Double-Channel Multi-layer Information Fusion for Text Matching

Guoxi Zhang, Yongquan Dong, Huafeng Chen

Double-Channel Multi-layer Information Fusion for Text Matching

Guoxi Zhang[1,2], Yongquan Dong[1,2] *, and Huafeng Chen[1,2]

[1] School of Computer Science and Technology, Jiangsu Normal University**
[2] Xuzhou Engineering Research Center of Cloud Computing

Abstract. Text matching is one of the fundamental tasks in natural language processing. Most of the existing models focus only on encoding the text itself but ignore other semantic information which may further improve matching accuracy. In this paper, we propose a novel model for text matching with double-channel multi-layer information fusion. It treats text and part-of-speech information of words in a sentence as double-channel information which is fused by multi-layer interactions. Meanwhile, our model uses a Siamese network structure to learn common and unique features of two sentences, which can improve its ability to learn the relationship between two sentences while reducing the parameter size and complexity. Experimental results on SNLI dataset show that our model can achieve better performance than baseline methods.

Keywords: Text matching · Neural network · Information fusion · Siamese net.

1 Introduction

With the continuous development of the Internet, big data and artificial intelligence, modern lifestyles are gradually becoming intelligent and automated. Accurate and rapid matching of semantic similarity between two texts has a profound impact on intelligent search, intelligent marking, intelligent translation and other applications[4]. Currently, text matching research has been applied to many fields, such as information retrieval, interpretation recognition and automatic question-answering tasks. The research on text matching has important theoretical significance and practical value for the further development of these fields[6].

Most traditional matching models focus only on the encoding of the text itself, but it is difficult for these approaches to take into account both local and global information. The models that can be achieved are generally large in scale, with high training consumption, and the original semantic information is lost after encoding. Therefore, how to represent text more effectively and make it more suitable for text matching is a key problem to be solved. In this paper,

* Corresponding author. E-mail: tomdyq@163.com. Phone number: +8615152816579.
** No.101 Shanghai Rd, Tongshan District, Xuzhou, Jiangsu, China. 221006

we embed the part-of-speech tag vector similarly, so that our model can obtain more input to improve the matching accuracy. At the same time, we explore and propose a new information fusion method due to the different semantic relations between the part-of-speech and the words themselves. Afterwards, we perform a series of tests to compare the accuracy of our model with that of the baseline model.

2 Related works

For text matching problem, deep learning models are one of the mainstream solutions, which can be divided into representation-based matching models and interaction-based matching models.

The representation-based matching models learn the representation of sentences A and B separately and then obtain the matching by defining the matching function, such as vector dot product, Euclidean distance, etc. The whole representation learning framework is a double tower structure, which means two sentences are processed individually. A classical representation-based matching model generally has three layers, respectively, input layer, representation layer and matching layer.

Typical representation-based matching models are as follows:

- DSSM[5] is a DNN-based model. The model uses word-hashing for encoding the two sentences at the input layer, and word-bag at the representation layer. Then the matching layer is used to calculate the vector distance between the two sentences, and finally obtain the matching score. The significance of this model lies in the three-layer paradigm of input-representation-matching.
- ARC-I[3] uses convolution and pooling as the representation layer based on DSSM to capture word order information in the sentence. Therefore, the representations can capture the word order information better than DSSM. However, the pooling is carried out in the local window, so the global information cannot be obtained to some extent.
- CNN-DSSM[9] adds word-trigrams in the input layer to extract the local information of word order, compared with DSSM. The convolution of the representation layer adopts the method of TextCNN to capture the context information of sentences A and B through the convolutional sliding window of $n = 3$. The maximum pooling can obtain the maximum value of each feature map extracted by the convolution, thus capturing the global context information to a certain extent. Compared with ARC-I, this model can maximize the pooling operation of the whole sentence in each feature map and obtain the global relationship.

Compared with the representation-based matching models, the interaction-based matching models do not directly learn the representation of sentences A and B. Instead, it first interacts the two sentences, then extracts the features through the interactive matching information, and finally learns the extracted

matching information with various network structures through integration to get the final matching score. The matching process of this model can be roughly divided into two steps: interaction and aggregation. The biggest difference between an interaction-based model and a representation-based model is that sentences A and B interact ahead of time, so most of the work of the model focuses on how to design the interaction between sentences A and B.

Typical interaction-based matching models are as follows:

- ARC-II[3] extracts the word vectors obtained by the convolution of N-gram in sentences A and B, and carried out element-wise calculation to obtain a matching matrix. Compared with ARC-I, it uses the matching matrix of sentences A and B text at the beginning, and obtains the interaction information of both in advance. It has a better ability to capture the information matching, and the convolution and pooling process retain the order information.
- ESIM[2] is an enhanced version of LSTM. It achieves better results through detailed sequential network design and considering both local inference and global inference. Two kinds of LSTM are used to extract and encode the vectors a into weighted vectors a' in the local inference. After that, it performs element-wise multiplication and subtraction with the original vector, and 4 groups of vectors are concatenated into one group with shape $(a', a, a' - a, a' \otimes a)$, which is equivalent to enrich the extracted information.
- BiMPM[10] regards two sentences as a bilateral relationship with consideration of the relationship both from A to B and from B to A of two sentences in the matching process. Therefore, four different attention methods are used to reflect the multi-perspective thought. However, the network structure is complex and has a lot of computation, which is slow for large-scale text matching computation.

3 Model

In this section, we present Double-Channel Multi-layer Information Fusion Model, i.e. DMIF for text matching.

The input of our model is a batch of tuples (w_a, w_b, l), where w_a and w_b denote sequences of words in two sentences, and l denotes the true matching label of two sentences. The goal of our model is to predict sentence-pairs' relation and divide them into certain classes.

As our model needs to get part-of-speech information, the dataset needs to be preprocessed. Specifically, it takes out the corresponding part-of-speech tags of each word in w_a and w_b, and combines them with original data. The combined data structure is of the shape $(w_a, w_b, pos_a, pos_b, l)$. Besides, there are usually numerous unknown words in the test set, so we generate the sub-word of the unknown words through N-gram, and try to embed the known words in the sub-words and average them, which can reduce the number of unknown words and further improve the matching performance.

3.1 Framework

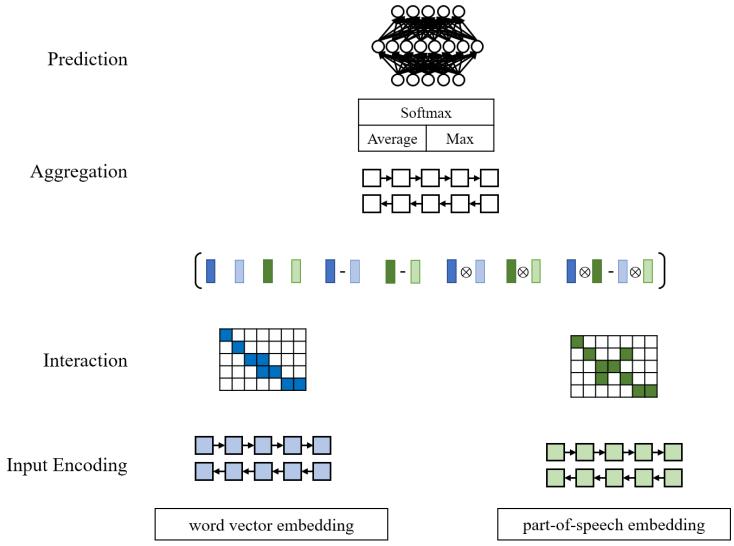

Fig. 1. Framework of Double-Channel Multi-layer Information Fusion Model

Fig. 1 is an overview of our model framework. It is an interaction-based model. Specifically, it can be divided into four layers from bottom to top: input encoding layer, interaction layer, aggregation layer and prediction layer. We will introduce our model more detailedly in the next subsections.

3.2 Input Encoding

Word-vector Encoding. We encode words in the sentences as vectors by GloVe embedding[8]. Then, we use bidirectional LSTM (BiLSTM) as the fundamental block of our model to represent each word in the input sequences.

$$\begin{aligned} \hat{a}_i^w &= \mathbf{BiLSTM}(w_a, i), \forall i \in [1, \ldots, l_a], \\ \hat{b}_i^w &= \mathbf{BiLSTM}(w_b, i), \forall i \in [1, \ldots, l_b]. \end{aligned} \tag{1}$$

where $\hat{a}_i^w$ is the i-th word's hidden state over the input sequence w_a generated by BiLSTM. $\hat{b}_i^w$ is generated similarly. l_a and l_b are lengths of the two sentences, similarly hereinafter.

Part-of-speech-vector Encoding. Our model encodes the part-of-speech tag of each word in the two input sequences by a similar method with word-vector

encoding, and the results are also involved in the subsequent processing together with the word-vector encoding.

$$\begin{aligned} \hat{a}_i^p &= \mathbf{BiLSTM}(pos_a, i), \forall i \in [1, \ldots, l_a], \\ \hat{b}_i^p &= \mathbf{BiLSTM}(pos_b, i), \forall i \in [1, \ldots, l_b]. \end{aligned} \tag{2}$$

where $\hat{a}_i^p$ is the i-th word's hidden state over the input sequence pos_a generated by BiLSTM. $\hat{b}_i^p$ is generated similarly as well.

Since the semantic features of word vectors and part-of-speech tags are often different, it is not possible to construct a single-pipeline network. Therefore, we decided to set two different input coding layers, and the two coding layers have similar structures. In order to capture the similarities between the two sentences and reduce the size of our model, we use Siamese structure to process the input data, which means the network shares parameter values when processing both sequences.

3.3 Interaction

In the interaction layer, in order to enhance the information, we use the attention mechanism used in the ESIM model[2], which balances attention on the bidirectional sequential encoding of the input.

Word Interaction. In particular, we use Equation (3) to compute the similarity of the implicit state tuples $(\hat{a}^w, \hat{b}^w)$ and $(\hat{a}^p, \hat{b}^p)$ between two sentences to calculate the concern weight.

$$\begin{aligned} e_{ij}^w &= (\hat{a}_i^w)^T \hat{b}_j^w \\ e_{ij}^p &= (\hat{a}_i^p)^T \hat{b}_j^p \end{aligned} \tag{3}$$

where e_{ij}^w and e_{ij}^p are attention weights to reflect the similarity of hidden state tuple $(\hat{a}_i^w, \hat{b}_j^w)$ and $(\hat{a}_i^p, \hat{b}_j^p)$, respectively.

Sequence Interaction. Word interaction is determined by the previously calculated attention weight e_{ij}, which is used to obtain a local correlation between two sentences. For the hidden state of the word in one sentence, that is, $\hat{a}_i$, which has encoded the word itself and its context, e_{ij} is used to identify and combine the relevant semantics in another sentence, as shown in Equation (4) and (5).

$$\begin{aligned} \tilde{a}_i &= \sum_{j=1}^{l_b} \frac{\exp\left(e_{ij}^w\right)}{\sum_{k=1}^{l_b} \exp\left(e_{ik}^w\right)} \hat{b}_j^w, \forall i \in [1, \ldots l_a] \\ \tilde{b}_j &= \sum_{i=1}^{l_a} \frac{\exp\left(e_{ij}^w\right)}{\sum_{k=1}^{l_a} \exp\left(e_{kj}^w\right)} \hat{a}_i^w, \forall j \in [1, \ldots l_b] \end{aligned} \tag{4}$$

$$\begin{aligned}\tilde{a}_i^p &= \sum_{j=1}^{l_b} \frac{\exp\left(e_{ij}^p\right)}{\sum_{k=1}^{l_b} \exp\left(e_{ik}^p\right)} \hat{b}_j^p, \forall i \in [1, \ldots l_a] \\ \tilde{b}_j^p &= \sum_{i=1}^{l_a} \frac{\exp\left(e_{ij}^p\right)}{\sum_{k=1}^{l_a} \exp\left(e_{kj}^p\right)} \hat{a}_i^p, \forall j \in [1, \ldots l_b]\end{aligned} \tag{5}$$

We expect that such an operation will help improve the information on local and sequential interactions between elements in a tuple to capture the relationship between two sentences better. To make our model further enhance the interaction information collected above, we use difference and element-wise products in series with the pre-interacted and post-interacted vectors $\hat{a}^w$ and $\tilde{a}^w$ or $\hat{b}^w$ and $\tilde{b}^w$ respectively, to enhance the interaction information. We deal with part-of-speech tag features, $\hat{a}^p$ and $\tilde{a}^p$ or $\hat{b}^p$ and $\tilde{b}^p$, in similar way, as shown in Equation (6).

$$\begin{aligned}m_a &= [\tilde{a}^w, \hat{a}^w, \tilde{a}^p, \hat{a}^p, \tilde{a}^w - \hat{a}^w, \tilde{a}^p - \hat{a}^p, \tilde{a}^w \otimes \hat{a}^w, \tilde{a}^p \otimes \hat{a}^p, \tilde{a}^w \otimes \tilde{a}^p - \hat{a}^w \otimes \hat{a}^p] \\ m_b &= \left[\tilde{b}^w, \hat{b}^w, \tilde{b}^p, \hat{b}^p, \tilde{b}^w - \hat{b}^w, \tilde{b}^p - \hat{b}^p, \tilde{b}^w \otimes \hat{b}^w, \tilde{b}^p \otimes \hat{b}^p, \tilde{b}^w \otimes \tilde{b}^p - \hat{b}^w \otimes \hat{b}^p\right]\end{aligned} \tag{6}$$

3.4 Aggregation

In order to determine the overall relationship between the two sentences, we use the aggregation layer to combine the interaction information of the two sentences enhanced by the aforementioned processing. We use BiLSTM to perform the aggregation in sequence according to Equation (7).

$$\begin{aligned}\tilde{m}_{a,i} &= \mathbf{BiLSTM}\left(m_a, i\right), \forall i \in [1, \ldots l_a] \\ \tilde{m}_{b,i} &= \mathbf{BiLSTM}\left(m_b, i\right), \forall i \in [1, \ldots l_b]\end{aligned} \tag{7}$$

The aggregation layer converts the resulting vector obtained above into a fixed-length vector by using Equations (8) and (9) to calculate the average and maximum pooling. All these vectors are then concatenated to form the final fixed length vector, which is fed into the final classifier to determine the overall relationship between the two sentences.

$$\tilde{m}_{a,\text{ave}} = \frac{1}{l_a} \sum_{i=1}^{l_a} \tilde{m}_{a,i}, \tilde{m}_{a,\max} = \max_{i=1}^{l_a} \tilde{m}_{a,i} \tag{8}$$

$$\tilde{m}_{b,\text{ave}} = \frac{1}{l_b} \sum_{j=1}^{l_b} \tilde{m}_{b,j}, \tilde{m}_{b,\max} = \max_{j=1} l_b \tilde{m}_{b,i} \tag{9}$$

$$v = [\tilde{m}_{a,\text{ave}}; \tilde{m}_{a,\max}; \tilde{m}_{b,\text{ave}}; \tilde{m}_{b,\max}] \tag{10}$$

3.5 Prediction

The prediction layer is to evaluate the label probability distribution of the two sentences. After the representation of the two sentences is obtained, the label probability distribution of the two sentences can be obtained through the multi-layer perception (MLP) classifier, and set the network size according to the actual situation, as an example shown in the following table 1, where the class corresponding to the maximum probability is the prediction result generated by DMIF.

Table 1. An example of MLP structure

Layer	Output dimension(s)
Fully-connected 1	600
Fully-connected 2	150
Fully-connected 3	3

4 Experimental Setup

4.1 Datasets

The **SNLI corpus**[1] is a corpus composed of 570K manually annotated English sentence pairs, which is balanced by annotated implication, contradiction and neutral classification, and supports natural language reasoning tasks. This is a benchmark dataset that is widely used to evaluate textual representational systems, especially those induced by representational learning methods. Here, we will use this data set to train the model and compare it with the original model.

4.2 Training Hyperparameters

To compare the accuracy between our model and the baseline model, we set the same hyperparameters as that in ESIM[2] model. In the experiments, we use Adam method for optimization. The first and second value of momentum are set to 0.9 and 0.999. The initial learning rate is set to 0.0004. When training, data loader pushes a batch of data with 32 groups of sentences. The hidden layers' dimension are set to 300. The optimal accuracy of 3 Bernoulli tests with 5 epochs of training each time will be used as the basis for evaluating the accuracy of the model.

5 Results and Discussion

5.1 Experimental Results

We use SNLI data set to train and test the model, and use ESIM as baseline. As ESIM uses non-public pre-training data, there are objective differences in

training environment. Therefore, we use the model reproduced by ourselves to set the same hyperparameters for testing under the same hardware environment. The experimental results are shown in the table 2. Although there is a certain gap between the accuracy of our model and the model reported in the paper [2], our model performs better than the re-implement model with 1% relative accuracy improved. Therefore, we believe that our model has better performance under the same conditions. At the same time, it can be found that our model adds a relatively independent channel to input part of speech information, while the model size does not increase exponentially. Therefore, it can be considered that our model does optimize the parameter size to some extent.

Table 2. Accuracies of different models on SNLI.

Model	Num_of_params ($\times 10^7$)	Train acc. (%)	Test acc. (%)
600D ESIM [2] (re-implement)	2.84	93.64	85.43
Siamese ESIM [7] (re-implement)	2.06	95.29	85.91
DMIF	2.92	95.20	**86.34**

5.2 Analysis and Discussion

Compared with the baseline model, we apply more than one method to improve the performance of text-matching mission. To confirm that the optimized model does have a performance improvement effect, we performed ablation tests on the model using a control variable method. The results are shown in the Table 3.

Table 3. Ablation tests on SNLI Dataset.

Model	Test Acc.
ESIM	85.43
ESIM+Siamese network	85.73
ESIM+POS embedding	85.68
DMIF	**86.34**

We also try to add some layers with higher complexity at the prediction layer, but we find that the accuracy did not improve with the increase of model parameters. Specifically, the experimental results are shown in the Table 4 below.

6 Conclusions and Future Work

Compared with the baseline model, the model proposed in this paper can obtain more input information by inputting part-of-speech tag sequence, reduce the number of parameters to be trained by using Siamese structure, and adjust

Table 4. More complex models' test on SNLI Datasets

Model	Num_of_params($\times 10^7$)	Test Acc.
baseline	2.84	85.43
DMIF	2.92	**86.34**
DMIF+resnet34	3.78	85.03
DMIF+resnet50	7.24	84.48

the information aggregation method. Experimental results show that our model performs better in the task of text matching.

In the future, we plan to put prior knowledge or statistical features into the model inputs and propose better feature construction strategies to improve the accuracy of the model. In addition, we will also try to improve the traditional model by strengthening the interaction between layers and across layers with the expectation of reducing the loss of the model as it propagates forward.

7 Acknowledgements

This work is supported by the National Natural Science Foundation of China (No. 61872168) and Graduate Research and Practice Innovation Program of Jiangsu Normal University (No. 2021XKT1380).

References

1. Bowman, S.R., Angeli, G., Potts, C., Manning, C.D.: A large annotated corpus for learning natural language inference. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics (2015)
2. Chen, Q., Zhu, X., Ling, Z.H., Wei, S., Jiang, H., Inkpen, D.: Enhanced lstm for natural language inference. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 1657–1668 (2017)
3. Hu, B., Lu, Z., Li, H., Chen, Q.: Convolutional neural network architectures for matching natural language sentences. In: Advances in neural information processing systems. pp. 2042–2050 (2014)
4. Hu, W., Dang, A., Tan, Y.: A survey of state-of-the-art short text matching algorithms. In: Tan, Y., Shi, Y. (eds.) Data Mining and Big Data. pp. 211–219. Springer Singapore, Singapore (2019)
5. Huang, P.S., He, X., Gao, J., Deng, L., Acero, A., Heck, L.: Learning deep structured semantic models for web search using clickthrough data. In: Proceedings of the 22nd ACM international conference on Information & Knowledge Management. pp. 2333–2338 (2013)
6. Huang, Z., Cao, L.: Deep learning for text matching: A survey. In: 2021 5th Annual International Conference on Data Science and Business Analytics (ICDSBA). pp. 66–70. IEEE (2021)

7. Liu, Y., Liang, X., Ren, F., Li, Y., Hou, Y., Zhang, Y., Pan, L.: An enhanced esim model for sentence pair matching with self-attention. In: CCKS Tasks. pp. 52–62 (2018)
8. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). pp. 1532–1543 (2014)
9. Shen, Y., He, X., Gao, J., Deng, L., Mesnil, G.: A latent semantic model with convolutional-pooling structure for information retrieval. In: Proceedings of the 23rd ACM international conference on conference on information and knowledge management. pp. 101–110 (2014)
10. Wang, Z., Hamza, W., Florian, R.: Bilateral multi-perspective matching for natural language sentences. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence. pp. 4144–4150 (2017)