Capturing Sentence Relations for Answer Sentence Selection with Multi-Perspective Graph Encoding

Zhixing Tian,1,2 Yuanzhe Zhang,1 Xinwei Feng,3 Wenbin Jiang,3 Yajuan Lyu,3 Kang Liu,1,2 Jun Zhao1,2

1Institute of Automation, Chinese Academy of Sciences, Beijing, China
2University of Chinese Academy of Sciences, Beijing, China
3Baidu Inc., Beijing, China
{zhixing.tian, yzzhang, kliu, jzhao}@nlpr.ia.ac.cn
{fengxinwei, jiangwenbin, lvyajuan}@baidu.com

Abstract

This paper focuses on the answer sentence selection task. Unlike previous work, which only models the relation between the question and each candidate sentence, we propose Multi-Perspective Graph Encoder (MPGE) to take the relations among the candidate sentences into account and capture the relations from multiple perspectives. By utilizing MPGE as a module, we construct two answer sentence selection models which are based on traditional representation and pre-trained representation, respectively. We conduct extensive experiments on two datasets, WikiQA and SQuAD. The results show that the proposed MPGE is effective for both types of representation. Moreover, the overall performance of our proposed model surpasses the state-of-the-art on both datasets. Additionally, we further validate the robustness of our method by the adversarial examples of AddSent and AddOneSent.

Introduction

Answer sentence selection is an important subtask of question answering, where given a question and a set of sentences, a model is required to select the most suitable sentence that can answer the question. In this task, most of the previous work (Wang and Nyberg 2015; Wang, Liu, and Zhao 2016; Tran and Niedéréé 2018) has only focused on capturing semantic relations between questions and candidate sentences which directly serves the goal of the task. Basically, they encode each candidate independently, and then conduct a semantic matching between the question and each candidate. This kind of method simply ignores the relations among the candidates which are also supportive for obtaining the answer.

To explain the importance of capturing the relations among candidate sentences, we show an example in Figure 1. In this example, if only focusing on the relations between the question and each candidate and understanding each candidate independently, the model will tend to select one of the distractors, S1 and S3, as the answer. This is because both of them share more lexical overlap with the question than does the true answer, S2. However, if the model captures the relations among the candidates, it will be quite supportive for obtaining the true answer. Concretely, there is coreference between the words from S1 or S2; by utilizing this relation, the model will know Cavill is Henry Cavill and the character is exactly the Superman mentioned in the question. Besides, the name of the film Man of Steel presented in S1, which is also a key phrase in the question, can be complementary information for understanding the character mentioned in S2 and facilitate building a closer connection between the answer S2 and the question. Therefore, capturing the relations among the candidates can help to understand each candidate in its context and further contributes to selecting the answer sentence indirectly. Tan et al. (2018) proposed a model which employed Gated Recurrent Unit (GRU) at the sentence level to capture the relations among the candidates. The model made considerable progress against previous work. However, as a variant of RNN, the ability of GRU is limited to modeling the sentence relation from the perspective of sequence, which is not...
enough to cover the complex and versatile relations that exist among candidate sentences.

In this paper, we propose a novel sentence encoder, Multi-Perspective Graph Encoder (MPGE), to capture the relations among the candidates. As shown in Figure 2, the MPGE builds relation graphs of the candidates from multiple perspectives, encodes the candidates based on those graphs, and finally outputs the representation that has aggregated contextual information. Specifically, we propose two strategies to build the graphs. One is static building strategy, which builds the graphs from three perspectives, including entity co-occurrence, sentence distance, and semantic similarity. The other is dynamic strategy, which builds the graph from the instance specific perspective. Compared with (Tan et al. 2018), our proposed MPGE captures the candidate relations more comprehensively, because it models the relations from more than one perspective. Moreover, we explicitly define the graphs to describe the candidate relations, which retains better interpretability than (Tan et al. 2018) that models the relations in a neural network implicitly.

Furthermore, by utilizing MPGE as a module, we construct two answer sentence selection models based on two types of representation. One is built on the traditional representation, which is obtained by a feedforward neural network without pre-train. The other is built on the pre-trained representation obtained from BERT (Devlin et al. 2018). Experimental results show that our proposed MPGE is effective for both types of representation and the overall performance surpasses the state-of-the-art on both datasets.

We demonstrate the effectiveness, universality and robustness of MPGE experimentally. Moreover, our overall performance surpasses the state-of-the-art on both datasets.

**Related Work**

**Answer Sentence Selection:** Previous work on this task has primarily focused on feature based methods. Wang, Smith, and Mitamura (2007) compared the question with candidate sentences according to their syntactical matching in parse trees. Heilman and Smith (2010) developed an improved Tree Edit Distance (TED) model to conduct matching by minimal edit sequences between dependency parse trees.

Severyn and Moschitti (2013) presented an automatic method to extract tree-edit features over parsing trees. Recently, deep learning and neural networks have been employed in this task and have shown promising results. Yu et al. (2014) applied a convolutional neural network (CNN) to encode the question and candidate sentences, and subsequently used logistic regression for prediction. In addition to CNN, another popular neural network in this field is Recurrent Neural Networks (RNN) including its variants such as Long-Short Term Memory network (LSTM) and GRU. Wang and Nyberg (2015) introduced LSTM to capture sequence information when encoding the sentences. Tan et al. (2015) combined CNN and LSTM into a hybrid architecture which utilizes the advantages of both architectures. Attention mechanisms play a critical role in the latest answer sentence selection research. Yin et al. (2016) presented a general attention based CNN for modeling a pair of sentences. Tran and Niedereéé (2018) proposed a sequential attention mechanism, which applies multiple steps of attention to learn representations for the candidate sentences.

**Graph Neural Network:** Graph Neural Network (GNN) is a kind of deep learning method that operates in the graph domain, and it has gained recent wide attention due to its convincing performance and high interpretability. Based on CNN and graph embedding, Kipf and Welling (2016) proposed Graph Convolutional Network (GCN) to conduct a semi-supervised learning on graph-structures. To focus on the important part of the graph, Velicković et al. (2017) introduced attention weight into graph computations, which is known as Graph Attention Networks (GAT). Wang et
S1: Cross section of sclerenchyma fibers in plant ground tissue.
S2: Microscopic view of a histologic specimen of human lung tissue ...
S3: In Biology, Tissue is a cellular organizational level intermediate ...
S4: A tissue is an ensemble of similar cells from the same origin that ...
S5: Organs are then formed by the functional grouping together of ...

Underlying Encoding Module

Candidate Sentences

S1: Cross section of sclerenchyma fibers in plant ground tissue.
S2: Microscopic view of a histologic specimen of human lung tissue ...
S3: In Biology, Tissue is a cellular organizational level intermediate ...
S4: A tissue is an ensemble of similar cells from the same origin that ...
S5: Organs are then formed by the functional grouping together of ...

Multi-Perspective Graph Encoder

To capture the relations among the candidates, we propose a sentence encoder, MPGE, as shown in Figure 3. By the static building strategy and the dynamic one, MPGE builds the relation graphs from four perspectives, including entity co-occurrence, sentence distance, semantic similarity and instance specificity. Based on those graphs, MPGE encodes the sentences by GCN, and obtain the sentence representation of different perspective. Finally MPGE fuse all of the sentence representation, and output the fused representation which has aggregated contextual information gathered from multiple perspective.

Input

One input of MPGE is the text of candidate sentences. The other is the original representation of the candidates \( H_0 \). The representations \( A_{ij}^{ent} \) are defined by:

\[
A_{ij}^{ent} = \begin{cases} 
1 & \text{if co-occurrence entity exists between } S_i \text{ and } S_j \\
0 & \text{otherwise}
\end{cases}
\]

Thus, we obtain the Entity Graph formulated by the adjacent matrix, \( A^{ent} \).

• **Distance Graph**: In view of the observation that sentences closer to each other tend to be more relevant in a passage, we take the distance into account when modeling the relation of two candidates. We employ the Gaussian Distribution to measure the distance. Thus in distance graph, the edge between the i-th sentence, \( S_i \), and the j-th sentence, \( S_j \), is calculated by

\[
A_{ij}^{dist} = \frac{1}{\sigma \sqrt{2 \pi}} e^{-\frac{(j-i)^2}{2\sigma^2}}
\]

where \( \sigma \) is a hyper-parameter. Thus, we get the Distance Graph formulated by the adjacent matrix, \( A^{dist} \).

• **Similarity Graph**: To build a connection between two candidates with similar semantics, and then conduct a rich information encoding for each of them, we define a similarity graph. Low dimension vectors, obtained by the pre-trained BERT model, are used to represent sentences. We

Static Building Strategy

The static building strategy is a kind of pre-defined strategy, which means it will not change during the training process. We build three static graphs from different perspective by this kind of strategy.

- **Entity Graph**: We link two sentences(nodes), if there is entity co-occurrence between them, which indicates that the two sentences are likely to describe the same entity and share a common topic. The edge between two sentences \( S_i \) and \( S_j \) is defined by:

\[
A_{ij}^{ent} = \begin{cases} 
1 & \text{if co-occurrence entity exists between } S_i \text{ and } S_j \\
0 & \text{otherwise}
\end{cases}
\]

Thus, we obtain the Entity Graph formulated by the adjacent matrix, \( A^{ent} \).

- **Distance Graph**: In view of the observation that sentences closer to each other tend to be more relevant in a passage, we take the distance into account when modeling the relation of two candidates. We employ the Gaussian Distribution to measure the distance. Thus in distance graph, the edge between the i-th sentence, \( S_i \), and the j-th sentence, \( S_j \), is calculated by

\[
A_{ij}^{dist} = \frac{1}{\sigma \sqrt{2 \pi}} e^{-\frac{(j-i)^2}{2\sigma^2}}
\]

where \( \sigma \) is a hyper-parameter. Thus, we get the Distance Graph formulated by the adjacent matrix, \( A^{dist} \).

- **Similarity Graph**: To build a connection between two candidates with similar semantics, and then conduct a rich information encoding for each of them, we define a similarity graph. Low dimension vectors, obtained by the pre-trained BERT model, are used to represent sentences. We
calculate the edge between two candidates, \( S_i \) and \( S_j \), by cosine similarity:

\[
A_{ij}^{simi} = \frac{r_i \cdot r_j}{\|r_i\|_2 \cdot \|r_j\|_2} \quad (3)
\]

where \( r_i \) and \( r_j \) are vector representation of \( S_i \) and \( S_j \) respectively. Thus, we obtain the adjacent matrix \( A^{simi} \) of the Similarity Graph.

**Dynamic Building Strategy** The static building strategy is a kind of general method of modeling the relations among the candidates for all of the instance, but it cannot cover some instance specific relations. As a complementary, we further present a dynamic building strategy, which is based on the self-attention mechanism and is trainable. We build the Dynamic Graph by this method. In the graph, the edge between two sentences, \( S_i \) and \( S_j \), is obtained by

\[
A_{ij}^{dyn} = \frac{\exp(\alpha_{ij})}{\sum_j \exp(\alpha_{ij})} \quad (4)
\]

\[
\alpha_{ij} = \sigma(w_s h_i^0)\sigma(w_t h_j^0) \quad (5)
\]

where \( \sigma \) is the activation function, and \( w_s \in \mathbb{R}^{d \times d} \) is a trainable weight matrix. Thus, we obtain the adjacent matrix \( A^{dyn} \), of the dynamic graph.

**Multi-Perspective GCN Encoding**

We employ GCN (Kipf and Welling 2016) to encode the sentences on the four graphs built from different perspectives. The layer-wise propagation rule of GCNs is formulated as

\[
H^{(t+1)} = \sigma \left( \tilde{D}^{-\frac{1}{2}} A \tilde{D}^{-\frac{1}{2}} H^{(t)} W^{(t)} \right) \quad (6)
\]

where \( H^t \in \mathbb{R}^{N \times d} \) is the input representation of the nodes of the graph, \( N \) is the number of nodes, and \( d \) is the dimensional size. \( A \in \mathbb{R}^{N \times N} \) is the original adjacency matrix of the graph. \( \tilde{A}_{ij} = \sum_j \tilde{A}_{ij} \) and \( W^{(t)} \) is a layer-specific trainable weight matrix, \( \sigma \) is the activation function. Note that as the edges \( \tilde{A}_{ij} \) have been calculated during building the graphs, we do not add the identity matrix \( I \) to the adjacent matrix \( A \), which is different from (Kipf and Welling 2016).

We pair the original sentence representation, \( H^0 \), with each of the four adjacent matrices, \( A^{ent}, A^{dist}, A^{simi}, \) and \( A^{dyn} \), and then input the four pairs into GCNs respectively. After that, we obtain four types of sentence representation, \( H_0^{ent}, H_0^{dist}, H_0^{simi}, H_0^{dyn} \in \mathbb{R}^{N \times d} \), which carry with the contextual information encoded from different perspectives.

**Fusion and Output**

To preserve the information from the original sentence representation, \( H^0 \), we employ a residual connection around the GCNs. After that, we aggregate the information from the outputs of the GCNs and the residual connection by applying a bi-directional GRU:

\[
H_{mpge}^{mpge} = BiGRU \left( H^{all} \right) \quad (7)
\]

\[
H^{all} = [H^0; H_0^{ent}; H_0^{dist}; H_0^{simi}; H_0^{dyn}] \quad (8)
\]

where \( H_{mpge}^{mpge} \in \mathbb{R}^{N \times d} \) is the final output of MPGE. Thus far we update the sentence representation to the one that contains multiple perspectives contextual information.

**MPGE Based Answer Sentence Selection**

We embed MPGE module into two answer sentence selection models. One is the traditional representation based model TR-MPGE-AS, shown in Figure 4, which obtains the original sentence representation, \( H^0 \), by a RNN based encoder and a pooling layer. The other, shown in Figure 5, is a BERT representation based model BR-MPGE-AS, which obtains \( H^0 \) by fine tuning the pre-trained representation.

**MPGE Encoding**

Here, the proposed MPGE is applied to update the representation for candidates, as described in the last section. Thus we acquire an updated sentence representation, \( H_{mpge}^{mpge} \in \mathbb{R}^{N \times d} \).

---

**Figure 4: Traditional Representation and MPGE based Answer sentence Selection model(TR-MPGE-AS). The part in the dashed box is the Underlying Encoding Module.**

**Figure 5: BERT-based Answer sentence Selection model(BR-MPGE-AS).**

**Traditional Representation & MPGE Based Model**

**Document Reader Encoder:** As shown in Figure 4, we employ the encoder of the Document Reader (Chen et al. 2017a), which is a simple and effective reading comprehension model, to encode the question and the candidates at word level. The encoder is composed of an embedding layer, an attention layer and a LSTM layer. By utilizing this encoder, we can obtain the word level representation for the question \( Q \in \mathbb{R}^{L_0 \times d} \), and that for each sentence. Specifically, for the \( k \)-th candidate sentence, we have \( S_k \in \mathbb{R}^{L_k \times d} \). \( L_0 \) and \( L_k \) are the sequence length of the question and the \( k \)-th candidate respectively.

**Sentence Attention Pooling:** We apply an attention pooling to obtain the sentence-level representation:

\[
h_k^0 = \sum_j \gamma_i S_{ki} \quad (9)
\]

\[
\gamma_i = \frac{\exp(\sigma(w_p S_{ki}))}{\sum_j \exp(\sigma(w_p S_{kj}))} \quad (10)
\]

where \( w_p \in \mathbb{R}^d \) is a trainable weight vector. \( S_{ki} \) is the \( i \)-th word in the \( k \)-th candidate. Thus, we obtain the original sentence-level representation, \( H^0 = \{h_0^0, h_1^0, ..., h_N^0\} \in \mathbb{R}^{N \times d} \), where \( N \) is the number of candidates. Meanwhile, we obtain the sentence-level representation, \( H^Q \in \mathbb{R}^d \), for the question.
**Bilinear Matching:** Finally, the score of each of the candidates can be calculated by a bilinear matching:

\[
score_{c_i} = \frac{\exp(H^{mpge}_i W_{sc} H^Q)}{\sum_i \exp(H^{mpge}_i W_{sc} H^Q)}
\]

where \(score_{c_i}\) is the normalized selection score of the i-th candidate. \(W_{sc} \in \mathbb{R}^{d \times d}\) is a trainable matrix.

**BERT Representation & MPGE Based Model:**

**Pre-trained BERT:** As shown in Figure 5, we employ the pre-trained BERT model as an underlying encoder. By using BERT, we obtain question aware candidate representation, \(H^0 \in \mathbb{R}^{N \times d}\).

**MPGE Encoding:** Next, the proposed MPGE is applied to update the representation for candidates to \(H^{mpge} \in \mathbb{R}^{N \times d}\).

**Sentence Scoring:** Finally, we obtain the score for each candidate by

\[
score_{c_i} = \frac{\exp(H^{mpge}_i W_{sd})}{\sum_i \exp(H^{mpge}_i W_{sd})}
\]

where \(score_{c_i}\) is the normalized selection score of the i-th candidate. \(W_{sd} \in \mathbb{R}^d\) is a trainable vector.

---

**Experiments and Analysis**

**Datasets and Metrics**

The datasets we choose are WikiQA and SQuAD. This is because the candidates of those two datasets compose a complete paragraph, which means there is a natural correlation between these sentences. To further validate the robustness of our method, we also introduce the test on adversarial examples for SQuAD, and the test sets are also known as AddSent and AddOneSent (Jia and Liang 2017).

**WikiQA:** a popular benchmark dataset for answer sentence selection, based on factual questions from Wikipedia and Bing search logs. For each question, Yang, Yih, and Meek (2015) selected Wikipedia pages and used sentences in the summary paragraph as candidates. Following the same preprocessing steps as (Yang, Yih, and Meek 2015), we exclude the questions with no correct candidate answers.

**SQuAD:** a reading comprehension dataset, where the answer to each question is a span of text from the corresponding passage. In order to evaluate our answer sentence selection task, we split the sentences from the passage using the spaCy\(^1\) toolkit and then treat the sentence, where the span of the correct answer is located in, as the answer sentence.

**AddSent and AddOneSent:** the adversarial test sets for SQuAD. The adversarial examples are built by inserting distracting sentence to the passages of the original examples of SQuAD. Specially, for one original example, there are several corresponding adversarial examples in AddSent, and one adversarial example in AddOneSent. Following the preprocess for SQuAD, we employ those sets for the adversarial test of answer sentence selection.

**Evaluation Metrics:** We use two common evaluation measures of answer sentence selection task, mean average precision (MAP) and mean reciprocal rank (MRR), for the WikiQA dataset. As the number of candidate sentences in SQuAD is relatively few, 5.3 in average, we use the TOP 1 accuracy and MAP for evaluation on SQuAD as well as AddSent and AddOneSent, instead of MAP and MRR, which follows (Min et al. 2018).

**Implementation Details**

We implement two answer sentence selection models with the proposed MPGE: one is based on traditional representation, named TR-MPGE-AS, as shown in Figure 4, and the other is the BERT based one, named BR-MPGE-AS, as shown in Figure 5. The above models are implemented on PaddlePaddle\(^2\).

**TR-MPGE-AS:** In the Document Reader Encoder, we use 300-dimensional Glove (Pennington, Socher, and Manning 2014) word embeddings, token features annotated by the spaCy toolkit, and 3-layer bidirectional LSTMs with hidden size of 128. We concatenate the hidden of all 3 layers, so the output dimension size of the Document Reader Encoder is 768. Dropout with \(p = 0.4\) is applied to word embeddings and all the hidden units of LSTMs. 2-layer GCNs are applied inside MPGE. We take the hidden of the last layer as the output of GCNs, and the dropout rate is set to 0.2 for the hidden of GCNs.

**BR-MPGE-AS:** We employ the pre-trained bert-base (Devlin et al. 2018) model, distinguished from the bert-large model, as the underlying encoder, whose hidden size is 768, which is the same as the TR-MPGE-AS model. Following (Devlin et al. 2018), the learning rate is set to \(3 \times 10^{-5}\), the model is fine tuned for 3 epochs, and the dropout rate of BERT is set to 0.1. We use the default tokenizer of BERT to preprocess the input sentences. The configuration of the GCNs is the same as that in TR-MPGE-AS.

---

\(^1\)spaCy is a Python library for natural language processing with support for part-of-speech tagging, sentence segmentation, named entity recognition, and word vector operations.

\(^2\)https://github.com/PaddlePaddle/Paddle
Overall performance

Table 1 reports the results on the WikiQA dataset. Our proposed BR-MPGE-AS model, which utilizes both BERT and MPGE, outperforms the state-of-the-art by a large margin, 10.4% in terms of MAP and 9.7% in terms of MRR. The results on the SQuAD dataset are shown in Table 2. The BR-MPGE-AS model outperforms the state-of-the-art by 2.9% in terms of MAP. Besides, our traditional representation based model TR-MPGE-AS, which obtain original representation by a simple encoder, also surpasses the state of the art on both WikiQA and SQuAD. Concretely, the MPGE based model TR-MPGE-AS outperforms the state-of-the-art 1.0% in terms of MAP on WikiQA, and 1.1% in terms of MAP on SQuAD.

Effectiveness and Universality of MPGE

As shown in Table 1 and Table 2, we find significant improvement from TR-AS to TR-MPGE-AS. This demonstrates that our proposed MPGE is effective on the traditional representation. Meanwhile, we notice similar improvement, when comparing the performance of BR-MPGE-AS and that of BR-AS on both datasets, which means MPGE also works well on the powerful pre-trained representation. Therefore, our proposed MPGE, which capturing relations among the candidates, is effective for both the weak representation (the traditional one) and the strong one (BERT). Note that the Universality is a critical difference between our proposed MPGE and the GRU based method presented by Tan et al. (2018), which also focuses on the candidate relations, but be only proved effective on single type of representation.

Robustness of MPGE

Table 3 reports the results on the adversarial test sets, AddOneSent and AddSent. Note that, following (Jia and Liang 2017), during testing on AddSent we pick the worst performance for a group of adversarial examples which is adapted from one original example. As shown in the table, the models with MPGE perform better than those without MPGE on both two adversarial test set, AddOneSent and AddSent. Furthermore, we notice that the performance gap between the MPGE based model and that without MPGE becomes wider when we change the test set from Original to AddOneSent and AddSent. Thus, on the adversarial examples which contain distracting sentences, the MPGE can better shows its strength. This phenomenon further validates the robustness of MPGE. To our best knowledge, we are the first that validates the robustness of the model by adversarial examples on the answer sentence selection task.

Effectiveness of Multi-Perspective Graphs

To obtain better insights of our proposed Multi-Perspective Graph Encoder (MPGE), and discuss the effect of the graphs built from multiple perspectives, we conduct an in-depth ablation study on the test set of WikiQA. Table 4 shows the results.

Static Graphs: We remove each static graph from MPGE. The degradation in performance of the model -entity, -distance, and -similarity verifies the effect of those three types of static graphs. Specifically, We notice that the entity graph and similarity graph, which model the sentence relation from the perspective of topic and semantics, play more important roles in TR-MPGE-AS than they do in BR-MPGE-AS. Meanwhile, the distance graph contributes similarity performance in TR-MPGE-AS and BR-MPGE-AS. We speculate the reason is that for the BERT representation, part of the priori linguistic knowledge like topic relevance and semantics similarity has been acquired by the large scale
Table 4: Effectiveness of each graph. -entity, - similarity, -distance, and -dynamic are the models which ablate entity graph, similarity graph, distance graph, and dynamic graph respectively from TR-MPGE-AS and BR-MPGE-AS.

Table 5: MAP score of different fusion layer. GRU Fusion and Linear Fusion refer to the models that use the GRU and Linear projection respectively as the fusion layer.

To study the effectiveness of the GRU in the Fusion layer, we replace the GRU by a linear projection and test the different fusion layer on WikiQA datasets. As shown in the Table 5. The performance degrades slightly, when we use the linear fusion layer instead of the GRU. The results demonstrate that the GRU in the fusion layer could be a complementary to capturing candidate relations, especially from the perspective of sequence.

Comparing with GRU Based Model

Last but not the least, to further compare with the method that capturing the relations among the candidates by GRU introduced by (Tan et al. 2018), we build the GRU based answer sentence selection model. Specifically, we replace MPGE module of BR-MPGE-AS and TR-MPGE-AS with a 2-layers bidirectional GRU at the sentence level. Thus we get two GRU based model TR-GRU-AS and BR-GRU-AS, whose performance on WikiQA is shown in Figure 6.

On one hand, TR-GRU-AS (BR-GRU-AS) is better than TR-AS (BR-AS), which confirms the effectiveness of capturing relations among the candidate sentences. On the other hand, compared with TR-MPGE-AS (BR-MPGE-AS), the performance of TR-GRU-AS (BR-GRU-AS) falls significantly, 3.5% in terms of MAP and 3.6% in terms of MRR (2.7% in terms of MAP and 2.0% in terms of MRR). This demonstrate that our proposed MPGE which capturing the sentence relations from multiple perspectives is more effective than the GRU which only modeling the relations from the perspective of sequence.

Conclusion

In this paper, we focus on the answer sentence selection task. We propose a novel candidate sentence encoder, MPGE, to capture the relations among the candidates from multiple perspectives. By utilizing MPGE as a module, we construct two answer sentence selection models, which are build on traditional representation and pre-trained representation, respectively. We conduct experiments on two datasets. Our overall performance surpasses the state-of-the-art on both datasets. We experimentally demonstrate that our proposed MPGE which capturing the sentence relations from multiple perspectives is more effective than the GRU which only modeling the relations from the perspective of sequence.

Acknowledgments

This work is supported by the National Key R&D Program of China under Grant 2018YFB1005100, the National Natural Science Foundation of China (No.61533018), and the
independent research project of National Laboratory of Pattern Recognition. This work is also supported by Baidu-CASIA Joint Project and Beijing Academy of Artificial Intelligence(BAAI). We would like to thank the anonymous reviewers for their valuable feedback.

References
Chen, D.; Fisch, A.; Weston, J.; and Bordes, A. 2017a. Reading Wikipedia to answer open-domain questions. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, 1870–1879.

Chen, Q.; Hu, Q.; Huang, J. X.; He, L.; and An, W. 2017b. Enhancing recurrent neural networks with positional attention for question answering. In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, 993–996.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

Heilman, M., and Smith, N. A. 2010. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, 1011–1019.

Jia, R., and Liang, P. 2017. Adversarial examples for evaluating reading comprehension systems. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, 2021–2031.

Kipf, T. N., and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.

Miller, A.; Fisch, A.; Dodge, J.; Karimi, A.-H.; Bordes, A.; and Weston, J. 2016. Key-value memory networks for directly reading documents. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, 1400–1409.

Min, S.; Zhong, V.; Socher, R.; and Xiong, C. 2018. Efficient and robust question answering from minimal context over documents. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, 1725–1735.

Pennington, J.; Socher, R.; and Manning, C. 2014. Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing, 1532–1543.

Rajpurkar, P.; Zhang, J.; Lopyrev, K.; and Liang, P. 2016. Squad: 100,000+ questions for machine comprehension of text. arXiv preprint arXiv:1606.05250.

Santos, C. d.; Tan, M.; Xiang, B.; and Zhou, B. 2016. Attentive pooling networks. arXiv preprint arXiv:1602.03609.

Severyn, A., and Moschitti, A. 2013. Automatic feature engineering for answer selection and extraction. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, 458–467.

Tan, M.; Santos, C. d.; Xiang, B.; and Zhou, B. 2015. Lstmbased deep learning models for non-factoid answer selection. arXiv preprint arXiv:1511.04108.