Applied Intelligence (2019) 49:3283–3292
https://doi.org/10.1007/s10489-019-01456-x

Enhancing sentence embedding with dynamic interaction

Jinsong Xie[1] · **Yongjun Li**[1] · **Qiwei Sun**[1] · **Yi Lin**[1]

Published online: 1 April 2019

Abstract

Sentence embedding is a powerful tool in many natural language processing subfields, such as sentiment analysis, natural language inference and questions classification. However, previous work just integrates the final states, which are the output of encoder of multiple-layer architecture, with average pooling or max pooling as the final sentence representation. Average pooling is simple and fast for summarizing the overall meaning of sentences, but it may ignore some significant latent semantic features considering that information is flowing through the multiple layers. In this paper, we propose a new dynamic interaction method for improving the final sentence representation. It aims to make the states of the last layer more conducive to the next classification layer by introducing some constraint from the states of the previous layers. The constraint is the product of dynamic interaction between states of intermediate layers and states of the upper-most layer. Experiments can surpass prior state-of-the-art sentence embedding methods on 4 datasets.

Keywords Sentence embedding · Sentiment analysis · Self-attention · Deep neural networks

1 Introduction

Sentence embedding plays an important role in many NLP tasks. Sentiment analysis is to study and analyze subjective information from some specific contexts and indicates how people feel about movies, music, products, etc. The discrimination of question type helps to label the questions in Q&A websites like Quora. The classification of subjective and objective sentences is one of the sentiment analysis tasks. Sentence embedding is devoted to capturing the latent semantic relationship of words in a sentence. Nowadays, sentence embedding has attracted considerable research interest and developed into a rapid growth stage. Early traditional methods like the Bag-of-Words or Skip-Gram model fail to capture the complete contextual information, because they just treat a sentence as some discrete collections of words and in that way, they lose a lot of latent semantic information because there is no difference between sentences that obey language syntax and sentences that just combine several words in any order. With the development of encoder such as LSTM [9], contextual information can be extracted from ordered words that obey natural language syntax when a sentence flows through the encoder. It's worth noting that substantial efforts have been made to sentence embedding in portability and representation. Up to date, there are generally two kinds of popular approaches for sentence embedding. The first one is Skip-thought vectors [16] by using unsupervised learning. Sentence embedding is a byproduct in unsupervised learning during which it learns to decide whether a coherent succession of sentences is in a sentence. Unsupervised learning methods work well in a wide range of fields and can apply on all corpora which contain a coherent succession of sentences. Skip-thought vectors works well as a classic example of unsupervised learning for sentence embedding. It makes some alteration to the skip-gram model proposed for word embedding which tries to predict the surrounding words of a given word, and focuses on predicting the sentences surrounding a sentence. The second one based supervised learning shows more abundant work such as recurrent neural networks (RNN) [1, 4] and convolution neural networks (CNN) [13, 15]. Supervised learning needs to confirm the specific task

✉ Yongjun Li
liyj@scut.edu.cn

Jinsong Xie
cspub1@mail.scut.edu.cn

Qiwei Sun
csqivigor@mail.scut.edu.cn

Yi Lin
201620131079@mail.scut.edu.cn

[1] School of Computer Science and Engineering, South China University of Technology, Guangzhou, 510006, China

with labeled datasets and requires the appropriate sizes of datasets for good-quality embedding. InferSent in [4] makes some efforts to change the opinion that supervised learning methods for sentence embedding provided lower-quality sentence embedding than unsupervised learning for a period of time. InferSent adapts a bi-directional LSTM encoder and groups its states with a max-pooling operator. The simplicity of InferSent's architecture makes it interesting and effective.

Recently, many variants of CNN and RNN taking word embedding as input dominate the way of constructing the sentence embedding. Ding et al. [5, 14] proposed densely connected RNN and LSTM where the original feature from the input will be retained to the last or uppermost layer. The sentence embedding is converted into a fixed-sized vector with average pooling that can represent the complete information of sentence and the vector will be sent into the final classification layer. However, RNN-based models, max or average pooling is usually used to integrate the hidden states at the uppermost layer to obtain the final sentence-encoding vector and CNN also often adopts the same approach for aggregating sentence embedding. To generalize the diversity of sentence, attention mechanism plays an important role in enhancing the understanding of natural language. Attention mechanism was firstly proposed in machine translation task, then it develops ability to handle more challenging NLP tasks such as question answering, paraphrase identification and recognizing textual entailment. Self-attention is some kind of simplify attention that focuses on its own words without participation of other sentences. It usually computes attention weighted vectors that help to discriminate the informative words. [3] made use of vector-based multi-head attention with average pooling or scalar self-attention and it achieved significant improvement on sentiment classification than other sentence-encoding-based models. However, the final fixed-sized vector is static during prediction. On the other hand, the model will be confused about the importance of words when it becomes deeper and deeper.

To improve the adaptability of the inference and the robustness of the model, we adopt and modify the dynamic-routing method in Capsule Network [26]. Based on densely connected Bi-LSTM [5, 38], we fully make use of the interaction between the states of the uppermost layer and the states of the intermediate layers. In this paper, we propose a new dynamic interaction method to enhance the semantic information of the sentence embedding. Our dynamic interaction method iteratively collects informative vector over the states of the uppermost layer by common inner product and element-wise multiplication with the weight vector of states of the rest layers. Before each iteration, we apply transformation to the states of the uppermost layer which is inspired by the multi-head attention mechanism [36]. In general, our approach achieves state-of-the-art results among sentence embedding methods on several tasks, including sentiment analysis, question type classification, and subjectivity classification.

Our main contributions are as follows:

- We proposed a dynamic interaction method which explores and extracts the beneficial information from the intermediate layers to correct the final sentence representation from the uppermost layer. This method is some kind of attention and makes the multi-layer network paying attention to the different aspects in a sentence. It is portable to other encoders of multiple layers. Our method is easier to implement than other self-attention approaches and doesn't introduce more complexity but better performance than the average pooling method for sentence embedding.
- We further improve the model performance by combining with densely connected LSTM and multi-head attention mechanism.
- We achieve state-of-the-art results of sentence embedding methods on 4 out of 6 tasks.

The rest of this paper is organized as follows : Section 2 presents the related work about this model. In Section 3, the basic concepts about recurrent neural networks and the reason why we choose the densely connected Bi-LSTM as the encoder layer are discussed in detail. We further develop our dynamic interaction algorithm on the output of encoder layer and explain the architecture of our model. In Section 4, we make a full comparison to other model and explain the portability of our method with supplementary experiments using a different encoder. The implementation and results of the experiments have been discussed in detail. Finally, we end the paper with further conclusion in Section 5.

2 Related work

Preparing pre-trained word vectors for the word embedding layer becomes popular since it's able to capture the intrinsic semantic feature from large scale unlabeled corpus [20, 24, 35]. Pre-trained word vectors such as Glove become one of the standard components of many state-of-the-art models, including sentiment analysis, natural language inference [3] and question answering [6].

Currently there exist many models for sentence embedding in natural language processing, mapping sentence into high dimensional vector space. The very first models are proposed based on recurrent neural network models with all kind of variants such as LSTM, Bi-RNN [27], TreeRNN [29]. With the further progress, CNNs [15, 21, 39]

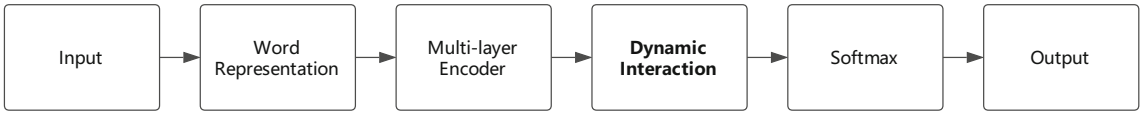

Fig. 1 The architecture of our model. The "Input" is the raw sentence and the "Output" expresses the predicted label of the given sample. The "Multi-layer Encoder" can by replaced by any multi-layer encoder and we adopt the densely connected Bi-LSTM encoder. The bold "Dynamic Interaction" represents our proposed method

develop new mechanisms to accumulate information over words or characters with convolution filters. Considering other aspects apart from modifying the inherent structure of common models such as [2, 4, 30, 37] develop max-pooling, internal interaction and generalized pooling on Bi-directional LSTM encoder.

In terms of so many experiments show that networks with deeper layers make more progress in image classification. The residual connection unit introduced in [7, 8, 33] has become a common and widely used trick in the deep network structures to stable the training process. Furthermore, [11] accelerated the information flowing from lower to upper layers without losing information of the lower layer by concatenation operation. Ding et al. [5] also proposed a multi-layer LSTM model called DC-Bi-LSTM. The model concatenates the current hidden states with inputs of the current layer to form the new inputs of the next layer, and it works well on several NLP tasks.

Recently, sentence embedding with self-attention [18, 19, 28, 37, 38] has introduced various dynamic weight vectors, which fully exploring the intrinsic feature of sentence or text. Motivated by these works and learning from the dynamic routing algorithm [26], we enhance the interaction of each layer by our dynamic interaction method.

3 Model

In this section, we describe the proposed model in detail. The architecture of the model is shown in Fig. 1. Firstly, we map the raw sentence into vector representation through word representation layer. Then, the underlying syntax features will be fully extracted by densely connected Bi-LSTM layer. For the coherence and acceptance of content, we introduce the key concepts of LSTM and then discuss the advantage of densely connected Bi-LSTM encoder. Then, we apply the dynamic interaction method on the states of multiple layers to extract the abstract features. Finally, we classify the raw sentence through the softmax layer.

3.1 Word representation

The input of the model is a variable-length sentence $S = \{w_1, \ldots, w_n\}$. Like many sentence embedding methods, our word representation concatenates the word embedding of each word and character embedding learned from character. For the word embedding we use the pre-trained word embedding vectors from Glove(42B). We construct character embedding with multi-filters 1D convolution for feature extraction and max-pooling for feature synthesis. We randomly initialize the character embedding as input. As the same way at sharing weights in traditional convolution layer, the character embedding parameter is shared. In terms of the powerful representation ability of LSTM used in the next part, we just simply concatenate the word embedding and character embedding as input for the next encoder layer without extra operations. When we input the raw sentence of n words into the word representation layer, the layer output the vector representation $X \in \mathbb{R}^{n \times d_x}$ of the sentence. For simplicity, we express X as follows:

$$X = \text{concat}([X_{\text{word}}, X_{\text{character}}]) \tag{1}$$

where X_{word} and $X_{\text{character}}$ is the the output of word embedding and character embedding.

3.2 Long short-term memory

Recurrent Neural Network(RNN) is a class of neural sequence model, processing words on variable-length input sentences in a cyclic way. It uses an internal memory mechanism taking word embedding $x = [x_1, \ldots, x_n]$ as input to construct the hidden state h_t in a recursive way. At time step t, the state h_t is a combination of projection on its previous hidden state h_{t-1} and current input x_t , a non-linear activation function f is following :

$$h_t = f(Wx_t + Uh_{t-1} + b) \tag{2}$$

where U, W and b are the trainable parameters of RNN, the hyperbolic tangent function usually express the non-linearity.

Long Short-Term Memory networks (LSTM) is enhanced variant of RNN. It was designed for dropping the vanishing or exploding gradient problems and can capture long-term relation of sentences. LSTM introduces memory cells and sets three gates to control information flowing in the model. For instance, we have the memory cell c_t, the input gate i_t, the output gate o_t, the forget gate f_t, the input at current time step x_t is in $\mathbb{R}^d$ and the current hidden state

h_t is in $\mathbb{R}^h$. At each time step, the LSTM state's updated rule is described as follows:

$$\begin{pmatrix} i_t \\ f_t \\ o_t \\ \tilde{h}_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} P \begin{pmatrix} x_t \\ h_{t-1} \end{pmatrix} \tag{3}$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \tilde{h}_t \tag{4}$$

$$h_t = o_t \circ \tanh(c_t) \tag{5}$$

where σ is the sigmoid function, $\circ$ is element-wise multiplication and P is affine functions with different weight matrix for different operators. For readability, we ignore the bias terms. Note the gates are all in $[0, 1]^h$, the input gate restricts how the input is selected, the forget gate controls how much the previous cell is dropped, the output gate controls how the new state is composed. The hidden state h_t at time t is finally obtained by selective reservation of information through multiple gated units. For the better illustration of the networks, the internal structure of cells of RNN and LSTM are shown in Fig. 2. The operator P is different through different branches.

3.3 Densely connected Bi-LSTM

The traditional stacked RNNs take the hidden state sequence of previous layer as the next input sequence and form the multiple RNN layers. Precisely, let x_t^l and h_t^l be the input and hidden state vector of l^{th} layer at time step t, and H_l plays the l^{th} layer's transformation, the stacked RNNs is compositions of L times of H_l. In our model, H_l will be replaced by Bi-LSTM which can get information from the states of past and future simultaneously. A traditional stacked RNNs can be expressed as follows:

$$h_t^l = H_l(x_t^l, h_{t-1}^l), \qquad x_t^l = h_t^{l-1}. \tag{6}$$

The number of layers in stacked RNNs is usually less than 5 since the networks are hard to train when they become deeper. However, many architectures have performed better by using deeper layers with more tricks. The residual connection [8] is commonly used to make robust and deeper networks. Furthermore, [11] accelerated the information flowing from lower to upper layers without losing information of the lower layers with the concatenation operation. However, the residual connection using the element-wise summation operation has been showed by experiments that may impede and slow down the information flow in the computation graph. Following [5, 14], we adopt the densely connection and concatenate the previous input with the generated hidden state as the next input:

$$\overrightarrow{h_t^l} = H_l(x_t^l, \overrightarrow{h_{t-1}^l}), \qquad x_t^l = [\overrightarrow{h_t^{l-1}}; x_t^{l-1}]. \tag{7}$$

$$\overleftarrow{h_t^l} = H_l(x_t^l, \overleftarrow{h_{t-1}^l}), \qquad x_t^l = [\overleftarrow{h_t^{l-1}}; x_t^{l-1}]. \tag{8}$$

$$h_t^l = [\overrightarrow{h_t^l}; \overleftarrow{h_t^l}]. \tag{9}$$

where the hidden state h_t^l of the l^{th} layer at time t is a concatenation of state of forward and backward . Given the number of units d_h^j of the j^{th} layer, the hidden states of the j^{th} layer is in $\mathbb{R}^{d_u^j \times n}$, where $d_u^j = 2d_h^j$.

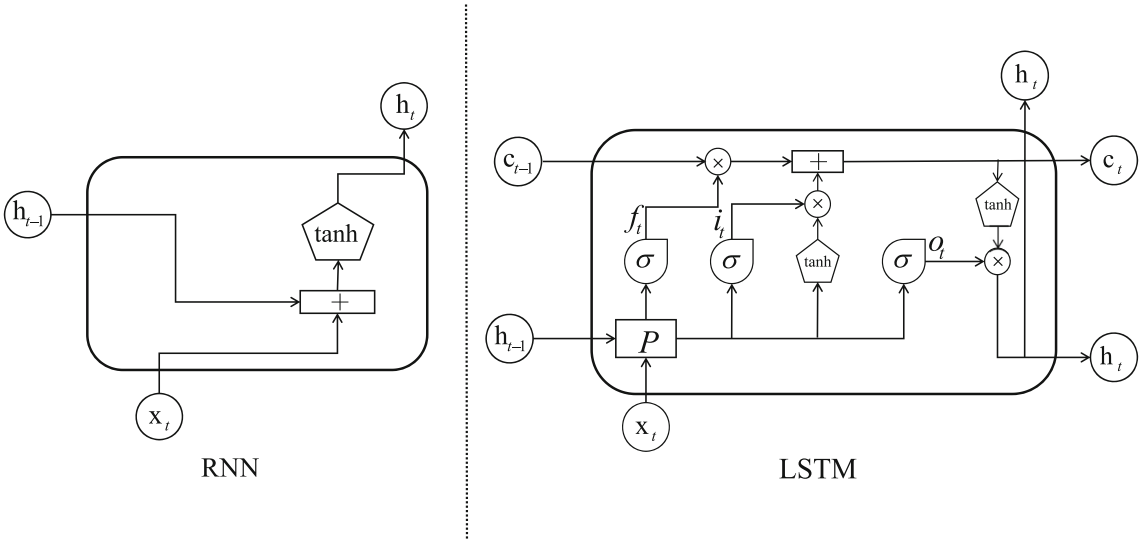

Fig. 2 The internal structure of cells of RNN and LSTM. The pentagon containing "tanh" and the "σ" are activation functions. The circles outsides the frame are input and output vectors. The "P", "i_t", "f_t" and "o_t" are the same ones in (3). The "$\times$" expresses the element-wise multiplication and the "+" is the element-wise summation or concatenation

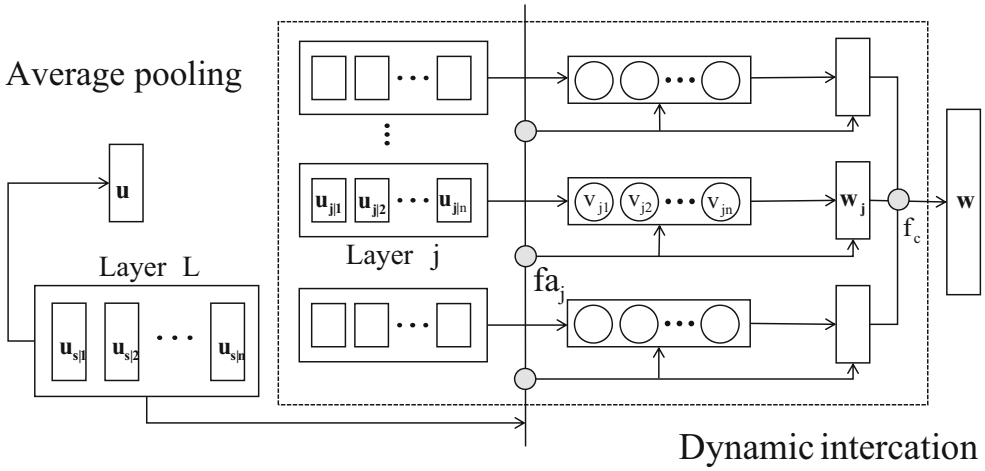

Fig. 3 The proposed dynamic interaction method and average pooling. The "$\mathbf{u_{s|i}}$", "$\mathbf{u_{j|i}}$" and "v_{ji}" are the same ones in Algorithm 1. The grey circles are attention functions f_a and concatenation operator f_c. Rectangle element is the representation of vector and the circle element is the representation of scalar. The dotted box contains the key elements in the iterations of Algorithm 1

3.4 Dynamic interaction between layers

Dynamic interaction between states of intermediate layers and states of the uppermost layer in multiple-layer network has not been noticed in previous work. Some with self-attention like [38] focused on the states of the uppermost layer and did not consider the existence of states of the intermediate layers. Our experiments show that the dynamic interaction method does help to extract the deeper semantic information from sentences. In [26] work, dynamic routing algorithm computed weighted vector between two continuous layers. In our model, the states U_s of the uppermost layer are considered as the most valuable information of sentence, and the states U_j of the intermediate layers help to correct the weights of each word by dynamically interact with U_s by being vectors. Given the number of training layers L, $j \in [1, L-1]$, the detail of dynamic interaction method is expressed as follows:

Algorithm 1 Dynamic interaction.

procedure INTERACTION($\mathbf{u_{j|i}}, \mathbf{u}_{s|i}, r$)
  for L-1 iterations **do**
    for n iterations **do**
      $\mathbf{u}_{s|i} = \text{LeakyReLU}(W_j \mathbf{u}_{s|i} + b_j)$
      $b_{ij} = 0$
    for r iterations **do**
      for all i: $c_{ij} = \frac{exp(b_{ij})}{\sum_k exp(b_{kj})}$
      $\mathbf{s_j} = \text{ReduceMean}([c_{1j}\mathbf{u_{j|1}}, \ldots, c_{nj}\mathbf{u_{j|n}}])$
      $\mathbf{v_j} = \text{Sigmoid}(\mathbf{s_j})$
      for all i: $b_{ij} = b_{ij} + \text{Sum}(v_{ji}\mathbf{u_{s|i}})$
    $\mathbf{w_j} = \text{ReduceMean}([v_{j1}\mathbf{u_{s|1}}, \ldots, v_{jn}\mathbf{u_{s|n}}])$
  return all $\mathbf{w}_j$

where $W_j \in \mathbb{R}^{d_c \times d_u}$ is a projection matrix of the j^{th} step which inspired by multi-head attention in the [36] work, $b_j \in \mathbb{R}^{d_c}$ is bias term of the j^{th} step, d_c is set to the half of d_u, where d_u is the number of double of hidden units of the L^{th} layer. The number of iteration is r, c_{ij} is the weighted coefficient of i^{th} state vector of the j^{th} layer, $\mathbf{u}_{j|i}$ is the i^{th} state vector of the j^{th} layer, and $\mathbf{u}_{s|i} \in \mathbb{R}^{d_u}$ is changing with different transformation when applied at a different step. Our dynamic interaction method is just enough to boost the previous memory for the entire training process through deeper layers. At step j, we obtain the interactive vector $w_j \in \mathbb{R}^{d_c}$ which is the result of interaction between the U_j and the j^{th} U_s. After $L-1$ rounds of interaction, we get $L-1$ vectors in total, then the final excepted vector representation of sentence embedding is expressed as follows:

$$w = \text{concat}[w_1, \ldots, w_{L-1}] \tag{10}$$

where $w \in \mathbb{R}^{(L-1)*d_c}$ represents the full semantic information of a sentence that has ability to make a appropriate respond to dynamic changes of semantics of sentences in different contexts. Then w_j will be sent to the next layer for different downstream tasks. To distinguish the difference between average pooling and our dynamic interaction method, we make a figure for the explanation. In Fig. 3, the process of Algorithm 1 is presented in the form of a graph where f_c is the concatenation operator and fa_j is the j^{th} attention for U_s. The left side of Fig. 3 shows the process of average pooling which averages up the output states of layer L to a fixed-sized vector u. The right side of Fig. 3 visualizes the key steps described in Algorithm 1. For simplicity, we only list the j^{th} step for explanation and the output states of layer L is also the input of dynamic interaction. At step j, we firstly apply attention function fa_j on the collection of $u_{s|i}$, $i \in \{1, \ldots, n\}$, and by the participation of layer j, we obtain informative weights v_j to form the valuable vector w_j. We repeat it by the participation of different layers in $L-1$ steps. The f_c operator generalizes these iterations and output the final vector w which is relative to the u of average pooling.

Table 1 Summary information of benchmark datasets

Data	Class	len	N	$\|V\|$	Test
MR	2	20	10662	18765	CV
Subj	2	23	10000	21323	CV
TREC	6	10	5952	9592	500
SST-1	5	18	11855	17836	2210
SST-2	2	19	9613	16185	1821
CR	2	19	3775	5340	CV

Note, class: Number of target labels. len: Average sentence length. N: Data set size. $|V|$: Vocabulary size. test: Test set size. CV means there is no standard train/test set division and 10-fold cross validation is used

3.5 Output layer

A softmax classifier is following the dynamic interaction of pooling-like layer and to predict our excepted labels. The softmax layer takes the final representation w as input and output the probability p_i belonging the class i for each class. The class with the maximum probability will be chosen as the final prediction result.

4 Experiments

4.1 Datasets

We evaluate our model on several data sets, the summary information is listed in Table 1.

- **MR**: Each review with one sentence for movie reviews. It labels the reviews with positive or negative. Pang and Lee [23]
- **Subj**: Subjectivity dataset that labels the sentence with subjective or objective. Pang and Lee [22]
- **TREC**: Question dataset where the task is to classify a question into 6 question types (location, person, numeric information, etc.) [17]
- **SST-1**: The Stanford Sentiment Treebank [31] is a sentiment dataset, containing 11,855 people's reviews about movie. Each review has five level of sentiment: very negative, negative, neutral, positive, and very positive. The training, validation, test set sizes are split as 8544,1101,2210.
- **SST-2**: The simplification of SST-1, the prediction is converted into binary classes from the five classes. The neutral sentences are dropped. The division of data is changed as 6920,872,1821.
- **CR**: Customer reviews of various products. Task is to label positive or negative reviews. Hu and Liu [10]

4.2 Implementation details

Here we describe our implementations in detail.

- Word representation: In the data processing stage, the case of words is ignored and other tricks are similar as the previous work doing with the same datasets. We use the pre-trained Glove vectors that is in the 42B-300D file. The word embedding is trainable at training time. For character embedding, character dimension is set to 50 and 1D convolution of 2-layer is applied. Word

Table 2 The Accuracy of our model against other state-of-the-art models

Model	SST-1	SST-2	TREC	Subj	MR	CR
SVM [32]	40.7	79.4	–	–	–	–
NB [32]	41.0	81.8	–	–	–	–
Standard-RNN [32]	43.2	82.4	–	–	–	–
DRNN [12]	49.8	86.6	–	–	–	–
LSTM [34]	46.4	84.9	–	–	–	–
Bi-LSTM [34]	49.1	87.5	93.6	93.0	81.8	–
LR-Bi-LSTM [25]	50.6	–	–	–	82.1	–
CNN-MC [15]	47.4	88.1	92.2	93.2	81.1	85.0
DCNN [13]	48.5	86.8	93.0	–	–	–
Single DSA [38]	50.6	88.5	–	–	–	–
BLSTM-2DCNN [40]	**52.4**	89.5	96.1	94.0	82.3	–
DC-Bi-LSTM [5]	51.9	89.7	95.6	94.5	82.8	–
DC-Bi-LSTM (ours)	50.1	89.3	94.8	94.8	**83.3**	85.3
DC-Bi-LSTM with dynamic interaction (ours)	51.0	**89.9**	**96.5**	**95.2**	83.1	**87.5**

Note, the number highlighted in bold is the highest accuracy of each dataset in all models. There are regular machine learning method, RNNs, CNNs models and LSTM with some tricks. In general, our result show the best in some of the datasets

Table 3 Stacked Bi-LSTM and stacked Bi-LSTM with dynamic interaction

Model	SST-2				TREC			
Bi-LSTM(4-layers)	87.2	86.2	86.9	86.8	94.3	94.0	93.7	93.9
Bi-LSTM with dynamic interaction(4-layers)	87.9	88.3	87.8	87.9	95.1	94.3	94.7	95.2

Note, there are four experiments on SST-2 and TREC respectively by stacked Bi-LSTM and stacked Bi-LSTM with dynamic interaction. The accuracy of stacked Bi-LSTM with dynamic interaction is higher at each experiment

representation is the concatenation of word embedding and character embedding.

– Weights and biases initialization: The Glorot uniform initializer, also called Xavier uniform initializer is applied on all the weights. It draws samples from a uniform distribution within a specific range of intervals. All biases are initialized with 0.0.
– Hyperparameters: To show the benefit of our method, we compare with the original densely connected Bi-LSTM [5]. In our experiments, we follow the detail of parameters setting of the model and also reproduce it. For the densely connected LSTM layers, we use 15 layers, the last layer has 100 hidden units and the others have 13 hidden units. The Adam optimizer to train our model with a learning rate of 0.005 is applied. For multi-head attention and the fully connected layer, we perform L2 regularization with a rate of 0.001 to improve the generalization of our model. we further apply dropout with a rate of 0.5 on the word representation and the output of dynamic interaction layer. Batch size is set to 200 for training.

4.3 Results and discussion

Our method applied on DC-Bi-LSTM shows better results than other basic state-of-the-art models, including the original DC-Bi-LSTM model. The detail of comparison of different models is listed in Table 2. We use accuracy as metric to measure the performance of each model. For reliability of the results, we reproduce the DC-Bi-LSTM model and then apply it for our method. As shown in Table 2, our reproduced DC-Bi-LSTM performs a little poorer than the original one. However, our DC-Bi-LSTM with dynamic interaction performs quite well in four tasks.

Furthermore, we conduct experiments on stacked Bi-LSTM and stacked Bi-LSTM with dynamic interaction. As the dynamic interaction method is based on the multi-layer structure, we choose a 4-layer Bi-LSTM that has 100 hidden units in each layer. For instance, we select results of four experiments on dataset SST-2 and TREC respectively and list them in Table 3. The accuracy of stacked Bi-LSTM with our method is higher at each experiment. We take Figs. 4 and 5 to further explain our results in detail. In these two figures, we adapt the batch size 200 and for each batch we validate our model with test sets. For convenience, we label our model as Di-LSTM.

– Performance on CR and TREC: Seen in Fig. 4, for convenience, we select CR and TREC out of six datasets for further description. Note the CR dataset, densely connected Bi-LSTM with average pooling (yellow line) and dynamic interaction(green line) have the similar speed getting to the accuracy of 80% about 50 batches. However, CR-Di-LSTM performs better than CR-DC-Bi-LSTM as the former has higher peaks

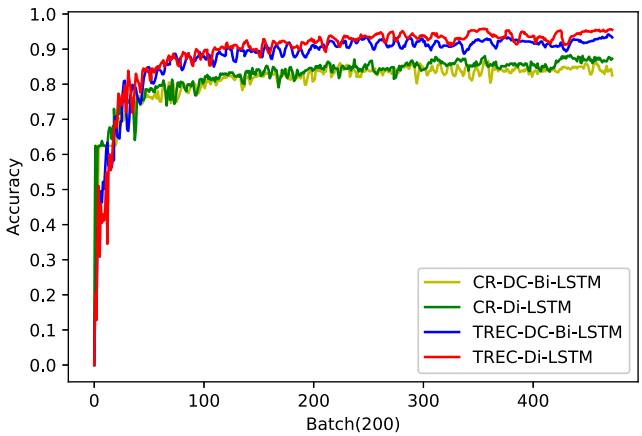

Fig. 4 The accuracy of DC-Bi-LSTM and DC-Bi-LSTM with dynamic interaction on dataset CR and TREC

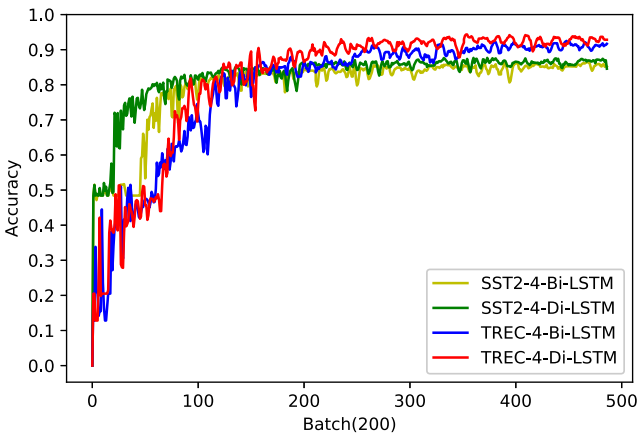

Fig. 5 The accuracy of stacked Bi-LSTM and stacked Bi-LSTM with dynamic interaction on dataset SST-2 and TREC

Table 4 Samples from dataset SST2 and CR

Source	Sentence	Label
SST2	a smart, sweet and playful romantic comedy.	positive
CR	this phone is not suitable for fast people, maybe only for old people.	negative

than the latter. It seems that dynamic interaction method introduces more opportunity for model to discover the true meaning of sentences. Observing another dataset TREC, the two models get to the accuracy of 90% within 100 batches. As we can seen in CR, TREC-Di-LSTM(red line) has higher peaks than TREC-DC-Bi-LSTM(blue line), moreover, the gap between them is more obvious. Our method works better than the general average pooling.

– Performance on SST-2 and TREC: Seen in Fig. 5, for the dataset SST-2, the stacked 4-layers Bi-LSTM with our dynamic interaction method(green line) got faster to the accuracy of 80% than the original stacked 4-layers Bi-LSTM(yellow line), less than 50 batches of training. At about 200 batches of training, the convergence speeds of the two models both tended gentle, however, the accuracy of SST2-4-Di-LSTM is higher than SST-4-Bi-LSTM most of the time. For the dataset TREC, at the first of training, the accuracy of the stacked 4-layers Bi-LSTM with our dynamic interaction method(red line) and the stacked 4-layers Bi-LSTM with general average pooling(blue) is similar. After 200 batches of training, our dynamic interaction method helps to keep the accuracy higher than the original 4-layers Bi-LSTM model. In two datasets, the model with dynamic interaction method performs better than the one with average pooling.

For the detail of what our method contributes to, we select source sentences from the test set of SST2 and dataset CR for visualization. The samples are listed in Table 4.

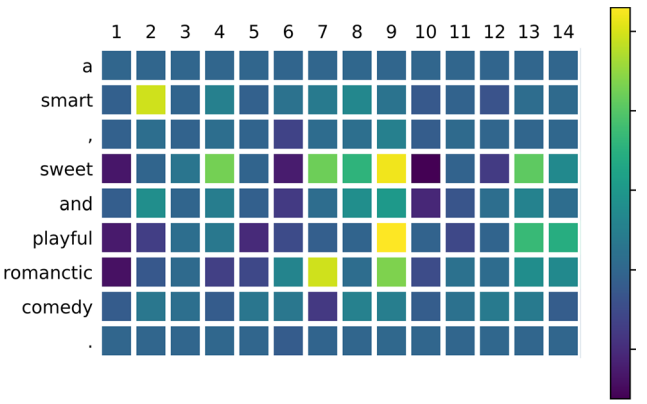

Fig. 6 The visualization of a source sentence from dataset SST2

We visualize the results in the form of a heat map as shown in Figs. 6 and 7. The y-axis indicates the words of a given sentence and the x-axis indicates the attending layers (L in our experiment is 15), and a heat map shows how much a specific layer (step) is concerned about a word. More specifically, as shown in Fig. 6, we can see that the important words have been noticed in a different step of Algorithm 1. The square in the figure becomes brighter when a step of Algorithm 1 shows more concern on a word of a sentence. The words of "smart", "sweet" and "playful" are bright which means they are important from a layer's perspective, and they are positive words. For another sentence as shown in Fig. 7, some steps of Algorithm 1 all focus on the key word "only", which may be an indirect negation of the product considering that CR is collection of customer reviews on various products.

In short, our method do make contributions in improving the performance of multi-layer network. As shown in Figs. 4 and 3, it's obvious that the dynamic interaction won't slow down the convergence speed and it achieves better accuracy. Our algorithm is human interpretable and helps to notice the different aspects of a sentence as the visualization shown in Figs. 6 and 7. Furthermore, our method is easy to implement, is portable to other multi-layer models and is suitable for more tasks.

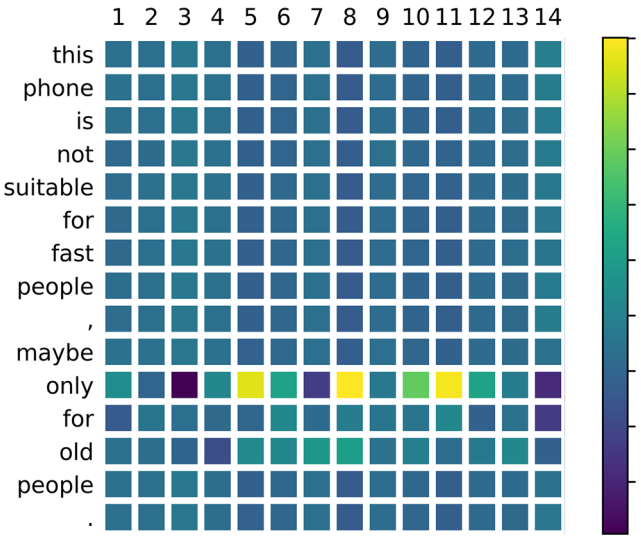

Fig. 7 The visualization of a source sentence from dataset CR

5 Conclusion

In this paper, we propose a new dynamic interaction method in multiple-layer RNN architecture, which fully combines the final states of the uppermost layer with previous information of the forward layers to achieve more valuable sentence embedding. With dynamic interaction, experimental results across some benchmark datasets show that our proposed method integrates more semantic information into the final sentence embedding representation and yields performance comparable or superior to the-state-of-art. Our future work will focus on the combination of the proposed method with more multiple-layer architecture for more challenging tasks.

Acknowledgments We thank the support from National Natural Science Foundation of China (Nos. 11771152); Science and Technology Foundation of Guangdong Province (Nos. 2015B010128008 & 2015B010109006).

References

1. Bowman SR, Angeli G, Potts C, Manning CD (2015) A large annotated corpus for learning natural language inference. In: EMNLP, pp 632–642
2. Chen Q, Ling Z-H, Zhu X (2018) Enhancing sentence embedding with generalized pooling. In: COLING, pp 1815–1826
3. Chen Q, Zhu X, Ling Z-H, Wei S, Jiang H, Inkpen D (2017) Enhanced LSTM for natural language inference. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics: long papers, vol 1. Association for Computational Linguistics, pp 1657–1668
4. Conneau A, Kiela D, Schwenk H, LBarrault L, Bordes A (2017) Supervised learning of universal sentence representations from natural language inference data. In: EMNLP, pp 670–680
5. Ding Z, Xia R, Yu J, Li X, Yang J (2018) Densely connected bidirectional LSTM with applications to sentence classification. In: NLPCC, pp 278–287
6. Gao J, Duh K, Liu X, Shen Y (2018) Stochastic answer networks for machine reading comprehension. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics: long papers, vol 1. Association for Computational Linguistics, pp 1694–1704
7. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: CVPR, pp 770–778
8. He K, Zhang X, Ren S, Sun J (2016) Identity mappings in deep residual networks. In: ECCV, pp 630–645
9. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
10. Hu M, Liu B (2014) Mining and summarizing customer reviews In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp 168–177
11. Huang G, Liu Z, van der Maaten L, Weinberger KQ (2017) Densely connected convolutional networks. In: CVPR, pp 2261–2269
12. Irsoy O, Cardie C (2014) Deep recursive neural networks for compositionality in language. In: Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems, pp 2096–2104
13. Kalchbrenner N, Grefenstette E, Blunsom P (2014) A convolutional neural network for modelling sentences. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: long papers, vol 1. The Association for Computer Linguistics, pp 655–665
14. Kim S, Hong J-H, Kang I, Kwak N (2018) Semantic sentence matching with densely-connected recurrent and co-attentive information. arXiv:180511360
15. Kim Y (2014) Convolutional neural networks for sentence classification. In: EMNLP, pp 1746–1751
16. Kiros R, Zhu Y, Salakhutdinov R, Zemel RS, Urtasun R, Torralba A, Fidler S (2015) Skip-thought vectors. In: Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems, pp 3294–3302
17. Li X, Roth D (2002) Learning question classifiers. In: COLING
18. Lin Z, Feng M, dos Santos CN, Yu M, Xiang B, Zhou B, Bengio Y (2017) A structured self-attentive sentence embedding. arXiv:170303130
19. Liu Y, Sun C, Lin L, Wang X (2016) Learning natural language inference using bidirectional LSTM model and inner-attention. arXiv:160509090
20. Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J (2013) Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems, pp 3111–3119
21. Mou L, Men R, Li G, Xu Y, Zhang L, Yan R, Jin Z (2016) Natural language inference by tree-based convolution and heuristic matching. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics: short papers, vol 2. The Association for Computer Linguistics
22. Pang B, Lee L (2004) A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In: Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, pp 271–278
23. Pang B, Lee L (2005) Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In: Proceedings of 43rd Annual Meeting of the Association for Computational Linguistics, pp 115–124
24. Pennington J, Socher R, Manning CD (2014) Glove: Global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, pp 1532–1543
25. Qian Q, Huang M, Zhu X (2016) Linguistically regularized lstms for sentiment classification. arXiv:161103949
26. Sabour S, Frosst N, Hinton GE (2017) Dynamic routing between capsules. In: Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems, pp 3859–3869
27. Schuster M, Paliwal KK (1997) Bidirectional recurrent neural networks. Trans Signal Process 45(11):2673–2681
28. Shen T, Zhou T, Long G, Jiang J, Pan S, Zhang C (2018) Disan: Directional self-attention network for rnn/cnn-free language understanding. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence
29. Socher R, Lin CC-Y, Ng AY, Manning CD (2011) Parsing natural scenes and natural language with recursive neural networks. In: Proceedings of the 28th International Conference on Machine L earning, pp 129–136
30. Liu T, Yu S, Xu B, Yin H (2018) Recurrent networks with attention and convolutional networks for sentence representation and classification. Appl Intell 48:3797–3806

31. Socher R, Perelygin A, Wu J, Chuang J, Manning CD, Ng AY, Potts C (2013) Recursive deep models for semantic compositionality over a sentiment treebank. In: EMNLP, pp 1631–1642
32. Socher R, Perelygin A, Wu J, Chuang J, Manning CD, Ng AY, Potts C (2013) Recursive deep models for semantic compositionality over a sentiment treebank. In: EMNLP, pp 1631–1642
33. Srivastava RK, Greff K, Schmidhuber J (2015) Highway networks. arXiv:150500387
34. Tai KS, Socher R, Manning CD (2015) Improved semantic representations from tree-structured long short-term memory networks. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing: long papers, vol 1. The Association for Computer Linguistics, pp 1556–1566
35. Turian JP, Ratinov L-A, Bengio Y (2010) Word representations: A simple and general method for semi-supervised learning. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pp 384–394
36. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2017) Attention is all you need. In: Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems, pp 6000–6010
37. Yang Z, Yang D, Dyer C, He X, Smola AJ, Hovy EH (2016) Hierarchical attention networks for document classification. In: NAACL, pp 1480–1489
38. Yoon D, Lee D, Lee S (2018) Dynamic self-attention : Computing attention over words dynamically for sentence embedding. arXiv:180807383
39. Zhang X, Zhao JJ, LeCun Y (2015) Character-level convolutional networks for text classification. In: Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems, pp 649–657
40. Zhou P, Qi Z, Zheng S, Xu J, Bao H, Xu B (2016) Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling. In: COLING, pp 3485–3495