Convolutional Deep Neural Networks for Document-based Question Answering

Jian Fu, Xipeng Qiu, Xuanjing Huang

School of Computer Science, Fudan University
825 Zhangheng Road, Shanghai, China
{12307130136,xpqiu,xjhuang}@fudan.edu.cn

Abstract. Document-based Question Answering, particularly Semantic Matching here, aims to compute the similarity or relevance between two documents. It's a typical and core task in NLP and considered as a touchstone of natural language understanding. Deep learning approaches have achieved a lot of success in many research due to its ability to automatically learn optimal feature representations for a given task , including modeling sentence pairs. In this article, we present a convolutional neural network architecture to learn feature representations of each pair of query and document and compute their match score then. By taking the interaction and attention between query and document into consideration, as well as word overlap indices, the empirical study on Chinese Open-Domain Question Answering Task (document-based) demonstrates the efficacy of the proposed model.

1 Introduction

Semantic Matching is a critical task for many applications in NLP. It aims to model sentences and then compute the similarity or relevance. And it's central to many tasks such as question answering ([3]), answer sentence selection ([22]), information retrieval, paraphrase identification and textual entailment [15, 16].

Modelling natural language sentences is very basic and essential in QA Task as natural language sentences usually have complicated structures, both sequential and hierarchical, that make them hard to understand. Among neural network models, we can use LSTM(long short-term memory, [9]), a improved version of Recurrent Neural Network, or Convolutional Neural Network (CNN, [14]) in this paper to model sentences and sentence pairs. Moreover, Bidirectional LSTM is a alternative ([8]). CNN is supposed to be good at extracting robust and abstract features of input and it's capable of local selecting ([13]), so we choose to use CNN to model sentence in this article as it's easy to do modification as well([11, 20]).

A successful sentence-matching algorithm need to not only primely model the internal structures of natural language sentences but also the interaction between them. The match score is supposed to be more accurate if the rich patterns in each pair of sentences can be well exploited. So based on a simple sentence pair architecture like ARC-I([10]), a proposed way to reach this target is to align the two sentences word by word just like ARC-II([10]) or do similarity matching ([19]) to compute a similarity unit that taken into consideration later. Apart from the alignment and similarity match, attention mechanism, a powerful mechanism in Neural Machine Translation([1]), can also be applied for modelling sentence pair by taking into account the interdependence between the two sentences([18]) since attention-based CNNs have been used in computer vision for visual question answering ([6]). By the way, attention mechanism and similarity match can be put into the top or bottom or both top and bottom layer of the neural network([21]). Also, semantic matching with attention has variations like [7].

Besides these, additional features which not require external knowledge sources make contributions to the performance of neural network (see experiment results), such as word overlap indices([22]) and IDF-weighted word overlap indices.

The contribution of this work lies in three folds:

1. We propose a CNN-based Semantic Match Architecture, which can not only model internal structure of sentence separately through layer-by-layer convolution and pooling, but also capture the rich matching patterns between query and document.
2. We perform empirical study on DBQA Task and put additional word overlap features which is computed straightforward without pre-processing or external resources into the neural network to improve the performance.
3. We apply attention mechanism to neural network and do evaluation by visualizing the attention matrices.

2 Convolutional Sentence Model

The convolutional sentence model of our ConvNet is shown on Fig. 1. It's inspired by many convolutional models([11]), but the goal of our distributional model is to learn good intermediate feature representations for each pair of query and document, which are then used for semantic match. As illustrated, it takes input as the words embeddings in the sentence alignment sequentially, and then distill the meaning of each sentence through several layers(or just a single layer) of convolution and max-pooling, reaching a fixed length vectorial representation which is their final intermediate feature representations. To be mentioned, there could be many filters and its number is a hyperparameter to be tuned.

In general, our sentence model consists of a single convolution layer which followed by non-linearity layer and max-pooling layer. There are 4 layers in detail as below and we now describe each in turn.

2.1 Embedding layer

The input of the network is a sequence of words : $[w_1, ..., w_l]$, and each word is derived from a vocabulary $\mathbf{V}$. Words are represented by distributional vectors $\mathbf{w} \in \mathrm{R}^{\mathrm{d}}$ which are drawn from a word embedding matrice $\mathbf{W} \in \mathrm{R}^{|\mathrm{V}| \times \mathrm{d}}$, a pre-trained word2vec([17]) embedding.

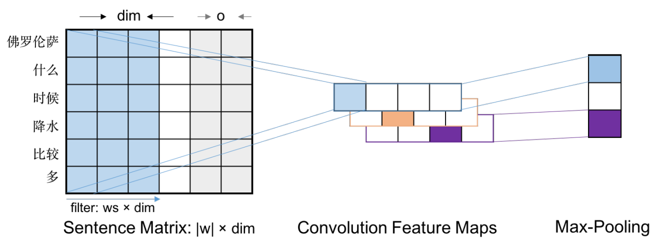

Fig. 1. The overall architecture of the convolution sentence model for mapping input sentences to intermediate feature representations.

Besides word2vec embedding, word overlap features is also added to the dimension of each word of a sentence which indicates essential matching information in Flatten layer and future match score computing(see Model Architecture in Fig. 1). For each question and answer, we get sentence matrix $S \in \mathrm{R}^{\mathrm{L}\times(\mathrm{d}+1)}$.

2.2 Convolution layer

The aim of convolution layer is to extract features or patterns. Given the sequence $q^{emb} = r^{w_1}, ..., r^{w_l}$, let us define the matrix $Z_q = [z_1, ..., z_l]$ as a matrice where each column contains a vector $\mathbf{z_i} \in \mathrm{R}^{\mathrm{d}w_s}$ which is the concatenation of a sequence of w_s word embeddings. The output of the convolution with c filters over the question q is computed as follows:

$$Q = WZ_q + b \tag{1}$$

where each row m in $\mathrm{Q} \in \mathrm{R}^{\mathrm{l}\times\mathrm{c}}$ contains features extracted in a context window around the m_{th} word of q. The matraces W and the vector b are parameters which to be learned. The number of convolution filters c, and the length or size of the word-level context window w_s are hyper-parameters that need be chosen manually by the user.

We then compute A in a similar manner.(the neural network parameters can be the same or not)

2.3 Non-linearity layer

To make the network enable to learn non-linear decision boundaries which make the representations to extract the features better, each convolution layer is followed typically by a non-linear activation function $\alpha()$. To be mentioned, it's applied element-wise to the output of preceding layer. In this paper, activation function is hyperbolic tangent *tanh* by default.

2.4 Max-pooling layer

The output of the convolution layer (after processed by the activation function) are then passed to a pooling layer that aggregate the information, and also, reduce the representation. Max pooling and average pooling, are commonly used to extract robust features from convolution. In this paper, we use max pooling, which selects the max of each filter, to extract patterns for future semantic matching.

3 Convolutional Matching Model

The sentence-pair matching model of our model is presented in Fig. 2. Our ConvNets-Based sentence models (described above), learn to map the input sentence pairs to vectors, which can proceed to be computed their similarity. It mostly take the conventional approach: firstly, it finds the representation matrice of each sentence, then deal with the representations for the two sentences with a multi-layer perceptron (MLP) ([2]), the Siamese architecture introduced in [4, 13].

However, different from previous work, we can compute more interaction between two sentences such as query-document similarity score or attentive pooling(see later).

In the following, we described how to computer similarity score by using the intermediate feature representations and some other remaining layers.

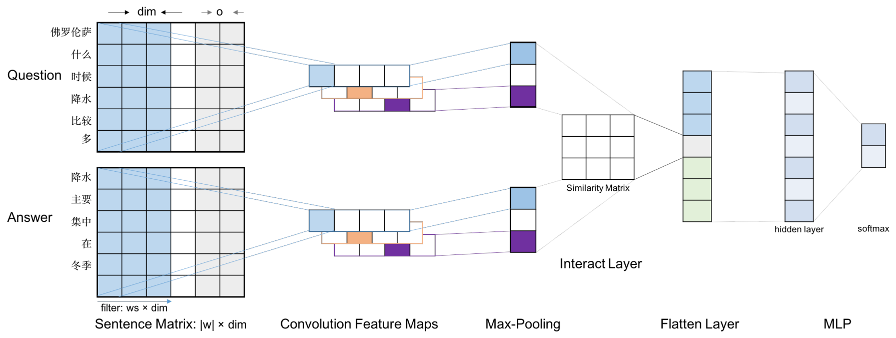

Fig. 2. Illustration of our whole Convolutional Deep Neural Network for Semantice Match.

3.1 Interact layer

Given the representations of our convNets after processing queries and documents, the resulting vector representations x_q and x_d, can be used to compute a similarity score between query-document pairs. Following the approach of [5], the similarity unit is defined as follows:

$$sim\_score(x_q, x_d) = x_q M {x_d}^T, \tag{2}$$

where $\mathrm{M} \in \mathrm{R}^{\mathrm{d}\times\mathrm{d}}$ is a similarity matrix and is optimized during the training.

3.2 Multi-layer perceptron

Multi-layer perceptron(MLP) is composed of a hidden layer and a logistic regression(softmax layer). Hidden layer computes the following transformation:

$$f(w_h x + b), \tag{3}$$

where w_h is the weight vector of the hidden layer and f is the non-linearity activation function. After then, we apply a logistic regression to the vector, the softmax function computes the probability distribution over the labels:

$$p(y = j|x) = \frac{e^{y_j}}{\Sigma e^{y_k}}, \tag{4}$$

3.3 Training

The model is trained to minimise the cross-entropy cost function:

$$L(y, o) = -\frac{1}{N} \Sigma_{n\in N} y_n \mathrm{log} o_n \tag{5}$$

4 Attentive Pooling

Attentive pooling [6, 18, 21] is an approach which enables the pooling layer to be sensitive to the current input pair, in a way that the information from question q can directly influence the representation of the answer a $\mathrm{r_a}$, and vice versa. It's an attention mechanism for model training to discriminate sentence pairs that can model sentence pairs into a common representation space where they can be computed and compared in a more plausible way.

In Fig. 3, we illustrate the application of attentive pooling over the output of the convolution layer to construct the representations $\mathrm{r_q}$ and $\mathrm{r_a}$. After we compute the representation matrices $\mathrm{Q} \in \mathrm{R}^{\mathrm{c}\times\mathrm{L_q}}$ and $\mathrm{A} \in \mathrm{R}^{\mathrm{c}\times\mathrm{L_a}}$ by convolution(weights shared), we compute the attention matrix $\mathrm{G} \in \mathrm{R}^{\mathrm{L_q}\times\mathrm{L_a}}$ as follows:

$$G = tanh(QUA^T) \tag{6}$$

where $\mathrm{U} \in \mathrm{R}^{\mathrm{c}\times\mathrm{c}}$ is a matrix of parameters which to be learned by the NN. After the convolution is used to compute the representations of Q and A, the matrix G contains the scores of an alignment between the w_s-size context windows of q and a after convolution.

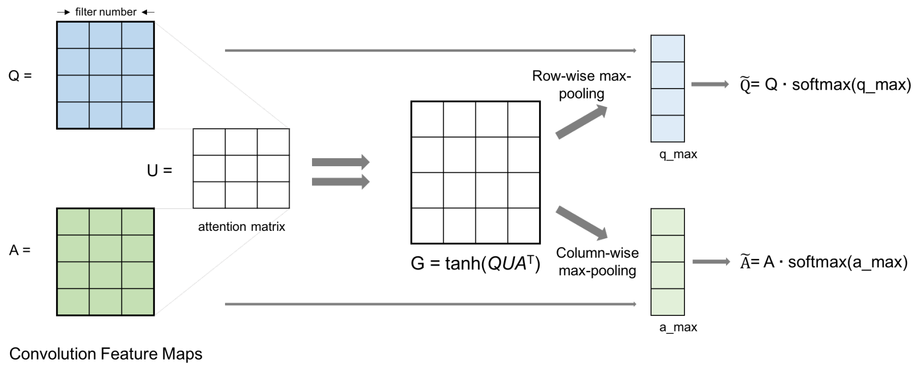

Fig. 3. Attentive pooling architecture which applys attention mechanism right after convolution layer.

Next, we apply row-wise and column-wise max-pooling over G to generate the vectors $g^q \in \mathrm{R}^{\mathrm{L_q}}$ and $g^a \in \mathrm{R}^{\mathrm{L_a}}$, respectively. And we apply the softmax function to the vectors g^q and g^a to create attention vectors σ_q and σ_a.

Finally, the representations of r^q and r^a are computed as dot product between the attention vectors σ_q and σ_a and the output of the convolution over q and a, respectively:

$$r^q = Q\sigma_q \tag{7}$$

$$r^a = A\sigma_a \tag{8}$$

5 Experiments

5.1 Dataset

We conduct experiments on Chinese Open-Domain Question Answering Task(document-based). The dataset is shown in Tab. 1. The QA-pairs of dbqa-train and dbqa-test is 181882 and 122531. After segmentation, the length of most question sentences is less than 20 and the one of most answer sentences is less than 40.

Table 1. Dataset of NLPCC DBQA Task, 2016

Dataset	QA-pairs
DBQA-train	181882
DBQA-test	122531

5.2 Evaluation metrics

DBQA Task is formalized as a ranking problem. Therefore, we use Mean Reciprocal Rank (MRR) as evaluation metrics, but accuracy in validation.

5.3 Embedding

We use word2vec to train word embeddings on corpus Chinese Wikipedia (zh-wiki), which contain more than 230 thousand Chinese articles. The pre-trained word embedding is much of importance to this task.

5.4 Results and discussion

Our experiments results are illustrated as Tab. 2. The table shows the performance of the baseline(CNN), and the improvement of interact layer and additional features. The interaction and word overlap indices play important roles in the performance. And additionally, attentive pooling(with word overlap features) gives a comparable performance. Related research showed that adding attention mechanism before or after convolution layer may have better performance for longer sentences. We didn't tune the hyperparameters very much on both neural networks and we'll do more research on it.

Table 2. Experimental results on the NLPCC-DBQA Task

Model	MAP	MRR
CNN_base	36.41	36.42
+interact	59.14	59.21
+interact&overlap	85.86	85.92
AttentivePooling	84.90	84.96

5.5 Attentive pooling Visualization

The Fig. 4 depict an attention heat map of a sample of QA-pair which is correctly answered by the NN. The darker of the color of a word in the question (answer), the larger the attention score in $\sigma_q(\sigma_a)$ of the trigram centred at that word. As shown in the pictures, the attentive pooling mechanism indeed puts more focus on the segments of the answer which have some relation or interaction with the question, and vice-verse.

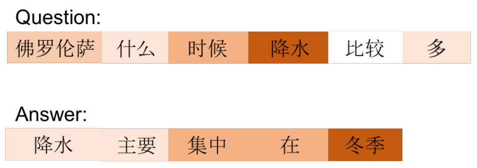

Fig. 4. Attention heat map of a sample of QA-pair that correctly answered by the Neural Network

6 Conclusion

In this paper, we proposed a deep convolutional architecture for sentence semantic matching, which can nicely combine the hierarchical modelling of individual sentences and the patterns of their matching. Empirical study on DBQA Task shows that our model can perform well with word overlap features. Additionally, we apply attention mechanism to it and do evaluation by visualizing the attention matrices.

References

1. Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.
2. Yoshua Bengio. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.
3. Adam Berger, Rich Caruana, David Cohn, Dayne Freitag, and Vibhu Mittal. Bridging the lexical chasm: statistical approaches to answer-finding. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 192–199. ACM, 2000.
4. Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. A semantic matching energy function for learning with multi-relational data. *Machine Learning*, 94(2):233–259, 2014.
5. Antoine Bordes, Jason Weston, and Nicolas Usunier. Open question answering with weakly supervised embedding models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 165–180. Springer, 2014.
6. Kan Chen, Jiang Wang, Liang-Chieh Chen, Haoyuan Gao, Wei Xu, and Ram Nevatia. Abc-cnn: An attention based convolutional neural network for visual question answering. *arXiv preprint arXiv:1511.05960*, 2015.
7. Yiming Cui, Zhipeng Chen, Si Wei, Shijin Wang, Ting Liu, and Guoping Hu. Attention-over-attention neural networks for reading comprehension. *arXiv preprint arXiv:1607.04423*, 2016.
8. Cícero Nogueira dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. Attentive pooling networks. *CoRR*, abs/1602.03609, 2016.
9. Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

10. Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. Convolutional neural network architectures for matching natural language sentences. In *Advances in Neural Information Processing Systems*, pages 2042–2050, 2014.
11. Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.
12. Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.
13. Yann LeCun and Yoshua Bengio. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.
14. Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
15. Pengfei Liu, Xipeng Qiu, Jifan Chen, and Xuanjing Huang. Deep fusion LSTMs for text semantic matching. In *Proceedings of Annual Meeting of the Association for Computational Linguistics*, 2016.
16. Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. Modelling interaction of sentence pair with coupled-lstms. *arXiv preprint arXiv:1605.05573*, 2016.
17. Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.
18. Cicero dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. Attentive pooling networks. *arXiv preprint arXiv:1602.03609*, 2016.
19. Aliaksei Severyn and Alessandro Moschitti. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 373–382. ACM, 2015.
20. Zhiguo Wang, Haitao Mi, and Abraham Ittycheriah. Sentence similarity learning by lexical decomposition and composition. *arXiv preprint arXiv:1602.07019*, 2016.
21. Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *arXiv preprint arXiv:1512.05193*, 2015.
22. Lei Yu, Karl Moritz Hermann, Phil Blunsom, and Stephen Pulman. Deep learning for answer sentence selection. *arXiv preprint arXiv:1412.1632*, 2014.