R-Transformer Network Based on Position and Self-attention mechanism for Aspect-level Sentiment Classification

Ziyu Zhou¹, Fang’ai Liu¹, Qianqian Wang¹

¹ School of Information Science & Engineering, Shandong Normal University, Jinan, 250014, China

Corresponding author: Fang’ai Liu (e-mail: lfa@sdnu.edu.cn).

This work was supported by the following grants: National Natural Science Foundation of China 61772321, Natural Science Foundation of Shandong Province ZR2016FP07, the Innovation Foundation of Science and Technology Development Center of Ministry of Education and New H3C Group 2017A15047 and CERNET Innovation Project NGII20170508.

ABSTRACT Aspect-level sentiment classification (ASC) is a research hotspot in natural language processing, which aims to infer the sentiment polarity of a particular aspect in an opinion sentence. There are three main influence factors in the aspect-level sentiment classification: the semantic information of the context; the interaction information of the context and aspect; the position information between the aspect and the context. Some researchers have proposed way to solve aspect-level sentiment classification. However, previous work mainly used the average vector of the aspect to calculate the attention score of the context, which introduced the influence of noise words. Moreover, these attention-based approaches simply used relative positions to calculate positional information for contextual and aspect terms and did not provided better semantic information. Based on these above questions, in this paper, we propose the PSRTN model. Firstly, obtaining the position-aware influence propagate between words and aspects by Gaussian kernel and generating the influence vector for each context word. Secondly, capturing global and local information of the context by the R-Transformer, and using the self-attention mechanism to obtain the keywords in the aspect. Finally, context representation of a particular aspect is generated for classification. In order to evaluate the validity of the model, we conduct experiments on SemEval2014 and Twitter. The results show that the accuracy of the PSRTN model can reach 83.8%, 80.9%, and 75.1% on three data sets, respectively.

INDEX TERMS aspect-level sentiment classification, Gaussian kernel, R-Transformer, self-attention mechanism

I. INTRODUCTION

When people experience products and services, they often make reviews. As the amount of review data continues to grow, people must process it before using. The results of data processing are valuable information for potential customers and suppliers. Sentiment analysis is an important task in the field of natural language processing (NLP) [1]. Aspect-level sentiment classification (ASC) is a fine-grain task in sentiment analysis, which aims to identify the polarity of sentiment (positive, neutral, negative) of different aspects in the context. Therefore, ASC provides a more complete sentiment expression. For example, given a sentence like "The pizza at this restaurant is very good, but the take-out pizza is very slow." The sentiment polarity of the aspect term "pizza" is positive, while the sentiment polarity of the aspect "take-out pizza" is opposite to the sentiment polarity of "pizza." The standard sentiment classification is to assign an emotional label to each sentence. Since the ASC needs to consider different aspects of the sentence compared with the standard sentiment classification, it is difficult to classify the aspect-level sentiment.

There are two ways to solve this task: traditional machine learning methods and deep learning techniques. The traditional classification methods mainly train classifiers such as support vector machine (SVM) by designing a set of words, emotional lexicon [2][3]. The quality of feature engineering is the key to traditional classification methods. The main reason that deep learning technology can effectively solve ASC is that deep learning can automatically learn effective features from high-dimensional data [4]-[6]. As a powerful sequence modeling technique, recurrent neural network (RNN) is widely used in NLP tasks [7][8]. Some RNN-based method has been used for ASC [9]. But RNN treats each context word equally. Inspired by the human visual attention mechanism, the attention mechanism
based on neural network has been well applied in many researches, such as image generation, machine translation, natural language reasoning and so on [10]-[12]. Recently, some attention-based RNN methods have been used for ASC. Chen [9] and Tang [13] used the average aspect vector to learn the attention weight of context words. Ma et al. averaged the hidden vectors of aspect and context respectively, and utilized the bidirectional attention to learn weights [7]. When inferring the sentiment polarity of one aspect of a sentence, other aspects and irrelevant context words become noise. Modeling the relationship between aspects and context and contextual semantic information becomes the focus of ASC.

When we calculate the relationship between aspects and context, the previous method mainly used the average pool of aspect hidden vectors as the final representation of the aspect to learn weights. However, if the aspect is a phrase, the average aspect vector may result in information loss and introduce noise words. For example, the term "mobile phone" in the aspect "one mobile phone" is more important, while "one" becomes a noise word. Each word in the aspect pays different attention to the context. If we use the keyword "mobile phone" in the aspect to calculate the attention score of the context, the accuracy of the sentiment classification will be improved. Ma et al. proposed the IAN model, which used the average vector of aspect terms to calculate the representation of context, so the accuracy of model classification was low [14]. Each word in aspect terms has a different impact on classification, so it is important to calculate the weight of each word in aspect terms. Therefore, we use the self-attention mechanism to calculate the weight of each word in the aspect terms and generate the feature representation of the aspect terms.

On the other hand, some previous work has clarified the position information between aspect terms and context, which will improve the classification accuracy [15]. Therefore, we use the Gaussian kernel function to calculate the positional relationship between the aspect terms and context. Similar to some of the previous work, we use the positional relationship to limit the aspect to pay more attention to neighboring words.

In addition, a lot of previous work is based on RNN to get the semantic information of the context. Since RNN is difficult to compute in parallel and cannot maintain long-term dependencies, the semantic information in the modeling context shows weaknesses. In this paper, we use the R-transformer proposed by Wang et al. to model the context [16]. It can obtain both long-term dependency information and local semantic information.

Based on the above ideas, we propose a based-on position and self-attention mechanism R-Transformer network (PSRTN). PSRTN considers three influencing factors for ASC. Our model uses Gaussian kernel function to calculate the positional relationship between aspect and context. In order to avoid noise words and better use of the keywords in aspect, PSRTN uses the self-attention mechanism to calculate the weight of each word in the aspect. In order to better obtain the semantic information of the context, we use R-Transformer to obtain global information dependence and local information dependence. We evaluate our model on SemEval2014 and Twitter dataset. The experimental results show that our model has a certain improvement on ASC.

The main contributions of our model are summarized as follows:

1) We consider three influencing factors for ASC: the keywords in the aspect term, the positional relationship between the aspect term and the context, and the semantic information of the context.

2) In order to overcome the shortcomings of RNN, we use R-Transformer to obtain the semantic information of the context.

3) In order to get an aspect-specific context representation, we use a self-attention mechanism to calculate the weight of each word in the aspect term.

4) We conduct experiments on the SemEval2014 and Twitter datasets. The experimental results show that our model is 0.7%, 1%, and 0.4% higher than the baseline model on the three data sets.

The main structure of our paper is as follows: In the second part, we review the related work of ASC. The third part shows the details of the PSRTN model. In the fourth part, experiments were conducted on three public data sets. The last part, summarizing our work and shortcomings.

II. RELATED WORK

The purpose of ASC was to determine the sentiment polarity of a particular aspect in a sentence. Recently, the ASC had been paid more and more attention by researchers. Many methods only used contextual features, but did not consider different aspects of the sentence. Jiang et al. found in the study that neglecting aspects would lead to 40% sentiment classification errors [17]. Now, we had a lot of ways to solve the sentiment classification problem. The machine learning method usually labeled the text, then extracted the features of the training data to construct a classifier such as SVM, and finally classified the unlabeled data by the classifier [18]. The quality of the feature determines the performance of the classifier. In recent years, neural networks had achieved remarkable results in the field of sentiment classification [22]. Many neural network structures had been proposed, including Convolutional Neural Networks [19], Recurrent Neural Networks [20], and Recursive Neural Networks [21]. Neural network-based methods could automatically learn semantic representations without complex feature engineering.

A. ASC BASED ON RNN

Recurrent neural networks (RNN) are capable of performing on various tasks in NLP. Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) are derived from RNN. Dong et al. proposed the AdaRNN model, which used an adaptive multi-combination layer in the Recurrent neural network, so that the sentiment of words could be propagated to the target according to the context and syntactic structure
Nguyen and Shirai proposed a new model, PhraseRNN, based on RNN and AdaRNN, which enriched the representation of the target by using the dependency from the sentence and the syntactic information of the composition tree [24]. Tang et al. proposed TD-LSTM, that divided the context into two parts from the target location. The context before the target is the left half, followed by the right half [25]. TD-LSTM used two long and short-term memory networks to model the left and right parts, and finally combined the two parts for classification. The structure of TC-LSTM was similar to TD-LSTM, which connected the average vector of the aspect to the word embedding as part of the input, thus making the model more concerned with the aspect information. Zhang et al. proposed two gated neural networks for target sentiment analysis, one for obtaining grammatical and semantic information and the other for modeling the relationship between left and right contexts of a given target [26]. Xue et al. proposed a model based on convolutional neural networks and gating mechanisms (GTRU). GTRU used two convolutional layers to model aspects and sentiment information separately and controlled sentiment flow based on given aspect information [27].

B. ASC BASED ON ATTENTION MECHANISM

Neural network based on attention mechanism had been successfully applied in machine translation and image generation. Recently, many works had applied attention mechanisms to the aspect sentiment classification. Wang et al. proposed an attention-based LSTM for aspect-level sentiment classification [28]. At first, he mapped the aspect to an embedding vector and then computed the relationship between the aspect and the context. Inspired by the deep memory network [29], Tang et al. designed MemNet, which used a deep memory network with multiple compute layers to capture the importance of each context word [13]. Chen et al. used a multi-layered attention mechanism model to capture long-distance sentiment features, and then nonlinearly combined different attentional features to obtain important information [9]. The difference between Tang [13] and Chen [9] was that they synthesized the features of a word sequence between the attention module and the input module by introducing a memory module. The previous approach was to model their context by generating target-specific representations, but none of these methods neglected to model the target separately. Ma et al. proposed an interactive attention network and interactive learned attention in target and context through two LSTM networks [14]. Finally, the combination of aspect representation and context representation was used as input to the classifier.

The previous method had achieved some success in the ASC, but they all regard the aspect as a whole. Gu et al. proposed a position-aware bidirectional attention network. Although they realized that location information was helpful for ASC, they simply introduced location information by relative distance [30]. Song et al. proposed an attention encoder network (AEN), which used multiple self-attention to obtain interaction information between each word and context in the aspect [31]. Although AEN overcame the inability of LSTM to process in parallel and maintained long-term dependency on sentences, it discarded recurrent structures that caused performance degradation and neglected local semantic information.

III. THE PROPOSED MODELS: PSRTN

In this section, we describe our propose model, PSRTN, in detail. PSRTN can effectively infer the sentiment polarity of the aspect. In Section 3.1, we describe the generation process of position vectors. Section 3.2, we use Bi-GRU and R-Transformer to obtain aspect and context semantic information, respectively. The aspect-specific context representation is taken in Section 3.3. Finally, we show the overall structure of the PSRTN in Section 3.4. Next, we will introduce each part one by one.

A. POSITION VECTORS

Through our previous analysis, more accurate location information is very effective for ASC. Inspired by the method of marked context position in [32] and [33], we propagate the influence of aspect to other locations through the Gaussian kernel and calculate the positional relationship between the aspect term and the context.

$$Kernel(\mu) = \exp\left(-\frac{\mu^2}{2\gamma^2}\right)$$

(1)

Where $\mu$ represents the distance between the aspect terms and the words in the sentence, and $\gamma$ represents the range of propagation. Obviously, the greater the distance, the smaller the position perception influence. In other words, the farther the distance is, the smaller the impact of words on the aspect. The value of $\gamma$ varies from word to word, so we will set $\gamma$ to a constant value.

To use $\mu$, we use a normal distribution to represent it as a vector. We use matrix $P$ to represent the influence of distance $\mu$ on each dimension. The effect of the $i^{th}$ dimension of distance $\mu$ is calculated as:

$$P(i, \mu) \sim N(Kernel(\mu), \sigma^2)$$

(2)

Where $N$ is a normal distribution and $\sigma$ represents a standard deviation. Each column in matrix $P$ represents an influence vector for a particular distance. For a clearer representation, we use $p_i \in \mathbb{R}^d$ in matrix $P$ as vector representation of between each word and aspect in a sentence. $d$ is the dimension of the position-aware influence vector. Figure 1 shows the generation process of a position-aware vector.
The local RNN organizes the original long sequence into a number of short sequences that include local information and that are independently and identically processed by a shared RNN. Specifically, the R-Transformer builds a local window of size M for each target position, which contains M consecutive locations and ends at the target position. Therefore, local RNNs only focus on local short-term dependencies. Figure 2 shows a schematic of the local RNN. Specifically, the position of a local short sequence of length M is given \( M = [x_{i-M-1}, x_{i-M-2}, \ldots, x_i] \). These short sequences are processed by the local RNN and output M hidden vectors, and then the last hidden vector represent as a feature of the local sequence.

\[
h_i = \text{LocalRNN}(x_{i-M-1}, x_{i-M-2}, \ldots, x_i)
\]

\[
h_1, h_2, \ldots, h_N = \text{LocalRNN}(x_1, x_2, \ldots, x_N)
\]

Then, the representation of the local hidden vector is input to the multi-head self-attention module to capture the long-term dependency information of the sentence [36]. Given the current representation \( h_i, h_2, \ldots, h_i \), enter it into the multi-head self-attention. The new representation \( u_i \) is calculated as follows:

\[
u_i = \text{MultiHeadAttention}(h_i, h_2, \ldots, h_i)
\]

\[= \text{Concatenation}(\text{head}_1(h_i), \text{head}_2(h_i), \ldots, \text{head}_d(h_i))W^o
\]

Where \( \text{head}_k(h_i) \) is the result of the \( k \)th attention head and \( W^o \) is a linearization mapping matrix. Each attention head is a weighted sum of a set of vectors, which is calculated as follows:

\[
\{\alpha_1, \alpha_2, \ldots, \alpha_d\} = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})W^o
\]

B. WORD REPRESENTATION

We define a sentence \( S \) with \( n \) words \( S = [w_1, w_2, \ldots, w_n] \) and an aspect term \( M \) contains \( m \) words \( M = [a_1, a_2, \ldots, a_m] \). Our goal is to determine the sentiment polarity of the aspect term \( M \) for sentence \( S \). We use the pre-trained embedded matrix GloVe and the pre-trained language model BERT to map each word in the aspect term and context to word embedding. The embedded vector of context and aspect terms can be expressed as:

\[
E^c = [e_1^c, \ldots, e_n^c] \quad \text{and} \quad E^m = [e_1^m, \ldots, e_m^m].
\]

The position influence matrix \( P \) and the word embedding matrix \( E^c \) are connected to obtain an embedding matrix \( R^c \).

\[
R^c = p_j \oplus e_i^c
\]

Context representation with R-Transformer

Because RNN cannot capture long-term dependencies and cannot perform parallel calculations on sequences. Recently, many sequence learning models abandon the recurrent structure and only rely on the attention mechanism [31]. Although the multi-attention mechanism allows each position in the sequence to be connected to any other positions. Therefore, information can flow to other position without any intermediate loss. However, there are problems with multi-head attention mechanism. In order to solve the loss of position sequence information, the multi-head attention mechanism introduces position information. Dai et al also pointed out that the impact of this position information was limited and we need to design effective position embedding or introduce them into the learning process in different ways [34]. In addition, wang et al proposed that multi-head attention mechanism could captured long-term dependencies, but it ignored the important local structures in natural language [35]. Inspired by [35], we use R-Transformer to get the hidden vector of the context. R-Transformer takes advantage of RNN and multi-head self-attention while avoiding their shortcomings.

The R-Transformer consists of three parts: the lowest layer is a local RNN, which is used to capture local information. The middle layer is a multi-head self-attention layer. The top layer is a nonlinear feature conversion layer.
Among them, Q, K, and V are the matrix of the query, key, and value, respectively. The vector \( q, k, v \) in the matrix is mapped by the mapping matrix. The mathematical formula is as follows:

\[
q, k, v = W^q h_i, W^k h_i, W^v h_i
\]

Where \( W^q, W^k, W^v \) are mapping matrices and different attention heads have different mapping matrices. Finally, the feedforward network is used to nonlinearly transform the features, and the residuals and layernorm are used to process the connection between the layers. The entire R-Transformer is shown in Figure 3.

\[
FeedForward(m_i) = \max(0, u_i W'_1 + b'_1) W'_2 + b'_2
\]

\[
\text{Multi-head Attention}(v_i, W'_1, W'_2) = \sum_{i=1}^{n} \alpha_i v_i
\]

\[
\text{FeedForward}(v_i, W'_1, W'_2) = \sum_{i=1}^{n} \beta_i v_i
\]

Among various NLP tasks, Bi-GRU has achieved remarkable results. It models the context in both directions for better semantic information. The forward GRU obtain the semantic information of the given context from left to right, and the reverse GRU obtains the semantic information of the given context from right to left. Finally, the two semantic information is linked and use as a representation of the entire context. Since the aspect contains fewer words, we use Bi-GRU to obtain hidden vectors for aspect.

\[
H^m_i = [h^1_i, h^2_i, ..., h^m_i]
\]

C. ATTENTION MECHANISM

The self-attention mechanism has been successfully applied in various tasks. Self-attention is the correlation calculation of every unit and all units in a sequence. It can directly calculate dependencies without considering the distance between words, and learn the internal structure of a sentence. We use the self-attention mechanism to capture the keywords in the aspect and generate aspect representations \( V'_i \).

\[
\gamma(h^i_j) = \tanh(w^1_i h^i_j + b^1_i)
\]

\[
\alpha^i_j = \frac{\exp(\gamma(h^i_j))}{\sum_{j=1}^{m} \exp(\gamma(h^i_j))}
\]

\[
V'_i = \sum_{j=1}^{m} \alpha^i_j h^i_j
\]

In order to get words that affect the polarity of aspect. Calculating the weight of aspect represent \( V'_i \) for the context. Finally, get an aspect-specific context representation \( V'_s \).

\[
\beta^i_j = \frac{\exp(\gamma(h^i_j, V'_i))}{\sum_{j=1}^{m} \exp(\gamma(h^i_j, V'_i))}
\]

\[
V'_s = \sum_{j=1}^{m} \beta^i_j h^i_j
\]

Other words in the aspect may also provide some information. In order to get more complete aspect-specific context information, we average the hidden vector of the aspect.

\[
M_{avg} = \sum_{i=1}^{m} h^i / m
\]
Then, the aspect-specific context representation $V'_a$ and the mean $M_{avg}$ of the aspect hidden vector are concatenated. The final representation $Z$ input the SoftMax layer after the connection.

$$Z = [M_{avg}, V'_a]$$

$$x = \tanh(w_xZ + b_x)$$

$$y = \text{soft max}(w_yx + b_y)$$

**D. MODEL ARCHITECTURE**

The overall architecture of the PSRTN is shown in Figure 4. It consists of three modules. The underlying module is a position vector module that uses the Gaussian kernel to obtain the position vector of the context and then joins it to the word embedding.

The intermediate module is a word representation module, and the embedding of aspects and contexts is input into Bi-GRU and R-Transformer respectively to obtain hidden vectors. R-Transformer can obtain both global and local dependencies.

The top module is the attention module, which uses a self-attention mechanism to capture the keywords in the aspect and then generates an aspect-specific context representation.

**E. MODEL TRAINING**

Our goal is to optimize all parameters in order to minimize the loss function as much as possible. We use cross entropy and $L_2$ regularization as loss function. The formula is as follows:

$$L = - \sum_{d \in D} \sum_{c=1}^{C} y_c(d) \cdot \log(g_c(d)) + \lambda L_2(\theta) \quad (17)$$

Where $D$ is the data set, $d$ is one of the samples, $C$ is the number of categories, and $y_c(d)$ is the true sentiment distribution. The output of $\log(g_c(d))$ is a vector that represents the probability of sentiment polarity. $\lambda$ is a regularization coefficient and $\theta$ contains all the parameters. To avoid overfitting, the dropout strategy is used.

**IV. EXPERIMENTS**

In this section, we describe the experimental setup and design several sets of experiments to demonstrate the validity of our model. First, we compare our model with some baseline models to prove the validity of our model. Then, we design several sets of ablation experiments to prove the validity of our module. Finally, we visualize the data sets in our experiments.

**A. EXPERIMENT SETTINGS**

We conduct experiment on Laptop and Restaurant and Twitter\(^1\). SemEval2014\(^2\) includes both Laptop and Restaurant. These data sets have three sentiment polarities: positive, neutral, and negative. Each review contains aspect terms and corresponding sentiment polarity. The number of each polarity is shown in table 1. We also count the number of words in aspect in table 2. From table 2, we can see that more

---

\(^1\)http://goo.gl/5Enpu7
\(^2\)http://alt.qcri.org/semeval2014/
than half of the data sets have multiple words in the aspect terms.

Table 1: Statistics of experimental data sets.

<table>
<thead>
<tr>
<th>Domain</th>
<th>Positive</th>
<th>Neutral</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Restaurant</td>
<td>Train</td>
<td>2148</td>
<td>631</td>
</tr>
<tr>
<td></td>
<td>Test</td>
<td>720</td>
<td>194</td>
</tr>
<tr>
<td>Laptop</td>
<td>Train</td>
<td>989</td>
<td>468</td>
</tr>
<tr>
<td></td>
<td>Test</td>
<td>337</td>
<td>167</td>
</tr>
<tr>
<td>Twitter</td>
<td>Train</td>
<td>1561</td>
<td>3127</td>
</tr>
<tr>
<td></td>
<td>Test</td>
<td>173</td>
<td>346</td>
</tr>
</tbody>
</table>

Table 2: Statistics of the number of words in aspect on three data sets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Len=1</th>
<th>Len=2</th>
<th>Len&gt;2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Restaurant-Train</td>
<td>2720</td>
<td>75.38%</td>
<td>604/16.74%</td>
</tr>
<tr>
<td>Restaurant-Test</td>
<td>801</td>
<td>71.52%</td>
<td>215/19.20%</td>
</tr>
<tr>
<td>Laptop-Train</td>
<td>1473</td>
<td>63.27%</td>
<td>649/27.88%</td>
</tr>
<tr>
<td>Laptop-Test</td>
<td>351</td>
<td>52.78%</td>
<td>209/31.43%</td>
</tr>
<tr>
<td>Twitter-Train</td>
<td>1893</td>
<td>30.25%</td>
<td>4360/69.67%</td>
</tr>
<tr>
<td>Twitter-Test</td>
<td>198</td>
<td>28.61%</td>
<td>492/71.10%</td>
</tr>
</tbody>
</table>

In our experiments, we use GloVe [37] and pre-trained language model word representation Bert [38] to initialize aspects and context word embedding. Glove sets the embedding dimension of each word to 300 and Bert to 768. The number of GRU hidden units is set to 300, and the R-Transformer is set to 128. Words outside the vocabulary are initialized with a uniform distribution U(-0.1,0.1), and the weight matrix is set to zero. The propagation range γ is set to 23 and the standard deviation is set to 0.1. In the experiment, the size of the batch is set to 25, the weight of the $L_2$ regularization is set to $10^{-5}$ and the initial learning rate of AdaDelta is set to 0.05.

B. COMPARATIV METHODS

To evaluate the effectiveness of our model, we compare our model with the following baseline model:

LSTM ignores the importance of aspects. It gets the hidden vector of each word in the sentence and takes the last hidden vector as the final representation of the sentence and then inputs the SoftMax layer [28].

TD-LSTM divides the entire context into the left half of the target context and the right half of the target context, and then models the left and right parts by two LSTM networks. Finally, the two parts are connected for classification [25].

TC-LSTM The difference between TC-LSTM and TD-LSTM is that it integrates the relationship between target words and context to enhance the importance of aspect representation [25].

AE-LSTM maps aspects to word embedding and then uses it as part of the training [28].

ATAE-LSTM is designed based on AE-LSTM, which emphasizes the importance of aspect by adding aspect embedding in the embedding of each word [28].

MemNet uses a deep memory network to model sentences and captures the association between context words and aspect terms [15].

IAN uses two LSTM networks to model aspect terms and words respectively, and then generates sentence representations and aspect representations through interactive attention. Finally, two representations are connected for prediction [14].

PBAN adds relative position information to each word embedding, and then uses Bi-GRU to model the interrelationship between words and aspect terms [30].

AOA-LSTM first models aspect terms and context through Bi-LSTM, and then uses an interaction module to obtain attention from aspect terms to context and from context to aspect terms to generate sentence representation for classification [39].

AEN avoids recurrent structures and models the relationship between aspect and context by multi head self-attention [31].

C. EXPERIMENT RESULTS

Table 3: Performance comparisons between different methods and the best results in bold.

<table>
<thead>
<tr>
<th>Model</th>
<th>Restaurant</th>
<th>Laptop</th>
<th>Twitter</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSTM</td>
<td>0.743</td>
<td>0.665</td>
<td>0.665</td>
</tr>
<tr>
<td>TD-LSTM</td>
<td>0.756</td>
<td>0.681</td>
<td>0.666</td>
</tr>
<tr>
<td>TC-LSTM</td>
<td>0.763</td>
<td>0.683</td>
<td>0.672</td>
</tr>
<tr>
<td>AE-LSTM</td>
<td>0.762</td>
<td>0.689</td>
<td>0.677</td>
</tr>
<tr>
<td>ATAE-LSTM</td>
<td>0.772</td>
<td>0.687</td>
<td>-</td>
</tr>
<tr>
<td>MemNet</td>
<td>0.787</td>
<td>0.708</td>
<td>0.685</td>
</tr>
<tr>
<td>IAN</td>
<td>0.781</td>
<td>0.721</td>
<td>-</td>
</tr>
<tr>
<td>PBAN</td>
<td>0.811</td>
<td>0.741</td>
<td>-</td>
</tr>
<tr>
<td>AOA-LSTM</td>
<td>0.812</td>
<td>0.745</td>
<td>-</td>
</tr>
<tr>
<td>AEN</td>
<td>0.831</td>
<td>0.799</td>
<td>0.747</td>
</tr>
<tr>
<td>PSRTN-GloVe</td>
<td>0.834</td>
<td>0.789</td>
<td>0.741</td>
</tr>
<tr>
<td>PSRTN-Bert</td>
<td><strong>0.838</strong></td>
<td><strong>0.809</strong></td>
<td><strong>0.751</strong></td>
</tr>
</tbody>
</table>

We use the accuracy to evaluate the validity of the model. The experimental results of the PSRTN model and some baseline models are shown in Table 3. To prevent the effects of different word representations, we perform experiments on GloVe and BERT, respectively. It can be clearly seen that the experimental results of our model are superior to all other models based on GloVe and BERT.

From the experimental results, we can see that the LSTM network performs the worst in all LSTM-based models because LSTM does not consider aspect. Since TC-LSTM considers the relationship between target and context, the accuracy rate is increased by 0.7% and 0.2%, 0.6% compared with TD-LSTM. AE-LSTM embeds aspects into the LSTM model for training, so AE-LSTM is more accurate than TD-LSTM and TC-LSTM. Because ATAE-LSTM, IAN, and MemNet use attention mechanisms to
model the relationship between aspects and contexts, they are more accurate than TD-LSTM and TC-LSTM. IAN is the first to use attention to model aspects. Comparing with AAE-LSTM, IAN improves 0.9% and 3.4% on Restaurant and Laptop, respectively. PBAN introduces relative position information, so the accuracy has been significantly improved, which indicates that position has an important role in ASC. AOA-LSTM and PBAN have a similar result.

Among all GloVe-based models, PSRTN-GloVe has the highest accuracy. Comparing with AOA-LSTM, PSRTN-GloVe increases by 2.2% and 4.4% on the restaurant and laptop, respectively. The main reason is that PSRTN-GloVe uses location information. Since PSRTN-GloVe obtains global and local information of the context, it is increased by 2.3% and 4.8% compared with PBAN. In addition, comparing with the AEN model, the PSRTN-Bert model increases by 0.7%, 1%, and 0.4% on Restaurant, Laptop, and Twitter. Although AEN can capture long-term dependency information, it discards the recurrent structure, which is the main reason why AEN is less accurate than PSRTN.

Table 4: Effect of position on model accuracy

<table>
<thead>
<tr>
<th>Model</th>
<th>Restaurant</th>
<th>Laptop</th>
<th>Twitter</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSRTN w/o p</td>
<td>0.805</td>
<td>0.784</td>
<td>0.729</td>
</tr>
<tr>
<td>PSRTN+Absolute</td>
<td>0.832</td>
<td>0.804</td>
<td>0.747</td>
</tr>
<tr>
<td>PSRTN+Gaussian</td>
<td><strong>0.838</strong></td>
<td><strong>0.809</strong></td>
<td><strong>0.751</strong></td>
</tr>
</tbody>
</table>

From Table 4, we can see that the PSRTN w/o p does not use any position information, so the model performs the worst. The comparison of PSRTN w/o p and PSRTN+Absolute shows that position information is very helpful for the classification results. Comparing with PSRTN+Absolute, PSRTN+Gaussian increases by 0.6%, 0.5%, and 0.4% on three data sets. Therefore, Gaussian kernel function is more suitable for our task.

Table 5 shows that PSRTN has higher classification accuracy by using aspect keywords than average aspect vector. Comparing with average pooling, self-attention increases by 0.9%, 1.1%, and 0.5%. In fact, the aspect does not have any sentiment information. Aspects that contain multiple words will weaken the sentiment features during the processing of the model and introduce noise. Therefore, the use of the self-attention mechanism can extract key words, thereby extracting sentiment words for the aspect.

Table 5: Impact of averaging pooling and self-attention on aspect.

<table>
<thead>
<tr>
<th>Model</th>
<th>Restaurant</th>
<th>Laptop</th>
<th>Twitter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aspect word with average pooling</td>
<td>0.829</td>
<td>0.798</td>
<td>0.746</td>
</tr>
<tr>
<td>Aspect word with self-attention</td>
<td><strong>0.838</strong></td>
<td><strong>0.809</strong></td>
<td><strong>0.751</strong></td>
</tr>
</tbody>
</table>

From the experimental results in Table 6, we can see that R-Transformer is more suitable for our model. Using the BiGRU module to obtain hidden vectors of context, the model has the lowest accuracy because it cannot maintain long-term dependencies and does not get better semantic information. Comparing to multi-head attention, R-Transformer increases by 0.9%, 0.8%, and 0.3% on Restaurant, Laptop, and Twitter, respectively. Since R-Transformer not only obtain long-term dependency information of context, but also obtain local dependency information, it achieves better results than multi-head attention.

Table 6: Effect of PSRTN with Bi-GRU, multi-head attention and R-Transformer module on accuracy.

<table>
<thead>
<tr>
<th>Model</th>
<th>Restaurant</th>
<th>Laptop</th>
<th>Twitter</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSRTN with Bi-GRU</td>
<td>0.819</td>
<td>0.768</td>
<td>0.721</td>
</tr>
<tr>
<td>PSRTN with multi-head attention</td>
<td>0.829</td>
<td>0.801</td>
<td>0.748</td>
</tr>
<tr>
<td>PSRTN with R-Transformer</td>
<td><strong>0.838</strong></td>
<td><strong>0.809</strong></td>
<td><strong>0.751</strong></td>
</tr>
</tbody>
</table>

D. ANALYZE THE PROPAGATION RANGE γ

![Figure 5](image_url)

**FIGURE 5.** The impact of propagation range γ on accuracy.

In the formula (1), γ controls the propagation range of the aspect. The position propagation influence decreases as γ increases. In Figure 5, we use accuracy to measure the impact of γ on three public data sets. The value of γ is set between 5 and 35. As can be seen from the Figure5(a), the growth rate is more obvious between 15 and 25.
In Figure 5(b), we magnify the effect of $\gamma$ on the accuracy between 20 and 25. We can see that the accuracy of $\gamma$ is basically stable between 20 and 25. When $\gamma = 21$, $\gamma = 23$, $\gamma = 23$, the accuracy of the three data sets reach the highest. We suspect that the length of the text mainly is between 20 and 25 in the three data sets. For shorter or longer texts, the effect of words on the aspect decreases as the distance increases.

E. THE EFFECT OF NUMBER OF ITERATIONS ON THE MODEL

Figure 6 shows the effect of the number of iterations of the three data sets on the model. The loss is relatively large at the beginning. As the number of iterations increases, the loss gradually decreases. After 30 iterations of the Restaurant, the model basically reached a convergence state, while Laptop and Twitter reached convergence at 18 and 49 respectively. Twitter has a slower convergence time than Restaurant and Laptop.

![Figure 6](image1.png)

**FIGURE 6.** The effect of the number of iterations.

F. CASE STUDY

![Figure 7](image2.png)

**FIGURE 7.** Attention weights are visualized on two sentences by PSRTN.

In order to better understand the PSRTN model, we use PSRTN model to predict the sentiment polarity of the review “Boot time is super-fast, around anywhere from 35 seconds to 1 minute, but quite unreasonably priced” for "Boot time" and "Priced". Figure 7 visualizes the attention of the word, the deeper the color, the greater the weight of attention.

The PSRTN model accurately infers the sentiment polarity of aspects in a given text. The sentiment polarity of "Boot time" is positive and the sentiment polarity of "priced" is negative. As can be seen from Figure 7, the model considers the key words in the aspect. For example, the aspect "Boot time" calculates the weight of two words by the self-attention mechanism, and the word "time" should give more attention in aspect. PSRTN model also considers position relationship. For example, when inferring the sentiment polarity of "Boot time", the word "super-fast" with a close distance is significant, while the words "seconds", "but", "quite" with a long distance are less weighted. Similarly, "priced" is also affected by the word "unreasonably". Even if there are multiple aspects in a given text, the PSRTN model can find relevant sentiment words based on a given aspect term. From the second example, "The menu is limited but almost all of the dishes are excellent,” we can also see that the PSRTN model calculates key words in aspect terms and focus on words that are close to aspect terms. Through the above analysis, the PSRTN model can correctly judge the sentiment polarity of the aspect.

V. CONCLUSION AND FUTURE WORK

In this paper, we present a PSRTN model for aspect-level sentiment classification. PSRTN effectively exploits the relationship between aspect and context and gets better contextual semantic information. The PSRTN considers the influence of position information, which obtains positional information between aspects and contexts through a Gaussian kernel function and generates a position vector. PSRTN uses the self-attention mechanism to obtain key words in the aspect, so that the model can better extract sentiment words for keywords. PSRTN also utilizes R-Transformer to obtain long-term dependency information...
and local dependency information. Finally, we conduct experiments on SemEval2014 and Twitter, and our results show that our model exceed all baseline models.

We will further study how to improve the performance of the PSRTN model and apply it to other NLP tasks. On the one hand, there may be a more efficient way to connect position vectors and word embedding. On the other hand, in addition to position features, studying other features is a problem worthy of study for improving the aspect-level sentiment classification.

REFERENCES


Ziyu Zhou is a master student in School of Information Science and Engineering, Shandong Normal University, China. Her research interest includes the natural language processing and data mining.

Fang’ai Liu is a professor, doctoral student guide Architect in the School of Information Science and Engineering, Shandong Normal University, China, he has received PhD in Computer Science from China Science College in 2002, and he is a CCF senior ember, his research interests include the Internet, parallel processing, recommendation system ,prediction system and data mining.

Qianqian Wang is a PhD candidate in School of Information Science and Engineering, Shandong Normal University, China. Her research interest includes the prediction system, recommendation system and data mining.