Stance Detection with Stance-Wise Convolution Network

Dechuan Yang[1], Qiyu Wu[2], Wei Chen[2(✉)], Tengjiao Wang[2], Zhen Qiu[3], Di Liu[3], and Yingbao Cui[3]

[1] Center for Data Science, Peking University, Beijing, China
yangdechuan@pku.edu.cn
[2] Key Lab of High Confidence Software Technologies (MOE), School of EECS, Peking University, Beijing, China
{wuqiyu,pekingchenwei,tjwang}@pku.edu.cn
[3] State Grid Information and Telecommunication Group, Beijing, China
{qiuzhen,liudi,cuiyingbao}@sgitg.sgcc.com.cn

Abstract. Stance detection aims at identifying the stance (*favor*, *against* or *neutral*) of a text towards a specific target of opinion. Recently, there is a growing interest in using neural models for stance detection, but there are still some challenges to be solved. Firstly, it is difficult to associate text with target because targets are not always discussed explicitly in texts. However, existing methods always roughly model the representations of text and target on task-specific and limited corpus without considering the indispensable external information. Secondly, different from categories in normal classification task, we find that stances in stance detection task are not independent to each other. We study this observation and find it would be more effective to learn each stance individually. But all previous approaches ignore the correlation. To address these two challenges effectively, we introduce a Stance-wise Convolution Network (SCN) including two novel modules. Specifically, we first use a Text-Target Encoder module to subtly incorporate the pre-trained BERT into our model to learn more reasonable text-target representations. Then we propose a Stance-wise Convolution module to better learn stances by absorbing the correlation between stances. We evaluate our method on real-world dataset and the experimental results show that our proposed method achieves the state-of-the-art performance.

Keywords: Stance detection · Stance-wise Convolution · Pre-trained model

1 Introduction

Recognizing different stances from user-generated texts is essential to analyze public opinion and user behaviour on social media. For example, we would like

D. Yang and Q. Wu—Contribute equally.

X. Zhu et al. (Eds.): NLPCC 2020, LNAI 12430, pp. 555–567, 2020.
https://doi.org/10.1007/978-3-030-60450-9_44

to find out from someone's tweet whether he supports a presidential candidate or not during U.S. general election. Stance detection is the task of automatically determining from text whether the author is in favor of, against or neutral towards a given target [9]. Early related work focuses on document-level opinion mining [10]. However, in stance detection, we are more concerned about the position of an author to a specific target rather than the polarity of the whole text. Hence accurate detection of stance toward a target from texts is crucial to perform such a task.

Previous work has proposed different methods for stance detection. Early work uses traditional sentiment analysis methods for stance detection [15,18]. However, these methods perform stance detection based on features extracted from text and ignore the target information. Typical work in [5] and [21] uses bidirectional LSTM and GRU [2] to extract high-level text features, and then learns to focus on important words of text given the specific target using attention mechanism. These RNN-based methods attempt to use target information to learn target-specific text representation. But they ignore text information when modeling target representation. Later work [14] attempts to model text and target jointly. It concatenates text and target into one sequence and used CNN to model them together. However, the interactive relation between text and target is learned on task-specific and limited training corpus, and the performance improvement is rather limited.

All these aforementioned approaches suffer from two common limitations. 1) Firstly, existing methods do not learn joint representations of text-target and only model the correlation between target and text on task-specific corpus. Since the given target is rarely discussed explicitly in the text, in such a rough way the joint correlation of target and text cannot be fully learned. Recent advances in natural language processing have produced pre-trained languages models [3,6,11,13]. So it's natural to expect that these well initialized models can capture each token's semantic meaning in a sentence and the interactive relation between text and target, which may lead to better solutions for stance detection. 2) Secondly, stances in stance detection task are different from categories in normal classification task. In normal multi-classification problem, categories (e.g. dog, cat and horse) are independent to each other, as depicted in Fig. 1 (*a*). But for stance detection, there is some correlation between stances, which should be considered. Stances are actually continuous and somehow related to each other, as depicted in Fig. 1 (*b*). Take a text-target pair below for example:

Target: Hillary Clinton
Text: *Hillary Clinton has some strengths and some weaknesses.*

We can deduce from the tweet that the author is neutral towards the target because both *Favor* and *Against* stances exist. Although there are *Favor* and *Against* stances in the text, the author may be still neutral towards the target as a whole. But in the normal multi-classification situation, it is unreasonable to infer a "horse" if the model capture strong features for both "cat" and "dog".

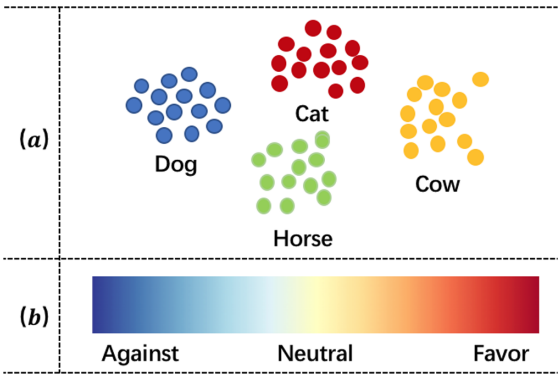

Fig. 1. (*a*) In normal classification problem, categories are independent to each other. (*b*) Instance detection, stances are continuous and somehow related to each other.

Apparently, typical correlation between stances like that should get more attention, but all existing methods ignore the crucial factor.

To address the two limitations, we build our proposed model consisting of a Text-Target Encoder (TTE) module and a Stance-wise Convolution (SC) module. These two modules focus on learning reasonable text-target representations and modeling stance-wise clues respectively.

Specifically, we first concatenate the text-target pair with a [SEP] token between them and then employ BERT to obtain joint text and target representations with respect to each other. Since BERT is trained for next sentence prediction, we believe that using this method will provide richer text and target representations than these aforementioned approaches. Then we use our Stance-wise Convolution (SC) module to consider the correlation between stances as mentioned before. SC utilises three independent convolution networks to extract stance-indicative features from text-target representations for the three stances individually. So every stance could be modeled adequately. After that, we consider synthetically the influence of these features and give every stance label a score. The stance label with the highest score is the prediction result. We use cross entropy loss function to train the classification task, and we also propose an extra stance-wise loss function to model the correlation explicitly. The details of TTE and SC will be discussed in Sect. 3.

The main contributions of our work can be summarized as follows:

- We propose to learn text and target representations simultaneously. By adopting the Text-Target Encoder, the text, target and their interactive information can be imparted together into a joint representation and the rich pre-trained information of relation between target and text is also subtly utilized.
- We study the difference between stance detection task and normal classification task, and first propose a Stance-wise Convolution Network as a better solution for stance detection.

– We devise a hybrid network that combines Text-Target Encoder and Stance-wise Convolution in a unified architecture. And experimental results show that our proposed method achieves the state-of-the-art performance.

The rest of this paper is organized as follows. Section 2 briefly introduces related work and Sect. 3 describes our proposed model. Section 4 discusses evaluation results. Finally, Sect. 5 concludes the paper.

2 Related Work

In this section, we will review related work on stance detection task and pre-trained models briefly.

2.1 Stance Detection

Stance detection has gained research attention in recent times. A seminal work by [9] results in starting wide-spread research in this area. In the past few years, various models, including machine learning methods such as SVM, deep learning methods such as convolution neural networks (CNN) and recurrent neural networks (RNN), were proposed for stance detection task. [20] won the first place in SemEval2016 stance detection task using transfer learning methods. [15] use SVM based model, and [18] use a CNN based model, and [1] use a bidirectional LSTM based model. However, all these methods ignore the target information.

Recently, attention mechanism has been used in stance detection task. [5] first proposed a target-specific attention network, which made full use of the target information in stance detection. [21] also used attention mechanism to focus on important words of text. And he used a GRU to extract high-level features from both text and target. Later, [4] proposed a two-phase LSTM-based model with attention and [17] proposed an end-to-end neural memory model via target and tweet interaction. More recently, [14] proposed a neural ensemble method that combines the attention based densely connected Bi-LSTM and nested LSTMs models with the multi-kernel convolution in a unified architecture. However, these methods don't utilize pre-trained information, and they don't consider the correlation between stances.

2.2 Pre-trained Models

Recently, in order to improve the performance of natural language processing systems, pre-trained models have become prevalent, namely the Embedding from Language Models (ELMo) [11], Generative Pre-training Transformer (GPT) [12, 13], Bidirectional Encoder Representations form Transformers (BERT) [3] and XLNet [19]. BERT is essentially a multi-layer bi-directional Transformer encoder [16]. It pre-trains bi-directional representations by conditioning on both left and right contexts jointly in all layers. Unlike ELMo and GPT, BERT is not pre-trained using traditional left-to-right or right-to-left language models. Instead,

BERT is pre-trained using two novel unsupervised prediction tasks: Masked Language Models (MLM) and Next Sequence Prediction (NSP). The first task is to predict randomly masked tokens in the input. And the second task is to predict whether a sentence following a given sentence is in the corpus or not. The authors claim that BERT allows to swiftly adapt for a downstream task with little modification to the architecture. And BERT improved the state-of-the-art for a range of NLP benchmarks by a significant margin. But existing models do not specially pre-train joint embedding for target and text. How to transfer the specific information from a general pre-trained model is a great challenge.

3 Methodology

In this section, we describe the details of our proposed BERT-based Stance-wise Convolution Network for stance detection. Figure 2 depicts an overview of our proposed framework. It consists of a Text-Target Encoder (TTE) layer and a Stance-wise Convolution (SC) layer. We first use the Encoder layer to extract joint text-target representations. After that, the Stance-wise Convolution layer extracts stance-indicative feature vectors from the feature sequence for three stances respectively and then determines the stance label. We describe the detail of the model in the following subsections.

3.1 Task Definition

We are given a set of samples D. For a sample $c \in D$, it contains a text S with n tokens $(s_1, s_2, ..., s_n)$ and an opinion target T with m tokens $(t_1, t_2, ..., t_m)$. For the opinion target T, it is also associated with a stance label y, which can be either *Against*, *Favor* or *Neutral*. So our task can be stated as follows: given D as training corpus, our goal is to learn a target-specific stance detection classifier so that it can predict the stance label for a specific target in an unseen sample.

3.2 Text-Target Encoder Layer

BERT [3] is a influential pre-trained model. It is pre-trained using two unsupervised prediction tasks: masked language models and next sequence prediction, which makes it able to learn both contextualized word representations of one sentence and alignment between two arbitrary sentences. BERT's input embedding is constructed by summing the positional, segment and token embeddings. It uses positional embedding to encode the word sequences without a need for recurrent architecture, and it uses segment embedding to differentiate two input sentences.

Our Text-Target Encoder (TTE) utilizes BERT to encode text S and target T simultaneously. We propose to pack text and target together into a single token sequence $([CLS], s_1, s_2, ..., s_n, [SEP], t_1, t_2, ..., t_m, [SEP])$. After that, our TTE constructs input embedding and transform it into a joint embedding as same as BERT [3]. Formally, the input embedding is denoted as $X = (x_1, x_2, ..., x_l)$,

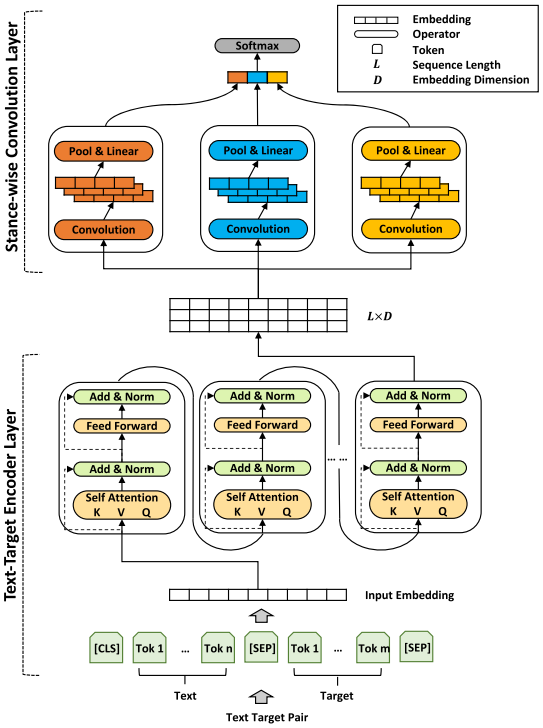

Fig. 2. Architecture of Stance-wise Convolution Network (SCN)

where $x_i \in \mathbb{R}^k$ is the embedding by summing up the token, segment and position embeddings, and l is the maximum length of the token sequence. After generating the input embedding X, TTE transforms it into a joint text-target embedding. At the j-th layer, the text-target embedding is denoted as $H^{(j)} = (h_1, h_2, ..., h_l)$, where $h_i \in \mathbb{R}^k$ has the same dimension with the corresponding input x_i. And $H^{(j)}$ can be obtained as follows:

$$\begin{aligned} Y &= H^{(j-1)} \\ head_i(Y) &= \text{softmax}(\frac{(YW_{Q_i})(YW_{K_i})^T}{\sqrt{d/m)}})(YW_{V_i}) \\ MultiHead(Y) &= Concat(head_1, ..., head_m)W_O \\ Z &= LN(Y + MultiHead(Y)) \\ H^{(j)} &= LN(Z + MLP(Z)), \end{aligned} \tag{1}$$

where $\{W_{Q_i}, W_{K_i}, W_{V_i}\} \in \mathbb{R}^{d \times \frac{d}{m}}$, $W_O \in \mathbb{R}^{d \times d}$ are learnable parameters, LN represents layer normalization, and $H^{(0)} = X$. The entire model stacks J such layers, and we choose the last layer $H^{(J)}$ as our final text-target embedding H.

3.3 Stance-Wise Convolution Layer

After the joint text-target embedding is obtained from the Text-Target Encoder, we use a Stance-wise Convolution (SC) layer to extract stance-indicative features. In this layer, we adopt the idea proposed by [7] to extract structural information of input sequence. Beyond it, we additionally utilize three independent convolution networks, which are Against Stance Convolution, Favor Stance Convolution and Neutral Stance Convolution respectively. And every Stance Convolution is expected to extract the corresponding stance-indicative features.

Specially, for every Stance Convolution network, the input is the joint text-target embedding from the Text-Target Encoder layer. Let $h_i \in \mathbb{R}^k$ be the k-dimensional output embedding corresponding to the i-th token in the input sequence of the Text-Target Encoder. A sequence of length l is represented as

$$H = h_1 \oplus h_2 \oplus ... \oplus h_l, \tag{2}$$

where $\oplus$ is the concatenation operator and $H \in \mathbb{R}^{l \times k}$. We then perform the convolution on it by using several filters. A convolution operation involves a filter $w \in \mathbb{R}^{h \times k}$, which is applied to a window of h tokens to produce a new feature. For example, a feature $c_i \in \mathbb{R}$ is generated from a window of tokens $h_{i:i+h-1}$ by

$$c_i = f(w \odot h_{i:i+h-1} + b), \tag{3}$$

where f is a non-linear function. This filter is applied to each possible window of tokens in the sequence to produce a feature map,

$$c = (c_1, c_2, ..., c_{n-h+1}). \tag{4}$$

We then apply a max-over-time pooling operation over the feature map and take the maximum value $\hat{c} = \max\{c\}$ as the feature corresponding to this particular filter. Finally, the feature generated from each filter are concatenated to form a single high-level feature vector,

$$\hat{h} = (\hat{c}_1, \hat{c}_2, ..., \hat{c}_p). \tag{5}$$

As a result, for every Stance Convolution, we get a feature vector $\hat{h} \in \mathbb{R}^p$, where p represents the number of filter. And We use $\hat{h}_0, \hat{h}_1, \hat{h}_2$ to represent *Against*-indicative feature vector, *Neutral*-indicative feature vector and *Favor*-indicative feature vector respectively.

The stance scores of the three stances are then defines as follows:

$$\begin{aligned} \alpha_0 &= W_0 \hat{h}_0 + b_0 \\ \alpha_1 &= W_1 \hat{h}_1 + b_1 \\ \alpha_2 &= W_2 \hat{h}_2 + b_2 \end{aligned} \tag{6}$$

where $W \in R^{1 \times d}$ and $b \in R$ defines a stance-specific linear transformation, with α_0, α_1 and α_2 reflects the confidence in *Against*, *Neutral* and *Favor* stances respectively. The higher the value of α_i, the more likely i is the true stance label.

We concatenate α_0, α_1 and α_2 to represent the stance scores vector. The confidence scores α are then normalized to probabilities using a softmax operation:

$$\begin{aligned} \alpha &= [\alpha_0, \alpha_1, \alpha_2] \\ \hat{y} &= \text{softmax}(\alpha) \end{aligned} \tag{7}$$

where $\hat{y} \in R^3$ is the vector of predicted probability for the three stances.

3.4 Stance-Wise Loss Function

As mentioned before, there is some correlation between stances, so we propose a stance-wise penalty function to model the correlation explicitly:

$$L(\hat{y}^i, y^i) = \begin{cases} \max(0, \hat{y}_2^i - \hat{y}_1^i) & y_0^i = 1 \\ |\hat{y}_0^i - \hat{y}_2^i| & y_1^i = 1 \\ \max(0, \hat{y}_0^i - \hat{y}_1^i) & y_2^i = 1, \end{cases} \tag{8}$$

where y^i is the one-hot representation of the ground-truth stance for the i-th sample, and $\hat{y}^i$ is the corresponding predicted probability vector as described in Eq. 7.

When the stance label for the sample is *Against* ($y_0^i = 1$), the probability of *Neutral* should be higher than that of *Favor*. And if not, there will be a penalty. It is the same with the situation that the stance is *Favor* ($y_2^i = 1$). When the stance is *Neutral* ($y_1^i = 1$), there should be a small gap between the probabilities of *Against* and *Favor*. If not, there will also be a penalty.

According to these penalties, the stance-wise loss function can be computed as:

$$J_{scn}(\theta) = \frac{1}{N} \sum_{i=1}^{N} L(\hat{y}^i, y^i), \tag{9}$$

where N is the number of training samples and θ is the set of model parameters.

3.5 Model Training

The parameters of the network are trained to classify the stances correctly, so we also use a cross-entropy loss function to train the model end-to-end. The cross-entropy loss function is defined as:

$$J_{ce}(\theta) = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{3} y_j^i \log \hat{y}_j^i \tag{10}$$

So the final loss function of our model can be written as:

$$J(\theta) = (1 - \gamma) J_{ce}(\theta) + \gamma J_{scn}(\theta), \tag{11}$$

where γ is a hyper-parameter.

Standard back propagation is performed to optimize the model parameters. And we use Adam [8] for parameter optimization.

4 Experiment

4.1 Experimental Settings

Data Description. We use the benchmark training and test data provided by the SemEval 2016 stance detection task [9]. This is the first released dataset for stance detection from English tweets. And training and test data belongs to five targets: "*Atheism*", "*Climate Change is a Real Concern*", "*Feminist Movement*", "*Hillary Clinton*" and "*Legalization of Abortion*". The distribution of the dataset is shown in Table 1, and the dataset has been separated into training and test set.

Hyper-parameters. We build our model on top of the pre-trained uncased BERT model released by [3]. We set the number of attention heads as 12, and the dropout rate as 0.1. Adam is used for our optimization method, and its learning rate is 5e−5, β_1 is 0.9, β_2 is 0.999. And the γ of the final loss function is 0.2. The model is trained by batch size of 32. And the parameters are initialized from the pre-trained BERT model.

Table 1. Distribution of instances in the SemEval2016 dataset.

		Number of samples in train				Number of samples in test			
Target	#Total	#Train	Favor	Against	Neutral	#Test	Favor	Against	Neutral
Atheism	733	513	92	304	117	220	32	160	28
Climate Change is a Real Concern	564	395	212	15	168	169	123	11	35
Feminist Movement	949	664	210	328	126	285	58	183	44
Hillary Clinton	984	689	118	393	178	295	45	172	78
Legalization of Abortion	933	653	121	355	177	280	46	189	45
Total	4163	2914	753	1395	826	1249	304	715	230

Evaluation Metrics. Following the SemEval-2016 Task 6-A benchmark, we employed the macro-average of F1-score for the *Against* and *Favor* stance classes as the evaluation measure. Firstly, the F1-score for *Against* and *Favor* stances for all instances in the dataset is calculated as:

$$\begin{aligned} F_{\text{favor}} &= \frac{2 \times P_{\text{favor}} \times R_{\text{favor}}}{P_{\text{favor}} + R_{\text{favor}}} \\ F_{\text{against}} &= \frac{2 \times P_{\text{against}} \times R_{\text{against}}}{P_{\text{against}} + R_{\text{against}}} \end{aligned} \tag{12}$$

where P is precision and R is recall. Then the average of F_{favor} and F_{against} is calculated as the final metrics:

$$F_{\text{avg}} = \frac{F_{\text{favor}} + F_{\text{against}}}{2} \tag{13}$$

4.2 Baselines

We validate the effectiveness of our proposed Stance-wise Convolution Network for target-based stance detection by comparing with the following baseline systems:

- Neural Bag-Of-Words (*NBOW*): The *NBOW* sums the word vectors within the sentence and applies a softmax classifier.
- *MITRE* [20]: As the SemEval-2016 best performing system, *MITRE* won the first place in SemEval-2016 stance detection task using transfer learning methods.
- *TAN* [5]: *TAN* is the first work to apply attention mechanism in stance detection. It makes use of target-based attention mechanism to focus on the important parts of the text sequence.
- *TGMN-CR* [17]: This method proposed an end-to-end neural memory model via target and tweet interaction, which leverages the power of target content information.
- *PNEM* [14]: *PNEM* is a neural ensemble method that combines CNN and LSTM in a unified architecture. It first models text and target jointly and achieves a state-of-the-art result.

Table 2. Performance comparison of our method against existing baselines.

Method	Overall			Target				
	F_{against}	F_{favor}	F_{avg}	Atheism	Climate Change	Feminist Movement	Hillary Clinton	Legal. of Abortion
NBOW	–	–	60.19	55.12	39.93	50.21	55.98	55.07
MITRE	76.33	59.32	67.82	61.47	41.63	62.09	57.67	57.28
TAN	–	–	68.79	59.33	**53.59**	55.77	65.38	63.72
TGMN-CR	76.55	65.52	71.04	64.60	43.02	59.34	66.21	**66.21**
PNEM	77.66	66.56	72.11	67.73	44.27	**66.76**	60.28	64.23
SCN(Ours)	**78.86**	**68.60**	**73.73**	**73.55**	48.41	61.36	**71.30**	65.34

4.3 Main Results

The performance of all baselines and our proposed method are listed in Table 2. For each baseline method, we copy the result reported from the original paper, which makes sure that they have the best performance. And for our result, each experiment was run 5 times with different random seeds, and we report the average results.

From the results, we can find that *NBOW* performs unsatisfactorily, because it only use features extracted from the text but ignore the target information. *TGMN-CR* and *PNEM* outperforms both *TAN* and *MITRE*. It indicates that the interaction between target and text is conductive to stance detection. For our

method, we performs the best on the target "*Atheism*" and "*Hillary Clinton*". We improve upon the best performing methods by 5.82% and 5.09% on the two targets respectively. Our method doesn't perform well enough on the target "*Climate Change*", and we think this is because the distribution of *Against* and *Favor* labels in the dataset is rather imbalanced and we don't use any trick to solve the problem. And in general, our method outperforms the best performing method *PNEM* by 1.62% and outperforms all other baseline methods by a large margin. It shows the effectiveness of our proposed *SCN* model.

4.4 Ablation Study

In order to estimate the effect of each module of our *SCN* model, we perform ablation study on the model to better understand their relative importance.

As mentioned before, target content information plays an important role in the stance detection task and we design a Text-Target Encoder (TTE) to learn joint representations for text and target. So we conduct experiment to demonstrate that we make use of target information effectively. We remove target sequence from the input sequence and then take the sequence into the model. For example, if previous input sequence is:

$$[CLS]\, Hillary\ is\ our\ best\ choice\ if\ we\ truly\ want\ to\ continue\ being\ a\ progressive\ nation\ [SEP]\, Hillary\ Clinton\ [SEP]$$

we now use a new input sequence instead as follows:

$$[CLS]\, Hillary\ is\ our\ best\ choice\ if\ we\ truly\ want\ to\ continue\ being\ a\ progressive\ nation\ [SEP]$$

Table 3 shows the experiment results. And from the results, we can see that performance will decrease by 0.61% when removing target information from input sequence.

We then conduct experiment to estimate the effect of our Stance-wise Convolution (SC) module. In this regard, we first remove the Stance-wise Convolution layer from the model and use a full connected (FC) layer instead. For the second comparative model, we use an ordinary convolution network (CNN) instead of our Stance-wise Convolution. From the Table 3, we can see that performance will decrease by 0.5% when removing Stance-wise Convolution. And if we use ordinary convolution network, performance will even decrease by 0.92%. The experimental result shows that stance-wise convolution could benefit stance detection.

Table 3 also compares our model with plain BERT model. Plain BERT model use text as the input and has no additional network. Our method outperform plain BERT model by 1.47%, which demonstrates that it is our TTE and SC, instead of BERT on its own, that brings these improvements.

Table 3. Comparative performance with different experimental settings. The best results are highlighted in boldface. BERT is a important baseline to be compared with. TTE+FC and TTE+CNN represent that we remove SC from our model and use FC or CNN instead respectively. SC represents that we remove TTE from our model. And TTE+SC represents our final model.

Method	BERT	TTE+FC	TTE+CNN	SC	TTE+SC
F_{against}	77.21	77.57	78.50	77.69	**78.86**
F_{favor}	67.30	68.89	67.11	68.55	**68.60**
F_{avg}	72.26	73.23	72.81	73.12	**73.73**

5 Conclusion

In this paper, we propose a neural architecture for stance detection task. It consists of two modules: 1) a BERT-based Text-Target Encoder to obtain joint text-target representations, which extracts abundant text-target relation information from pre-trained model. 2) a Stance-wise Convolution module to extract stance-indicative features, which improves performance by distinguishing normal classification and stance detection. Benefited from more particular factors for stance detection, our proposed model performs better on this task. This research provides new insights to the study of stance detection from both encoder and predictor views. Experimental results show that our proposed model achieves state-of-the-art performance.

Acknowledgements. This work is supported by State Grid Technical Project (No. 52110418002W).

References

1. Augenstein, I., Rocktäschel, T., Vlachos, A., Bontcheva, K.: Stance detection with bidirectional conditional encoding. arXiv preprint arXiv:1606.05464 (2016)
2. Cho, K., et al.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078 (2014)
3. 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)
4. Dey, K., Shrivastava, R., Kaushik, S.: Topical stance detection for twitter: a two-phase LSTM model using attention. In: Pasi, G., Piwowarski, B., Azzopardi, L., Hanbury, A. (eds.) ECIR 2018. LNCS, vol. 10772, pp. 529–536. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-76941-7_40
5. Du, J., Xu, R., He, Y., Gui, L.: Stance classification with target-specific neural attention networks. In: International Joint Conferences on Artificial Intelligence (2017)
6. Howard, J., Ruder, S.: Universal language model fine-tuning for text classification. arXiv preprint arXiv:1801.06146 (2018)

7. Kim, Y.: Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882 (2014)
8. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
9. Mohammad, S., Kiritchenko, S., Sobhani, P., Zhu, X., Cherry, C.: Semeval-2016 task 6: detecting stance in tweets. In: Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), pp. 31–41 (2016)
10. Pang, B., Lee, L., et al.: Opinion mining and sentiment analysis. Found. Trends® Inf. Retrieval **2**(1–2), 1–135 (2008)
11. Peters, M.E., et al.: Deep contextualized word representations. arXiv preprint arXiv:1802.05365 (2018)
12. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving language understanding by generative pre-training (2018). https://s3-us-west-2.amazonaws.com/openai-assets/researchcovers/languageunsupervised/languageunderstandingpaper.pdf
13. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners. OpenAI Blog **1**(8), 9 (2019)
14. Siddiqua, U.A., Chy, A.N., Aono, M.: Tweet stance detection using an attention based neural ensemble model. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 1868–1873 (2019)
15. Sobhani, P., Mohammad, S., Kiritchenko, S.: Detecting stance in tweets and analyzing its interaction with sentiment. In: Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics, pp. 159–169 (2016)
16. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)
17. Wei, P., Mao, W., Zeng, D.: A target-guided neural memory model for stance detection in twitter. In: 2018 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. IEEE (2018)
18. Wei, W., Zhang, X., Liu, X., Chen, W., Wang, T.: pkudblab at SemEval-2016 task 6: a specific convolutional neural network system for effective stance detection. In: Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), pp. 384–388 (2016)
19. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., Le, Q.V.: Xlnet: generalized autoregressive pretraining for language understanding. arXiv preprint arXiv:1906.08237 (2019)
20. Zarrella, G., Marsh, A.: Mitre at semeval-2016 task 6: transfer learning for stance detection. arXiv preprint arXiv:1606.03784 (2016)
21. Zhou, Y., Cristea, A.I., Shi, L.: Connecting targets to tweets: semantic attention-based model for target-specific stance detection. In: Bouguettaya, A., et al. (eds.) WISE 2017. LNCS, vol. 10569, pp. 18–32. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68783-4_2