, wrongly labeled and wrongly written entity mentions.
- Propose an attention-based multi-granularity interaction model (ABMGIM). Multi-granularity approach for text embeddings is applied. A nested character-level and word-level approach is used to concatenate the pre-trained embedding of a character with corresponding embedding on word-level. Furthermore, a two-layer Bi-GRUs with element-wise connections structure is incorporated to obtain better hidden representations of the given question, and attention mechanism is utilized for a fine-grained alignment between characters for relation selection.

2. Related Work

The primary goal of open-domain KBQA is to automatically answer the given question by selecting a fact from knowledge base which can best match. According to the characteristics of the methods, the ways to tackle this problem can be divided into three categories: semantic parsing based methods [5,11,14,15], information retrieval methods [8,16] and deep learning models [6,10,17]. Semantic parsing methods depend on linguistics rules to construct a semantic parser. They map natural language questions into structured expressions, such as logical forms. However, in such methods, important vocabularies are generally artificially generated, and such vocabularies usually lack domain adaptability. As for information retrieval methods, they convert semantic parsing problem into retrieval problem. They search all relative resources conveyed in questions from knowledge bases, and uses ranking algorithm to select the best fact from candidate answers. It is relatively easy to implement, and also does not have to design vocabularies manually. Bordes et al. [8] show that information retrieval method can also achieve good performance compared to semantic parsing. Recently, deep learning methods are widely applied in KBQA and gain a significant success. They can automatically extract features, and the results have gradually outperformed the traditional methods. Thus, we consider conducting our experiments through deep learning models.

According to the method process, many researchers handle KBQA in accordance with the following two procedures: topic entity extraction and relation selection. For topic entity extraction task, Bordes et al. [8] and Golub et al. [9] search all n-gram words of the given question and then link to knowledge base, and Yang et al. [18] also use all phrases appear in question text to extract linguistic features for classification. They both require a large searching space. Berant et al. [5] present an effective approach which relies on linguistic tools. However, it needs predefined rules and handcraft features. Xie et al. [19] use convolutional neural network to do entity extraction, which is similar to sequence labeling model. The disadvantage of this model is that it is hard to process variable length input sequence. In order to improve the efficiency, Dai et al. [17] use the golden entity to label mention as training data and construct a Bi-GRU-CRF tagging model to do mention detection. Yin et al. [12] also introduce a Bi-LSTM-CRF tagging model to improve the performance of the approach.

The relation selection task is the main part of the whole KBQA task. Bordes et al. [20] first apply deep learning to relation selection of KBQA, and since then various models are developed. Most of these methods are in accordance with the encoding-comparing paradigm, which maps questions and relation candidates to vectors respectively, then calculate the similarity between vectors as their semantic similarity. Dai et al. [17] propose a conditional focused neural network-based approach and initialize the relation token with pre-trained vector learned by TransE [21]. Golub et al. [9] consider character-level representation due to its advantages in handling OOV words and a smaller size of parameters. Yin et al. [12] propose an attentive convolutional neural network which uses CNN to encode questions and relations. In order to better match the predicate, the network applies attentive max-pooling mechanism to put emphasis on the relation description part of the given question. Jain [22] proposes a Factual Memory Network, which extracts and infers relevant facts from the knowledge base to obtain answers. Yu et al. [13] represent a hierarchical recurrent neural network with different abstract levels to detect relations in knowledge bases and combine word-level and relation-level to obtain relation representations. Xie et al. [19] utilize CNN-based deep structured semantic models (DSSM) to do the answer selection between questions and candidate relations, and variants of DSSM are developed such as extending it by Bi-LSTM and integrating CNN with Bi-LSTM in order to get rich representations. Yang et al. [18] train several answer ranking models, both CNN and information retrieval models are included. Stacking method is used as re-ranking ways to select the results of the base ranking model and output the final answer. The current state-of-the-art system of Chinese KBQA task is shown in the study of Lai et al. [23]. They propose an algorithm of subject predicate extraction. It is able to identify the subject-predicate pair which the question refers to, and translate it to knowledge base query to search the candidates. Furthermore, methods based on word vector similarity, answer patterns and predicate attention are imported to rate the candidate predicates.

However, these introduced pattern rules highly depend on the specific dataset. It may require new handcraft features when generalizing the way to other knowledge bases.

3. Our Approach

Figure 1 illustrates the architecture of our KBQA system. In entity extraction stage, we import named entity recognition methods to carry out entity detection, and propose a Levenshtein Ratio entity linker to improve the entity linking result and obtain candidate relations. Enlightened from the study of Yu et al. [13], in relation selection stage, we build a two-layer Bi-GRU model to measure the similarity between given question and candidate relations. Furthermore, multi-granularity text embeddings are proposed to enrich semantic information and attention mechanism is employed for a fine-grained alignment between characters. We select the relation with the highest confidence and obtain the predicted answer.

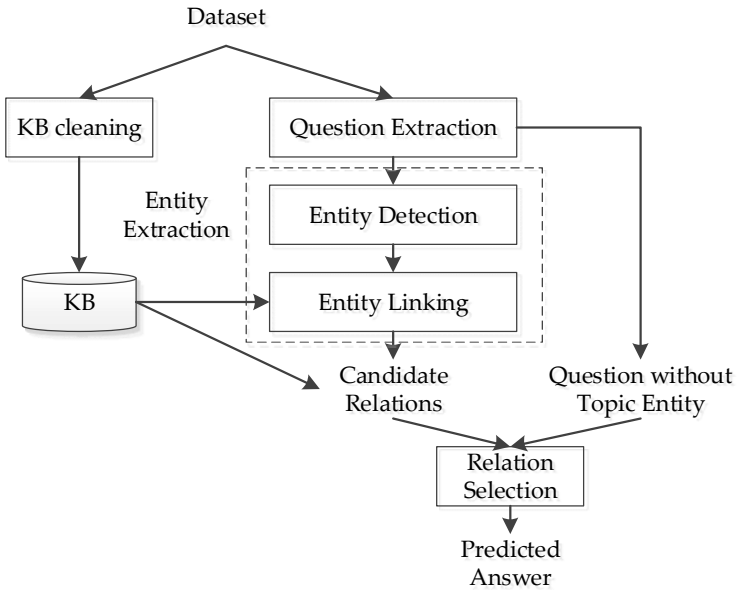

Figure 1. Overview of our knowledge base question answering (KBQA) system framework.

In this section, we first describe our entity extraction method for natural language question in Section 3.2. Then the framework and details of relation selection are illustrated in Section 3.3.

3.1. Task Definition

Given a question, topic entity extraction aims to find its mention and link it to knowledge base to get the topic entity and relation candidates $\boldsymbol{C} = \left\{ rel_1, rel_2, \ldots, rel_{|C|} \right\}$ in knowledge base. The purpose of relation selection is to identify the relation mentioned in a question, namely find the chain of relations that connects the topic entity and the answer in the knowledge base. Relation selection task is formulated as a pairwise ranking problem. For each relation r in the relation candidate set $\boldsymbol{C}$, the model computes its hidden representation semantic similarity with the representation of corresponding question q, and the relation with the highest score is selected to be the final predicate, formally:

$$r^+ = argmax\ S(\boldsymbol{q}; \boldsymbol{r}) \tag{1}$$

3.2. Topic Entity Extraction

Topic entity extraction of questions is a significant part in KBQA task. Given a single-relation factual question, our entity linker extracts the main entity mention which the question contains, and then links it to the knowledge base referring to the mention. It requires topic entity detection and entity

linking method, and the result can directly influence relation candidate retrieval. Some linguistic tools like name entity recognition tools are the key elements of traditional question answering models in topic entities extraction. However, these tools may not be applicable with Chinese because their quality varies when dealing with different language. And unlike English entity extraction task, sentences in Chinese need word segmentation and entity mention boundary is not clear as that of English. Besides, some data noise like entity mention with spelling mistakes are found in the dataset, which increases the difficulty of topic entity extraction. In this study, a topic entity extraction model, which contains entity detection model and entity linking model is proposed in order to extract topic entities in questions.

3.2.1. Entity Detection Model

Inspired by prior named entity recognition work, our entity detection model is implemented through sequential labeling to detect the mention of a question. In order to match the golden entity, we need to train an effective model to label the question text span for topic entity. For instance, Dai et al. [17] use the golden entity to label mention as training data and construct a Bi-GRU-CRF tagging model to do mention detection. Yin et al. [12] also introduce a Bi-LSTM-CRF tagging model to improve the performance of the approach. Similar to their work, we adopt Bi-LSTM-CRF model to conduct entity detection experiment.

LSTM is proposed in [24] and it is a variant of RNN. With memory cell and input gate, forget gate and output gate to manage the information flow, LSTM avoids gradient exploding and vanishing problem and is capable of capturing long range dependencies. By using these gates, LSTM can control both the extent that the input gives to the memory cell and the extent to forget from the previous state. However, one main disadvantage of unidirectional LSTM lies in that only the information before a particular word is considered while that after it is not taken into consideration. In order to avoid this disadvantage, bidirectional LSTM is applied like Bahdanau et al. [25] do. It is superior to unidirectional LSTM due to its ability to catch the information both before and after a word. Thus, this approach is applied in this study to solve the problem of entity extraction. In a typical process, a hidden representation $\overrightarrow{\boldsymbol{h}}_i$ of the left context is generated at every word while $\overleftarrow{\boldsymbol{h}}_i$ of the right context can be acquired by reading the same sequence reversely. Finally, the forward hidden representation $\overrightarrow{\boldsymbol{h}}_i$ and the backward representation $\overleftarrow{\boldsymbol{h}}_i$ are concatenated resulting in $\boldsymbol{h}_i = \begin{bmatrix} \overrightarrow{\boldsymbol{h}}_i \\ \overleftarrow{\boldsymbol{h}}_i \end{bmatrix}$.

In addition, tagging decisions are modeled with the aid of a conditional random field as suggested by Lafferty et al. [26]. Given an input text $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)$, the bidirectional LSTM network outputs the score matrix $P \in \mathbb{R}^{n \times k}$, where k denotes the number of output tags, $\boldsymbol{P}_{i,j}$ denotes the probability of the i-th word labeled as the j-th tag in $\boldsymbol{X}$. Given an output sequence $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)$, the score can be expressed as of

$$s(\boldsymbol{X}, \boldsymbol{y}) = \sum_{i=0}^{n} \boldsymbol{A}_{y_i, y_i+1} + \sum_{i=1}^{n} \boldsymbol{P}_{i, y_i} \tag{2}$$

where $\boldsymbol{A}$ is a transition score matrix of size $k+2$ considering the start and end tags of a sentence. The probability of the output sequence y can be obtained by applying softmax operation to all tag sequences:

$$p(\boldsymbol{y}|\boldsymbol{X}) = \frac{e^{s(\boldsymbol{X}, \boldsymbol{y})}}{\sum_{\hat{\boldsymbol{y}} \epsilon \boldsymbol{Y}_{\boldsymbol{X}}} e^{s(\boldsymbol{X}, \hat{\boldsymbol{y}})}} \tag{3}$$

where $\boldsymbol{Y}_{\boldsymbol{X}}$ denotes the candidate set of tag sequences for $\boldsymbol{X}$. In training procedure, the optimal tags can be reached by maximum the log-probability of the correct tag sequence:

$$log(p(\boldsymbol{y}|\boldsymbol{X})) = s(\boldsymbol{X}, \boldsymbol{y}) - log(\sum_{\hat{\boldsymbol{y}} \epsilon \boldsymbol{Y}_{\boldsymbol{X}}} e^{s(\boldsymbol{X}, \hat{\boldsymbol{y}})}) \tag{4}$$

Therefore, the prediction of the output tag sequence is given by:

$$\boldsymbol{y}^* = \underset{\hat{\boldsymbol{y}} \epsilon \boldsymbol{Y}_X}{argmaxs}(\boldsymbol{X}, \hat{\boldsymbol{y}}) \tag{5}$$

The architecture of our entity detection model is shown in Figure 2. The model is a Bi-LSTM neural network with a CRF layer. A sequence of Chinese characters is projected into a sequence of dense vectors, and concatenated with extra features as the inputs of a recurrent layer. Here, we employ one-hot vectors representing word boundary features for illustration. The recurrent layer is a bidirectional LSTM layer, outputs of forward and backward vectors are concatenated and projected to score of each tag. A CRF layer is used to overcome label-bias problem. Given the labeled data, parameters are trained to maximum Equation (4) of observation sequence from corpus.

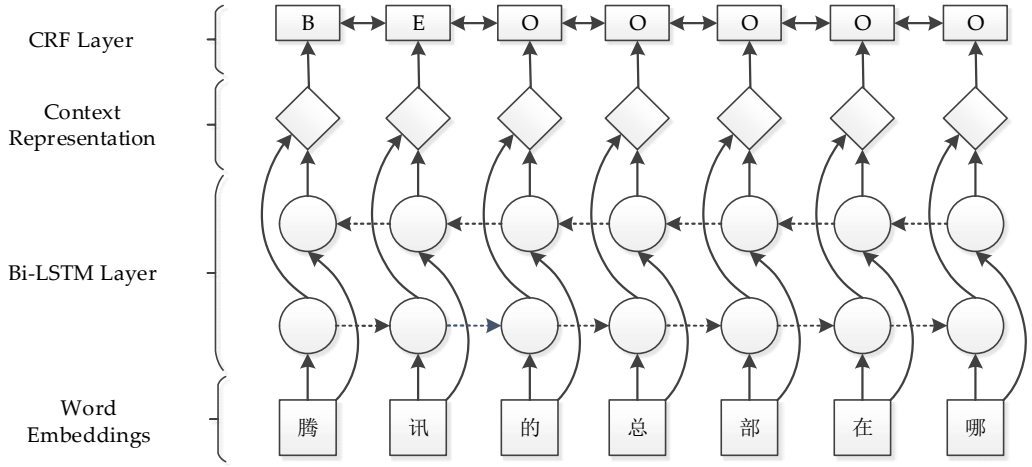

Figure 2. Main architecture of entity detection model. The input Chinese character sequence is "Where is the headquarters of Tencent".

3.2.2. Entity Linking Model

According to observation, there are three main types of obstacles that we encounter in entity linking: (1) wrong entity mention that the entity detection model labeled; (2) the entity mentions are abbreviation of some entity names; (3) wrongly written Chinese characters that appear in entity mentions. Thus, the main idea of entity linking model is carried out by tackling the problems above. We present a Levenshtein Ratio entity linker that utilizes Levenshtein Distance [27], which aims to improve the entity linking rate comparing to literally matching.

Entity names are short text. For short text strings, Levenshtein Ratio is a good measurement to compare similarity between them. The Levenshtein Ratio of two entity mentions m_i and m_j (of length $|m_i|$ and $|m_j|$ respectively) is defined as follows:

$$LevenshteinRatio(m_i, m_j) = 1 - \frac{LevenshteinDistance(m_i, m_j)}{|m_i| + |m_j|} \tag{6}$$

where Levenshtein Distance shows in Equation (6) is the minimum number of operation to transform m_i to m_j, including insertions, deletions or substitutions. Given the collection of all the entities $\boldsymbol{C}_e$ and the entity detection mention m, the following steps are performed to link entities to the knowledge base. First we lowercase all the English letters that appear in entity name collection and the detection entity mention. For every entity candidate e in $\boldsymbol{C}_e$, we compute its Levenshtein Ratio with mention m, then retrieve the entity who has the highest Levenshtein Ratio score. In this paper, top 1 entity is kept for the question. Specifically, in our experiment, even if the entity recognition result is not so accurate,

such as wrong boundary of the question text span is detected, this linking method can also link to the correct entity. For instance, to the question "«纸牌屋»都有什么演员啊？ || Who are the actors of *House of Cards*?" the entity mention we detected is "«纸牌屋» || *House of Cards*", which contains a book title mark in Chinese. However, the target entity name in the knowledge base is "纸牌屋 || House of Cards" without book title mark. The Levenshtein Ratio is calculated to be 0.75, which is the highest score, thus we consider the mention and the entity are linked. For abbreviations in entity names, such as "中科院 || CAS" which refers to "中国科学院 || Chinese Academy of Sciences", Levenshtein Ratio entity linker also has good performance. We also compare our entity linker with retrieval method, namely plain matching the mention strings to prove the effectiveness of Levenshtein Ratio entity linker. Details are illustrated in Section 4.5.1.

3.3. Relation Selection

Through entity linker, entity with the highest confidence is selected to generate predicate candidates. However, it is challenging to measure the similarity between the question and the relation because the expression of predicate in question text is always different from it. We present an attention-based multi-granularity interaction model, which represents the question dynamically according to different answer aspects. The architecture of our model is shown in Figure 3. A two-level hierarchical matching Bi-GRU encoder is adopted to represent question text, and a Bi-GRU encoder is used to get hidden representation of relation. In the representation, our model combines both character-level and word-level in order to get richer semantic information. We finally consider cosine as pairwise semantic relevance function to compute the semantic similarity between the representation of question and relation after the max-pooling operation.

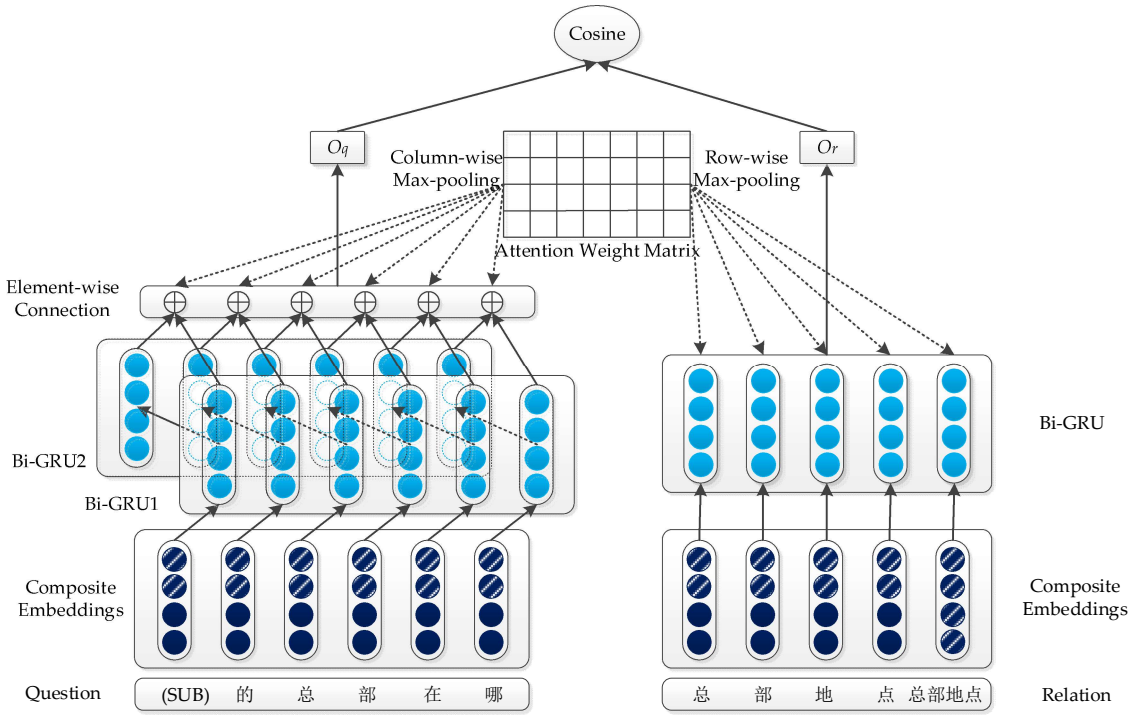

Figure 3. Main architecture of relation selection model. The input question character sequence is "Where is the headquarters of (SUB)", and the input relation sequence is "headquarter address" with its character form and whole token form.

3.3.1. Embedding Layer

Given a question text q or a relation text r, we consider how to map it to the vector representation by fully utilizing semantic information. Different from English, Chinese characters usually contain

specific meaning. Thus, we propose an approach exploiting both character-level and word-level information of given question. In the following, how to construct vector representation of the question $\boldsymbol{q}$ is illustrated in detail.

We employ Word2Vec [28] vectors for character embeddings and word embeddings. The pre-trained embeddings implicitly contains the inferred character or word semantics from a large text corpus. In other words, it means that words having similar meanings appear in similar contexts. In our case, terms with similar meanings are translated into similar vectors. In this paper, character embeddings are the base embeddings while word embeddings are the additional embeddings of character embeddings. Thus, the embeddings of the i-th character in the sentence c_i is constructed with two parts: initial character embeddings, and the corresponding embeddings of word that the character belongs to. The initial character embedding of c_i resulting in the d_c-dimensional vector representation $\overrightarrow{\boldsymbol{v}}_i^c$ can be formally described as follows. The characters c_i, $i = 1, 2, \ldots, n$ is embedded by

$$\overrightarrow{\boldsymbol{v}}_i^c = \boldsymbol{W}_c^T \boldsymbol{v}_i^c \tag{7}$$

where $\boldsymbol{W}_c^T \epsilon \mathbb{R}^{|V_c| \times d_c}$ is the character embedding matrix with a vocabulary size $|V_c|$, and $\boldsymbol{v}_i^c$ denotes the one-hot representation of character c_i. The added word-level embeddings $\overrightarrow{\boldsymbol{v}}_i^w$ are similarly embedded by

$$\overrightarrow{\boldsymbol{v}}_i^w = \boldsymbol{W}_w^T \boldsymbol{v}_i^w \tag{8}$$

with word embedding matrix $\boldsymbol{W}_w^T \epsilon \mathbb{R}^{|V_w| \times d_w}$, where $|\boldsymbol{V}_w|$ is the vocabulary size of words while d_w denotes the dimension of word vectors. $\boldsymbol{v}_i^w$ is the one-hot representation of corresponding word. Because of the limited coverage of word embedding, if the words that occur in the question are not included in pre-trained embedded vocabularies, we consider randomly initializing the word vectors. We use concatenation operation to join the embeddings in order to get the final representation:

$$\boldsymbol{v}_i^{com} = \begin{bmatrix} \overrightarrow{\boldsymbol{v}}_i^w \\ \overrightarrow{\boldsymbol{v}}_i^c \end{bmatrix} \tag{9}$$

In order to illustrate clearly, we name it composite embeddings. The method is similar to the char2word model proposed by Ling et al. [29] with the difference that word embeddings are added to enrich the semantic representation. Figure 4 is a schematic diagram of the overall representation network.

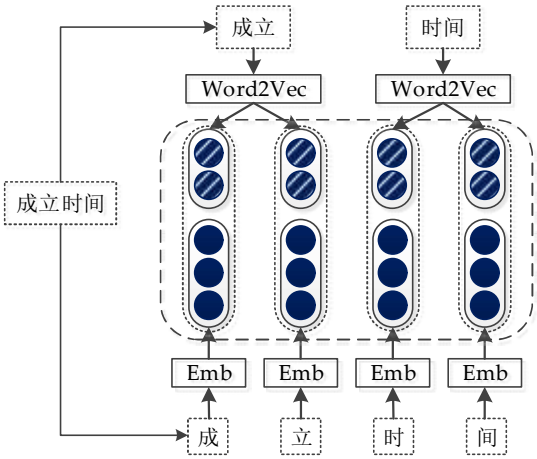

Figure 4. Composite embedding with example. The sample sequence is "Established time" with its character form and word form.

In our experiment, we also explore other combination of character-level and word-level representation, and the result shows that character-level representation combining the word-level representation outperforms others. We also compare composite embeddings with word embeddings and character embeddings respectively, which proves that effective combination of character-level and word-level significantly improves our system and helps us get the competitive results. Detailed results are illustrated in Section 4.5.2.

3.3.2. Relation Representation Layer

In relation aspects, we consider different granularities to represent the feature: composite embeddings and relation-level representation. Composite embeddings combine character-level and word-level information, which we have already introduced above. Relation-level representation treats each relation name as a unique token, such as "出生日期 | | date_of_birth". Character-level divides the relation into single Chinese characters. Word-level treats the relation as a sequence of words from the tokenized relation name. The three types of relation representation contain different levels of abstraction, all these levels of granularity have their own pros and cons. Relation-level focuses more on global information (long phrases and skip-grams) but suffers from data sparsity because some relations are absent from the training data and their relation representation is initialized randomly during inference. Word-level focuses more on local information (words and short phrases). However, these both levels suffer from OOV problem, character-level has no such issues, and usually achieves high accuracy in predicting the correct entity and relation. Thus, a multi-granularity approach for KB relation representation is utilized in our model, for a candidate relation, our approach matches the input relation to composite embeddings and relation embeddings to get the final representation.

Therefore, a relation $\boldsymbol{r}$ is finally represented as $\{\boldsymbol{r}_1^{com}, \boldsymbol{r}_2^{com}, \ldots, \boldsymbol{r}_{|r|}^{com}\} \cup \{\boldsymbol{r}^{rel}\}$, where $|r|$ is the number of relation characters. The first $|r|$ tokens are characters, and the last token is relation names, and we denote the total number of tokens in the representation as $|R|$. We transform each token of relation from one-hot representation to corresponding composite embedding vectors of d_r dimension. Note that we have the composite embedding vectors $\boldsymbol{V} \in \mathbb{R}^{|\boldsymbol{V}| \times d_r}$, and the relation embedding vectors $\boldsymbol{V}_{rel} \in \mathbb{R}^{|\boldsymbol{V}_{rel}| \times d_r}$, where $|\boldsymbol{V}_{rel}|$ are the vocabulary size and the number of relations in the knowledge base respectively.

Since we get relation embeddings, a Bi-GRU layer is used to represent its context. GRU is proposed by Cho et al. [30]. As a variant of LSTM [31], it can function in the same way as LSTM, modulating the information flow within the unit via gating units and enabling adaptive capture dependencies of different time scales. The GRU unit does not have to use a memory unit to control the flow of information like the LSTM unit. It can directly make use of the all hidden states without any control. GRUs have fewer parameters and thus can train a bit faster and need less data to generalize. The structure of the GRU cell is illustrated in Figure 5.

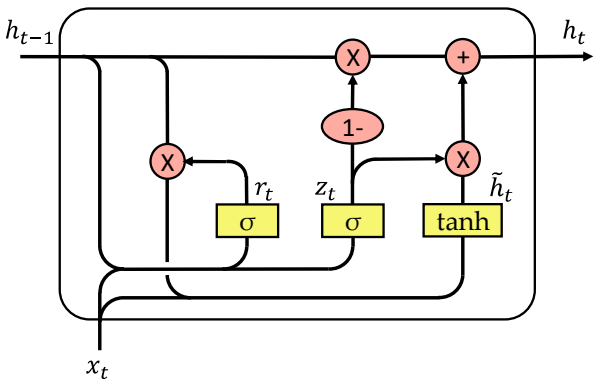

Figure 5. The structure of Gated Recurrent Unit (GRU) cell.

The forward GRU cell outputs the encoding result based on the input $\boldsymbol{x}_t$ and the output of last time $\overrightarrow{\boldsymbol{h}}_{t-1}$. Here we denote the representation procedure in the cell as $\overrightarrow{\boldsymbol{h}}_t = gru(\boldsymbol{x}_t, \overrightarrow{\boldsymbol{h}}_{t-1})$. GRU integrates the gates of LSTM such as forget gate $\boldsymbol{f}_t$, input gate $\boldsymbol{i}_t$ and output gate $\boldsymbol{o}_t$ into update gate $\boldsymbol{z}_t$ and reset gate $\boldsymbol{r}_t$. The update gate $\boldsymbol{z}_t$ determines the amount of content the unit renews, or the extent to which the activation is updated. It is calculated by

$$\boldsymbol{z}_t = \sigma(\boldsymbol{W}_z \boldsymbol{x}_t + \boldsymbol{U}_z \overrightarrow{\boldsymbol{h}}_{t-1} + \boldsymbol{b}_z) \tag{10}$$

where $\boldsymbol{W}_z \in \mathbb{R}^{d_u \times d}$, $\boldsymbol{U}_z \in \mathbb{R}^{d_u \times d_u}$ and $\boldsymbol{b}_z \in \mathbb{R}^{d_u}$ are parameters to be learned. Hyper-parameter d_u is the dimension of GRU unit. Like LSTM, it calculates a linear sum between existing state and new state. However, the difference with LSTM is that GRU lacks systematic control over the extent of state exposure. The reset gate $\boldsymbol{r}_t$ determines how the previous information combines with the current input. When it is off, namely $\boldsymbol{r}_t$ is close to 0, the reset gate effectively frees the previously computing state, functioning as if it is reading from the beginning of the sequence. $\boldsymbol{r}_t$ is calculated as:

$$\boldsymbol{r}_t = \sigma(\boldsymbol{W}_r x_t + \boldsymbol{U}_r \overrightarrow{\boldsymbol{h}}_{t-1} + \boldsymbol{b}_r) \tag{11}$$

Similar to [25], the candidate activation $\widetilde{\boldsymbol{h}}_t$ is calculated:

$$\widetilde{\boldsymbol{h}}_t = \tanh(\boldsymbol{W}\boldsymbol{x}_t + \boldsymbol{U}(\boldsymbol{r}_t \odot \overrightarrow{\boldsymbol{h}}_{t-1})) \tag{12}$$

where $\odot$ is an element-wise multiplication. Finally, the activation $\overrightarrow{\boldsymbol{h}}_t$ is decided by the previous activation $\overrightarrow{\boldsymbol{h}}_{t-1}$ and the candidate activation $\widetilde{\boldsymbol{h}}_t$:

$$\overrightarrow{\boldsymbol{h}}_t = (1 - \boldsymbol{z}_t)\overrightarrow{\boldsymbol{h}}_{t-1} + \boldsymbol{z}_t \widetilde{\boldsymbol{h}}_t \tag{13}$$

Similarly, the backward GRU is also represented as

$$\overleftarrow{\boldsymbol{h}}_t = gru(\boldsymbol{x}_t, \overleftarrow{\boldsymbol{h}}_{t+1}) \tag{14}$$

For input vector sequence $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N)$ with length N, forward GRU encodes the input $\boldsymbol{x}_t$ with context from $\boldsymbol{x}_1$ to $\boldsymbol{x}_{t-1}$ into vector $\overrightarrow{\boldsymbol{h}}_t$, while backward GRU encodes $\boldsymbol{x}_t$ to $\overleftarrow{\boldsymbol{h}}_t$ considering the future contextual information from $\boldsymbol{x}_N$ to $\boldsymbol{x}_{t+1}$. Concatenating $\overrightarrow{\boldsymbol{h}}_t$ and $\overleftarrow{\boldsymbol{h}}_t$, the Bi-GRU encodes the input $\boldsymbol{x}_t$ with both the past and future information from the sentence in consideration. Then Bi-GRU layer can be denoted by

$$\boldsymbol{H} = Bi - GRU(\boldsymbol{X}) = \left[\begin{bmatrix} \overrightarrow{\boldsymbol{h}}_1 \\ \overleftarrow{\boldsymbol{h}}_1 \end{bmatrix}, \begin{bmatrix} \overrightarrow{\boldsymbol{h}}_2 \\ \overleftarrow{\boldsymbol{h}}_2 \end{bmatrix}, \ldots, \begin{bmatrix} \overrightarrow{\boldsymbol{h}}_N \\ \overleftarrow{\boldsymbol{h}}_N \end{bmatrix} \right] \tag{15}$$

In this paper, the context aware representation of relation can be formally defined as follows:

$$R = Bi - GRU\left(\left[\boldsymbol{r}_1^{com}, \boldsymbol{r}_2^{com}, \ldots, \boldsymbol{r}_{|r|}^{com}, \boldsymbol{r}^{rel} \right] \right) = \left\{ \boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_{|R|} \right\} \tag{16}$$

where $\boldsymbol{R} \in \mathbb{R}^{d_r \times |R|}$, d_r is the dimension of GRU unit for the relation representation.

3.3.3. Question Representation Layer

After entity extraction, we then replace the mention in the question with <SUB> sign. Then the target is to identify the relation that most closely matches the description of the question. Usually, different parts of a relation correspond to different sections of a question. The whole relation names

often match longer phrase while relation words correspond to shorter ones. Therefore, in order to enrich the semantics and catch different granularity information, a two-layer deep Bi-GRUs is utilized on questions to address such issue. The first layer of Bi-GRU deals with the composite embeddings of question $\boldsymbol{q} = \{\boldsymbol{q}_1^{com}, \boldsymbol{q}_2^{com}, \ldots, \boldsymbol{q}_{|Q|}^{com}\}$ where $|Q|$ denotes the total number of characters in given question, and hidden representations are obtained as below:

$$\boldsymbol{Q}^{(1)} = Bi - GRU\left(\left[\boldsymbol{q}_1^{com}, \boldsymbol{q}_2^{com}, \ldots, \boldsymbol{q}_{|Q|}^{com}\right]\right) = \left\{\boldsymbol{q}_1^{(1)}, \boldsymbol{q}_2^{(1)}, \ldots, \boldsymbol{q}_{|Q|}^{(1)}\right\} \tag{17}$$

The second GRU layer subsequently functions on the hidden representations $\boldsymbol{Q}^{(1)}$ and obtains $\boldsymbol{Q}^{(2)}$:

$$\boldsymbol{Q}^{(2)} = Bi - GRU\left(\left[\boldsymbol{q}_1^{(1)}, \boldsymbol{q}_2^{(1)}, \ldots, \boldsymbol{q}_{|Q|}^{(1)}\right]\right) = \left\{\boldsymbol{q}_1^{(2)}, \boldsymbol{q}_2^{(2)}, \ldots, \boldsymbol{q}_{|Q|}^{(2)}\right\} \tag{18}$$

More abstract information is to be learned in the second-layer GRU as it is based on the first layer. A typical way to fulfill hierarchical matching is to calculate similarity between each layer of $\boldsymbol{Q}$ and $\boldsymbol{R}$ individually and the weighted sum between the two scores. However, this approach will make the training much harder and usually leads to a much higher of converged training loss than a single-layer baseline model. A major reason is that deep Bi-GRUs cannot guarantee that the training for both layers achieve the best simultaneously. In addition, deeper architectures require more difficult training. To address such issues, hierarchical matching by adding element-wise connections between two Bi-GRU layers [13] is employed in our model. Each $\boldsymbol{Q}^{(1)}$ and $\boldsymbol{Q}^{(2)}$ are connected to obtain a $\boldsymbol{q}_i = \boldsymbol{q}_i^{(1)} + \boldsymbol{q}_i^{(2)}$ for each position i, resulting in the hidden representation of the question $\boldsymbol{Q}$.

3.3.4. Attention Layer

Since we get the extracted features, the representation results are merged to vectors with attention mechanism. Similar to the study of Cui et al. [32] and Zhang et al. [33], attention weights for questions are calculated by column-wise max-pooling

$$\hat{a}_i = max(a_{i,1}, a_{i,2}, \ldots, a_{i,|R|}) \tag{19}$$

where $a_{i,j}(1 \leq i \leq |Q|, 1 \leq j \leq |R|)$ is the element of attention weight matrix. And apply softmax operation we get

$$\alpha_i = \frac{e^{\hat{a}_i}}{\sum_{m=1}^{|Q|} e^{\hat{a}_m}} \tag{20}$$

Then the vector representation of the question is

$$\boldsymbol{o}_q = \sum_{i=1}^{|Q|} \alpha_i \boldsymbol{q}_i \tag{21}$$

where q_i is the i-th column of final question representation $\boldsymbol{Q}$. In the same way, attention weights of the relation are calculated by

$$\hat{b}_j = max(a_{1,j}, a_{2,j}, \ldots, a_{|Q|,j}) \tag{22}$$

And

$$\beta_j = \frac{e^{\hat{b}_j}}{\sum_{n=1}^{|R|} e^{\hat{b}_n}} \tag{23}$$

Then relation's vector representation is

$$\boldsymbol{o}_r = \sum_{j=1}^{|R|} \beta_j \boldsymbol{r}_j \tag{24}$$

3.3.5. Output Layer

The output layer computes the semantic similarity between the question and the relation as follows:

$$S(\boldsymbol{q};\boldsymbol{r}) = \cos(\boldsymbol{o}_q, \boldsymbol{o}_r) \tag{25}$$

where cos is the cosine similarity which is defined as $\cos(\boldsymbol{a},\boldsymbol{b}) = \frac{\boldsymbol{a}\cdot\boldsymbol{b}}{|\boldsymbol{a}||\boldsymbol{b}|}$.

4. Experiments

4.1. Datasets

Evaluation of our approach is carried out on NLPCC-ICCPOL 2016 KBQA dataset, which is the largest public Chinese KBQA dataset at present. The dataset contains approximately 43 million subject-predicate-object (SPO) triples in the knowledge base, where there are about 6 million entities. The triples of the knowledge base are mostly collected from Baidu Encyclopedia, and extracted from item in fobox. In the dataset, there are 14,609 training question-answer pairs and 9870 testing pairs. The questions are provided by Microsoft researchers and the corresponding answers are labeled manually, and both questions and answers are with some noises, especially in relations. Thus, before we conduct experiments, pre-processing on the knowledge base is necessary. The details of KB cleaning are explained in Table 1.

Unlike some English KBQA dataset such as Simple Questions [8], in the training set of NLPCC-ICCPOL 2016 KBQA dataset, the corresponding knowledge triple of each question is not provided. In order to conduct entity linking and relation selection experiment, golden knowledge triple needs to match with each question, so question-entity pairs and question-relation pairs need to be generated. In this paper, we use an iterative way to obtain subtask training sets from original training data. First, the answer is used to retrieve objects of SPO triples from the knowledge base. Refer to the subjects of candidate triples, we then map the most relevant subject back to the question text to label the entity mention of the given question. Since subjects of knowledge triples may differ from entity mention, the training data initially obtained cannot extract all the entity mentions of questions. To get high-quality training data as much as possible, we use the data initially obtained to train the entity recognition model and apply it to the rest of the questions of the original training set and search the knowledge triples again. Finally, we get 14,165 questions with golden triples.

Table 1. Knowledge base cleaning rules.

Type	Times	Instance	Disposal
Space in predicate between Chinese characters	367,218	别 名/Alias	Delete space
Predicate prefixes "-" or "."	163,285	- 行政村数/- Number of administrative villages	Delete prefixes
Appendix labels in predicate	9110	人口 (2009) [1]/Population (2009) [1]	Delete appendix labels
Predicate is the same as object	193,716	陈祝龄旧居|||天津市文物保护单位|||天津市文物保护单位/Former Residence of Chen Zhuling|||Tianjin heritage conservation unit|||Tianjin heritage conservation unit	Delete the record

4.2. Training and Inference

A pairwise training is performed with the generated training data. The training loss is given as follows:

$$L_{\boldsymbol{q},\boldsymbol{r}^+,\boldsymbol{r}^-} = \sigma(S(\boldsymbol{q};\boldsymbol{r}^-|\theta) - S(\boldsymbol{q};\boldsymbol{r}^+|\theta)) \tag{26}$$

where θ denotes parameters of the network. Then θ consists of composite embeddings, relation embeddings, parameters in the GRU network for relation and question representation. The intuition of

this training procedure is to ensure that positive question-answer pairs are rated higher than negative ones with a margin. The object function is as follows:

$$min \sum_{\boldsymbol{q}_i \epsilon \mathbb{Q}} \sum_{\boldsymbol{r}^- \epsilon \mathbb{N}_q} L_{\boldsymbol{q},\boldsymbol{r}^+,\boldsymbol{r}^-} \tag{27}$$

where $\mathbb{Q}$ denotes the questions in the training set, $\mathbb{N}_q$ is the false candidate relation set. The back propagation method is adopted to update the parameters. Formally, the parameters in θ are updated by

$$\theta = \theta - \lambda \frac{\partial L}{\partial \theta} \tag{28}$$

where λ is the learning rate. Adadelta optimizer [34] is adopted to adjust the learning rate. Dropout is applied to the output of embeddings, GRU layer in order to avoid over-fitting problems.

In the testing stage, the semantic similarity $S(\boldsymbol{q};\boldsymbol{r}|\theta)$ is calculated for each candidate relation, and the relation with highest semantic similarity score is regarded as the corresponding relation.

4.3. Evaluation Metrics

The evaluation of a KBQA system is generally considered by precision, recall, averaged F1 and accuracy@N. For entity detection task, the precision, recall and F1 are utilized to judge the performance of the model. Precision is defined as follows:

$$P = \frac{1}{|\mathbb{Q}|} \sum_{i=1}^{|\mathbb{Q}|} P_i \tag{29}$$

P_i denotes the precision for question Q_i computed based on the generated answer set and the golden answers A_i. P_i equals to 0 when C_i for Q_i is empty or does not overlap with A_i for Q_i. In rest circumstances, P_i is computed as follows:

$$P_i = \frac{\#(C_i, A_i)}{|C_i|} \tag{30}$$

where $\#(C_i, A_i)$ denotes the answers number that both C_i and A_i contain, while $|C_i|$ and $|A_i|$ denote the answers number occur in C_i and A_i respectively. Similarly, recall is defined as follows:

$$R = \frac{1}{|\mathbb{Q}|} \sum_{i=1}^{|\mathbb{Q}|} R_i \tag{31}$$

where R_i is the recall for question Q_i calculated based on C_i and A_i. It equals to 0 when C_i for Q_i is empty or does not overlap with the golden answers A_i for Q_i. Similarly in other cases, recall for question Q_i is computed as follows:

$$R_i = \frac{\#(C_i, A_i)}{|A_i|} \tag{32}$$

Averaged F1 is defined as follows:

$$\text{Averaged } F1 = \frac{2 \cdot P \cdot R}{P + R} \tag{33}$$

The result of entity linking or relation selection is selection of the candidate of highest confidence, which is the top 1 answer of a ranking model, so we have accuracy

$$Acc = P = R = F1 \tag{34}$$

We also import accuracy@N to evaluate a ranking model. It is defined as follows:

$$\text{Accuracy@}N = \frac{1}{|\mathbb{Q}|}\sum_{i=1}^{|\mathbb{Q}|}\delta(C_i^N, A_i) \tag{35}$$

where C_i^N is the answer set which generated top N answers, and $\delta(C_i^N, A_i)$ is set to 1 if C_i^N contains at least one answer appears in A_i, otherwise $\delta(C_i^N, A_i)$ equals to 0.

4.4. Experiment Setup

All the experiments are carried out on a machine with Intel Core i7-6700 CPU @3.4 GHz and NVIDIA GTX1080 GPU, and neural networks are implemented in Keras with Tensorflow as the backend.

4.4.1. Topic Entity Extraction Model

For training of this entity detection model, we use back-propagation algorithm to update the parameters on training examples. Embedding vectors are trained with gensim version of Word2Vec on Chinese WiKi corpus. Different from the results reported by Lample et al. [35] in English, 50 dims achieve 95.21% F1 and are not enough to represent Chinese characters. The result of 100 dims achieves 2.15% better than 50 dims, but no more improvement is observed when we use 200 dims achieving 96.95%. Thus, we use 100 dims in the following experiments. The dropout rate 0.5 is selected according to the study of Dong et al. [36]. When dealing with the dimension of LSTM, we refer to the study of Greff [37] and Reimers [38], selecting {100, 200, 300} as the searching space. Result of 100 dims is 97.36% F1, compared to 97.27% F1 and 97.01% F1 when the dimension is 200, 300, respectively. Detailed hyper parameters are illustrated in Table 2 below.

There are 14,165 questions for training and 9870 questions for testing. Our training batch size is 20 and we train our model for 50 epochs. The training time is 843 s. Our testing batch size is 100 and testing time is 4.39 s.

Table 2. Hyper parameters for entity detection experiment.

Hyper Parameter	Batch Size	Gradient Clip	Embedding Size	Dropout Rate	Learning Rate
Value	20	5	100	0.5	0.001

4.4.2. Relation Selection Model

The hyper-parameters of our model are summarized in Table 3. In the experiment, composite embeddings are initialized with the Word2Vec with $d = 200$, with per-trained character and word vector size 100. Embeddings of relations and words that are out of vocabulary are randomly initialized by sampling values uniformly from $(-0.25, 0.25)$. The values of embeddings are updated during the training process. The dimension of GRU hidden units is similar to that of LSTM. According to Reimers' work [38], we try 50, 100, 150, 200, and get the lowest F1 score 79.80% when it is 50 and the highest 81.74%. Dropout rate is also a significant hyper-parameter. The best result is achieved when we use 0.35. The difference to not using dropout can be as high as ΔF1 $= -1.71\%$. for relation selection task.

Table 3. Hyper parameters for relation selection experiment.

Parameter	Search Space	Value
Embedding dim. d	{200}	200
Dim of GRU d_q, d_r	{50, 100, 150, 200}	150
Dropout rate	{0.2, 0.3, 0.35, 0.4, 0.5}	0.35
Batch size	{64, 128, 256, 512}	256

There are 188,165 question-predicate pairs for training and 118,092 question-predicate pairs for testing. Our training batch size is 256. We observe that after about only 20 epochs our model already reaches decent performance on the validation set. Afterwards, the accuracy continues to increase slowly, starting to stagnate around 50 epochs. The training time is 557 s. Our testing batch size is 1024 and testing time is 3.4 s.

4.5. Result

4.5.1. Topic Entity Extraction

The entity extraction performance is determined by entity detection result and entity linking result. In entity linking experiment, the raw accuracy is measured by information retrieval method, namely match the mention string to the knowledge base literally. Results of accuracy@N are given by Levenshtein Ratio entity linker. We select accuracy@1 of the Levenshtein Ratio entity linker as the standard performance of our entity extraction model.

Experimental results of entity extraction are listed in Table 4. In the testing dataset, F1 of entity detection is 97.36%, which proves the effectiveness of Bi-LSTM with CRF layer in Named Entity Recognition task. The information retrieval method only reaches 96.56% accuracy, while Levenshtein Ratio entity linker outperforms retrieval method by 2.16% when we select top 1 entity as the linking result. When there are top 3 candidates, accuracy reaches 99.41%. The overall entity extraction precision is 96.16%, which generates positive data for relation selection task.

Table 4. Performance of entity detection, linking and overall extraction results. The accuracy@1 result of Levenshtein Ratio entity linker is selected as the final result of entity linking stage.

Entity Detection			**Entity Linking**				**Overall Entity Extraction**		
P_{ED}	R_{ED}	$F1_{ED}$	**Raw** A_{EL}	A_{EL}**@1**	A_{EL}**@2**	A_{EL}**@3**	P_{EE}	R_{EE}	$F1_{EE}$
97.41	97.32	97.36	96.56	98.72	99.05	99.41	96.16	96.07	96.11

4.5.2. Relation Selection

An ablation experiment is performed to illustrate the effectiveness of our model. The advantages of our approach is proved by comparing it with other methods.

Table 5 shows the ablation experiment results of our proposed method. We can see that the task benefits from the two-layer Bi-GRU encoder hierarchical matching on question representation. The composite embeddings and attention mechanism also contribute a lot. Three group ablation experiments are conducted. Experiments about hierarchical matching are as follows.

Table 5. Ablation experiment results.

Analysis Content	**Model**	**Acc.**
Hierarchical matching framework	replace hierarchical matching framework with single-layer Bi-GRU question encoder	80.39
	replace hierarchical matching framework with two-layer Bi-GRUs without element-wise connections	79.26
	replace hierarchical matching framework with two single-layer Bi-GRUs with element-wise connections	76.54
Structure unit	model without attention	79.92
	replace Bi-GRU with Bi-LSTM	81.51
	replace Bi-GRU with CNN	79.03
Text embeddings	replace composite embeddings with word embeddings	78.36
	replace composite embeddings with character embeddings	79.58
	ABMGIM (Our approach)	81.74

- Single-layer Bi-GRU question encoder: we also use composite embeddings. One single-layer Bi-GRU is adopted to perform the question context aware representation instead of our two-layer hierarchical matching framework, and the representation results of question and relation are merged to vectors with attention mechanism.
- Two-layer Bi-GRUs without element-wise connections: composite embeddings are adopted. we still use two-layer deep Bi-GRUs network to get the hidden representation of questions but without element-wise connections. Attention mechanism is applied on the second layer Bi-GRU hidden representation.
- Two single-layer Bi-GRUs with element-wise connections: we replace the deep Bi-GRU question encoder with two single-layer Bi-GRUs, with element-wise connections between their hidden states. Other architectures of the network like composite embeddings and attention mechanism remain the same.

The first part of Table 5 gives the experiment results about hierarchical matching framework. First, our proposed model outperforms the comparing model with single-layer Bi-GRU question encoder by 1.35%, which proves that two-levels of question hidden representations with element-wise connections structure has better performance in relation selection task. Furthermore, our model benefits from hierarchical matching in comparison with deep Bi-GRU without element-wise connections, because the accuracy drops to 79.26% when there are not element-wise connections between question hidden representations. Note that the accuracy of two-layer Bi-GRU is lower than the 80.39% achieved by a single-layer one. Finally, two single-layer Bi-GRUs with element-wise connections converges to 76.54%, which results in a large performance drop. It shows that hierarchical matching promotes the learning of different levels of abstraction by hierarchical architecture, and is rather than a simple combination of two Bi-GRUs with element-wise connections. This group ablation experiment proves that the good execution of hierarchical matching is ascribed to both the element-wise matching and deep structures.

Ablation experiments are also carried out to study the effectiveness of some structure unit we apply in our proposed model. LSTM and CNN network are used to replace GRU unit, considering different structure units may have different performance in relation selection task. We also compare the model without attention mechanism. Specific ablation experiments are introduced as follows.

- Model without attention: relations are represented with Bi-GRU layers and questions are represented with two-layer hierarchical Bi-GRUs. The semantic similarity is measured by the cosine similarity between final hidden representations: $S(\boldsymbol{q};\boldsymbol{r}) = \cos(\boldsymbol{q},\boldsymbol{r})$.
- Replace Bi-GRU with Bi-LSTM: simply replace the Bi-GRU layers of question and relation with Bi-LSTM, other structures remain the same.
- Replace Bi-GRU with CNN: unlike GRU that depends on the computations of the previous time step, CNN enables parallelization over every element in a sequence, so it is capable of making full use of the parallel architecture of GPU. We study the performance of fully CNN network on the relation selection of KBQA. The GRU layer for question and relation preprocessing is replaced with a multi-kernel CNN layer, and the dimension of the CNN output is consistent with that of the original GRU layer.

Experimental results with different structure units are given in the second part of Table 5. The first result shows that attention mechanism plays an important role in the whole model. It enables the network to focus on important parts of the sequence and get a better representation. From the second result, we can see that LSTM has similar performance compared with GRU (81.51% vs. 81.74%). However, The GRU layer has quick convergence and fewer parameters in the experiment. Furthermore, the GRU layer, which is capable of learning long range dependency, outperforms the CNN by 2.71%. However, CNN does not rely on the computations of previous time step, so they can fully utilize the computational capability of GPU and are faster to be trained and perform inference.

We also explore the influence of text embeddings in our experiments. Results are given in the last part of Table 5. It is showed that using only word or character embeddings causes a performance

drop on datasets compared to using composite embeddings, which proves combination of word and character embeddings can improve the semantic representation of basic embeddings unit. Note that character embeddings outperform the word embeddings, which is mainly because that the Chinese characters do carry important semantic information when compared to English characters, and the error of Chinese word segmentation may also influence the precision of word embeddings.

We also compare our proposed model with several strong baselines that are representative of Chinese KBQA.

- SPE & Pattern Rule [23]: subject predicate extraction algorithm with several pattern rules. A linear combination of pattern rules including answer patterns, core of questions, question classification method and posttreatment rules for alternative questions is employed to pick up golden answers.
- NBSVM & CNN [18]: NBSVM-based ranking model and CNN-based ranking. N-gram co-occurrence features are extracted to train an SVM model with Naive Bayes features, and CNN-based ranking firstly maps the question and relation as vectors by CNN and then merges two output vectors to get a score. Stacking method is used to ensemble two model to get the final result.
- DSSM Combination [19]: a combination of CNN-based deep structured semantic models and some variant, including Bi-LSTM-DSSM, Bi-LSTM-CNN-DSSM. Bi-LSTM-DSSM extends DSSM by applying bi-directional LSTM, while Bi-LSTM-CNN-DSSM is developed by integrating CNN with Bi-LSTM layer. Finally, cosine similarity is used to measure the matching degree between question and candidate predicates. The three models own different weights in order to give a composite lexical matching score.

The comparison results are listed in Table 6. Our approach obtains a similar result which is as good as the state-of-the-art model SPE & Pattern Rule (81.74% vs. 82.47%). Note that the state-of-the-art model introduces a lot of patterns or artificial features that we mention above. Therefore, our model can have more robustness and generalization ability comparing to it. Our model outperforms the rest of the Chinese KBQA model reported on the datasets. It is worth mentioning that our method applies a single model in relation selection task and achieves better results while Yang et al. and Xie et al. combine multiple models to improve performance.

Table 6. Comparison of accuracy with other baselines.

Model	Acc.
SPE & Pattern Rule (Lai et al., 2016)	82.47
NBSVM & CNN (Yang et al., 2016)	81.59
DSSM Combination (Xie et al., 2016)	79.57
ABMGIM (Our approach)	81.74

4.6. Error Analysis and Discussion

In order to gain some insight into the deficiency of our approach, thorough error analysis on our model is necessary. Since our experiment is conducted on the current largest Chinese KBQA dataset, an error analysis of the dataset is also performed, which can benefit those who utilize the same dataset.

We randomly choose 100 questions and inspect them from the testing data. Statistical results are shown in Table 7. Types of errors include missing entities, wrong entities, wrong predicates, ambiguity and some dataset caused errors. We can see that nearly half of the errors are caused by the dataset, which in fact do not belong to real mistakes. We will discuss these errors later in detail. Among the rest errors resulting in wrong predicate are the most frequent type, which means topic entity is linked correctly, but corresponding relation is wrongly chosen. This is mainly limited by the ABMGIM. Missing entities and wrong entities are due to the bad performance of our entity extraction model. It leads to the situation when the model cannot identify the mention in a question or link to the wrong entity in the knowledge base, which restricts the performance of ABMGIM. Ambiguity means that the entity of a question has insufficient information to conduct entity disambiguation. One such example

is that "刘勇是从哪个学校毕业的？ || Which school did Liu Yong graduate from?", while a lot of people whose name called Liu Yong are in the given knowledge base and there is no other clue to identify which one the question refers to. We leave all these situation to our future work.

Table 7. Counts of errors that our model makes on sampled data.

Cause of Error	Counts
Missing entities	2
Wrong entities	5
Wrong predicates	34
Ambiguity	16
Dataset caused errors	43
Total	100

For errors caused by the dataset, we manually inspect the matching results and findings are presented to show its properties. The main format problems are about 33%. For example, for the question " 太子山国家森林公园的绿化率是多大？ || What is the green coverage rate of *Taizi Mountain National Forest Park*?" the corresponding labeled answer is "80.40%", while answer in knowledge base is "80.4%". Typos in entities also contribute about 23% in dataset caused errors, fortunately our entity linker can handle part of this situation and gain some improvement in accuracy. Other situation mainly contains 7% wrong labeled answer, 16% aliases of entities, and 5% incomprehensible questions. While 16% of them still remain unclassified. If we take the testing samples which are wrongly judged, the accuracy of our proposed model on testing questions would rise, and the whole performance will also be improved.

5. Conclusions

In this article, we present an effective way to handle Chinese KBQA task, leveraging an attention-based multi-granularity interaction model. Two main contributions are made. In topic entity extraction stage, a Bi-LSTM-CRF model is trained to do the entity detection. Levenshtein Ratio entity linker is proposed to conduct effective entity linking. In relation selection, we combine character-level and word-level information for text embeddings to enrich semantic representation, and relation-level representation is also utilized to catch global information in relation representation. We further apply the hierarchical matching network for question representation. Attention mechanism is utilized for a fine-grained alignment between question and relation. Finally, we measure the questions and relations by cosine similarity. The experimental results demonstrate that our model achieves competitive performance and generally outperforms most of other Chinese KBQA model.

For future work, the investigation of the end-to-end neutral network approach is considered. Because the results of the subtasks may be the bottleneck of the whole pipeline method, end-to-end system makes decisions all by the model, which effectively avoids the error propagation. We will also explore the transfer learning between traditional relation extraction task and relation selection of KBQA in order to further improve the performance of our system.

Author Contributions: C.S. conceived and designed the algorithms and performed the experiments and analyzed the results; C.S. and X.L. wrote and revised the manuscript; T.H. and F.L. discussed the data and corrected the manuscript; T.H. and K.F. provided relevant information and instructions during the design of experiments. All authors have read and approved the final manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Bollacker, K.; Evans, C.; Paritosh, P.; Sturge, T.; Taylor, J. Freebase: A collaboratively created graph database for structuring human knowledge. In Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, Vancouver, BC, Canada, 9–12 June 2008; pp. 1247–1250.

2. Auer, S.; Bizer, C.; Kobilarov, G.; Lehmann, J.; Cyganiak, R.; Ives, Z. Dbpedia: A nucleus for a web of open data. In Proceedings of the 6th International Semantic Web and 2nd Asian Conference on Asian Semantic Web Conference, Busan, Korea, 11–15 November 2007; pp. 722–735.
3. Niu, X.; Sun, X.; Wang, H.; Rong, S.; Qi, G.; Yu, Y. Zhishi.Me: Weaving Chinese linking open data. In Proceedings of the 10th International Conference on The Semantic Web—Volume Part II, Bonn, Germany, 23–27 October 2011; pp. 205–220.
4. Wang, Z.-C.; Wang, Z.-G.; Li, J.-Z.; Pan, J.Z. Knowledge extraction from Chinese wiki encyclopedias. *J. Zhejiang Univ. SCI. C* **2012**, *13*, 268–280. [CrossRef]
5. Berant, J.; Chou, A.; Frostig, R.; Liang, P. Semantic parsing on freebase from question-answer pairs. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Seattle, WA, USA, 18–21 October 2013; pp. 1533–1544.
6. Dong, L.; Wei, F.; Zhou, M.; Xu, K. 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, Beijing, China, 26–31 July 2015; pp. 260–269.
7. Yu, L.; Hermann, K.M.; Blunsom, P.; Pulman, S. Deep learning for answer sentence selection. *arXiv* **2014**, arXiv:1412.1632.
8. Bordes, A.; Usunier, N.; Chopra, S.; Weston, J. Large-scale simple question answering with memory networks. *arXiv* **2015**, arXiv:1506.02075.
9. Golub, D.; He, X. Character-level question answering with attention. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, TX, USA, 1–5 November 2016; pp. 1598–1607.
10. Zhang, Y.; Liu, K.; He, S.; Ji, G.; Liu, Z.; Wu, H.; Zhao, J. Question answering over knowledge base with neural attention combining global knowledge information. *arXiv* **2016**, arXiv:1606.00979.
11. Yih, W.-T.; Chang, M.-W.; He, X.; Gao, J. Semantic parsing via staged query graph generation: Question answering with knowledge base. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, Beijing, China, 26–31 July 2015; pp. 1321–1331.
12. Yin, W.; Yu, M.; Xiang, B.; Zhou, B.; Schütze, H. Simple question answering by attentive convolutional neural network. *arXiv* **2016**, arXiv:1606.03391.
13. Yu, M.; Yin, W.; Hasan, K.S.; dos Santos, C.; Xiang, B.; Zhou, B. Improved neural relation detection for knowledge base question answering. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, BC, Canada, 30 July–4 August 2017; pp. 571–581.
14. Cai, Q.; Yates, A. Large-scale semantic parsing via schema matching and lexicon extension. In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, Sofia, Bulgaria, 4–9 August 2013; pp. 423–433.
15. Liang, P.; Jordan, M.I.; Klein, D. Learning dependency-based compositional semantics. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologie, Portland, OR, USA, 19–24 June 2011; pp. 590–599.
16. Yao, X.; Van Durme, B. Information extraction over structured data: Question answering with freebase. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Baltimore, MD, USA, 22–27 June 2014; pp. 956–966.
17. Dai, Z.; Li, L.; Xu, W. CFO: Conditional focused neural question answering with large-scale knowledge bases. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Berlin, Germany, 7–12 August 2016; pp. 800–810.
18. Yang, F.; Gan, L.; Li, A.; Huang, D.; Chou, X.; Liu, H. Combining deep learning with information retrieval for question answering. In *Natural Language Understanding and Intelligent Applications*; Springer International Publishing: Cham, Switzerland, 2016; pp. 917–925.
19. Xie, Z.; Zeng, Z.; Zhou, G.; He, T. Knowledge base question answering based on deep learning models. In *Natural Language Understanding and Intelligent Applications*; Springer International Publishing: Cham, Switzerland, 2016; pp. 300–311.
20. Bordes, A.; Weston, J.; Usunier, N. Open question answering with weakly supervised embedding models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases—Volume 8724*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 165–180.

21. Bordes, A.; Usunier, N.; Garcia-Dur, A.; Weston, J.; Yakhnenko, O. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*; Curran Associates Inc.: Lake Tahoe, NV, USA, 2013; pp. 2787–2795.
22. Jain, S. Question answering over knowledge base using factual memory networks. In Proceedings of the NAACL Student Research Workshop, San Diego, CA, USA, 13–15 June 2016; pp. 109–115.
23. Lai, Y.; Lin, Y.; Chen, J.; Feng, Y.; Zhao, D. Open domain question answering system based on knowledge base. In *Natural Language Understanding and Intelligent Applications*; Springer International Publishing: Cham, Switzerland, 2016; pp. 722–733.
24. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
25. Bahdanau, D.; Cho, K.; Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv* **2014**, arXiv:1409.0473.
26. Lafferty, J.D.; McCallum, A.; Pereira, F.C.N. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Proceedings of the 18th International Conference on Machine Learning, San Francisco, CA, USA, 28 June–1 July 2001; pp. 282–289.
27. Levenshtein, V.I. Binary codes capable of correcting deletions, insertions, and reversals. *Sov. Phys. Doklady* **1966**, *10*, 707–710.
28. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.; Dean, J. Distributed representations of words and phrases and their compositionality. In Proceedings of the 26th International Conference on Neural Information Processing Systems, Lake Tahoe, NV, USA, 5–10 December 2013; pp. 3111–3119.
29. Ling, W.; Luís, T.; Marujo, L.; Astudillo, R.F.; Amir, S.; Dyer, C.; Black, A.W.; Trancoso, I. Finding function in form: Compositional character models for open vocabulary word representation. *arXiv* **2015**, arXiv:1508.02096.
30. Cho, K.; Merrienboer, B.V.; Bahdanau, D.; Bengio, Y. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv* **2014**, arXiv:1409.1259.
31. Hochreiter, S.; Schmidhuber, J. Lstm can solve hard long time lag problems. In Proceedings of the 9th International Conference on Neural Information Processing Systems, Singapore, 18–22 November 1996; pp. 473–479.
32. Cui, Y.; Chen, Z.; Wei, S.; Wang, S.; Liu, T.; Hu, G. Attention-over-attention neural networks for reading comprehension. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, BC, Canada, 30 July–4 August 2017; pp. 593–602.
33. Zhang, H.; Xu, G.; Liang, X.; Huang, T. An attention-based word-level interaction model: Relation detection for knowledge base question answering. *arXiv* **2018**, arXiv:1801.09893.
34. Zeiler, M.D. Adadelta: An adaptive learning rate method. *arXiv* **2012**, arXiv:1212.5701.
35. Lample, G.; Ballesteros, M.; Subramanian, S.; Kawakami, K.; Dyer, C. Neural architectures for named entity recognition. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, CA, USA, 13–15 June 2016; pp. 260–270.
36. Dong, C.; Zhang, J.; Zong, C.; Hattori, M.; Di, H. Character-based lstm-crf with radical-level features for chinese named entity recognition. In *Natural Language Understanding and Intelligent Applications*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 239–250.
37. Greff, K.; Srivastava, R.K.; Koutník, J.; Steunebrink, B.R.; Schmidhuber, J. Lstm: A search space odyssey. *IEEE Trans. Neural Netw. Learn. Syst.* **2017**, *28*, 2222–2232. [CrossRef] [PubMed]
38. Reimers, N.; Gurevych, I. Optimal hyperparameters for deep lstm-networks for sequence labeling tasks. *arXiv* **2017**, arXiv:1707.06799.