Text Semantic Matching Research Based on Parallel Dropout

Zhuangzhuang Li[1,2], Zengzhen Shao[2](✉), Jianxin Xiao[1,2], Zixiao Yu[1,2], and Xu Zhang[1,2]

[1] School of Information Science and Engineering, Shandong Normal University, Jinan 250014, China

[2] School of Data Science and Computer Science, Shandong Women's University, Jinan 250002, China

shaozengzhen@163.com

Abstract. Text semantic matching is a fundamental task in natural language understanding, and has a wide range of applications in information retrieval, question and answer systems, reading comprehension, and machine translation. Currently, a better way to do text semantic matching tasks is to extract word vectors or sentence vectors with BERT and then fine-tuning them, but the commonly used fine-tuning methods suffer from the overfitting problem. We propose a bi-directional long short-term memory-parallel dropout model (BiLSTM-PD), which combines word vectors and sentence vectors to improve feature vectors quality and uses parallel dropout to reduce overfitting. First, word vectors and sentence vectors are generated using the pre-trained model, BiLSTM converts the word vectors into sentence vectors, and the sentence vectors generated by BiLSTM are combined with the sentence vectors generated by the pre-trained model to form the final sentence vectors representation; then, four dropout functions are used to randomly discard a portion of the neurons of the sentence vectors to obtain four subsets, and then a linear layer is used to transform the four subsets of dimensionality and calculate the average value, and then use the Softmax and Argmax functions to calculate the predicted value of each batch to know whether the two sentences are similar. Experiments on two text semantic matching datasets and detailed analyses demonstrate the effectiveness of our model.

Keywords: Text semantic matching · BERT · BiLSTM · Parallel dropout

1 Introduction

Text semantic matching refers to comparing the semantic similarity between two pieces of text to determine whether they have the same meaning or express similar meanings. In recent years, significant progress has been made in the application of pre-trained models for the semantic matching of text pre-trained models can learn linguistic representations by training on large-scale data to provide higher-quality semantic features for downstream tasks. Among them, feature-based matching [21] and interaction-based matching [18] are the two main applications of pre-trained models for text semantic matching.

The feature-based matching method refers to encoding two sentences separately to obtain their sentence vectors representations and then processing these sentence vectors by simple fusion to obtain the final matching result. The feature-based matching approach can effectively avoid the problem of information overload and has good computational efficiency at the same time. However, the independent encoding of two sentences in a sentence pair may lead to the neglect of the interaction information between sentences, which affects the matching accuracy. We usually use interactive matching methods to better use the interaction information between sentences. The interactive matching method refers to splicing two texts together as a single text for classification. This method can obtain richer semantic information and thus improve the matching accuracy. For example, in question-answer systems, the correlation between the question and the text can be better captured by stitching the question and the text together.

In this study, we propose the BiLSTM-PD model. The model combines word vectors and sentence vectors to improve the quality of feature vectors, uses parallel dropout to reduce overfitting, and uses interactive matching methods to obtain richer semantic information. Experiments on two semantic matching datasets show that the present model achieves excellent performance. The contribution of this paper includes three parts:

1. Combining word vectors and sentence vectors to ensure a higher quality representation of feature vectors.
2. Parallel dropout is proposed to reduce overfitting by parallel operations.
3. The results show that the BiLSTM-PD model can improve the performance of semantic matching and can also be easily integrated into other interaction-based text semantic matching models to improve accuracy.

2 Related Work

2.1 Text Semantic Matching

Text semantic matching aims to determine the semantic relationship between two text sequences. In earlier work, researchers mainly used keyword-based matching methods such as TF-IDF [12] and BM25 [11]. These methods rely on manually defined features and often fail to assess the semantic relevance of the text. With the development of deep learning techniques, researchers have started to propose various neural models to solve the text semantic matching problem. These models use deep learning techniques such as recurrent neural networks (RNN) [9] and convolutional neural networks (CNN) [10] to encode text sequences and then compare the encoded text sequences to determine the similarity between them. Their recurrent neural networks are mainly used for sequence modeling and can handle variable-length input sequences adaptively. They are widely used in the field of text semantic matching. There are also some improved models based on RNNs, such as Siamese [17], which classifies two text sequences by encoding them separately through the same RNN and then stitching them together for the task of text semantic matching. With the emergence of pre-trained models, the field of text semantic matching has also started to apply this technique. Pre-trained models can automatically learn rich semantic information by performing unsupervised learning on a large-scale text corpus to improve the performance of text semantic matching. BERT [5] is a representative model among them, which achieves extremely high performance by learning bidirectional contextual representations through a joint training task. Subsequent researchers have also proposed various improved models based on BERT, such as RoBERTa [15] and ALBERT [13].

2.2 Dropout

Dropout is a regularization technique commonly used in deep learning to reduce the complexity of the network by dropping some random neurons during training, thus reducing the risk of overfitting. Dropout was first proposed by Hinton et al. [8], one of the reasons for model overfitting is that the relationship between neurons is too complex, and dropout can prevent the relationship between neurons from being too complex by randomly dropping some neurons. Dropout is implemented in a simple way, i.e., some neurons are randomly dropped with a certain probability p in each training so that they are not involved in forward and backward propagation. This allows the network to be more robust during training and improves generalization. In addition to the original dropout technique, there are some improved methods. For example, DropConnect [19], which replaces random dropout with random disconnections, can increase the capacity of the network while reducing overfitting, and DropBlock [6], which replaces random dropout with random block dropout, can further reduce the risk of overfitting. However, these methods generally use only one dropout, while the parallel dropout we use works better by using multiple dropouts to randomly discard a portion of the neurons in the sentence vectors.

3 Method

Text semantic matching can be viewed as a classification task to find labels $y \in Y = \{similar, dissimilar\}$ for a given sentence pair (S_a, S_b). Figure 1 shows our model BiLSTM-PD for this task, and the model structure includes an input layer, a feature extraction layer, a dropout layer, and an output layer. In the following, we describe the components of the model.

3.1 Input Layer

Given two text sequences $S_a = \{w_1^a, ..., w_l^a\}$ and $S_b = \{w_1^b, ..., w_l^b\}$, we need to add an $[cls]$ at the beginning of the sentence and $[sep]$ in the middle and at the end of the two sentences, $S_{a,b} = [[cls]\, S_a\, [sep]\, S_b\, [sep]]$. Here, w_i^a and w_j^b represent the i-th and j-th word in the sequences; $[cls]$ denotes the beginning

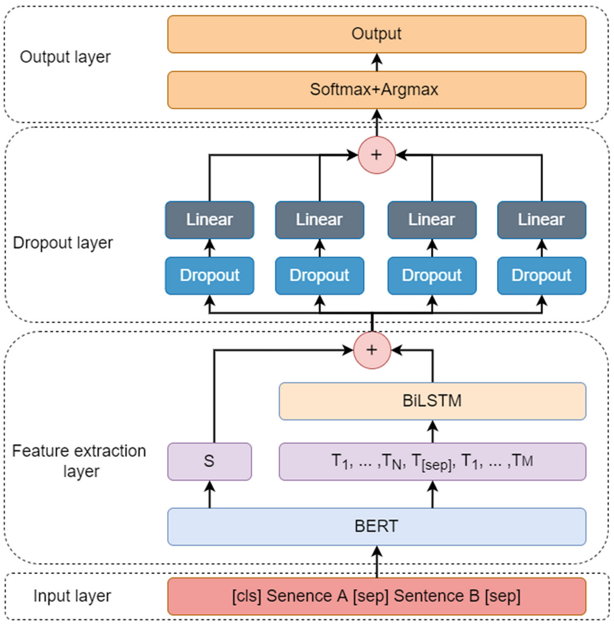

Fig. 1. General framework of the model

of the sequence and is used for the classification task; [*sep*] is used to separate two sentences or paragraphs in the input sequence so that BERT can distinguish them. We use $S_{a,b}$ as the input of BERT.

3.2 Feature Extraction Layer

The feature extraction layer consists of two parts, BERT and BiLSTM. The main work of this layer is that BERT generates word vectors and sentence vectors, BiLSTM converts the word vectors into sentence vectors, and the sentence vectors generated by BiLSTM are combined with the sentence vectors generated by the pre-trained model to form the final sentence vectors representation the structure of BiLSTM is shown in Fig. 2.

BERT: BERT is a pre-trained natural language processing model with a Transformer-based neural network at its core, whose goal is to map text data into vector representations by learning large amounts of unlabeled text data. We input $S_{a,b}$ to the BERT layer to obtain a vector representation S of the entire sentence pair and a vector representation x_t of each word, $x_t = [T_1, ..., T_N; T_1, ..., T_M]$, and we use x_t as input to the BiLSTM.

BiLSTM: The BiLSTM [7] consists of a combination of a forward LSTM [23] and a backward LSTM, using the forward LSTM and the backward LSTM to

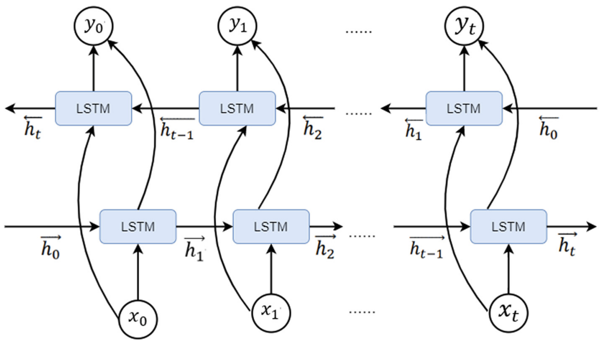

Fig. 2. BiLSTM structure

traverse the end and the beginning of the sequence, respectively, with the aim of efficiently capturing the global information of the input sequence. In this paper, we use BiLSTM to transform word vectors into sentence vectors. The BILSTM is calculated as follows:

$$\overrightarrow{h_t} = LSTM\left(\overrightarrow{h_{t-1}}, x_t\right) \tag{1}$$

$$\overleftarrow{h_t} = LSTM\left(\overleftarrow{h_{t-1}}, x_t\right) \tag{2}$$

$$H = \left[\overrightarrow{h_t}, \overleftarrow{h_t}\right] \tag{3}$$

where x_t is the input data at moment t, $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ is the output of the forward LSTM hidden layer and the output of the reverse LSTM hidden layer, respectively, and H is the output after the two are connected. The sentence vectors H is obtained by BILSTM processing, and the sentence vectors H is combined with the sentence vectors S generated by BERT to obtain the sentence vectors C, $C = [S, H]$.

3.3 Dropout Layer

The dropout layer consists of four dropout functions and four linear layers. Since the four dropout functions randomly discard a part of the neurons of the sentence vectors, the sentence vectors produce four subsets after the dropout layer. The dropout layer is computed as follows:

$$v_i = d_i\left(C\right), i \in [1, 2, 3, 4] \tag{4}$$

$$u_i = l_i\left(v_i\right), i \in [1, 2, 3, 4] \tag{5}$$

$$z = mean\left(u_1, u_2, u_3, u_4\right) \tag{6}$$

C represents the sentence vectors, d_i represents the four dropout functions; v_i represents the four subsets of the sentence vectors; l_i represents the four linear layers; u_i represents the four probability vectors mapped to the predicted label space. u_i is averaged to obtain z, and the probability vectors z goes to the output layer for the next processing step.

3.4 Output Layer

The output layer consists of a Softmax function and an Argmax function.

$$f = argmax\left(softmax\left(z\right)\right) \tag{7}$$

Softmax is the function that generates the vectors of probability distribution and Argmax is the function that determines the location of the maximum value. Z obtains the probability distribution vectors by Softmax function, and then obtains the predicted label vectors f for each Batch by Argmax function, with the value of f being 0 or 1. The accuracy of this Batch is obtained by comparing the predicted labels with the true labels.

Table 1. Datasets Statistics.

Datasets	Type	Train	Dev	Test
BQ	Positive	50.0k	5.0k	5.0k
	Negative	50.0k	5.0k	5.0k
LCQMC	Positive	138.5k	4,4k	6.2k
	Negative	100.1k	4,4k	6.2k

Table 2. Parameter Setting.

Hyperparameter	Value
Batch size	32
Epoch number	10
Optimizer	ADAMW
Learning rate	2e-05
Lr decay rate	0.01
Max char len	50
BiLSTM output size	384
Word embedding size	768

4 Experimental Setup

4.1 Datasets

The model was evaluated on two semantic matching datasets, including BQ [1], and LCQMC [14]. BQ is a question matching dataset for the banking and finance domain with data from question pairs in the customer service logs of online banking, while LCQMC is a large-scale open-domain Chinese question matching dataset constructed from Baidu Know. All datasets are divided into training, validation, and test data. Datasets statistics of BQ and LCQMC are shown in Table 1.

4.2 Parameter Setting

The parameters are set as shown in Table 2. The batch size of the dataset is fixed at 32. The initial learning rate is 2e-05, which decreases at a rate of 0.01 during the training period. Also, we adjust the output dimension of the BiLSTM hidden layer to 384. Finally, we use the ADAMW optimizer to modify all trainable parameters.

4.3 Evaluation Metrics

To evaluate the effectiveness of the model, the evaluation metrics for text semantic matching include accuracy (ACC) rate and F1 score (F1).

4.4 Contrast Model

We compare our model with recent work, including state-of-the-art neural network models and BERT-based approaches. DIIN [16] extracts relevant information from input sequences by a self-attention mechanism in the absence of RNNs or CNNs. ESIM [2] extracts information from text sequences using a bidirectional LSTM and models the relationship between sequences by a self-focusing mechanism. BiMPM [20] uses multi-angle matching and fusion. RE2 [22] employs richer features for the alignment process to improve performance. For the pre-trained approach, we consider BERT, RoBERTa, PERT [4], and MacBERT [3].

5 Experimental Results

5.1 Comparison Results and Discussion

Table 3 shows the results of the BQ dataset. All baselines are divided into two groups, the first group is four neural network-based methods and the second group is four pre-trained model-based methods. The pre-trained model-based methods show superior performance compared to the traditional neural matching models. For example, BERT outperformed ESIM by 1.95% and 0.95% in terms of accuracy and F1, respectively. MacBERT outperformed RE2 by 4.47% and 3.53% in terms of accuracy and F1, respectively. And the BiLSTM-PD model using MacBERT as the base model outperformed all eight models. For example, BiLSTM-PD is 1.82% and 2.30% higher than RoBERTa in terms of accuracy and F1, respectively. The possible reasons are that our model uses BiLSTM, which can better integrate the contextual information of the input sequences,

Table 3. Experimental results of accuracy on the BQ datasets.

Model	Acc(%)	F1(%)
DIIN [16]	82.45	83.19
ESIM [2]	81.74	82.25
BiMPM [20]	80.47	81.82
RE2 [22]	80.23	80.74
BERT [5]	83.69	83.20
RoBERTa [15]	83.48	83.00
PERT [4]	84.08	83.63
MacBERT [3]	84.70	84.27
BiLSTM-PD(MacBERT)	**85.30**	**85.30**

combining word vectors and sentence vectors can extract richer semantic features.

Table 4 shows the results of the LCQMC dataset. The method based on pre-trained models showed superior performance compared to the traditional neural matching models. For example, RoBERTa outperforms DIIN in terms of accuracy and F1 by 1.83% and 0.60%, respectively. PERT outperforms BiMPM in terms of accuracy and F1 by 2.71% and 1.35%, respectively. And the BiLSTM-PD model using MacBERT as the base model outperformed all eight models. For example, BiLSTM-PD outperforms MacBERT in terms of accuracy and F1 by 0.63% and 0.68%, respectively. Overall, our models achieved the best results on the LCQMC dataset.

Table 4. Experimental results of accuracy on the LCQMC dataset.

Model	Acc(%)	F1(%)
DIIN [16]	83.57	84.23
ESIM [2]	84.34	85.48
BiMPM [20]	83.72	84.63
RE2 [22]	84.23	84.97
BERT [5]	86.23	85.72
RoBERTa [15]	85.40	84.83
PERT [4]	86.43	85.98
MacBERT [3]	86.69	86.22
BiLSTM-PD(MacBERT)	**87.32**	**86.90**

To explore the effectiveness of our models, we combined BiLSTM-PD with the four pre-trained models to compare the results with the original pre-trained models and calculated the improvement in accuracy and F1, with bolded numbers indicating significant changes. As can be seen in Table 5 and Table 6, the accuracy and F1 of all pretrained models steadily improved on both datasets, and the improvement was particularly significant on the two pretrained models, RoBERTa and MacBERT. The results show that converting word vectors into sentence vectors by BiLSTM and combining the sentence vectors generated by BiLSTM with those generated by BERT and using four dropout functions to randomly discard a portion of the neurons of the sentence vectors can effectively improve the model and can be well integrated with the pre-trained model.

Table 5. Experimental results of accuracy on the BQ and LCQMC datasets.

Model	BQ	LCQMC
	Ori.→BiLSTM-PD(change)	Ori.→BiLSTM-PD(change)
BERT	83.69 →84.14 (**0.45**)	86.23 →86.25 (0.02)
RoBERTa	83.48 →83.96 (**0.48**)	85.40 →86.87 (**1.47**)
PERT	84.08 →84.11 (0.03)	86.43 →86.46 (0.03)
MacBERT	84.70 →85.30 (**0.60**)	86.69 →87.32 (**0.63**)

Table 6. Experimental results of F1 on the BQ and LCQMC datasets.

Model	BQ	LCQMC
	Ori.→BiLSTM-PD(change)	Ori.→BiLSTM-PD(change)
BERT	83.20 →83.41 (0.21)	85.72 →86.01 (0.29)
RoBERTa	83.00 →83.56 (**0.56**)	84.83 →85.52 (**0.69**)
PERT	83.63 →83.87 (0.24)	85.98 →86.24 (0.26)
MacBERT	84.27 →85.30 (**1.03**)	86.22 →86.90 (**0.68**)

5.2 Ablation Experiments

The main modules of the BiLSTM-PD model are the BiLSTM and the dropout layer, and the dropout layer has two important parameters, so we designed two ablation experiments.

Number of Dropout Samples: We choose 6 quantities of 0, 1, 2, 4, 8, and 16 respectively. We can see from Fig. 3 that the error rate tends to decrease and then increase as the number of dropout samples increases, which indicates that too many or too few dropout samples will make the model less effective. When the number of dropout samples is 4, the error rate of both data sets is the lowest, so it is better to choose 4 as the number of dropout samples in the experiment.

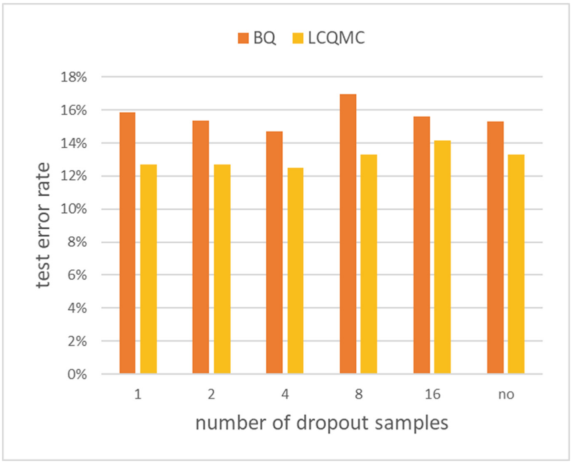

Fig. 3. Error rate of different dropout sample numbers

Dropout Ratio: We choose five probabilities, 10%, 30%, 50%, 70%, and 90%. Here we show how the multiple samples loss of BQ and LCQMC datasets works with different loss rates:{10%, 10%, 10%, 10%}, {10%, 10%, 30%, 30% }, {30%, 30%, 30%, 30%}, {10%, 30%, 50%, 70%}, {10%, 30%, 70%, 90% }, {30%, 50%, 70%, 90% }, {70%, 70%, 70%, 70% }, {70%, 70%, 90%, 90% }, {90%, 90%, 90%, 90% }, with mean values of 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, and 90%, respectively. Figure 4 shows the test set error rates for different dropout ratios. Regardless of the dropout ratio setting, parallel dropout outperforms no dropout. Dropout ratio from 10% to 90%, the Test error rate shows a wave-like

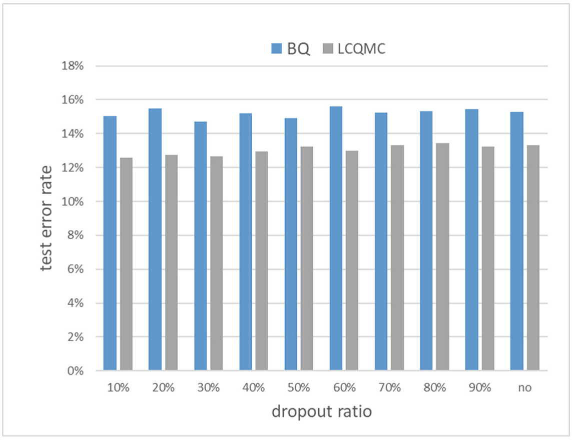

Fig. 4. Error rate of different dropout ratio

growth followed by a decline, and the test set error rate reached its lowest at an average dropout ratio of 30%. The overall change in the test error rate is not significant, which indicates that the parameter of dropout ratio has little effect on the model.

6 Conclusion

In this study, we propose a text semantic matching model that combine word vectors and sentence vectors to improve feature vectors quality and uses parallel dropout to reduce overfitting. The method is simple and effective, and easy to combines with pre-trained models. Experiments on two semantic matching datasets show that the proposed method outperforms the previously proposed model. However, the pre-trained models and datasets used in our work are all in Chinese, and since different languages have different characteristics and processing methods, future work may focus on modifying the models appropriately according to language characteristics to apply to other language datasets.

References

1. Chen, J., Chen, Q., Liu, X., Yang, H., Lu, D., Tang, B.: The BQ corpus: a large-scale domain-specific Chinese corpus for sentence semantic equivalence identification. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 4946–4951 (2018)
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. Cui, Y., Che, W., Liu, T., Qin, B., Wang, S., Hu, G.: Revisiting pre-trained models for Chinese natural language processing. In: Findings of the Association for Computational Linguistics: EMNLP 2020, pp. 657–668 (2020)
4. Cui, Y., Yang, Z., Liu, T.: PERT: pre-training BERT with permuted language model, arXiv preprint arXiv:2203.06906 (2022)
5. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805 (2018)
6. Ghiasi, G., Lin, T.Y., Le, Q.V.: Dropblock: a regularization method for convolutional networks. In: Advances in Neural Information Processing Systems, vol. 31 (2018)
7. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural Netw. **18**(5–6), 602–610 (2005)
8. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R.: Improving neural networks by preventing co-adaptation of feature detectors, arXiv preprint arXiv:1207.0580 (2012)
9. Hou, B.J., Zhou, Z.H.: Learning with interpretable structure from gated RNN. IEEE Trans. Neural Netw. Learn. Syst. **31**(7), 2267–2279 (2020)

10. 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)
11. Kadhim, A.I.: Term weighting for feature extraction on Twitter: a comparison between BM25 and TF-IDF. In: 2019 International Conference on Advanced Science and Engineering (ICOASE), pp. 124–128. IEEE (2019)
12. Kim, D., Seo, D., Cho, S., Kang, P.: Multi-co-training for document classification using various document representations: TF-IDF, LDA, and Doc2Vec. Inf. Sci. **477**, 15–29 (2019)
13. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: Albert: a lite BERT for self-supervised learning of language representations. In: International Conference on Learning Representations (2019)
14. Liu, X., et al.: LCQMC: a large-scale Chinese question matching corpus. In: Proceedings of the 27th International Conference on Computational Linguistics, pp. 1952–1962 (2018)
15. Liu, Y., et al.: RoBERTa: a robustly optimized BERT pretraining approach, arXiv preprint arXiv:1907.11692 (2019)
16. Mirakyan, M., Hambardzumyan, K., Khachatrian, H.: Natural language inference over interaction space: ICLR 2018 reproducibility report. arXiv preprint arXiv:1802.03198 (2018)
17. Mueller, J., Thyagarajan, A.: Siamese recurrent architectures for learning sentence similarity. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 30 (2016)
18. Rao, J., Liu, L., Tay, Y., Yang, W., Shi, P., Lin, J.: Bridging the gap between relevance matching and semantic matching for short text similarity modeling. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 5370–5381 (2019)
19. Wan, L., Zeiler, M., Zhang, S., Le Cun, Y., Fergus, R.: Regularization of neural networks using DropConnect. In: International Conference on Machine Learning, pp. 1058–1066. PMLR (2013)
20. 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)
21. Wu, Z., et al.: An efficient Wikipedia semantic matching approach to text document classification. Inf. Sci. **393**, 15–28 (2017)
22. Yang, R., Zhang, J., Gao, X., Ji, F., Chen, H.: Simple and effective text matching with richer alignment features. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 4699–4709 (2019)
23. Yu, Y., Si, X., Hu, C., Zhang, J.: A review of recurrent neural networks: LSTM cells and network architectures. Neural Comput. **31**(7), 1235–1270 (2019)