# Enhanced Embedding based Attentive Pooling Network for Answer Selection

Zhan Su[1], Benyou Wang[1,3], Jiabin Niu[1], Shuchang Tao[1], Peng Zhang*[1], and Dawei Song[1,2]

[1] Tianjin Key Laboratory of Cognitive Computing and Application, School of Computer Science and Technology, Tianjin University, Tianjin, P.R. China
shuishen112@gmail.com,wabywang@tencent.com,{niujiabin,taoshuchang,pzhang}@tju.edu.cn
[2] Department of Computing and Communications, The Open University, UK
[3] Tencent, Shenzhen, China
dawei.song2010@gmail.com

**Abstract.** Document-based Question Answering tries to rank the candidate answers for given questions, which needs to evaluate matching score between the question sentence and answer sentence. Existing works usually utilize convolution neural network (CNN) to adaptively learn the latent matching pattern between the question/answer pair. However, CNN can only perceive the order of a word in a local windows, while the global order of the windows is ignored due to the window-sliding operation. In this report, we design an enhanced CNN[4] with extended order information (e.g. overlapping position and global order) into inputting embedding, such rich representation makes it possible to learn an order-aware matching in CNN. Combining with standard convolutional paradigm like attentive pooling, pair-wise training and dynamic negative sample, this end-to-end CNN achieve a good performance on the DBQA task of NLPCC 2017 without any other extra features.

## 1 INTRODUCTION

Recently, deep learning approaches have been successfully applied to a variety of Natural Language Processing (NLP) tasks, such as Sentiment Analysis [1], Automatic Conversation [2] and Paraphrase Identification [3]. Compared with traditional approaches [4, 5], which require manual features and rely on domain experience, deep learning approaches have ability to automatically learn optimal feature representation. For Question Answering, deep learning approaches have also achieved good performance [6–8] in both English and Chinese datasets.

In this paper, our focus is Document-based Question Answering (DBQA), also known as Answer Selection (AS), which is a typical subtask of Question Answering. Given a question, DBQA task is to find accurate answers from a pool of pre-selected answer candidates [9] and the selection process is based on

[*] Corresponding author: pzhang@tju.edu.cn

[4] https://github.com/shuishen112/pairwise-deep-qa

the similarity matching between question and answers. Due to the sentences of DBQA are short texts, we utilize Convolutional Neural Network (CNN) architecture to model the sentences.

Although CNN has a strong ability to extract robust features, CNN is still unable to find out all the useful information, such as overlap which has been proved efficient for our QA task [7]. In addition, different words contribute different weights to the sentence. Despite of the meaning of words, the position of tokens in the sentence is also important. In our previous work, we have given a detailed explanation about the importance of the position information [10]. In order to tackle the above problems, we enchance the CNN by encoding position information and word overlap into word representation by additional dimensions [11]. For a typical text matching task, the representation not only contain the information of the text itself, but also the interdependence between the question/answer. The comparative effective method to model the relation of question/answer is attention mechanism at present [3, 12]. In this paper, we also investigate this mechanism to our architecture.

In addition to the sentence representation, the ranking approach is also a key. The common used ranking approaches are pointwise and pairwise strategies. Compared with the pointwise approach, the pairwise approach take advantage of more information about the ground truth [7]. For the sampling strategy in pairwise, Dynamic Negative Sampling (DNS) will largely improve the effect of the models.

Thus, the main characteristics of our model are as follow: First, we take position information and word overlap into consideration to obtain rich representation. Second, we utilize attention mechanism to exploit the interdependence between question/answer. Third, we employ pairwise ranking approach and DNS to improve the performance of our model.

# 2 MODEL ARCHITECTURE

## 2.1 Convolution Neural Network

In this work, we apply two kinds of CNN architectures into QA task. One is a simple QA-CNN, and the other is the attentive pooling network which has attract great attention in QA task. Both architectures can not capture the position information when the convolution filter slide through the sentence matrices. Convolutions and pooling operations will lose information about the local order of words. To tackle this deficiency, we extend word embedding with additional dimensions, such as overlap and position information. The approach can make the model more suitable for the Chinese DBQA task.

As is shown in the Fig. 1. Given a QA pair$(q, a)$, we truncate or pad the text sentence to a fixed length so that the sentence matrices have the same dimension as shown in the Fig. 1. The first layer of our model contains two sequences of word embeddings, $q^{emb} = r^{w_1}, ..., r^{w_M}$ and $a^{emb} = r^{w_1}, ..., r^{w_L}$, where the length of question is M and the length of answer is L. Then we equip our model with an overlap embedding and a sense of order by embedding the position of input tokens. Then we obtain the input element representations $e = (w_1 + o_1 + p_1, ..., w_m + o_m + p_m)$ where the $o$ is the overlap embedding and the $p$ is the position embedding.

In the second layer, we typically use convolution filters, whose width is the same as the width of the input matrix, to slide over the sentence matrix. The height may vary, but sliding windows over 2-5 words at a time is typical. Then we apply a max-pooling to the output of convolution filters, which convert the matrix to vector representations $r^q$, $r^a$. In the last layer we compute the cosine similarity between these two representations.

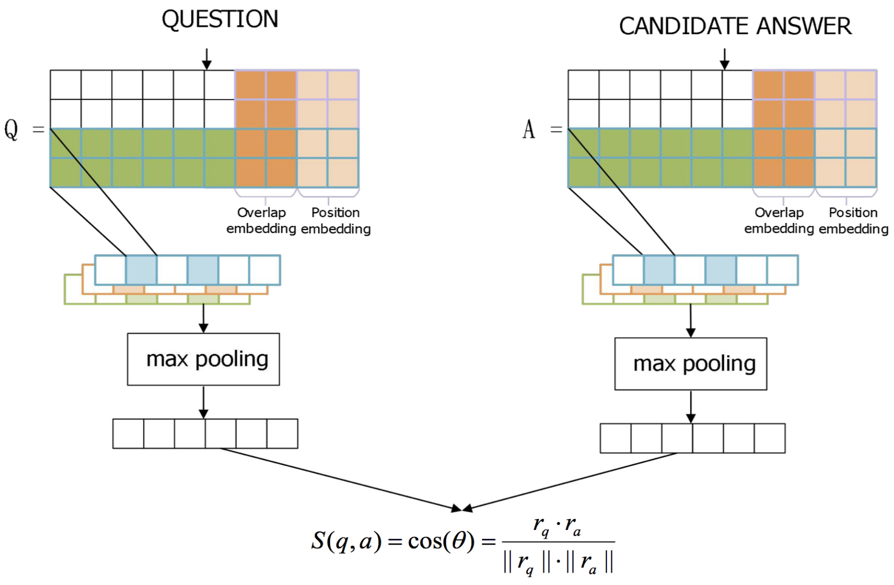

**Fig. 1.** Basic architecture of CNN

### 2.2 Attentive Pooling Neural Network

The simple QA-CNN learn representation of input individually. Instead of using max pooling, we use the attentive pooling networks so that the representation of the question and answer can be learned by the QA pairs. As shown in Fig 2, the output of convolution are matrices $Q \in R^{c \times M}$ and $A \in R^{c \times L}$. The matrix $G \in R^{M \times L}$ can be computed as follows:

$$G = tanh(Q^T U A) \tag{1}$$

$U \in R^{c \times c}$ are parameters and can be learned by our model. $G$ is soft alignment between the $k$-size context windows of $Q$ and $A$, We can obtain important-score vectors $g^q \in R^M$ and $g^a \in R^L$ after applying column-wise and row-wise max-pooling over G. Then the attention vectors $\delta^q$ and $\delta^a$ can be computed by applying softmax function over importan-score vectors. The final representation of $q$ and $a$ are as follows:

$$r^q = Q\delta^q \tag{2}$$

$$r^a = A\delta^a \tag{3}$$

We will compute the cosine similarity between the $r^q$ and $r^a$ like in the simple QA-CNN.

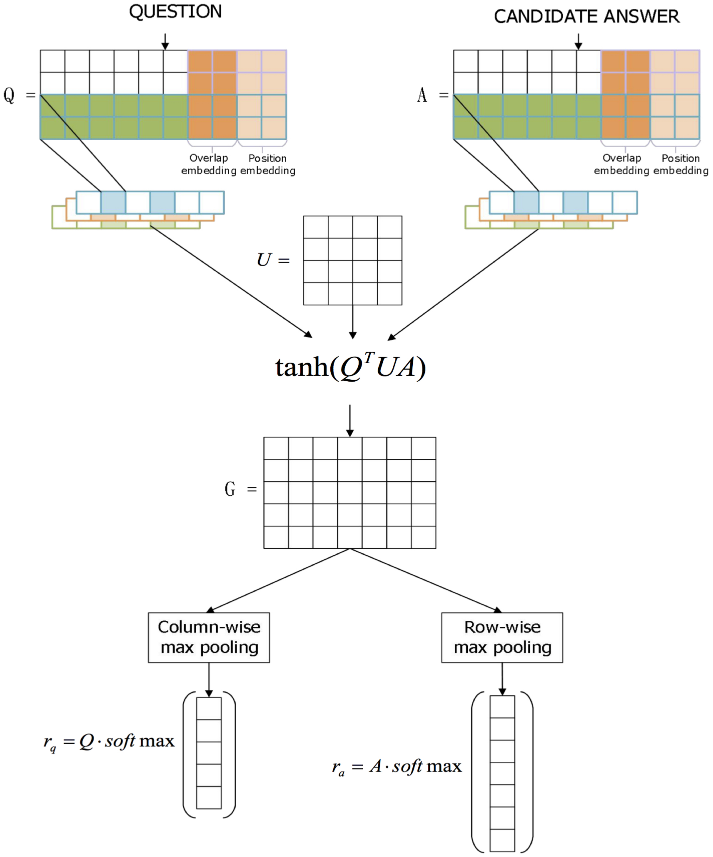

**Fig. 2.** Attentive pooling network with additional dimension

### 2.3 Triplet ranking loss function

Instead of treating the task as a pointwise classification problem, our input pairs are triplet items *(question,positive answer,negative answer)*. Given a question $q$,we can sample positive pairs $(q, a^+)$ and negative pairs $(q, a^-)$ where $a^+, a^-$ donate the positive and negative answer, respectively. Our goal is to learn a representation function $f(.)$ which can make the score of positive pairs is larger than the negative pairs.

$$f(q, a^+) > f(q, a^-), \forall q, a^+, a^- \tag{4}$$

we use triplet ranking hing loss

$$L = max(0, m - f(q, a^+) + f(q, a^-)) + \lambda \parallel W \parallel^2 \tag{5}$$

where $\lambda$ is a regularization parameter, and $W$ is the parameters of CNN model.

### 2.4 Sampling Strategy

In our DBQA task,we use two sampling strategies which has proven to be effective in QA task[13].

**Random Sampling** : Given a question, we randomly select one negative answer for each positive answer.

**Dynamic Negative Sampling** : In general, what confuse our model are some of the confusing negative cases rather than those obvious wrong answers. Thus, instead of using random strategy, we can use the most competitive negative answer. In each epoch, we compute the similarity between the question and negative answers. We pick the highest score negative answer as the most competitive sample.

## 3 EXPERIMENTAL EVALUATION

### 3.1 Dataset

The DBQA task in nlpcc 2017 provides three datasets. The number of QA pairs in training data is 181882, the number in test1 and test2 data is 122531 and 47372, respectively. The unique questons in training, test1, test2 datasets is 8772, 5997, 2550. We utilize the pynlpir tool to segment the sentences. The max length of question tokens is 40, while the max length of answer tokens is 1076. The length is shown in the Fig 3. Since the length of most of the answers is less than 75, we truncate the answer length to 75.

### 3.2 Embedding

The embedding in this task is very important. We train our own 300 dimension embedding by word2vec tool [14]. The raw corpus is Chinese Wikipedia. After putting all the tokens in the dataset into a dictionary, we can find 50 percent

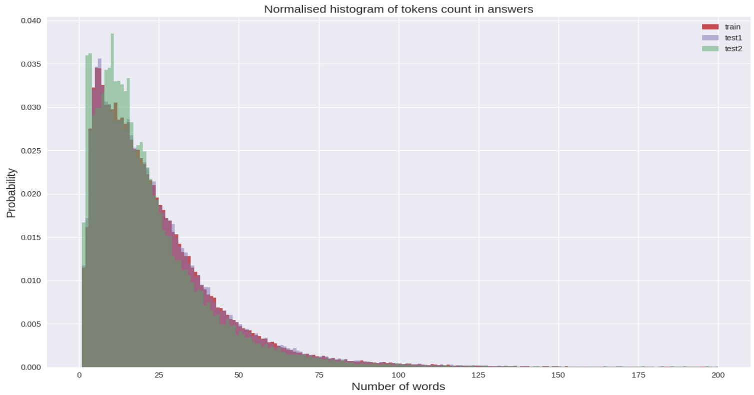

**Fig. 3.** the length of answer tokens in datasets

tokens in our pretrained embedding. There are a large number of places, names, numbers which we can't obtained by the raw corpus. So We assign a random vector between -0.5 and 0.5 for these tokens which means we will lose some important information.

### 3.3 Result

We implement our model using the open source tools tensorflow [5] and train the model in 50 epochs. The performance of our model is shown in Table 1:

**Table 1.** Result

| method | pooling | loss | MAP(test1) | MAP(test2) |
|---|---|---|---|---|
| CNN-base | max | pointwise | 0.408 | 0.371 |
| CNN-base | max | pairwise | 0.782 | 0.657 |
| CNN-base | max | pairwise | 0.784 | 0.661 |
| CNN-base | attentive | pairwise | 0.772 | 0.646 |
| +overlap | max | pointwise | 0.820 | 0.553 |
| +overlap | max | pairwise | 0.828 | 0.674 |
| +overlap | attentive | pairwise | 0.811 | 0.672 |
| +positon,overlap | max | pointwise | 0.815 | 0.554 |
| +positon,overlap | attentive | pairwise | 0.819 | **0.675** |
| +positon,overlap | max | pairwise | **0.834** | **0.679** |

*CNN-base* means that we do not use any additional feature embedding. Compared with pointwise *CNN-base*, we can see that pairwise *CNN-base* has a better

[5] https://www.tensorflow.org/

result. The dynamic negative sampling and random sampling both contribute to our model. We regard the *CNN-base* model as a baseline. The *+overlap* is our enhanced model with extended overlap embedding, The *+position* is our model with extended position information. The result indicate that the enhanced model is much better than the *CNN-base* model especially for test1. All the work we did depends on test1 which result in ranking 5th among the 21 submissions.

**Table 2.** Sensitivity analysis

| **loss** | **em-dim** | **extend-dim** | **region-size** | **MAP(test1)** | **MAP(test2)** |
|---|---|---|---|---|---|
| pointwise | 50 | 2 | 1,2,3,5 | 0.801 | 0.513 |
| pointwise | 50 | 5 | 1,2,3,5 | 0.814 | 0.540 |
| pointwise | 50 | 1 | 1,2,3,5 | 0.827 | 0.563 |
| pointwise | 300 | 10 | 1,2,3,5 | 0.820 | 0.553 |
| pointwise | 50 | 20 | 1,2,3,5 | 0.824 | 0.530 |
| pairwise | 50 | 2 | 1,2,3,5 | 0.795 | 0.639 |
| pairwise | 50 | 5 | 1,2,3,5 | 0.795 | 0.620 |
| pairwise | 50 | 10 | 1,2,3,5 | 0.807 | 0.629 |
| pairwise | 50 | 20 | 1,2,3,5 | 0.800 | 0.624 |
| pairwise | 300 | 2 | 1,2,3,5 | 0.822 | 0.6560 |
| pairwise | 300 | 5 | 1,2,3,5 | 0.826 | 0.654 |
| pairwise | 300 | 10 | 1,2,3,5 | 0.834 | 0.679 |
| pairwise | 300 | 20 | 1,2,3,5 | 0.831 | 0.653 |
| pairwise | 300 | 10 | 1,2 | 0.813 | 0.653 |
| pairwise | 300 | 10 | 2,3 | 0.817 | 0.657 |
| pairwise | 300 | 10 | 3,4 | 0.816 | 0.655 |
| pairwise | 300 | 10 | 4,5 | 0.816 | 0.647 |
| pairwise | 300 | 10 | 9,10 | 0.812 | 0.627 |
| pairwise | 300 | 50 | 1,2,3,5 | 0.820 | 0.629 |

To improve the performance, We tune some hyperparameters and present the result in Table 2. The *Em-dim* is dimension of our pretrained embedding. The *extend-dim* is dimension of our additional feature embedding. The *region-size* is a hyperparameter of convolution filter shape.

## 4 Conclusion

In this paper, we implement an enhanced convolution neural network by extending our word embedding with additional feature, such as overlap and position information. Instead of treating the task as pointwise classfication, we use a pairwise ranking approach with a triplet ranking loss function. The results demonstrate pairwise ranking approach is more suitable for NLPCC DBQA task than pointwise. We utilize the max pooling and attentive pooling network with dynamic negative sample strategy. In the future, we will add more features to our convolution neural network to improve the performance on DBQA task.

## References

1. Y. Kim, “Convolutional neural networks for sentence classification,” *arXiv preprint arXiv:1408.5882*, 2014.
2. B. Hu, Z. Lu, H. Li, and Q. Chen, “Convolutional neural network architectures for matching natural language sentences,” in *Advances in neural information processing systems*, 2014, pp. 2042–2050.
3. W. Yin, H. Schütze, B. Xiang, and B. Zhou, “Abcnn: Attention-based convolutional neural network for modeling sentence pairs,” *arXiv preprint arXiv:1512.05193*, 2015.
4. A. Severyn, “Automatic feature engineering for answer selection and extraction,” in *EMNLP*, 2013.
5. W. T. Yih, M. W. Chang, C. Meek, and A. Pastusiak, “Question answering using enhanced lexical semantic models,” in *Meeting of the Association for Computational Linguistics*, 2013, pp. 1744–1753.
6. L. Yu, K. M. Hermann, P. Blunsom, and S. Pulman, “Deep learning for answer sentence selection,” *arXiv preprint arXiv:1412.1632*, 2014.
7. A. Severyn and A. Moschitti, “Learning to rank short text pairs with convolutional deep neural networks,” in *SIGIR*. ACM, 2015, pp. 373–382.
8. J. Fu, X. Qiu, and X. Huang, “Convolutional deep neural networks for document-based question answering.” in *NLPCC/ICCPOL*, 2016, pp. 790–797.
9. Y. Yang, W.-t. Yih, and C. Meek, “Wikiqa: A challenge dataset for open-domain question answering.” in *EMNLP*, 2015, pp. 2013–2018.
10. B. Wang, J. Niu, L. Ma, Y. Zhang, L. Zhang, J. Li, P. Zhang, and D. Song, “A chinese question answering approach integrating count-based and embedding-based features,” 2016.
11. A. Severyn and A. Moschitti, “Modeling relational information in question-answer pairs with convolutional neural networks,” 2016.
12. C. D. Santos, M. Tan, B. Xiang, and B. Zhou, “Attentive pooling networks,” 2016.
13. J. L. Jinfeng Rao, Hua He, “Noise-contrastive estimation for answer selection with deep neural networks,” in *CIKM*, 2016.
14. T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in *Advances in neural information processing systems*, 2013, pp. 3111–3119.