2018 Tenth International Conference on Advanced Computational Intelligence (ICACI)
March 29–31, 2018, Xiamen, China

Sentence Representation and Classification Using Attention and Additional Language Information

Hongli Deng
College of Computer Science
Sichuan University
Chengdu, P.R.China
hongli_deng@foxmail.com

Lei Zhang*
College of Computer Science
Sichuan University
Chengdu, P.R.China
leizhang@scu.edu.cn

Lituan Wang
College of Computer Science
Sichuan University
Chengdu, P.R.China
wanglituan@stu.scu.edu.cn

***Abstract*—Sentence representation is one of the foundational tasks in natural language processing, and long short term memory (LSTM) is a widely used tool to deal with the variable-length sentence. In this paper, a new LSTM-based sentence representation model is proposed for sentence classification task. By introducing a self-supervised method in the process of learning the hidden representation of the sentence, the proposed model automatically capture the syntactic and semantic information from the context and used as additional language information to learn better contextual hidden representation. Moreover, instead of using the final hidden representation of LSTM or the max (or average) pooling of the hidden representations over all the time step, we propose to generate the global representation of the sentence by combining all contextual hidden representations in an element-wise attention manner. We evaluate our model on three sentence classification tasks: sentiment classification, question type classification, and subjectivity classification. Experimental results show that the proposed model improves the accuracy of sentence classification compared to other sentence representation methods in all of the three tasks.**

Keywords*—*Attention; LSTM; language information; sentence representation; sentence classification

I. Introduction

Sentence representation is very crucial in the field of natural language processing and has attracted much research interest in recent years [1] [2] [3]. The main objective is to learn the representation of the sentences which are inputs of various natural language related tasks, such as sentiment analysis, document summarization, machine translation, and so on. In general, the representation of a sentence is generated by combining the vectors of the words in the sentence. Therefore, the meaning of a sentence is determined by its word vectors and the combination rules. At present, the most successful technique for word vectorization is the word embedding [4] [5], and the existing sentence representation methods are usually based on combining the word embeddings of all the words in the sentence to generate a fixed dimensional representation.

Neural networks have been studied for many years [6] [7] [8]. In various neural networks, LSTM [9] is one of the successful sequence combination models and it has been widely applied in combining a sequence of words for sentence representation [10] [11]. The LSTM-based sentence representation usually uses the final hidden representation generated by LSTM [10] or the max(or average) pooling of all the hidden representations [11]. However, because each word of the sentence sequentially overwrites the activation of the hidden unit over the unfolded time step of LSTM, the final hidden representation of LSTM may be more biased towards the latter words of the sentence. As for the pooling combination, the max combination just contains the max hidden representations but discards others, and the average pooling does not consider the different importance of each hidden representation.

In recent years, the attention mechanism has been used to combine a sequence of words for sentence representation in many natural language-related tasks [12] [13] [14]. But the work that uses attention for LSTM-based sentence representation is still limited. In [15], attention mechanism is applied to merge the history, future, and local contexts at the same position, but it is not proposed as the combination rule of a sequence of words. The authors in [3] proposed to calculate an attention matrix to extract the representation of sentences in different semantic aspects, in order to meet the requirements of subsequent different tasks. However, most of the data is prepared for single task, where only one aspect of the semantics is needed. For example, in the task of sentiment classification, we focus on the sentiment of the sentence, but do not need the subject-related semantic information. In those cases, the calculation of the attention matrix leads to unnecessary complexity. In this paper, we propose to use a group of self-attentive vectors to combine the hidden representations across all the time steps of LSTM. And the attention-based combination is executed element by element to generate more accurate sentence representations.

Furthermore, although LSTM can obtain better performance than simple recurrent neural networks, the supervised training of LSTM-based model usually requires a large number of labeled datasets. Unfortunately, the labeled language datasets are often limited. The strategies to overcome this difficulty generally fall into two categories. The first consists of universal semi-supervised methods. These methods divide the network training into two stages, the unsupervised pre-training phase, in which a large amount of unlabeled data is used to learn the

This work was supported by National Key R&D Program of China (grant 2016YFC0801800); National Natural Science Foundation of China (grants 61332002,61772353); Fok Ying Tung Education Foundation (grant 151068).

word embedding, and the supervised training phase using the pre-trained word embeddings as the initial vectors of words. This approach has now been widely used and achieved good results [1] [2] [15]. In addition to the existing tagged data, this kind of methods introduce additional textual information from unlabeled data to improve the sentence representation. Another category is to provide more human annotated data, such as the stanford sentiment treebank (SST) dataset used in [16], where the phrase-level supervised data is added. However, the manual labeling tasks is labor intensive and time consuming.

This paper presents a self-supervised method to solve the problem of insufficient data. Inspired by the ability of the language model (LM) [4] [17] that it can extract the combined representation of a sequence of words from its context, the proposed model automatically capture the syntactic and semantic information from the context of the sentence itself and used as additional language information to improve the learning of hidden representation at each time step. The sentence serves as the label of itself to provide more labeled data. Taking the task of LSTM-based sentence classification as example, the traditional model trains LSTM just using the sentence-level label as the target. But the proposed method further improves the hidden representation at each time step by using the next word as the word-level target, making it more in line with the current context. This strategy brings a large amount of labeled data from the data itself, but without extra labeling tasks.

We validate our model on several sentence classification tasks. The experimental results show that the proposed method achieves the best results on several benchmark datasets. The main contributions of this paper are as follows:

- The introduction of the self-supervision method at word level enable the model to make full use of the syntactic and semantic information of the sentence itself, improving the learning capacity of LSTM to obtain better contextual hidden representations.
- The element-wise attentive combination of all the contextual hidden representations enables the model to generate more accurate sentence representation.
- Experimental results on three sentence classification tasks show that the proposed method obtains more accurate classification performance than other methods.

The rest of this paper is organized as follows: Section 2 provides the detailed description of the proposed model. In Section 3, the proposed model is evaluated on three sentence classification tasks. Finally, Section 4 offers the conclusions and future works.

II. The proposed model: AttBLSTM-AL

Sentence encoding is the core of this model. To generate better sentence representations, we employed a two-step strategy to encode sentences. The first step is to use the bidirectional LSTM with additional language information (Bidirectional LSTM-AL) module to encode the input sentence into a sequence of hidden representations. Then, based on the attention mechanism, a group of self-attentive vectors that denote the correlation between each hidden representation and the downstream task are calculated, and the sentence representation is obtained by combining all the hidden representations using the self-attentive vectors. Finally, a classifier is added on top of the sentence encoding module for sentence classification. Fig.1 shows the architecture of the proposed model. In this section, we describe this model in detail.

A. Bidirectional LSTM-AL

Due to the limited number of supervised training datasets, the model often falls into the overfitting problem. In recent years, researchers usually use a large amount of unlabeled data to pre-train word embeddings, which greatly improves the representation of the words. However, all of these pre-trained word embeddings are universal representations, which do not consider the specific context in current sentence. For the LSTM-based sentence representation models, the LSTM layer is the key to obtain the contextual hidden representation from the universal word embeddings. In order to improve the learning capacity of LSTM layer, the proposed model combines bidirectional LSTM and additional language (AL) information to encode a sequence of words recurrently and generate better contextual hidden representations at each time step.

In detail, given a sentence of length T, $\mathbf{x} = <x_1, x_2, ..., x_T>$, two LSTM-ALs are used to encode this sentence in forward and backward direction, respectively, where x_i is the i-th word of the sentence. We denote the LSTM-ALs in two directions as forward LSTM-AL and backward LSTM-AL, respectively. Each word x_i is expressed as one-hot form that is a vector with the same length as the size of the vocabulary. Only one element of this vector is 1 and others are zeros. Each LSTM-AL includes three layers: Embedding, LSTM and AL layers.

a) Embedding layer: First, forward LSTM-AL and backward LSTM-AL read the words in the sentence from x_1 to x_T and from x_T to x_1, respectively, one word a time. Then, the embedding layer transforms the one-hot form of each word into a continuous real-valued word embedding. Word embeddings are encoded by column vectors in an embedding matrix $W_{emb} \in \mathbb{R}^{d_w \times |V|} (d_w \ll |V|)$, where d_w is the dimension of the word embedding and V is the size of vocabulary. Each column of W_{emb} corresponds to the word embedding of one word in the vocabulary. Each word x_t $(t = 1, 2, \cdots, T)$ is transformed into its word embedding by using the matrix-vector product:

$$\hat{x}_t = W_{emb} x_t, \quad (1)$$

where the matrix W_{emb} is a parameter to be learned, and the dimension of word embedding d_w is a hyper-parameter to be chosen by the user.

b) LSTM layer: Two LSTM layers are used to recurrently compute the history-related and future-related hidden representation at each time step in both directions, respectively. As shown in Fig.1, the forward LSTM processes the sentence from left to right, and the backward LSTM from right to left. The history-related hidden representation of x_t, $\overrightarrow{h_t}$, is

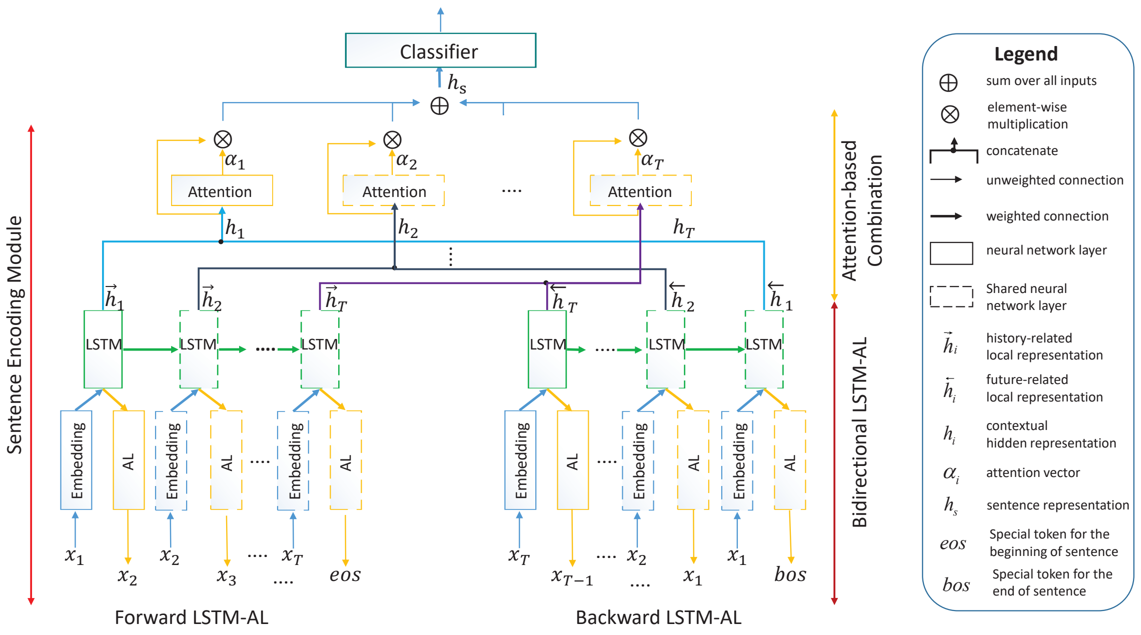
Classifier
h_s
α_1
α_2
α_T
Attention
Attention
....
Attention
h_1
h_2
h_T
Sentence Encoding Module
Attention-based Combination
Bidirectional LSTM-AL
$\overrightarrow{h}_1$
$\overrightarrow{h}_2$
$\overrightarrow{h}_T$
$\overleftarrow{h}_T$
$\overleftarrow{h}_2$
$\overleftarrow{h}_1$
LSTM
Embedding
AL
x_1 x_2 x_2 x_3 x_T eos
x_T x_{T-1} x_2 x_1 x_1 bos
Forward LSTM-AL
Backward LSTM-AL
Legend
sum over all inputs
element-wise multiplication
concatenate
unweighted connection
weighted connection
neural network layer
Shared neural network layer
$\vec{h}_i$ history-related local representation
$\overleftarrow{h}_i$ future-related local representation
h_i contextual hidden representation
α_i attention vector
h_s sentence representation
eos Special token for the beginning of sentence
bos Special token for the end of sentence

Fig. 1. Architecture of AttBLSTM-AL.

computed by the forward LSTM layer based on the previous representation $\overrightarrow{h_{t-1}}$ and the word embedding of current input word x_t . The backward LSTM layer computes the future-related hidden representation of x_t, $\overleftarrow{h_t}$, according to the succeeding hidden representation $\overleftarrow{h_{t+1}}$ and the word embedding of x_t.

$$\begin{cases} \overrightarrow{h_t} = \overrightarrow{LSTM}(\hat{x}_t, \overrightarrow{h_{t-1}}), \\ \overleftarrow{h_t} = \overleftarrow{LSTM}(\hat{x}_t, \overleftarrow{h_{t+1}}), \end{cases} \tag{2}$$

where $\overrightarrow{LSTM}$ and $\overleftarrow{LSTM}$ correspond to the forward and backward LSTM, respectively.

Then, the history-related and future-related hidden representations of x_t, $\overrightarrow{h_t}$ and $\overleftarrow{h_t} \in \mathbb{R}^{d_h}$, are concatenated as the contextual future-related representation of x_t,

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

where [ ] denotes the concatenation, d_h is the dimension of the LSTM layer and $h_t \in \mathbb{R}^{2d_h}$.

c) AL layer: The AL layer is proposed to improve the learning capacity of LSTM layer by using the language information from the context of the given sentence, so as to obtain a better hidden representation. Specifically, based on the idea of language model, the AL layer uses the sentence itself as its target, providing more labeled data to guide the LSTM to extract better hidden representation related to the given context in the process of sequence coding.

For a sentence **x**=$<x_1, x_2, ..., x_T>$, besides the sentence-level label, the forward LSTM is also trained through using the word-level target $<x_2, x_3, ..., x_T, eos>$. Similarly, backward LSTM is trained using $<x_{T-1}, x_{T-2}, ..., x_1, bos>$ as the word-level target. The AL layer here uses the softmax function. At time t, the two AL layers of forward LSTM-AL and backward LSTM-AL take $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ as input, respectively, to compute the probability distribution over each word in the vocabulary as,

$$\begin{cases} \overrightarrow{y_t} = softmax(W_{yh}^{(1)}\overrightarrow{h_t} + b_y^{(1)}), \\ \overleftarrow{y_t} = softmax(W_{yh}^{(2)}\overleftarrow{h_t} + b_y^{(2)}), \end{cases} \tag{4}$$

where $W_{yh}^{(1)}, W_{yh}^{(2)} \in \mathbb{R}^{|V| \times d_h}$ are the weight matrices from forward and backward LSTM layers to corresponding AL layers, respectively. $b_y^{(1)}$ and $b_y^{(1)} \in \mathbb{R}^{|V|}$ are the bias vectors. $\overrightarrow{y_t}$ and $\overleftarrow{y_t}$ are the output vectors of size $|V|$. $softmax$ function ($softmax(z_k) = \frac{e^{z_k}}{\Sigma_p e^{z_p}}$) ensures that $\overrightarrow{y_t}$ and $\overleftarrow{y_t}$ form a valid probability distribution over each word in the vocabulary, i.e., all elements of $\overrightarrow{y}_t$ and $\overleftarrow{y}_t$ are between 0 and 1, and the sum of all the elements is 1. Each element of $\overrightarrow{y_t}$ and $\overleftarrow{y_t}$ is the estimated conditional probability of a word in the vocabulary. The conditional probabilities that the two AL layers output their targets are

$$\begin{cases} P_{AL}(x_{t+1}|x_1, x_2, \cdots, x_t) = \overrightarrow{y_t} \otimes x_{t+1}, \\ P_{AL}(x_{t-1}|x_T, x_{T-1}, \cdots, x_t) = \overleftarrow{y_t} \otimes x_{t-1}. \end{cases} \tag{5}$$

B. Attention-based combination

In order to combine all the contextual hidden representations and to effectively distinguish their different significance, the attention layer is added on top of the bidirectional LSTM-AL to compute a self-attentive vector for each contextual hidden representation. These self-attentive vectors operate like gates to control the flow of the hidden representations into the global representation of the sentence. We present to use a feedforward neural network layer to recurrently produce a group of self-attentive vectors,

$$\alpha_t \quad = \sigma(W_{att}h_t + b_{att}), \quad (1 \leq t \leq T), \tag{6}$$

where $W_{att} \in \mathbb{R}^{2d_h \times 2d_h}$ and $b_{att} \in \mathbb{R}^{2d_h}$ are the weight matrix and bias vector of this layer, respectively. The feedforward neural network layer is shared by the contextual hidden representation at each time step. The self-attentive vector $\alpha_t \in \mathbb{R}^{2d_h}$ determines which hidden representation should be put more attention on than other hidden representation when predicting the sentence. All the contextual hidden representations are then combined in a element-wise weighted sum to generate the representation of the input sentence,

$$h_s \quad = \sum_{t=1}^{T} \alpha_t \otimes h_t. \tag{7}$$

C. Classifier

The sentence encoding module is connected by a classifier to estimate the probability that the input sentence belongs to each category. Supposed that the class label of input sentence **x** is k, the output probability distribution over all the classes is computed as,

$$P_c \quad = softmax(W_{ca}h_s + b_c), \tag{8}$$

and the estimated probability that **x** is attributed to the k-th class is P_c^k, where W_{ca} is the weight matrix of the connection from attention layer to the classifier. P_c is the estimated probability distribution over each word in the vocabulary. P_c^k denotes the k-th element of P_c.

D. Model training

Given a training set $\{(\mathbf{x}^{(n)}, l^{(n)})\}$ with N samples, AttBLSTM-AL with parameter set Θ is trained by minimizing the cross-entropy of the estimated and true probability distributions over all the samples to search the optima of the parameters for this model. The cross-entropy consists of two parts: the conventional categorical cross-entropy and that in the two LC layers. Then the cost is defined as the cross-entropy of the two parts over $\mathbf{x}^{(n)}$ $(1 \leq n \leq N)$, i.e.,

$$\begin{aligned} J(\Theta) = -\frac{1}{N}\sum_{n=1}^{N}\Bigg(&\sum_{k=1}^{K} 1\{l^{(n)} = k\} logP_c^k \\ &+ \beta\sum_{t=1}^{T_n}\bigg(logP_{AL}\left(x_t^{(n)}\middle| x_1^{(n)}, x_2^{(n)}, \cdots, x_{t-1}^{(n)}\right) \\ &+ logP_{AL}\left(x_t^{(n)}\middle| x_T^{(n)}, x_{T-1}^{(n)}, \cdots, x_{t+1}^{(n)}\right)\bigg)\Bigg), \end{aligned} \tag{9}$$

where $1\{\cdot\}$ is the indicator function that $1\{$a true statement$\}$=1 and $1\{$a false statement$\}$=0. $x_t^{(n)}$ is the one-hot form of the t-th word of $\mathbf{x}^{(n)}$. $l^{(n)}$ is the class label of $\mathbf{x}^{(n)}$. $P_{AL}(x_t^{(n)} | x_1^{(n)}, x_2^{(n)}, \cdots, x_{t-1}^{(n)})$ and $P_{AL}(x_t^{(n)} | x_T^{(n)}, x_{T-1}^{(n)}, \cdots, x_{t+1}^{(n)})$ are the conditional probability of the word $x_t^{(n)}$ given its history and future context, respectively. T_n is the length of the n-th sentence $\mathbf{x}^{(n)}$. β is the ratio of additional language information.

III. Experiments

To evaluate the effectiveness of the proposed AttBLSTM-AL, a series of experiments were conducted on the widely used benchmark corpora, including the tasks of sentiment classification, question type classification, and subjectivity classification. The detailed experiments and results are described in this section.

A. Implementation Details and Hyperparameters

The proposed AttLSTM-LC for sentence classification was implemented in Lua using Torch, and ran on an NVIDIA TESLA K20 GPU. Publicly available word embeddings trained from Google News were used as the pre-trained word embeddings. The dimension of each word embedding is 300. For all the tasks, the word embeddings were fine-tuned during the training phase. The size of the LSTM layer was set to 128 and the two LSTM layers shared the same parameters. The weights and biases of the model were randomly initialized over a uniform distribution in [-0.05,0.05]. 50% dropout was used for regularization. The back-propagation algorithm with AdaGrad stochastic optimization method was used to train the network through time with learning rate of 0.05. The weights are updated after a mini-batch of 20 sentences. After each training epoch, the network is tested on validation set. Other hyperparameters for each task were tuned on the validation set. We used the classification accuracy as the evaluation metric to measure the performance. In the training, a bad counter B_c and the maximum number of bad count that can be tolerated $maxB_c$ were used. Once the accuracy of the validation set obtained in current epoch was less than that of previous epoch, $Acc_{valid} < Acc_{valid-prev}$, then $B_c = B_c+1$. All the networks stopped training when $B_c > maxB_c$, and the parameters with the best validation performance was selected as the final parameters to be evaluated the model.

B. Sentiment Classification

In this task, we execute a group of experiments focused on classifying the sentiment of movie reviews through three datasets:

- MR [18]: Movie Reviews polarity dataset with one sentence per review. It contains 5331 positive reviews and 5331 negative ones. The objective of this dataset is to classify each review into either positive or negative by its overall sentiment polarity. 10-fold cross validation was used for testing.
- SST-5 [16]: Stanford Sentiment Treebank dataset, an extension of MR dataset. It contains 11,855 sentences that are split into 8544/1101/2210 for train/dev/test, respectively. Each sentence is fine-grained labeled as very positive, positive, neutral, negative, or very negative.
- SST-2: The binary labeled version derived from SST-5, in which the neutral reviews are removed. It contains 9313 sentences split into 6920/872/1821 for train/dev/test.

We also give the statistics of the datasets in Tab. I for more comparisons.

TABLE I. Statistics of Movie Review Datasets. K, $|V|$ and N are The Number of Classes, Vocabularies, and Sentences, Respectively. Avg-l and Max-l Denote The Average and Maximum Sentence Length. CV Denotes the Cross Validation.

Dataset	K	$\|V\|$	N	Avg-l	Max-l	Train/Valid/Test
MR	2	18765	10662	20	56	10-fold CV
SST-5	5	21701	11855	18	53	8544/1101/2210
SST-2	2	21701	9613	19	53	6920/872/1821

The performance of the proposed method is compared with several basic baseline methods and state-of-arts approaches. The comparison systems are described as follows:

- cBoW: Continuous Bag-of-Words. It combines all the word embeddings of the words in the sentence into the representation of the sentence by performing max-pooling or average-pooling operation.
- NB, NB-SVM, and MNB: Naive Bayes, Naive Bayes SVM, and Multinomial Navie Bayes models [19], respectively. They are three models related to Naive Bayes with unigram or bigram features as inputs. The word embeddings are not used for these models.
- RAE, MV-RNN, and RNTN: Recursive Auto-encoder [20], Matrix-vector Recursive Neural Network [21], and Recursive Neural Tensor Network [16], respectively. These models are recursive neural networks-based models that rely on the parse tree to combine the word embeddings into a vector representation of sentence.
- CNN, TBCNN, and DSCNN: Standard Convolutional Neural Network (CNN) [1], tree-based CNN [22], and Dependency Sensitive CNN [23], respectively. CNN generate sentence representation using convolution and pooling operation over word embeddings. TBCNN is an extension of CNN based on tree-structured network topologies. DSCNN builds the contextual representation by processing word embeddings via LSTM, and subsequently uses CNN to achieve the combination.
- LSTM, BiLSTM, and Tree-LSTM: Standard Long Short-Term Memory neural network, Bidirectional LSTM, and Tree-Structure LSTM, respectively [24]. LSTM and BiLSTM compose all the word embeddings by taking into account the order of the word in the sentence, and use the last hidden output of each LSTM network as the sentence representation. Tree-LSTM is a generalization of LSTM to tree-structured network topologies.

The comparison results are given in Tab. II. The accuracies of comparison systems are retrieved from their original papers. It shows that AttBLSTM-AL consistently outperforms the compared models in the tasks of sentiment classification. We attribute the success of AttBLSTM-AL to its power in modeling better contextual hidden representation of each token, and then the better sentence representation is generated from the attention-based composition. The results further show that the improvements obtained on the SST-5 dataset are more obvious compared to that on MR and SST-2. This is because: as shown in Tab. I, SST-5 has the shortest average sentence length in the three datasets. Due to the problem of gradient vanishing in RNN networks, the shorter the sequence is, the easier it is to train the network. Thus, the supervised training for SST-5 can generate better hidden representation at each time step, relative to other two datasets that with longer sentences.

TABLE II. Accuracy (%) of Sentiment Classification.

Model	MR	SST-5	SST-2
cBoW	77.2	42.8	81.5
NB	-	41.0	81.8
NB-SVM	79.4	-	-
MNB	79.0	-	-
RAE	77.7	43.2	82.4
MV-RNN	79.0	44.4	82.9
RNTN	-	45.7	85.4
CNN	81.5	48.0	87.2
TBCNN	-	51.4	-
DSCNN	82.2	50.6	88.7
LSTM	-	46.4	84.9
BiLSTM	-	49.1	87.5
Tree-LSTM	-	51.0	88.0
AttBLSTM-AL	82.8	52.2	89.3

C. *Question Type Classification*

The objective of this experiment is to classify a question into one of many question types. The TREC questions dataset [25] is used in this experiment, which includes six different question types (person, location, numeric information, etc.). This data has 9592 different vocabularies. The longest sentence has 37 words and the average length of all the sentences is about 10 words. The training and test sets consist of 5452 and 500 labelled questions, respectively. We take 90% of the training sets as the validation set.

The experimental results are presented in Tab. III. Because that not all the baselines in sentiment classification experiment provided the results for this task, our model is just compared with some of the baselines in this experiment. We can see that the improvement on this dataset is the most significant compared to other datasets. It may be own to that the average and maximum sentence length in the TREC dataset is the smallest of all the datasets.

TABLE III. Accuracy of Question Type Classification on TREC Dataset.

Model	Accuracy(%)
cBoW	87.3
SVM	95.0
DCNN	93.0
TBCNN	96.0
LSTM	89.4
DSCNN	95.6
AttBLSTM-AL	98.4

D. *Subjectivity Classification*

In this section, the subjectivity dataset SUBJ [26] is used to classify a sentence as being subjective or objective. It contains 10000 sentences, and 21323 different vocabularies. The average and maximum sentence length of this dataset is

23 and 120, respectively, which are both larger than that of the datasets in other two tasks. There is no standard train/test split and thus 10-fold corss validation is used.

Tab. IV presents the experimental results on SUBJ dataset. The comparison shows that AttBLSTM-AL obtained some improvements over other methods. However, some sentences in SUBJ dataset are very long, and even the longest sentence length reaches 120 tokens. Although the method proposed in this paper improves the learning capacity of LSTM layer, the training of the network is still difficult due to the too long sentences. As a result, the improvement on SUBJ is not obviously compared to other tasks.

TABLE IV. Accuracy of Subjectivity Classification on SUBJ Dataset.

Model	Accuracy(%)
cBoW	91.3
NB-SVM	93.2
MNB	93.6
CNN	93.4
DSCNN	93.9
AttBLSTM-AL	94.2

IV. Conclusion

In this paper, a new sentence representation model for sentence classification task is described. The introduction of the language information contained in the sentence is able to enhance the learning capacity of LSTM, promoting the model to learn better contextual representation at each time step. Moreover, the element-wise attentive combination rule is proposed, which combines all the contextual representations to generate more accurate sentence representation. The experiments on three different sentence classification tasks verify that the proposed model outperforms most of the existing competing approaches. Although our model improves the learning capacity of LSTM layer, it is still difficult to learn better hidden representation of LSTM for the time step in tail. Therefore, in future work we will further investigate new method to solve this difficulty. We will also execute vocabulary clean to remove the tokens that are not words, just as the CNN-based baselines did.

References

[1] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[2] C. N. Dos Santos and M. Gatti, "Deep convolutional neural networks for sentiment analysis of short texts." in *COLING*, 2014, pp. 69–78.

[3] Z. Lin, M. Feng, C. N. d. Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio, "A structured self-attentive sentence embedding," *arXiv preprint arXiv:1703.03130*, 2017.

[4] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of machine learning research*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[5] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[6] L. Zhang and Z. Yi, "Dynamical properties of background neural networks with uniform firing rate and background input," *Chaos, Solitons & Fractals*, vol. 33, no. 3, pp. 979–985, Aug 2007.

[7] Q. Guo, J. Jia, G. Shen, L. Zhang, L. Cai, and Z. Yi, "Learning robust uniform features for cross-media social data by using cross autoencoders," *Knowledge-Based Systems*, vol. 102, pp. 64–75, Jun 2016.

[8] L. Wang, L. Zhang, and Z. Yi, "Trajectory predictor by using recurrent neural networks in visual tracking," *IEEE Transactions on Cybernetics*, vol. 47, no. 10, pp. 3172–3183, Oct 2017.

[9] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[10] H. Margarit and R. Subramaniam, "A batch-normalized recurrent network for sentiment classification," *Advances in Neural Information Processing Systems*, 2016.

[11] J. Y. Lee and F. Dernoncourt, "Sequential short-text classification with recurrent and convolutional neural networks," *arXiv preprint arXiv:1603.03827*, 2016.

[12] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[13] A. M. Rush, S. Chopra, and J. Weston, "A neural attention model for abstractive sentence summarization," *arXiv preprint arXiv:1509.00685*, 2015.

[14] A. P. Parikh, O. Täckström, D. Das, and J. Uszkoreit, "A decomposable attention model for natural language inference," *arXiv preprint arXiv:1606.01933*, 2016.

[15] Y. Zhang, M. J. Er, R. Venkatesan, N. Wang, and M. Pratama, "Sentiment classification using comprehensive attention recurrent models," in *Neural Networks (IJCNN), 2016 International Joint Conference on*. IEEE, 2016, pp. 1562–1569.

[16] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the 2013 conference on empirical methods in natural language processing*, 2013, pp. 1631–1642.

[17] H. Deng, L. Zhang, and L. Wang, "Global context-dependent recurrent neural network language model with sparse feature learning," *Neural Computing and Applications*, pp. 1–13, 2017.

[18] B. Pang and L. Lee, "Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales," in *Proceedings of the 43rd annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2005, pp. 115–124.

[19] S. Wang and C. D. Manning, "Baselines and bigrams: simple, good sentiment and topic classification," in *Meeting of the Association for Computational Linguistics: Short Papers*, 2012, pp. 90–94.

[20] R. Socher, J. Pennington, E. H. Huang, A. Y. Ng, and C. D. Manning, "Semi-supervised recursive autoencoders for predicting sentiment distributions," in *Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John Mcintyre Conference Centre, Edinburgh, Uk, A Meeting of Sigdat, A Special Interest Group of the ACL*, 2011, pp. 151–161.

[21] R. Socher, B. Huval, C. D. Manning, and A. Y. Ng, "Semantic compositionality through recursive matrix-vector spaces," in *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 2012, pp. 1201–1211.

[22] L. Mou, H. Peng, G. Li, Y. Xu, L. Zhang, and Z. Jin, "Discriminative neural sentence modeling by tree-based convolution," 2015.

[23] R. Zhang, H. Lee, and D. Radev, "Dependency sensitive convolutional neural networks for modeling sentences and documents," pp. 1512–1521, 2016.

[24] K. S. Tai, R. Socher, and C. D. Manning, "Improved semantic representations from tree-structured long short-term memory networks," *Computer Science*, vol. 5, no. 1, p. : 36., 2015.

[25] X. Li and D. Roth, "Learning question classifiers," in *Proceedings of the 19th international conference on Computational linguistics-Volume 1*. Association for Computational Linguistics, 2002, pp. 1–7.

[26] B. Pang and L. Lee, "A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts," in *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2004, p. 271.