Siamese Network cooperating with Multi-head Attention for semantic sentence matching

Zhao Yuan
School of Internet of things Engineering
Jiangnan University
Wuxi,China
e-mail: zhaoyuansir@foxmail.com

Sun Jun
School of Internet of things Engineering
Jiangnan University
Wuxi,China
e-mail: junsun@jiangnan.edu.cn

***Abstract*—To compare a pair of sentences is a fundamental technology in many NLP tasks. According to the difference between the pair of sentence, we divide semantic sentence matching into two situations: Situation A is that the pair of sentences are worded with a context relationship, Situation B is that two are equal in semantics. Models for Situation A works in Situation B too, so prior deep work mostly model each sentence's representation considering the interaction of the other sentence simultaneously. However, models designed for Situation A bring redundant information for Situation B. In this paper, for sentence pairs with equivalence, we present a deep architecture with comparison-interaction separated to match two sentences, which based on Siamese network for comparison and multi-head attention for interaction information between sentence pairs. Experimental results on the latest Chinese sentence matching datasets outline the effectiveness of our approach.**

Keywords- Siamese network; Multi-Head Attention; Semantic Matching;

I. INTRODUCTION

The purpose of modeling a pair of sentences is to distinguish the relationship between two sentences by comparing the semantics of them. In different tasks[1-2], we want to divide them into two situations: Situations A is the context relationship in Natural Language Inference, Answer Selection; Situations B is the equal relationship in Paraphrase Identification, Semantic Textual. Sentence pairs in Situations A are different in semantics, but they are related to each other, sentence pairs in Situations B are same in semantics. There is a key difference between the two situations: the percentage of same words in sentences pair in Situation A are higher than B. Table 1 shows examples of both cases.

TABLE 1. DIFFERENT SAMPLES IN SITUATION A AND B

Situations	Tasks	Sentence A	Sentence B
Situation A	*Natural Language Inference*	A soccer game with multiple males **playing**.	Some men are **playing** a sport.
	Answer Selection	How to participate in meetings on online d uring the outbreak?	You can use a cloud-based peer-to-peer software platform called Zoom.
Situation B	*Paraphrase Identification*	What shoud **I do** to avoid **sleeping in class?**	**How do I** not **sleep in** a boring **class?**
	Semantic Textual Similarity	**一般我**要等**几天?**	**一般几天**会联系**我?**

The division of semantic matching tasks into these two situations has never actually been mentioned before. The reason we split two situations is the need for more lightweight and effective models for Situation A. We will consider both the semantic differences and the interaction between the two sentences simultaneously, but the way we use differs from some existing models[3-6] etc. We propose a model which construct the "Siamese" architecture by two bi-directional GRU to encode two sentences in the same embedding space, and then aggregate the two sentences vectors using Jaccard distance. At the same time, we apply a mechanism called Multi-Head Attention to interact the word vectors of the two sentences. Finally we get the final decision by combing the aggregated vectors both from "Siamese" architecture and Multi-Head Attention. We call it SNMA shorted for Siamese Network cooperating with Multi-head Attention. Fig. 1 gives an outline of SNMA based on Siamese network for comparison and multi-head attention for interaction information between sentence pairs.

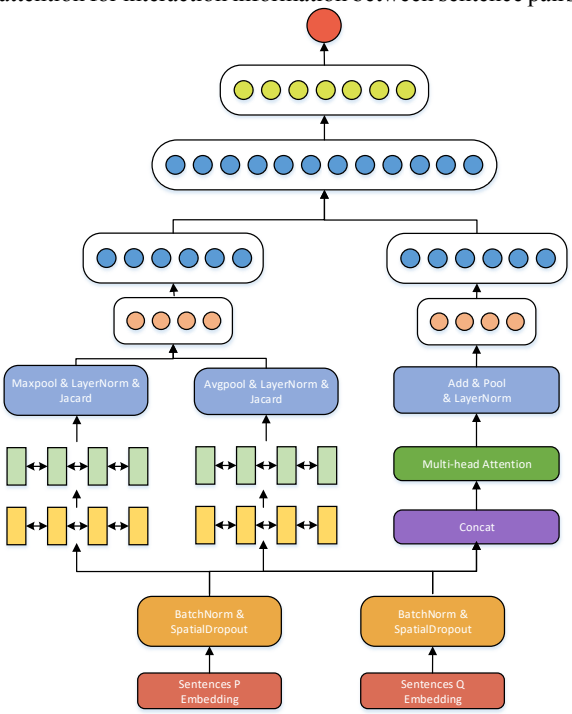

Figure.1 Overview of our approach SNMA: Siamese Network cooperating with Multi-head Attention.

2473-3636/20/$31.00 ©2020 IEEE
DOI 10.1109/DCABES50732.2020.00068

II. OUR APPROACH

In our notation, we have two sentences P = {p1, p2,⋯, pm} and Q = {q1, q2, ⋯, qn} where P is a sentence with m words and where Q is a sentence with n words, the pi or qi will be encoded into an embedding of 300-dimensional vector. The goal is to predict a label y∈{0, 1} that indicates the logic relationship between P and Q, 0 means synonym pairs and 1 means non-synonym pairs.

A. Input layer

In this layer, we encode the sentences with pre-trained vectors and do some works on the vectors.

1) Embedding: Since the datasets are designed for Chinese short text semantic matching task, we can token the sentences into words or characters. It has been proved that the model based on word-level in Chinese datasets is better than that based on character-level [8]..

We use the jieba [9] to do word segmentation for the two Chinese sentences, and we get the pre-trained word vectors on the relevant corpus in advance using Word2Vec [13]. We map each word or character to a high-dimensional vector space using these pre-trained word vectors. We found that trainable embedding layer may led to overfitting during training, so we set the embedding layer fixed. Fixed embedding can reduce the difference of word vectors between train-datasets and test-datasets. Now we have changed two sentences to $V_P \in R^{m\times d}$ and $V_Q \in R^{n\times d}$, where *d* is the dimension of the word vector.We set *d*=300 for both word vectors and character vectors in order to try the mixture of two kinds of vectors.

2) Normalization: The representation P and Q is passed to a Batch Normalization [10] layer and a Dropout [11] layer, and the Dropout is SpatialDropout [12] actually which is proposed in image field. SpatialDropout set the elements in same dimension zero. Different dimensions in word vectors represent different semantics, so we can get different combinations of semantic information by zeroing different dimensions of word vectors. We get the representation $V_{\overline{P}} \in R^{m\times d}$ and $V_{\overline{Q}} \in R^{n\times d}$:

$$V_{\overline{P}} = \text{SpatialDropout (BatchNorm } (V_P)) \quad (1)$$

$$V_{\overline{Q}} = \text{SpatialDropout (BatchNorm } (V_Q)) \quad (2)$$

B. Comparison: Siamese Network with Bi-GRU

In this layer, we extract the difference of sentences pairs with 'Siamese' architecture.

1) Encoding: We employ two Bi-GRU to encode the two sentences in order to learn the context around each word and get the representation of each word:

$$h_i^P = \text{BiGRU } (h_{i-1}^P, V_i^{\overline{P}}),\ i = 1,..., M \quad (3)$$

$$h_i^Q = \text{BiGRU } (h_{i-1}^Q, V_i^{\overline{Q}}),\ i = 1,..., M \quad (4)$$

Where $V_i^{\overline{P}}$ denotes the *i*-th word vector of *P* passed through input layer, $V_i^{\overline{Q}}$ denotes the *i*-th word vector of *Q* passed through input layer, h_i^P denotes the *i*-th element vector of *P* passed through two Bi-GRU encoding, h_i^Q denotes the *i*-th element vector of *Q* passed through two Bi-GRU encoding. We add two Bi-GRU encoding vectors for each word to get the final representations.

2) Pooling and Normalization: Pooling layer is used to extract features and minimize the dimensions. Considering the effect of sentence's length on summation mechanism and the robustness of SNMA, both global average pooling and global max pooling are used at the same time. We can get the vectors V_{avg}^P, V_{max}^P, V_{avg}^Q, V_{max}^Q:

$$V_{avg}^P = \sum_{i=0}^{m} \frac{V_i^P}{m} \quad (5)$$

$$V_{max}^P = max_{i=0}^{m} V_i^P \quad (6)$$

$$V_{avg}^Q = \sum_{i=0}^{n} \frac{V_i^Q}{n} \quad (7)$$

$$V_{max}^Q = max_{i=0}^{n} V_i^Q \quad (8)$$

Where V_i^P denotes the *i*-th word vector of *P* passed through the upper layer, V_i^Q denotes the *i*-th word vector of *Q* passed through the upper layer; V_{avg}^P represents the sentence vector of *P* after global average pooling, V_{max}^P represents the sentence vector of *P* after global max pooling; V_{avg}^Q represents the sentence vector of *Q* after global average pooling, V_{max}^Q represents the sentence vector of *Q* after global max pooling.

Then we adopt a LayerNormalization [16] to optimize easily and get a better result. With Layer Normalization, the neuron inputs of same layer have the same mean and variance; while with Batch Normalization, the neuron inputs of same batch have the same mean and variance. Due to Layer Normalization being insensitive to batchsize and the length of a sequence, we can get better result instead Batch Normalization. The LayerNormalization formula is as follows:

$$u_i = \frac{1}{m}\sum_{j=1}^{m} x_{ij} \quad (9)$$

$$\sigma_i^2 = \frac{1}{m}\sum_{j=1}^{m} (x_{ij} - u_i)^2 \quad (10)$$

$$\hat{x_{ij}} = \frac{x_{ij} - u_i}{\sqrt{\sigma_i^2 + \varepsilon}} \quad (11)$$

3) Aggregation: Now we have four sets of pooling vectors. In this layer, we need to aggregate and compare sentences. Instead of using four-arithmetic operations between sentences vectors, we only use element-wise Jaccard distance to measure the distance between sentences vectors in each dimension. The Jaccard distance formula is as follows:

$$\text{f}(V_i^P, V_i^Q) = \frac{\sum x_i y_i}{\sum x_i^2 + \sum y_i^2 - \sum x_i y_i} \quad (12)$$

$$V_{max} = \text{f}(V_{max}^P, V_{max}^Q) \quad (13)$$

$$V_{avg} = \text{f}(V_{avg}^P, V_{avg}^Q) \quad (14)$$

V_{max} represents the aggregated vectors from V_{max}^P and V_{max}^Q calculated by Jaccard distance, V_{avg} represents the aggregated vectors from V_{avg}^P and V_{avg}^Q calculated by Jaccard distance.

We concatenate both V_{avg}^P and V_{avg}^Q as V_{concated}, then we use activation function ReLU [16] on it:

$$V_{\text{merged}} = \text{ReLU}(V_{\text{concated}}) \quad (15)$$

V_{merged} represents the sentences pair aggregation representations, it will be aggregated will another aggregation representations from the block we are going to explain.

C. Interaction: Multi-Head Attention

In this layer, we extract the interactive features of sentences pairs with Multi-Head Attention.

1) Encoding: As shown in Fig. 1, we concatenate $V_{\overline{P}} \in R^{m\times d}$ and $V_{\overline{Q}} \in R^{n\times d}$ as $V_{text} \in R^{(m+n)\times d}$, then we pass it through a multi-head attention[16] to capture the interaction between sentences pair. The attention we applied is 'Scaled Dot-Product Attention'*:*

$$\text{Attention}(Q,K,V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \quad (16)$$

$$\text{Attention}(V_{text}, V_{text}, V_{text}) = \text{softmax}(\frac{V_{text}V_{text}^T}{\sqrt{d_k}})V_{text} \quad (17)$$

where Q,K,V represent query, key and value respectively, d_k is a hyperparameter. We use the self-attention to encode the concatenated sentences, so Q,K,V all equal to V_{text}. Multi-head attention can use different attention to represent different locations, get better semantic information, and prevent overfitting effectively:

$$V_{inter} = \text{Multi-Head}(Q,K,V) = \text{Concat}(\text{head}_1,\text{head}_2,\ldots,\text{head}_8)W^o \quad (18)$$

$$\text{head}_i = \text{Attention}(QW_i^Q; KW_i^K; VW_i^V) \quad (19)$$

where W_i^Q, W_i^K, W_i^V are all trainable weights.

2) Pooling: Now we have the vectors V_{inter} with interaction of sentences pairs. Similar to the pooling layer of the comparison module, but only one sentence vector is passed through here, we need to reduce the dimensions of V_{inter}. We use both global average pooling and global max pooling, obtain the aggregation representations by concatenating the results of two pooling layers:

$$V_{avg}^{inter} = \sum_{i=0}^{m} \frac{V_i^{inter}}{m} \quad (20)$$

$$V_{max}^{inter} = max_{i=0}^{m} V_i^{inter} \quad (21)$$

$$V_{inter} = [V_{avg}^{inter};\ V_{max}^{inter}] \quad (22)$$

D. Output

We get the final representation of the comparison module and the interaction module by concatenating the V_{merged} and V_{inter}:

$$V_{all} = \text{Concat}(V_{\text{merged}}; V_{inter}) \quad (23)$$

The way we get V_{all} has both the simplicity of the "Siamese" architecture and the explicit interaction of sentence pairs. We calculate the label using a final multilayer perceptron (MLP) classifier cooperating with ReLU activations and a output layer: sigmoid.

E. Loss Function and Metrics

1) Loss Function: In this paper, we use the sigmoid cross-entropy loss for training. The goal is to get the network's parameters matching the minimum cross-entropy between the true and predicted labels:

$$L(\hat{y}, y) = -\frac{1}{N}\sum_x [y \log \hat{y} + (1-y)\log(1-\hat{y})] \quad (24)$$

where $\hat{y}$ is prediction probabilities; y is the ground-truth label. The entire networks are trained end-to-end with backpropagation.

2) Metrics: We evaluate all contests by accuracy and F1-score. Before calculating, we get definitions: True Positive which can be abbrevated to TP, its value denotes the amount of correct synonym judgments; False Positive which can be abbrevated to FP, its value denotes the amount of wrong synonym judgments; TN, short for True Negative, its value means the number of correct non-synonym judgments; FN, short for False Negative, its value means the number of wrong non-synonym judgments. Then we can calculate the accuracy, and F1-score as follows

a) precision rate = TP/ (TP + FP);
b) recall rate = TP / (TP + FN);
c) accuracy = (TP + TN) / (TP + FP + TN+ FN);
d) F1-score = 2 * precision rate * recall rate / (precision rate + recall rate).

The higher accuracy and F1-score indicate the better performance of a model, we use both.

III. EXPERIMENTS

A. Implementation details

We use the jieba [12] for word segmentation, employ Word2Vec [16] to train the embedding vectors of Chinese word and character respectively and each dimension of vector is 300. During training, we keep the embedding layer fixed, while the out-of-vocabulary words are initialized randomly. The SpatialDropout rate is set as 0.2. The numbers of hidden nodes in the first GRU is set to 384, and the numbers in the second GRU is set to 256. There are 8 heads in multi-head attention, and each dimension of attention nodes is 16. We employ ReLU as the activation function and Nadam [17] whose learning rate is 0.0008 as the optimizer for back-propagation. When the F1-score had not improved for 3 epochs, we will try to offer the learning rate 20 percent discount. Besides, early-stopping is applied when the F1-score had not improved for 7 epochs. The batchsize of training stage is 256.

B. Experiments on ATEC Dataset

The dataset is derived from Ant Financial Artificial Competition(ATEC)[7], and all of them came from the actual application conditions of Ant Financial Brain. The contest is about question similarity calculation, and the contestants need to judge whether sentences pairs given by users in customer service have same meanings. The competition provides 100,000 pairs of annotated data as training data, including synonym pairs and non-synonym pairs. Each row in the dataset is a sample. The format is as follows:

TABLE 2 THE SAMPLE OF ATEC DATASET

index	sentence 1	sentence 2	label
1	如何还款花呗	我怎么还我的花被呢	1

TABLE.3 EXPERIMENTALS ON ATEC DATASETS

Model name	#para	Acc of dev	F1 of dev	F1 of test
Match Pyramid	8.1M	81.67	48.32	49.05
Decomposable Attention	2.0M	69.93	40.16	37.71
BCNN	1.6M	78.80	51.30	48.38
ABCNN	2.0M	75.14	44.57	43.36
ESIM	8.4M	81.32	49.89	50.72
SNMA	4.0M	**82.26**	**53.92**	**54.16**

TABLE 4 ABLATION ON THE ATEC DEVELOPMENT DATASET

Ablation Model	F1 of dev
Base model with chinese char	**53.92**
with chinese word instead	41.08
+chinese word	53.48
-max pool	52.45
-avg pool	48.00
-one bilstm layer	51.83
-multi-head attention	53.88

We split these 100,000 sentences pairs into training dataset and test dataset with 9:1 ratio, and then divide the training dataset into new training dataset and development dataset with 8:2 ratio.

The experimental results of different models for ATEC datasets are shown in the table 3. All models do not use additional manual features. In addition to our own model SNMA, the other five models in the table are reproduced from relevant papers that have achieved great results in English datasets in the past. Our model achieves better results on this dataset than other models.

We can see from the ablation on the ATEC development dataset above, the embedding of word vectors we pre-trained is so bad that even we cannot improve the F1-score of the model when we mix the embedding of word and char vectors, although mixxing improves score the of Chinese words. Global average pooling play a more important role than global max pooling in comparison phase in ATEC dataset. As for multi-head attention phase, it works with light contribution due to the reason of splitting Situation A and B we talked before.

IV. CONCLUSION

We proposed a novel model for Chinese semantic sentence matching, the experimental results on the latest Chinese sentence matching datasets outline the effectiveness of our approach. Our model SNMA mainly consists of a comparison and an interaction. The comparison is used to extract the difference of sentences pair and the interaction is applied to extract the interactive features. The model is designed on the fact that Chinese semantic sentence matching datasets with short sentences do not fit the interaction module of complex models completely. However, the disadvantage of "Siamese" architecture reminds us to add the interaction. We build our "Siamese" architecture cooperating with multi-head attention as SNMA. Semantic sentence matching always needs both the difference and interaction of sentences pair.

ACKNOWLEDGMENT

The work described in this paper was fully supported by a grant from the National Key R&D Program of China (No. 2018YFC1603303).

REFERENCES

[1] Zhiguo Wang, Haitao Mi, and Abraham Ittycheriah. Sentence similarity learning by lexical decomposition and composition. In COLING, 2016.

[2] Wenpeng Yin, Hinrich Schutze, Bing Xiang, and Bowen Zhou. ABCNN: Attention-based convolutional neural network for modeling sentence pairs. arXiv preprint arXiv:1512.05193, 2015.

[3] Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, Hui Jiang, Diana Inkpen. Enhanced LSTM for Natural Language Inference. arXiv preprint arXiv: 1609.06038, 2016

[4] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, Xueqi Cheng. Text Matching as Image Recognition. arXiv preprint arXiv: 1602.06359v1 2016

[5] Wang, Zhiguo & Hamza, Wael & Florian, Radu. Bilateral Multi-Perspective Matching for Natural Language Sentences. 4144-4150. 10.24963/ijcai.2017/579. 2017

[6] Parikh, Ankur & Täckström, Oscar & Das, Dipanjan & Uszkoreit, Jakob. A Decomposable Attention Model for Natural Language Inference. 2249-2255. 10.18653/v1/D16-1244. 2016

[7] Ant Financial. Ant Financial Artificial Competition.

[8] Meng, Yuxian & Li, Xiaoya & Sun, Xiaofei & Han, Qinghong & Yuan, Arianna & Li, Jiwei. Is Word Segmentation Necessary for Deep Learning of Chinese Representations? arXiv preprint arXiv: 1905.05526. 2019.

[9] Junyi S. jieba. https://github.com/fxsjy/jieba

[10] Sergey Ioffe, Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift, arXiv:1502.03167 [cs.LG]. 2015

[11] Srivastava N, Hinton GE, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 15(1):1929–1958. 2014

[12] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, Christopher Bregler. Efficient Object Localization Using Convolutional Networks. arXiv preprint arXiv: 1411.4280. 2014

[13] Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean. Efficient Estimation of Word Representation in Vector Space. arXiv preprint arXiv: 1301.3781. 2013

[14] Jimmy Lei Ba, Jamie Ryan Kiros, Geoffrey E. Hinton. Layer Normalization. arXiv preprint arXiv: 1607.06450. 2016

[15] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin. Attention is all your need. arXiv preprint arXiv: 1706.03762. 2017

[16] Glorot X, Bordes A, Bengio Y. Deep sparse rectifier neural networks. In: Proceedings of the 14th International Conference on Artificial Intelligence and Statistics, pp 315–323. 2011

[17] Timothy Dozat. Incorporating nesterov momentum into adam. . *ICLR Workshop*, (1):2013–2016.