IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 22, 2020, accepted November 1, 2020, date of publication November 6, 2020, date of current version November 18, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3036455

Adaptive Feature Extractor of Global Representation and Local Semantics for Text Classification

CHAOFAN WANG[1], SHENGGEN JU[1], YUEZHONG LIU[2], RUN CHEN[1], AND XIAOMING HUANG[3]

[1]College of Computer Science, Sichuan University, Chengdu 610065, China
[2]Enterprise Service, Commonwealth Bank of Australia, Sydney, NSW 2000, Australia
[3]Shenzhen CyberAray Network Technology Company Ltd., Shenzhen 518000, China

Corresponding author: Xiaoming Huang (apride@gmail.com)

This work was supported in part by the National Natural Science Foundation under Grant 61972270, in part by the Sichuan New Generation Artificial Intelligence Major Project under Grant 2018GZDZX0039, and in part by the Sichuan Key Research and Development Project under Grant 2019YFG0521.

ABSTRACT A hybrid model of Convolutional Neural Network (CNN) and self-attention has achieved remarkable results in text classification fields. In previous researches, text local semantics(captured by CNN) and global representation (extracted by self-attention) play equally important roles for each input. However, the importance of the two varies greatly with complex linguistic backgrounds. In this paper, we take an adaptive approach to automatically determine the contribution degree of each model to classification, according to specific structure and grammar information of the text. This strategy can make the most of two models. To better extract variable-size features of a word, multi-scale feature attention is introduced into our hybrid model. The attention focus on assigning larger weights to those multi-scale features that are important to a word. In addition, for fine-grained emotion classification tasks, a new type of loss function is also established. Experiment results show that, the proposed model makes noticeable improvements over hybrid models. Metrics are 0.3 to 1.5 percentages higher than previous methods. And results also prove that the new loss function further improves the performance of our model in all fine-grained emotion classification datasets.

INDEX TERMS CNN, self-attention, multi-scale feature, hybrid model, text classification.

I. INTRODUCTION

Text classification is a significant subtask of natural language processing applications, such as question classification [1], topic classification [2], and sentiment analysis [3]. Until now, the most popular approaches of text classification are generally based on CNN, RNN and self-attention. There are also some hybrid models such as CNN and RNN. In 2019, Chia and Witteveen [4] proposed a hybrid model of CNN and self-attention for natural language understanding. And this model makes great improvements over single CNN or self-attention models because of its capacity in fusing the advantages of two models (capture both local semantics and global representations of a sentence [5]).

The associate editor coordinating the review of this manuscript and approving it for publication was Bin Liu.

To better illustrate what are local semantics and global semantic representations [5] of a sentence, we give two examples in Table 1. The first and second sentences in Table 1 are examples of question classification. The first sentence should be classified as <numeric>. The key phrase "rate" that can be well captured by CNN owing to its position-invariance, determines the category is numeric. On the contrary, the second sentence should be classified as <describe>. The most vital expression is "what does ××× mean ?", which can be easily extracted by self-attention due to its efficiency in capturing long-term dependencies.

Since then, a variety of hybrid models based on CNN and self-attention emerged in text classification field. For example, Chia and Witteveen [4] in 2019 proposed a hybrid model, which utilizes the hierarchical CNN with Transformer [26]. However, most of hybrid models take the output

TABLE 1. Cases of question classification.

1: What's the conversion rate between pounds and dollars? (numeric)
2: What does bank bond mean? (describe)

of self-attention as the input of CNN, or just to concatenate respective outputs of the two as the final features representations of text. That means that local semantics and global representations will play equal roles for each sentence. However, the importance of the two varies sharply with complicated structure and grammar information. Just treat them equally may seriously affect the performance of hybrid model. In this paper, we take an adaptive approach proposed by Zhao *et al.* [5] to decide importance degree of local semantics and global representations to classification. Concretely, both of two feature representations will be assigned proper probability value according to a context feature vector. Because context feature vector represents the whole text, the probability value can reflect the grammar structure of the text in some degree. Larger value means more importance. So, this method can take advantages of two models in a more reasonable way. What's more, we also choose CNN with multi-scale feature attention [10] as our local semantics feature extractor because of its efficiency in extracting variable-size features.

In addition, it's unreasonable for fine-grained emotion classification task to take the same cross-entropy loss function as ordinary text classification tasks. Because the effect of classifying a piece of five-star review into a one star is quite different from that of classifying a piece of five-star review into a four star. It is obvious that mistake the former makes is more serious than the latter do although both of them belong to misclassification cases. But models with normal cross-entropy possibly get the same loss values in two cases.

The contributions of this paper can be summarized into two parts:

1) We propose MulCNN-Att to effectively fuse local semantics and global representation of text. It can automatically determine the contribution degree of each model to classification, according to specific structure and grammar information. Thus MulCNN-Att improves the performance of text classification. We also introduce multi-scale feature attention into our hybrid model. The attention mechanism is efficiency in extracting more accurate local semantics in text.

2) For fine-grained emotion classification tasks, we also establish a new kind of loss function which leads to further improvements in each fine-grained emotion classification datasets.

II. RELATED WORK

At present, almost all text classification work are based on RNN,CNN, or self-attention mechanism. Next, we will give a simple introduction to some important works.

A. CONVOLUTIONAL NEURAL NETWORKS

CNN has been utilized in natural language processing tasks for several years. In 2014, Kim [6] first proposed to view CNN with multiple kinds of convolutional filters as a feature encoder of a sentence to classify sentence. And then, Zhang *et al.* [7] introduced an empirical method to explore character-level CNN in text classification tasks. Shallow CNN cannot act well as a long-term information encoder. So, Conneau *et al.* [8] try to use a very deep CNN in text classification and reach an excessively high classification accuracy. Similarly, Johnson and Zhang [9] construct a deep pyramid CNN model which not only improves experimental results but accelerates the training process. Wang *et al.* [10] proposed a multi-scale feature CNN to capture more accurate local semantics features of the text through attentions weights among the different sizes of convolution filters. We can easily see that although CNN has achieved great success in task of text classification, it cannot capture the long-term dependencies among words of a sentence. So, many works start trying to utilize a hybrid model of CNN and RNN(or self-attention) to help model acquire the ability of capturing global features representation in a sentence.

B. HYBRID MODEL OF CNN AND RNN

C-LSTM [11] and DSCNN [12] are hybrid models of CNN and RNN. C-LSTM first utilized CNN to extract a sequence of phrase-level feature representations of a sentence. Then the representations would be fed into an LSTM encoder to gain sentence feature representation with a long-term dependency structure. Besides, DSCNN utilized CNN to extract features representations from hidden states of an LSTM layer, which ought to be capable of catching a long-term dependency structure at a certain level. Zhao *et al.* [5] proposed a model named SA-SNN to get more comprehensive representations of the text by adaptively deciding the proportions of global representation to local semantics in classification tasks. Their experimental results indicated that the hybrid model is much better than the single model. However, RNN is very time-consuming, some researchers began to replace RNN in hybrid model with self attention mechanism.

C. HYBRID MODEL OF CNN AND SELF-ATTENTION

Self-attention [13] is a variety of attention mechanism, which reduces the dependencies on external information and is better at capturing the internal correlation of data or features. It has been widely used in various text mining tasks such as textual entailment, reading comprehension, machine translation and other NLP tasks. Compared with RNN, self-attention not only owes ability in capturing long-term dependencies among words but is less time-consuming. Chia and Witteveen [4] first utilized a hybrid model of CNN and self-attention with fewer parameters for text classification tasks. However, this hybrid model just concatenate the two kind of features representation produced by CNN and self-attention respectively to realize the fusion of the two

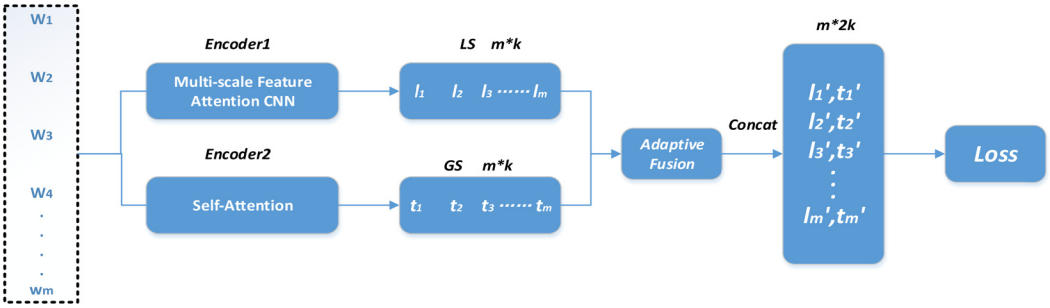

FIGURE 1. MulCNN-Att model architectures.

models. That means that local semantics and global representations will be extracted in the same way for each sentence. However, the structure and grammatical features of each sentence may varies with complex linguistic backgrounds. In this paper, we take an adaptive approach proposed by Zhao *et al.* [5] to distribute the proportions of local semantics to global representations. And in order to extract variable-size features in a sentence, we integrate CNN with multi-scale feature attention [10] into hybrid model to construct a more robust local semantics feature extractor.

For fine-grained emotion classification tasks, we also propose a new type of loss function, which is proven to be efficient in the following experimental results.

III. METHOD

In order to address the problems discussed in Section 2, we are inspired to propose our novel methods. As Figure 1 shows, input text was fed into two encoders: Multi-scale Features CNN encoder and Self-Attention encoder. (1) Endoer1: build the local semantics of sentences; (2) Endoer2: capture the global representations of text; (3) Adaptive Fusion: fuse the local semantic and global representations in an adaptive way; (4) Loss function: train the model with new cross-entropy objective function. The core can be formulated as $M = \phi\ (GS;\ LS)$, where ϕ is a fusion function that combines the local semantic representation $\boldsymbol{LS}$ with the global semantic representation $\boldsymbol{GS}$.

A. ENCODER1 (MULTI-SCALE FEATURE ATTENTION CNN)

1) CNN

Suppose $\boldsymbol{w_t}$ be the d-dimensional word vector of the t-th word in a sentence whose length is m, $w_{t-h+1:t}$ indicates the concatenation of t words $w_{t-h+1}, x_{t-h+2}, \ldots, x_t$ with k number and size h of filters are applied to the input sentence sequence to produce features representation. Formally, filters $\boldsymbol{W_f}$ are applied to window $w_{t-h+1:t}$ to compute $\boldsymbol{x_t}$:

$$x_t = Conv(w_{t-h+1}, w_{t-h+2}, \ldots, w_t) \quad (1)$$

$$= relu(W_f w_{t-h+1:t} + b_f) \quad (2)$$

By the same padding, filters are applied to L possible window sizes in the sequence and the global representation can be

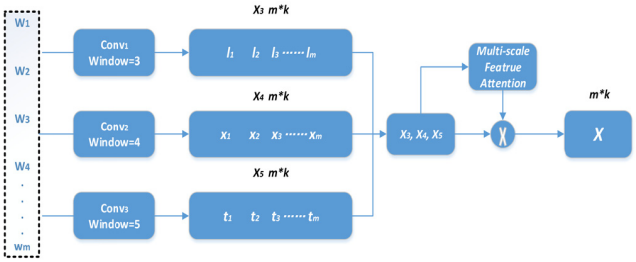

FIGURE 2. Multi-scale feature attention.

represented as $\boldsymbol{h}$:

$$h = [x_1; x_2; \ldots; x_m] \quad (3)$$

2) MULTI-SCALE FEATURE ATTENTION

Multi-scale feature attention [10] is used to construct a more powerful local semantics extractor in our model.

The attention mechanism is mainly composed of two operations: **Filter ensemble** and **Scale reweight**. Filter ensemble mainly aims at developing scalar descriptors s_l^i to represent sentence features of each scale $\boldsymbol{x_l^i}$ at position i and scale l. In order to reweigh these features from different scales, descriptors s_l^i will be utilized by Scale reweight as the input to output a softmax probability distribution of attention weights, i.e., $\boldsymbol{x_1^i}, \boldsymbol{x_2^i}, \boldsymbol{x_3^i}, \ldots, \boldsymbol{x_L^i}$.

3) FILTER ENSEMBLE

As described in Figure 2: According to the experience of previous works, we set window size to 3, 4 and 5. In each convolutional block, we use k filters and then we will obtain $\boldsymbol{X_l} = [\boldsymbol{x_l^1}, \boldsymbol{x_l^2}, \ldots, \boldsymbol{x_l^m}]_{m\times k}$. For each $\boldsymbol{x_l^i} \in \mathrm{R}_k$ indexed by i, it denotes the k-dimensional features representation in the i-th position of a sentence at convolutional layer whose window size equals to l. Then a scalar s_l^i is used to represent each feature vector $\boldsymbol{x_l^i}$:

$$s_l^i = F_{ensem}(x_l^i) = \sum_{j=0}^{k} x_l^i(j) \quad (4)$$

where $F_{ensem}(\cdot)$ indicates a sum function that sums all k components of the input vector. The scalar s_l^i can be utilized as a descriptor of the feature vector representations captured by CNN, because $\boldsymbol{x_l^i}$ is generated by applying k filters on the preceding feature maps and each value in $\boldsymbol{x_l^i}$ is a neural

activation value. Therefore, we can view the sum value of all the components in $\boldsymbol{x_l^i}$ as feature salience.

4) SCALE REWEIGHT

Through filter ensemble, we have obtained s_l^i. Next, all the scalar will be used as input to produce attention weights. In order to adaptively reweight the features representations from different scales, we define $\boldsymbol{x_{atten}^i}$ (the final feature representation) and α_l^i (attention weights) as follows:

$$\mathrm{x}_{atten}^i = \sum_{l=1}^{L} a_l^i x_l^i \sum_{l=1}^{L} a_l^i = 1 \quad \forall i,\ 1 \le i \le m \tag{5}$$

where $\boldsymbol{x_l^i}, \boldsymbol{x_{atten}^i} \in \mathrm{R}_k$, and α_l^i are attention weights. Note that feature maps at different layers have correspondence to the scale of features. For instance, when $l = 2$, $\boldsymbol{X_2}$ corresponds to unigram features, and $\boldsymbol{X_3}$ to trigram features. The attention weights are produced as follows:

$$a_i = soft\max(MLP(s_i)) \tag{6}$$
$$s_i = [s_1^i, s_2^i \ldots, s_L^i] \tag{7}$$
$$a_i = [a_1^i, a_2^i \ldots, a_L^i] \tag{8}$$

where $\boldsymbol{s^i}$ is defined by Eq.(7), and ***MLP*** represents a multi-layer perceptron. After being processed by the attention module, the final local semantics representation: $LS = \boldsymbol{X_{atten}} = [\boldsymbol{x_{atten}^1}, \boldsymbol{x_{atten}^2}, \ldots, \boldsymbol{x_{atten}^m}] \in R_{m\times k}$ is generated, and will be fed into the next adaptive fusion layer.

B. ENCODER2 (SELF-ATTENTION MECHANISM)

In encoder2, we fed m words embeddings ($m\times k$) into the self-attention mechanism. And each embedding vector will be mapped into three vectors: ***query***, ***key*** and ***value***. The output of self-attention is computed as a weighted sum of the values, in which the weight distributed to each value is computed by a compatibility function of the query with its corresponding key. Specifically, We compute the dot products of the query and all keys, divide each by $\sqrt{d_k}$($\sqrt{d_k}$ plays a role of scaling), and apply a softmax function to acquire the weight of each value. Practically, we compute the attention on a set of query vectors simultaneously, packed together into a matrix Q. Similarly, the values and keys are also packed together into matrices V and K. The matrix of outputs are as follows:

$$Attention(Q, K, V) = soft\max((QK^T/\sqrt{d_k})/V) \tag{9}$$

Finally, the out **Attention**($\boldsymbol{Q}$, $\boldsymbol{K}$, $\boldsymbol{V}$)($m\times k$) will be represented as the global representation of the whole text (GS) and fed into the next layer: Adaptive Fusion Layer.

C. ADAPTIVE FUSION LAYER

In this layer, we adopt the method proposed by Zhao *et al.* [5] as the fusion function. Now, we have obtained the local semantics representation LS and the global representation: GS of the text. In this section, we will show how the fusion of these two kinds of representations achieves. The weights of semantic and global information vary according to the sentence itself. For instance, "*The dog the stick the fire burned beat bit the cat*" is rich in global information, and we should pay more attention to its global information. First of all, we average the original word embeddings of the sentence to obtain a concise but available sentence features representation $\boldsymbol{S}$. Secondly, linear transformations are utilized to transform the feature representations into semantic space features representation $\boldsymbol{S_{ls}}$ and global space features representation $\boldsymbol{S_{gs}}$. The transformed dimension is d. Finally, we compute the similarity degree of representations through an inner product. Lastly, softmax is used to normalize weights. The computational steps are listed as follows:

$$(att_{ls}, att_{gs}) = soft\max(p_{ls}, p_{gs}) \tag{10}$$
$$p_{ls} = \rho(S_{ls}, LS) \tag{11}$$
$$p_{gs} = \rho(S_{gs}, GS) \tag{12}$$
$$S_{ls} = W_{ls} \times S + b_{ls} \tag{13}$$
$$S_{gs} = W_{gs} \times S + b_{gs} \tag{14}$$
$$S = (w_1 + w_2 + \ldots + w_m)/m \tag{15}$$

where $\boldsymbol{w_i}$ is a d-dimensional word embedding; $\boldsymbol{S}$ is a features representation of the sentence, computed by averaging the word embeddings of all words, size of d; $\boldsymbol{S_{ls}}$ and $\boldsymbol{S_{gs}}$ are the transformed semantic and global representations, size of n; $\boldsymbol{W_{gs}}$ and $\boldsymbol{W_{ls}}$ are the projection matrixes, size of $n \times d$; $\boldsymbol{b_{ls}}$ and $\boldsymbol{b_{gs}}$ are the biases, size of d; ρ is an average inner product operator; att_{ls} and att_{gs} are the attention weights respectively. Then, we concatenate weighted semantic and global representations to get the final sentence representation:

$$X = [att_{ls} \times LS; att_{gs} \times GS] \tag{16}$$

which will be fed into the final classification layer.

D. LOSS FUNCTION

After getting both semantic and global representations, a fully-connected layer follows. In order to learn the model parameters, most models minimize the following cross-entropy objective function:

$$Loss = -1/m \sum_{i=0}^{m} y_i \times \log(\hat{y_i}) \tag{17}$$

where y_i is the one-hot vector representation of the real label of the i-th sentence; $\hat{y_i}$ is the predicted probability distribution representation and m indicates the number of samples.

As for the fine-grained sentiment classification tasks, we think it's not enough to use cross-entropy as the loss function. For example, for a movie review "I think the movie is fantastic", we intuitively star it 4 or 5 (**1**: **very negative**; **2**: **negative**; **3**: **neutral**; **4**: **positive**; **5**: **very positive**). Because we think the review can't be negative or neutral. In other words, when we humans do such classification tasks, we either finish it correctly, or classify it into the nearest category of the true label, such as classify 1 as 2, 4 as 5.

Motivated by this idea, we suppose machine also act as humans in this kind of task. So, we conduct experiments

on several fine-grained sentiment classification datasets and find about 90% misclassification was made by classifying the instance into its nearest category. So why don't we impose more penalties on the sample whose predicted probability representation weights comparatively more in its nearest category, even if the final classification result is still correct? Below is our proposed loss function:

$$Loss = a_1 * loss_1 + (1 - a_1) * loss_2 \tag{18}$$

$$loss_2 = -1/m \sum_{j=0}^{m} (y^{j\prime}_{true}) \times \log(y^{j\prime}_{pred}) \tag{19}$$

$$y^{j\prime}_{true} = H(y^{j}_{true}) \tag{20}$$

$$y'_{pred} = (y_{pred} \times y_{true}) \times one + (one - y_{true}) \times y_{pred} \tag{21}$$

where $Loss$ is the new loss function, $loss_1$ is the normal cross-entropy; $loss_2$ is the second loss function we establish; m is the amounts of samples; $\boldsymbol{y^{j\prime}_{true}}$ is variation of true label of j-th sample; H means to set the element of 0 in the vector to a number less than 1 and keep their sum to 1; $\boldsymbol{y^{j\prime}_{pred}}$ is variation of predicted probability representation of j-th sample; one is all 1 vector. We suppose that a training sample $\boldsymbol{v_j}$, whose true label is [$\boldsymbol{0}$, $\boldsymbol{1}$, $\boldsymbol{0}$, $\boldsymbol{0}$, $\boldsymbol{0}$] and predicted probability representation is [$\boldsymbol{0.2}$, $\boldsymbol{0.4}$, $\boldsymbol{0.3}$, $\boldsymbol{0.1}$, $\boldsymbol{0}$]. After the above calculation, $\boldsymbol{y^{j\prime}_{true}}$ would be turned to [$\boldsymbol{0.4}$, $\boldsymbol{1}$, $\boldsymbol{0.4}$, $\boldsymbol{0.2}$, $\boldsymbol{0}$], and $\boldsymbol{y^{j\prime}_{pred}}$ would be turned to [$\boldsymbol{0.8}$, $\boldsymbol{0}$, $\boldsymbol{0.7}$, $\boldsymbol{0.9}$, $\boldsymbol{1}$]. It's obvious that the value of cross-entropy of $\boldsymbol{y'_{true}}$ and $\boldsymbol{y'_{pred}}$ would increases if its predicted probability representation weights more in its nearest category. So, our new loss function can maximize classification accuracy theoretically while minimizing nearest neighbor misclassifications.

IV. EXPERIMENTS SETTINGS

A. DATASETS INTRODUCTION

We use 7 popular large-scale text classification datasets proposed by Zhang and Lee [1] in our experiments. We list the concise information of all datasets in II. AG is a news corpus and DBPedia is an ontology datasets which comes from Wikipedia. Yelp and Amazon are reviews corpus for which we ought to predict the sentiment. Concretely, P. indicates that we only need to predict the polarities of the reviews, while F. means that we need to predict the star (1-5) number of the datasets. Besides, Yahoo! Answers (Yah. A.) is a question answering dataset. It can easily be found that these datasets come from various domains and contains different sizes. This is helpful to verify the generalization ability of model.

B. PARAMETERS SETTINGS

We limit the vocabulary and convolution kernel size that we set for these datasets in Table 3. 300D GloVe 840B vectors [14] was adopted as our pre-trained word embeddings. Word embeddings are updated along with other parameters while training model. We utilize Zeiler and Adadelta [15] as our optimizer to optimize all the trainable parameters. All the input vectors dimensions and hidden states are set to 100 or 128. And batch size is set to 128.

TABLE 2. Experimental settings.

Corpus	Kernel Size	Vocabulary Size
AG	(1,3,5,7)	100k
DBP.	(1,3,5,7)	500k
Yelp. P.	(1,3,5,7,9)	200k
Yelp. F.	(1,3,5,7,9)	200k
Yahoo	(1,3,5,7,9)	500k
Amz. F.	(1,3,5,7,9)	500k
Amz. P.	(1,3,5,7,9)	500k

C. BASE MODELS

As Table 4 demonstrates, the block at the top shows the traditional approaches and some other neural networks model that are not based on RNN or CNN such as the linear model [1], and Capsnets [16]. The RNN based models are placed in the second block and CNN based models follows. The fourth block lists some hybrid models based on RNN, CNN and self-attention.

D. EXPERIMENT INDEX

Our experimental index is error rate.

V. EXPERIMENTS RESULTS AND ANALYSIS

Table 4 and 6 show the classification experimental results of the performance of MulCNN-Att and baseline methods on seven public text datasets. It can be seen that the performance of MulCNN-Att is far greater than all other approaches. Moreover, the results show that MulCNN-Att has better performance after the introduction of new loss function in Section 3.4. According to the text classification results (As shown in Table 6), MulCNN- Att outperforms other approaches in accuracy (its metrics are 0.5 percentages higher than basic CNN and RNN models).

Therefore, the classification performance is also proved to be advanced by the introduction of the new loss function.

A. COMPARISON WITH SINGLE MODELS

As Table 4 shows that MulCNN-Att outperforms all the single models (listed in the first, second and the third blocks) based on CNN and self-attention and other models in 6 datasets. Especially in Amz. P. and Amz. F, which are relatively difficult task datasets due to longer sequence and variable grammar information. MulCNN-Att even achieve an increase of 1 to 2 percentage points in accuracy. Experiments results prove the advantages of hybrid models over single models in text classification.

In addition, we also conduct additional four experimental results as the ablation experiments to prove the validity of MulCNN-Att. Especially, CNN, SA and MSA respectively represents the basic CNN model, self-attention mechanism and CNN with multi-scale feature attention.

As to CNN-SA and CNN-MSA, both achieve about 0.5% increase compared with basic CNN. So, it is obvious both self-attention and multi-scale feature attention work.

TABLE 3. Datasets information (SA represents sentiment analysis, and QA represents question answering).

	AG	DBP	Yah. A.	Yelp P.	Yelp F.	Amz. F.	Amz. P.
Task types	News	Ontology	QA	SA	SA	SA	SA
Train dataset	120k	560k	1.4M	560k	650k	3.6M	3M
Test dataset	7.6k	70k	60k	38k	50k	400k	650k
Average Lengths	45	55	112	123	155	93	91
Classes Number	4	14	10	2	5	5	2

TABLE 4. Experiments results (evaluation index: error rate).

	AG	DBP.	Yelp P.	Yelp F.	Yah. A.	Amz. F.	Amz. P.
(Zhang et al., 2015 [1])	7.64	1.31	4.36	40.14	28.96	44.74	7.98
(Joulin et al., 2017 [17])	7.5	1.4	4.3	36.1	27.7	39.8	5.4
(Qiao et al., 2018 [18])	7.2	1.1	4.7	35.1	26.3	39.1	4.7
(Ren et al., 2018 [16])	7,6	1.7	3,5	**34.1**	26.1	39.0	5.0
(Wang et al., 2018 [19])	7.55	0.98	4.69	35.91	**22.58**	-	-
(Yogatama et al., 2017 [20])	7.9	1.3	7.4	40.4	26.3	-	-
(Yang et al., 2016 [21])	-	-	-	-	24.2	36.4	-
(Zhang et al., 2015 [7])	9.51	1.55	4.88	37.95	28.80	40.43	4.93
(Zhang et al., 2015 [1])	8.55	1.37	4.60	39.58	28.84	42.39	5.51
(Conneau et al., 2017 [8])	8.67	1.29	4.28	35.28	26.57	**37.00**	4.28
(Duque et al,. 2019 [25])	9.45	-	4.80	36.80	-	-	-
(Giannakopoulos et al., 2019 [22])	-	0.84	-	-	-	-	-
(Chia et al., 2019 [4])	8.8	1.5	-	-	29.0	-	-
(Zhao et al., 2018 [5])	6.90	1.04	4.02	35.13	25.44	38.04	4.59
CNN	7.25	1.30	4.45	37.21	27.84	40.40	4.95
CNN-SA	7.13	1.12	4.63	36.72	25.03	39.45	4.73
CNN-MSA	7.11	1.03	4.33	36.53	24.40	39.00	4.52
CNN-SA-MSA	7.03	0.95	4.04	35.25	24.47	37.93	4.30
MulCNN-Att	**6.94**	**0.73**	**3.78**	**34.10**	23.95	**37.00**	**4.23**

TABLE 5. Adaptive weights analysis.

Sentence	Att_{ls}	Att_{gs}
1 : What's the conversion rate between pounds and dollars? (numeric)	**0.81**	0.19
2: What does bank bond mean? (describe)	0.22	**0.78**

Self-attention can help CNN to capture long-terms dependencies among words, leading to a more comprehensive sentence representation. And multi-scale feature attention can help CNN to capture variable-size features, which will make the final text features representations more accurate.

It should be noted that, in Yelp. F and Yahoo, the results of our model is relatively poor. The reason is that those better models adopt some special skills such as Region Embedding and hierarchical CNN [10], which can help to improve the performance of model in large-scale datasets and multi-class datasets. So, in Yahoo, they reach higher accuracy.

B. COMPARISON WITH HYBRID MODELS

As we can see, the fourth part of Table 4 is the experimental results of the hybrid models. Our hybrid model outperforms all other hybrid models based on CNN, RNN and self-attention. Especially, compared with CNN-SA-MSA, MulCNN-Att increases about 0.3%-1% accuracy in each dataset. It proves that taking an adaptive approach to automatically determine the contribution degree of each model to classification indeed works in text classification tasks. With the adaptive approach, the model finds a balance between the global representation and local semantics of the text.

In order to show visually how this adaptive approach works, we list experiment results in Table 5. For the first sentence, its weight att_{ls} (local semantics weights) is larger than att_{gs} (global representation weights), because the key phrase "rate" is more important than "what's....", so our model distribute more weights to the local semantics of the text captured by CNN. Conversely, for the second sentence, the expression "what does. . .mean" is more vital to classification tasks. Therefore, MulCNN-Att assign larger

weights to att_{gs}. This means global features extracted by self-attention will play a leading role when classifying a sentence to some category.

C. NEW LOSS FUNCTION ANALYSIS

1) EFFECTIVENESS ANALYSIS

In Table 6, we list the experimental results conducted on three fine-grained emotion classification tasks. As we can see, each model with our new loss function (**SL**) makes some improvement in accuracy, up to 0.7% most, in every subtask. In order to better explain how our method works, we make a quantitative analysis in Table 7. The numbers in Table 7 represent the number of misclassified samples (1, 2, 3, 4 respectively represent the absolute value of the difference between predicted category and true label). As for each dataset, the first row indicates experimental results of our model without the new loss function and the second row indicates the results with the new loss function. We can easily see that the amounts of whose absolute difference is 1 declines a lot when the new loss function is added. In Amz. F. and Yelp. F, the number reduced by about 3000 and 1000. Figure 3 and Figure 4 are a graphical representa-tion of our new loss function experimental results. For Figure 3, the ordinate is the proportion of the samples divided into nearest neighbor categories. And red line, blue line and yellow line respectively corresponds to the experiment results of $\boldsymbol{a_1} = 0$, 1 and 0.6. We can see that for the model $\boldsymbol{a_1} = 1$ (without new loss function), the proportion is about 91%. But it drops to about 90% when we set $\boldsymbol{a_1} = 0.6$. For Figure 4, the ordinate represents the error rate of classification. The model works best on all datasets when $\boldsymbol{a_1}$ is set to 0.6. Therefore, we can draw a conclusion from these two figures: the new loss function not only reduces the amounts of nearest neighbor misclassification samples, but also reduces the total volume of misclassification samples. Therefore, accuracy reach a higher level.

TABLE 6. New loss function experiment results.

	SST-1	Yelp F.	Amaz F.
CNN	51.3	37.21	40.40
CNN+SL	**50.93**	**36.66**	**39.93**
LSTM	54.1	36.90	41.13
LSTM+SL	**53.66**	**36.54**	**40.46**
Ours	51.13	34.63	37.59
Ours+SL	**50.81**	**34.10**	**37.00**

2) WEIGHT ANALYSIS

$\boldsymbol{a_1}$ is actually a harmonic coefficient between the new loss function and the normal cross entropy function ($\boldsymbol{a_1} = 1$ indicates the model don't adopt the new loss function, $\boldsymbol{a_1} = 0$ means the model don't adopt the normal cross entropy function). We can notice that from Figure 3, 4 when we set $\boldsymbol{a_1} = 0$, both misclassification rate and proportion of the samples divided into their nearest neighbor categories decline compared to the model of $\boldsymbol{a_1} = 1$ and $\boldsymbol{a_1} = 0.6$. The reason

TABLE 7. Quantitative analysis of new loss function.

	1	2	3	4	test sample
SST-1	1031	73	16	10	
	1002	76	26	19	2210
Amaz-F	220634	21001	1804	896	
	218029	22226	1703	297	650000
Yelp-F	16044	901	316	74	
	15290	950	498	270	50000

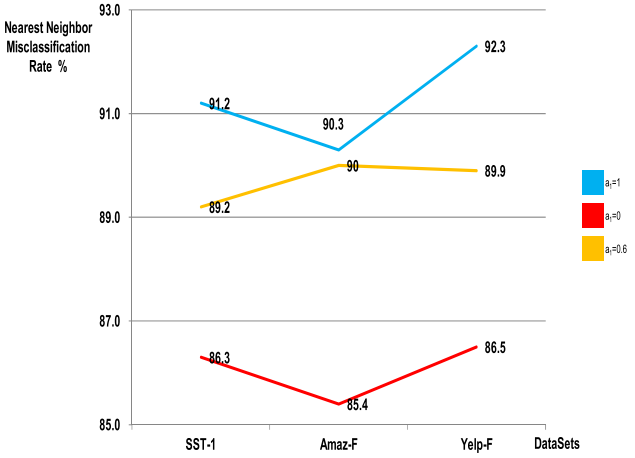

FIGURE 3. Visualization of nearest neighbor misclassification results.

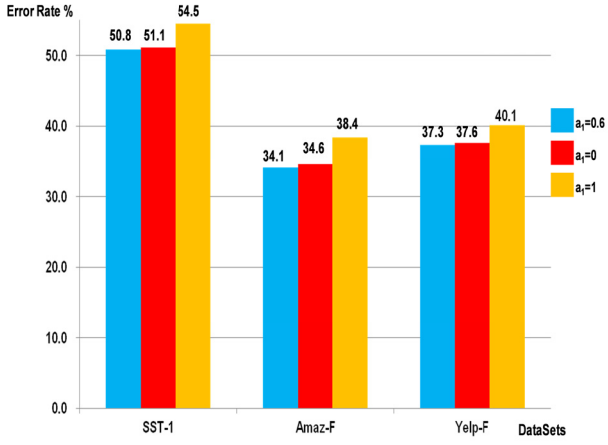

FIGURE 4. Visualization of new loss function experiments results.

is that when $\boldsymbol{a_1} = 0$, our model would impose more penalties on the samples with relatively high probability distribution on theirs nearest neighbor categories. So, although the amounts of nearest neighbor classification error samples is smaller, the overall classification effect will become very poor at the same time.

The experimental results prove the correctness of our idea: with the help of the new loss function, the model does impose more penalties on the sample with relatively high probability distribution in its nearest category. This method not only reduces the amounts of nearest neighbor misclassification samples, but also brings about total misclassification

samples loss. Therefore, new hybrid model achieves better experiments results.

VI. CONCLUSION

In this paper, we present a new hybrid model—MulCNN-Att of CNN and self-attention. Under complex linguistic backgrounds, MulCNN-Att can adaptively determine which model is more or less important to classification tasks. In this way, we can maximize the advantages of this two models. In addition, multi-scale feature attention is also introduced into our hybrid model. The attention can automatically select task-friendly and effective multi-gram features from texts. Besides, as for fine-grained emotion tasks, we also establish a new loss function, which aims at maximizing classification accuracy while minimizing nearest neighbor misclassifycations. Experiments results demonstrate that classification accuracy increases by up to 2 percentage point compared with classic hybrid models based on CNN and self-attention.

REFERENCES

[1] D. Zhang and W. S. Lee, "Question classification using support vector machines," in *Proc. 26th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2003, pp. 26–32.
[2] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," 2014, *arXiv:1406.1078*. [Online]. Available: http://arxiv.org/abs/1406.1078
[3] Z. Jianqiang and G. Xiaolin, "Comparison research on text preprocessing methods on Twitter sentiment analysis," *IEEE Access*, vol. 5, pp. 2870–2879, 2017.
[4] Y. Ken Chia, S. Witteveen, and M. Andrews, "Transformer to CNN: Label-scarce distillation for efficient text classification," 2019, *arXiv:1909.03508*. [Online]. Available: http://arxiv.org/abs/1909.03508
[5] J. Zhao, "Adaptive learning of local semantic and global structure representations for text classification," in *Proc. 27th Int. Conf. Comput. Linguistics*, 2018, pp. 2033–2043.
[6] Y. Kim, "Convolutional neural networks for sentence classification," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2014, pp. 1746–1751.
[7] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 649–657.
[8] A. Conneau, H. Schwenk, L. Barrault, and Y. Lecun, "Very deep convolutional networks for text classification," in *Proc. 15th Conf. Eur. Chapter Assoc. Comput. Linguistics*, 2017, pp. 1107–1116.
[9] R. Johnson and T. Zhang, "Deep pyramid convolutional neural networks for text categorization," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, 2017, pp. 562–570.
[10] S. Wang, M. Huang, and Z. Deng, "Densely connected CNN with multi-scale feature attention for text classification," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 4468–4474.
[11] C. Zhou, C. Sun, Z. Liu, and F. C. M. Lau, "A C-LSTM neural network for text classification," 2015, *arXiv:1511.08630*. [Online]. Available: http://arxiv.org/abs/1511.08630
[12] Z. Li, A. Ren, J. Li, Q. Qiu, Y. Wang, and B. Yuan, "DSCNN: Hardware-oriented optimization for stochastic computing based deep convolutional neural networks," in *Proc. IEEE 34th Int. Conf. Comput. Des. (ICCD)*, Oct. 2016, pp. 678–681.
[13] K. M. Yoo, Y. Shin, and S.-G. Lee, "Improving visually grounded sentence representations with self-attention," 2017, *arXiv:1712.00609*. [Online]. Available: http://arxiv.org/abs/1712.00609
[14] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1532–1543.
[15] M. D. Zeiler, "ADADELTA: An adaptive learning rate method," 2012, *arXiv:1212.5701*. [Online]. Available: http://arxiv.org/abs/1212.5701
[16] H. Ren and H. Lu, "Compositional coding capsule network with k-means routing for text classification," 2018, *arXiv:1810.09177*. [Online]. Available: http://arxiv.org/abs/1810.09177
[17] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," in *Proc. 15th Conf. Eur. Chapter Assoc. Comput. Linguistics*, 2017, pp. 427–431.
[18] C. Qiao, "A new method of region embedding for text classification," in *Proc. ICLR*, 2018, pp. 1–9.
[19] G. Wang, C. Li, W. Wang, Y. Zhang, D. Shen, X. Zhang, R. Henao, and L. Carin, "Joint embedding of words and labels for text classification," 2018, *arXiv:1805.04174*. [Online]. Available: http://arxiv.org/abs/1805.04174
[20] D. Yogatama, C. Dyer, W. Ling, and P. Blunsom, "Generative and discriminative text classification with recurrent neural networks," 2017, *arXiv:1703.01898*. [Online]. Available: http://arxiv.org/abs/1703.01898
[21] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, 2016, pp. 1480–1489.
[22] A. Giannakopoulos, M. Coriou, A. Hossmann, M. Baeriswyl, and C. Musat, "Resilient combination of complementary CNN and RNN features for text classification through attention and ensembling," in *Proc. 6th Swiss Conf. Data Sci. (SDS)*, Jun. 2019, pp. 57–62.
[23] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, Z. Fang, and H. Lu, "Dual attention network for scene segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3146–3154.
[24] J. Chung, "Empirical evaluation of gated recurrent neural networks on sequence modeling," in *Proc. NIPS*, Dec. 2014, pp. 1–5.
[25] A. B. Duque, "Squeezed very deep convolutional neural networks for text classification," in *Proc. Int. Conf. Artif. Neural Netw.* Cham, Switzerland: Springer, 2019, pp. 193–207.
[26] A. Vaswani, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.
[27] R. Collobert, "Natural language processing (almost) from scratch," *J. Mach. Learn. Res.*, vol. 12, pp. 2493–2537, Aug. 2011.
[28] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," 2014, *arXiv:1404.2188*. [Online]. Available: http://arxiv.org/abs/1404.2188
[29] R. Wang, Z. Li, J. Cao, T. Chen, and L. Wang, "Convolutional recurrent neural networks for text classification," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2019, pp. 2267–2273.
[30] Z. C. Lipton, J. Berkowitz, and C. Elkan, "A critical review of recurrent neural networks for sequence learning," 2015, *arXiv:1506.00019*. [Online]. Available: http://arxiv.org/abs/1506.00019
[31] Q. T. Thieu, M. Luong, J.-M. Rocchisani, N. M. Sirakov, and E. Viennet, "Efficient segmentation with the convex local-global fuzzy Gaussian distribution active contour for medical applications," *Ann. Math. Artif. Intell.*, vol. 75, nos. 1–2, pp. 249–266, Oct. 2015.
[32] Y. Shi, "Deep LSTM based feature mapping for Query classification," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, 2016, pp. 1501–1511.
[33] Z. Tu, Z. Lu, Y. Liu, X. Liu, and H. Li, "Modeling coverage for neural machine translation," 2016, *arXiv:1601.04811*. [Online]. Available: http://arxiv.org/abs/1601.04811
[34] Y. Xiao and K. Cho, "Efficient character-level document classification by combining convolution and recurrent layers," 2016, *arXiv:1602.00367*. [Online]. Available: http://arxiv.org/abs/1602.00367
[35] W. Yin, K. Kann, M. Yu, and H. Schütze, "Comparative study of CNN and RNN for natural language processing," 2017, *arXiv:1702.01923*. [Online]. Available: http://arxiv.org/abs/1702.01923

CHAOFAN WANG is currently pursuing the M.S. degree. His research interest includes natural language processing.

SHENGGEN JU received the Ph.D. degree. He is also a Professor. His research interests include natural language processing and data mining.

YUEZHONG LIU research interests include natural language processing and data mining.

RUN CHEN is currently pursuing the Ph.D. degree. She is also an Assistant Professor. Her research interest includes data mining.

XIAOMING HUANG received the master's degree. He was a Senior Engineer. He is currently a Deputy General Manager with Shenzhen CyberAray Network Technology Company Ltd. His research interests include cognitive domain security, cryptography and information security theory, trusted computing and trusted network technology, and computer and communication security issues. He is a member of the Sichuan Electronic Information Expert Group.