Syntax-Aware Graph Attention Network for Aspect-Level Sentiment Classification

Lianzhe Huang, Xin Sun, Sujian Li, Linhao Zhang and Houfeng Wang
MOE Key Lab of Computational Linguistics
Peking University, Beijing, 100871, China
{hlz,sunx5,lisujian,zhanglinhao,wanghf}@pku.edu.cn

Abstract

Aspect-level sentiment classification aims to distinguish the sentiment polarities over aspect terms in a sentence. Existing approaches mostly focus on modeling the relationship between the given aspect words and their contexts with attention, and ignore the use of more elaborate knowledge implicit in the context. In this paper, we exploit syntactic awareness to the model by the graph attention network on the dependency tree structure and external pre-training knowledge by BERT language model, which helps to model the interaction between the context and aspect words better. And the subwords of BERT are integrated into the dependency tree graphs, which can obtain more accurate representations of words by graph attention. Experiments demonstrate the effectiveness of our model.

1 Introduction

Aspect-level sentiment analysis (ABSA) (Liu, 2012), as a fundamental natural language processing task, has attracted lots of attentions recently. Its typical application is to classify the target aspects of a product in a customer review into the sentiment polarities of *positive*, *neutral*, or *negative* according to the contexts (Hu and Liu, 2004) . Taking the sentence "*The performance of this laptop is excellent, but the screen is terrible*" as example, the correct ABSA results would be to judge the aspect word *performance* as *positive*, and *screen* as *negative*.

The early approaches of aspect-level sentiment analysis mainly use handcrafted features to train a statistical classifier (Kiritchenko et al., 2014). With the rapid progress of deep neural network (DNN), lots of work based on DNN have been proposed (Dong et al., 2014; Tang et al., 2015b; Ma et al., 2017). These studies mostly depend on recurrent neural networks (RNNs) (Bahdanau et al., 2014) to model the semantic relationship between the aspect words and their contexts. However, their performances are affected by the RNN's limited ability of capturing long-distance dependencies (Werbos, 1990) and loss of syntax information (Sun et al., 2019). To overcome the shortages of RNN-based models mentioned above, there have been a lot of work using various methods like BERT language model and graph neural networks recently.

Since BERT language model (Devlin et al., 2018) based on Transformer can alleviate long-distance dependence of RNN, some studies have designed BERT-based models for the ABSA task and achieved outstanding results (Song et al., 2019; Rietzler et al., 2019; Zhao et al., 2019). To the best of our knowledge, most of these work just utilize BERT simply as an embedding layer, which sequentially models the position embedddings of the context and cannot utilize the syntactic information. Another improvement is the use of GNN in ABSA (Sun et al., 2019), since GNN has the capability to model linguistic structures like dependency trees other than only sequential data. With such a capability, the model can build the connection of some non-adjacent keywords (e.g., *desk* and *like* in Figure 1) and improve their interactions. However, this kind of research does not go far in exploiting how to model the graph elaborately. For example, Sun et al. (2019) treat all the child nodes of a parent node equally. As

Proceedings of the 28th International Conference on Computational Linguistics, pages 799–810
Barcelona, Spain (Online), December 8-13, 2020

Figure 1: An example of the graph in our model. The complete sentence is *They like the desks in their dormitories*. The aspect word is **desk** (colored red), and the keyword which gives out sentiment information is like (colored yellow). The distance between the two words is 1 in the dependency tree shorter than 2 in sequence. And the meaning of **desk** is more important than *they* for the word like (colored yellow).

shown in Figure 1, *they* and *desk* are both child nodes of *like* and should have different positions in the ABSA task.

Through the survey above, we hope to take a further step in solving the long-distance dependence problem and introducing syntactic information for the ABSA task. In this paper, we propose a novel model named **S**yntax-**A**ware **G**raph **AT**tention Network (SAGAT) which fully exploits the potentials of BERT and carefully models the syntactic relations between different words. Specifically, SAGAT first obtain the representation of each subword in the sentence from the BERT. Unlike other BERT-based models (Zhao et al., 2019; Rietzler et al., 2019; Song et al., 2019) which merge the representation of subwords mostly by mean pooling to form word-level embeddings, we retain all the representations of subwords and use them in the subsequent dependency tree graph to obtain more accurate representations of words by graph attention performed later. Following (Sun et al., 2019), we also build a graph for each sentence according to its dependency tree. In this kind of graph, we state above that a common parent word may treat its different child words differently as in Figure 1. To tackle this difference, we design the Graph Attention Network to pay different attentions to different children. Finally, we use the fusion and alignment layers to interact context and aspects, respectively. The fusion layer performs max-pooling on context and aspects to get the most significant features first, then the hidden states of each token in both context and aspects are updated by those captured features. The alignment layer utilizes attention mechanism to capture the most relevant features, and these features are also used for updating the hidden states of tokens. After all operations above, the representation of context and aspects concentrated to pass through a linear layer to get the final output. Experiments demonstrate the effectiveness of our model. The main contributions of this paper are presented as follows:

- To the best of our knowledge, SAGAT is the first model to combine both syntactic information and external pre-training knowledge for the sentiment classification task.
- We use syntax-guide graph attention to reduce the distance between keywords to ease RNN's long distance dependency, and tell the differences between various child nodes.
- We preserve subwords of BERT in the graph to fully release the power of BERT.
- We evaluate our method on the SemEval 2014 datasets and experiments show that SAGAT achieves outstanding performance.

2 Related Work

In this section, we will briefly review works on aspect-level sentiment analysis, graph neural network, and BERT language model.

2.1 Aspect-Level Sentiment Analysis

Aspect-level sentiment is widely used in scenarios like e-commerce and social network (Zhang, 2008). Researchers usually focus on the fusion of context and aspects to obtain corresponding results. Traditional methods utilize handcrafted features like sentiment lexical features and bag-of-words features to

train sentiment classifiers (Rao and Ravichandran, 2009). With the development of Deep Neural Network, more DNN-based models have been proposed. There are mainly two categories models including semantic-based and syntactic-based methods. Semantic-based models (Wang et al., 2016; Ma et al., 2017; He et al., 2018; Song et al., 2019) usually use the attention mechanism to capture and amplify the key semantic information of sentences and aspects. However, most of the previous works neglect the power of syntax information. Syntactic-based methods (Sun et al., 2019; Zhao et al., 2019; Huang and Carley, 2019; Zhang et al., 2019; Lin et al., 2019) introduce the results of dependency parsing into the DNN models to shorten the distance between the aspect and the keyword and introduce syntactic information. DNN-based models including semantic-based and syntactic-based methods can generate dense vectors of sentences without handcrafted features.

2.2 Graph Neural Network

Graph neural networks have recently become very popular in NLP research. GNN was first proposed in (Scarselli et al., 2009) and has been used in many tasks in NLP including text classification (Defferrard et al., 2016), sequence labeling (Zhang et al., 2018), neural machine translation (Bastings et al., 2017), and relational reasoning (Battaglia et al., 2016). Tai et al. (2015) first attempts to use GNN in the sentiment classification task. Recently, Zhao et al. (2019) proposed to model sentiment dependency within one sentence by building graphs between them, and Sun et al. (2019) introduced dependency parsing and implemented simple graph convolution to propagate their graph. They all achieve outstanding performance by applying GNN in their model. While Zhao et al. (2019) must parse out all the aspects first in the sentence, and no syntax information is introduced. Sun et al. (2019) makes use of syntax information, but they only implement simple graph convolution on their syntax graph, which can not tell the differences between nodes, and they simply using mean pooling to get final results after GCN with deeper integration between aspect and context.

2.3 BERT

BERT is one of the key innovations in the recent progress of contextualized representation learning (Peters et al., 2018; Devlin et al., 2018). Traditional word embeddings like Glove (Mikolov et al., 2013; Pennington et al., 2014) are trained among large scale of corpora, while all these methods finally get the superposition of different meanings of words. BERT adopts a fine-tuning mechanism that almost needs no specific requirement architecture for each end task. Recently, some BERT-based models (Zhao et al., 2019; Song et al., 2019) have been adopted in sentiment analysis task. But almost all of these models utilize BERT as an embedding layer, which is better than Glove.

3 Methodology

In this section, we will introduce the details of all the layers in our model separately. Figure 2 shows the overall architecture of the proposed **S**yntax-**A**ware **G**raph **AT**tention Network (SAGAT), which consists of an encoding layer, graph attention layer, fusion layer, alignment layer, and output layer.

3.1 Encoding

The raw text sequences of contexts and aspects are first represented as embedding vectors to feed a pre-trained BERT. The input context sequence $S_c = \{w_c^1, w_c^2, ..., w_c^m\}$ with length m and aspect sequence $S_a = \{w_a^1, w_a^2, ..., w_a^n\}$ with length n are first cut into subword tokens and concatenated, which become:

$$S = \{[\text{CLS}], w_c^1, w_c^{1\#1}, w_c^2, ..., w_c^{m\#k}, [\text{SEP}], w_a^1, ..., w_a^{n\#k'}, [\text{SEP}]\} \quad (1)$$

where $w^{n\#k}$ represents the k-th subword of w_n, [CLS] and [SEP] are special token of BERT.

Then the transformer encoder captures the semantic information of each subword through self-attention, and the encoder produces an embedded sequence with semantic information.

We notice that almost all the previously proposed BERT-based models merge subwords by mean pooling after encoding because their subsequent layers cannot deal with the structure of subwords. It's indisputable that the importance of different subwords is not consistent, so merge them here by mean

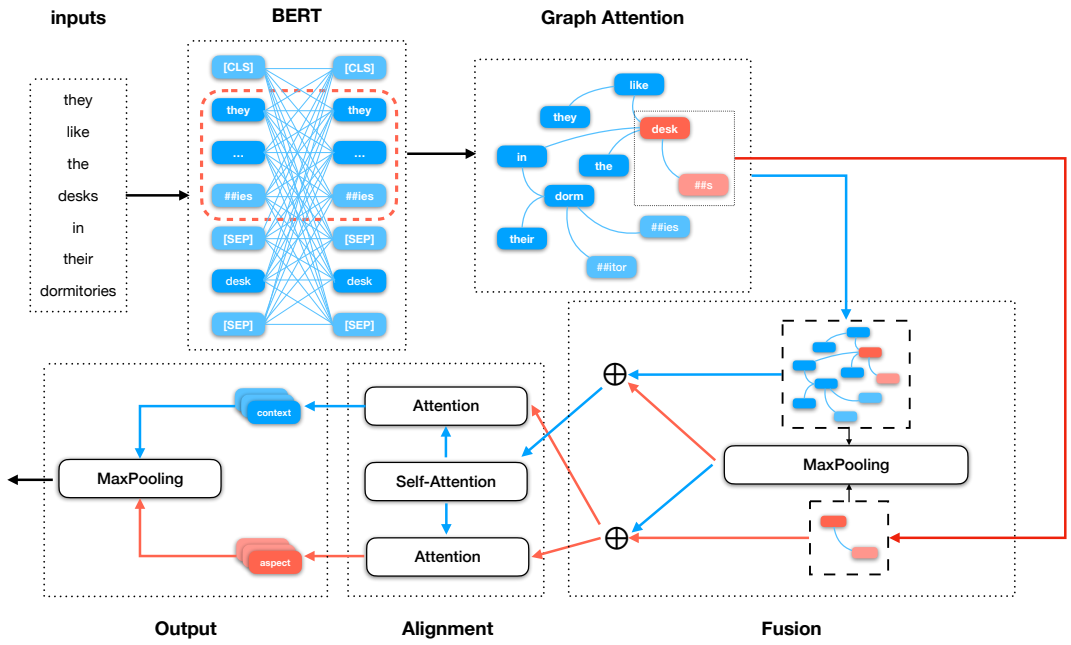

Figure 2: The overview of our model.

pooling is not appropriate. It is worth mentioning that we do not merge the subwords after BERT encoding layer, and they will be processed more effectively later by the graph attention layer that we introduce later.

So the final output of encoding layer is:

$$X = \{x^{1\#1}, ...x^{1\#l_1}, x^{2\#1}, ..., x^{(n+m)\#1}, ..., x^{(n+m)\#l_{n+m}}\} \tag{2}$$

where $x \in \mathbb{R}^d$, d is the dimension of encoding vectors.

3.2 Graph Attention

To get the syntax information of the text, we first perform dependency parsing on the input sequence S. The result of dependency parsing can be expressed as a list of tuples, and each tuple represents a pair of parent-child relationships on the parse tree. The parsing list can be notated as $P = [(w_{s1}, w_{e1}), (w_{s2}, w_{e2}), ..., (w_{sp}, w_{ep})]$, where w_{sp} ,w_{ep} means the p-th start and the end word respectively, p is the length of parsing results.

After obtaining the parsing results, we start to build graphs for each input sequence. Each token (subword) has a corresponding node in the graph, as we have mentioned in Section 3.1. The hidden states of nodes are filled with their encoding outputs. All the words are connected according to the dependency parsing results P, and the subwords $w_n^{\#k}$ are connected to its original word w_n. Concretely, the graph of input sequence S defined as:

$$N = X \tag{3}$$

$$E = P + \{(x_1^{\#1}, x_1), ..., (x_n^{\#k}, x_n)\} \tag{4}$$

where P is the parsing results mentioned before.

Then we perform graph attention on the graph we've built before. Since the children of a node can be its subwords or children in the dependency tree with different meanings to the parent node, attention can capture those differences much better than convolution. The definition of graph attention is described as

follows:

$$z_i = Wx_i \tag{5}$$

$$e_{ij} = \text{LeakyRelu}(A(z_i; z_j)) \tag{6}$$

$$a_{ij} = \frac{\exp(e_{ij})}{\sum_{k \in \mathcal{N}_i} \exp(e_{ik})} \tag{7}$$

$$x'_i = \sigma\Big(\sum_{j \in \mathcal{N}_i} a_{ij} z_j\Big) \tag{8}$$

where ; represents concatenation operation, $W \in \mathbb{R}^{d \times d}, A \in \mathbb{R}^{d \times 2d}$ are learnable parameters of the model, $\mathcal{N}_i$ means all the neighbors of node i and σ is the activation function.

After passing through the graph attention layer, the nodes containing aspect words within the context are duplicated into a set N_a, and the nodes of context make up another set N_c.

3.3 Fusion

It's essential to obtain the interaction information between aspect and context in aspect level sentiment analysis. Note that we attach the aspect to the end of the context when we put the sequence into the BERT model as Eq.1, there would be two aspect words in one sequence. The aspect word in the context contains more contextual information than the one at the end. Therefore, we would fetch the aspect representation in the context to fuse with the context in this layer.

We fetch the aspect representation from context, and then we perform max-pooling on them to get the most meaningful information. After passing through a linear layer, the fusion vector of aspect F_a is obtained. Finally, we directly add the fusion vector to each vector in N_c to get the fused representation of context N'_c. The way to get the fused representation of aspect N'_a is a mirror operation of obtaining N'_c. The definition of the fusion layer is shown as follows:

$$F_c = W_c \,\text{MaxPooling}(N_a) + b_c \tag{9}$$

$$F_a = W_a \,\text{MaxPooling}(N_c) + b_a \tag{10}$$

$$N'_a = N_a + F_c \tag{11}$$

$$N'_c = N_c + F_a \tag{12}$$

where $W_a, W_c \in \mathbb{R}^{d \times d}, b_a, b_c \in \mathbb{R}^d$ are learnable parameters of the model, the shape of N is $(\mathrm{B}, \mathrm{S}, \mathrm{D})$ where B is batch size, S is the sequence length and D is the embedding dimension, the $\mathrm{MaxPooling}$ is performed on the second dimension S.

3.4 Alignment

Aspect words and context sequences can bring the most meaningful features to each other through the fusion layer we've discussed before. With the fusion layer only, we can not get the most accurate aspect and context representation because there is no interaction between them. Therefore we introduce the alignment layer in our model.

In this layer, we perform self-attention on context to enhance contextual information. Then aspect words and context are interacted with each other by attention. After all these operations, context and aspect are fully integrated and ready for output. The definition of the alignment layer is shown as follows:

$$N'_c = \text{Attention}\big(\text{Attention}(N'_c, N'_c), N'_a\big) \tag{13}$$

$$N'_a = \text{Attention}(N'_c, N'_a) \tag{14}$$

in which the $\mathrm{Attention}$ function is defined as follows and the score of $\mathrm{Attention}$ is calculated by dot product:

$$\text{Attention}(\mathbf{k}, \mathbf{q}) = \text{Softmax}\big(f_{score}(\mathbf{k}, \mathbf{q})\big)\mathbf{k} \tag{15}$$

$$f_{score} = W_k \mathbf{k} (W_q \mathbf{q})^\top, W_k, W_q \in \mathbb{R}^{d \times d} \tag{16}$$

Dataset	Positive		Neural		Negative	
	Train	Test	Train	Test	Train	Test
Twitter	1561	173	3127	346	1560	173
Restaurant	2164	728	637	196	807	196
Laptop	994	341	464	169	870	128

Table 1: Datasets overview.

3.5 Output

We get the final representations of the previous outputs by max pooling, concatenate them, and use a fully connected layer to project the concatenated vector into the space of the targeted C classes.

$$P_c = \text{MaxPooling}(N'_c) \tag{17}$$

$$P_a = \text{MaxPooling}(N'_a) \tag{18}$$

$$\text{logits} = W_o[P_c; P_a] + b_o \tag{19}$$

$$y = \text{Softmax}(\text{logits}) \tag{20}$$

where ; represents concatenation operation, $W_o \in \mathbb{R}^{d \times d}, b_o \in \mathbb{R}^d$.

4 Experiments

In this section, we describe our experimental setup and report our experimental results.

4.1 Experimental Setup

For experiments, we utilize three datasets, including SemEval 2014 Task 4 dataset composed of Restaurant reviews and Laptop reviews (Manandhar, 2014) and ACL 14 Twitter gathered by Dong et al. (2014). All the cases in these datasets are labeled with three sentiment polarities *positive, negative* and *neutral*. The details of these datasets are shown in Table 1.

We fine-tune pre-trained BERT[1] in our model. Embedding dim d is set to 768. We utilize Adam (Kingma and Ba, 2014) as optimizer with initial learning rate 2×10^{-5}. Dropout with a keep probability of 0.9 is applied after the dense layer. The batch size of our model is 32, and the max sequence length is set to 128. The number of training epochs is 100. We use Spacy (Honnibal and Montani, 2017) as the dependency parsing tool.

The loss function $\mathcal{L}$ to be optimized in our model is the cross-entropy loss, which can be defined as:

$$\mathcal{L} = -\sum_{i=1}^{C} \hat{y}^i \log(y^i) \tag{21}$$

4.2 Baselines

We compare our model with the following baseline models:

- **TD-LSTM**: Tang et al. (2015a) utilizes two dependent LSTM network to model the left context with aspect and the right context, respectively. The left and right representations are concatenated for predicting the sentiment polarity.
- **ATAE-LSTM**: Wang et al. (2016) appends the aspect words embeddings with each context word embeddings. They use LSTM and attention to get the final representation for prediction.
- **MemNet**: Tang et al. (2016) utilizes multi-hops attention on the context word for sentence representation to tell the importance of each context word.

[1]We use uncased BERT-base from https://github.com/google-research/bert

Models	Twitter		Restaurant		Laptop	
	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1
TD-LSTM	0.7080	0.6900	0.7563	-	0.6813	-
ATAE-LSTM	-	-	0.7720	-	0.6870	-
MemNet	0.6850	0.6691	0.7816	0.6583	0.7033	0.6409
IAN	-	-	0.7860	-	0.7210	-
RAM	0.6936	0.6730	0.8023	0.7080	0.7449	0.7135
TNet	0.7312	0.7101	0.8079	0.7084	0.7654	0.7175
HSCN	0.6960	0.6610	0.7780	0.7020	0.7610	0.7250
CDT	0.7466	0.7366	0.8230	0.7402	0.7719	0.7299
SDGCN-BERT	-	-	0.8357	0.7647	**0.8135**	**0.7834**
AEN-BERT	0.7471	0.7313	0.8312	0.7376	0.7993	0.7631
Our Model	**0.7540**	**0.7417**	**0.8508**	**0.7794**	0.8037	0.7694

Table 2: Main results. The results of baseline models are from published papers. "-" means not reported.

- **IAN**: an RNN-based approach proposed by (Ma et al., 2017), which encodes contexts and aspects words by LSTM and interacts them with attention to generate the representations for aspects and contexts concerning each other.
- **RAM**: Chen et al. (2017) proposed a method based on Memory Network and represents memory with LSTM. Then a gated recurrent unit network is applied to concatenate all the outputs for sentence representation from attention.
- **TNet**: Li et al. (2018) uses BiLSTM embeddings as target-specific embeddings, and utilizes a CNN model to extract final embeddings.
- **HSCN**: Li et al. (2019) selects target words and extracts target-specific contextual representation to measure the deviation between target-specific contextual representation and target representations by capturing interactions between the context and target.
- **CDT**: a method based on graph neural network proposed by (Sun et al., 2019). CDT builds graph by dependency parsing, which is similar but not the same to us and propagates graphs by graph convolution network.
- **SDGCN**: also a GNN-based method by (Zhao et al., 2019), which first considers the sentiment dependencies between aspects by employing GCN to effectively capture the sentiment dependencies between different aspects in one sentence.
- **AEN-BERT**: Song et al. (2019) designs an attentional encoder network to draw the hidden states and semantic interactions between target and context words. And they apply pre-trained BERT to this task, which enhances the performance of the basic BERT model and obtains better results.

It is noted that the results of the baseline models are directly retrieved from published papers.

4.3 Main Results

Table 2 reports the main results of our model against other baseline models. We can see that our model generally achieve state-of-the-art performance in *Restaurant* and *Twitter* datasets. It is worth mentioning that although our model performs slightly worse than SDGCN-BERT on the Laptop dataset, which will discuss later, we still achieve the best average performance of *Restaurant* and *Laptop*, which is exactly the coverage of the SemEval 2014.

Generally speaking, graph-based models would introduce some additional information compared to traditional methods. SDGCN (Zhao et al., 2019) connects different aspects in one sentence to build two

kinds of graphs, nodes in one graph are connected in the order of their appearance in the sentence, and the other are all connected in pairs, which does not utilize the dependency information between aspects. CDT (Sun et al., 2019) builds graphs by dependency parsing, which is similar to us. The advantage of this method is that it can shorten the distance from keywords to aspect words as shown in Figure 1. However, graphs built by SDGCN still comes from sequence information; no syntax information is introduced. And CDT utilizes native GCN rather than attention mechanism when propagates the graph, which would cause child nodes of the same parent node in the dependency tree to be precisely in the same position when updating the graph state, which is incorrect in actual language scenarios. As shown in Figure 1, for the word *like*, the meaning of *desk* (underlined) should be more influential than *they*. In our model, syntax information is introduced by the graph based on dependency parsing. And differences between child nodes are also well distinguished by graph attention network.

We can also find that BERT-based models perform better overall due to the strong ability of BERT. However, the previous BERT-based models like SDGCN (Zhao et al., 2019) and AEN (Song et al., 2019) did not take advantage of "subwords" into their models, and just merged the subwords. That means they used BERT as an embedding layer only, which obviously could not play all of BERT's strength. In our model, the subwords can be used as the child nodes of its parent word and participate in the calculation of the rest of our models. Besides, with the help of GAT, subwords can also affect the meaning of its parent word with different weights corresponding to their importance.

We noticed that our model has a little performance degradation on the laptop dataset. We compared the laptop dataset with the other two datasets. From the perspective of data distribution differences, the reason for this phenomenon may be too many numbers and professional terms in the cases of the laptop dataset. The parser is more likely to obtain parsing errors when facing these professional terms. We take a case in laptop datasets as an example:

*bluetooth (2.1) , fingerprint reader , full 1920x1080 screen - integrated mic/webcam * - dual touchpad mode is interesting , and easy to use -5 usb ports - runs about 38-41c on idle , up to 65 (for me) on load - very quiet - i could go on and on .*

The example above contains a lot of numbers and technical terms, and the parser will usually get confusing results when parsing such sentences.

4.4 Ablation Study

To investigate the effects of different components in our model, we conduct the following ablation study on our model. The results of the ablation study are shown in Table 3.

(1) **w.o. GAT**: we remove the graph attention network in our model; no syntax information is introduced into the model.

(2) **w.o. Fusion**: we remove the fusion layer and remain other layers in our model the same as the original one.

(3) **w.o. Alignment**: the same settings as (2) but removed alignment layer

(4) **with GCN**: we use graph convolution with mean-pooling to propagate the graph instead of graph attention.

It is worth mentioning that we did not treat BERT as a component for ablation study since we introduced the subword of BERT into our model instead of simply using BERT as the embedding layer like other model of this task .

As expected, results for the simplified models all drop a lot, which demonstrates the effectiveness of these components.

When graph attention is removed in Experiments (1), vectors encoded by BERT are send to the subsequent layers directly.Thus our model would not be able to obtain the syntax information from the dependency tree. The model naturally degenerates to a similar result to other models using BERT as the embedding layer.

	Twitter		Restaurant		Laptop	
	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1
Original Model	0.7540	0.7417	0.8508	0.7794	0.8006	0.7625
(1) w.o. GAT	0.7467	0.7360	0.8453	0.7595	0.7849	0.7443
(2) w.o. Fusion	0.7351	0.7207	0.8480	0.7657	0.7828	0.7505
(3) w.o. Alignment	0.7481	0.7330	0.8454	0.7758	0.7817	0.7417
(4) with GCN	0.7510	0.7404	0.8481	0.7726	0.7943	0.7586

Table 3: Ablation Study results.

#	Sentence	Orig	w.o. GAT
(*i*)	the **specs** are pretty <u>good</u> too	✔	✖
(*ii*)	since the machine's **slim** **profile** is critical to me , that was a <u>problem</u>	✔	✖
(*iii*)	**working** with mac is so much <u>easier</u> , so many cool features .	✔	✖
(*iv*)	so <u>nice</u> . <u>stable</u> . <u>fast</u> . now i got my **ssd**	✖	✖

Table 4: (*i*)-(*iii*) are three examples taken from Laptop dataset. The given aspect words are **bold**, keywords that determine the classification result are <u>underlined</u>. The solid lines between words represent the edges on the dependency tree. For the sake of display,we only draw the edges between the aspect words and the keywords in (*i*) to (*iii*) . In (*iv*) we draw all the edges on the dependency tree. Orig and w.o. GAT mean the classification results from original model and model without graph attention layer, respectively.

Experimental results in Experiments (2) show that the fusion layer, which fuses the most prominent features between context and aspect by max-pooling, is simple but effective, as we have mentioned in Section 3.3.

In Experiment (3), when the alignment is removed but the fusion is retained, performance degradation still occurs, which indicates that the alignment layer and the fusion layer play different roles in our model. Since the operations of these two layers on the context and aspect are mirrored, let's take the context as an example, the fusion layer would send the most significant features of the aspect to the context. In contrast, the alignment layer mixes the most relevant features of the aspect with the current context.

To verify the actual performance of GAT, we designed Experiment (4). In Experiment (4), we use GCN instead of GAT for graph propagation, which means that nodes no longer use attention to assign neighbors' weights before updating their hidden states, but only average their neighbors' hidden states as the new one. The nodes would not be able to tell the different importance of their neighbor nodes. The experimental results drop with our expectations. But overall, it is still better than Experiment (1), where GAT is moved completely.

4.5 Case Study

To better explore the role of the graph attention layer, we conducted case study between SAGAT without graph attention layer and itself. We selected some cases from the classification results of these two models for discussion. They are shown in Table 4.

In example (*i*) to (*iii*), the original model classifies correctly but gets wrong when removing the GAT layer. We found that these cases have something in common, that is the aspect word and keywords are far away in sequence, but much closer on the dependency tree. We can see that the distance in the dependency tree is significantly reduced, no matter in short sentences like example (*i*) or long sentences like (*ii*) and (*iii*). From this, we can observe that there are two benefits of introducing graph based on the dependency tree. One is to obtain the syntax information that may be missing in the previous encoding procedure, and the other is to make the distance between keywords closer, thus deepen their mutual influence.

Besides, there are some cases with less formal expressions in the dataset, like example (*iv*). Due to the incorrect usage of punctuation, the sentence is split into multiple dependency trees, and keywords become inaccessible on the dependency tree. Thus the distance between keywords cannot be reduced. For these representations, they get some extra syntax knowledge without more benefits. Therefore, the classification result is wrong.

5 Conclusion

In this paper, we proposed a model named SAGAT for aspect-level sentiment classification. To fully obtain both syntax and semantic information in the sequence, we utilize graph attention network and BERT in our model. Experiments on three datasets demonstrate the effectiveness of our model. We also perform some additional experiments to show the power of components in SAGAT.

Acknowledgments

Our work is supported by the National Key Research and Development Program of China under Grant No.2017YFB1002101 and National Natural Science Foundation of China under Grant No.62036001 and No.61876009. The corresponding author of this paper is Houfeng Wang.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Simaan. 2017. Graph convolutional encoders for syntax-aware neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1957–1967.

Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. 2016. Interaction networks for learning about objects, relations and physics. In *Advances in neural information processing systems*, pages 4502–4510.

Peng Chen, Zhongqian Sun, Lidong Bing, and Wei Yang. 2017. Recurrent attention network on memory for aspect sentiment analysis. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 452–461.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*, pages 3844–3852.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Li Dong, Furu Wei, Chuanqi Tan, Duyu Tang, Ming Zhou, and Ke Xu. 2014. Adaptive recursive neural network for target-dependent twitter sentiment classification. In *Proceedings of the 52nd annual meeting of the association for computational linguistics (volume 2: Short papers)*, pages 49–54.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2018. Effective attention modeling for aspect-level sentiment classification. In *Proceedings of the 27th international conference on computational linguistics*, pages 1121–1131.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177.

Binxuan Huang and Kathleen M Carley. 2019. Syntax-aware aspect level sentiment classification with graph attention networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5472–5480.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Svetlana Kiritchenko, Xiaodan Zhu, Colin Cherry, and Saif Mohammad. 2014. Nrc-canada-2014: Detecting aspects and sentiment in customer reviews. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, pages 437–442.

Xin Li, Lidong Bing, Wai Lam, and Bei Shi. 2018. Transformation networks for target-oriented sentiment classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 946–956.

Zheng Li, Ying Wei, Yu Zhang, Xiang Zhang, and Xin Li. 2019. Exploiting coarse-to-fine task transfer for aspect-level sentiment classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4253–4260.

Peiqin Lin, Meng Yang, and Jianhuang Lai. 2019. Deep mask memory network with semantic dependency and context moment for aspect level sentiment classification. In *IJCAI*, pages 5088–5094.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2017. Interactive attention networks for aspect-level sentiment classification. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 4068–4074.

Suresh Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Delip Rao and Deepak Ravichandran. 2009. Semi-supervised polarity lexicon induction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 675–682. Association for Computational Linguistics.

Alexander Rietzler, Sebastian Stabinger, Paul Opitz, and Stefan Engl. 2019. Adapt or get left behind: Domain adaptation through bert language model finetuning for aspect-target sentiment classification. *arXiv preprint arXiv:1908.11860*.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2009. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80.

Youwei Song, Jiahai Wang, Tao Jiang, Zhiyue Liu, and Yanghui Rao. 2019. Attentional encoder network for targeted sentiment classification. *arXiv preprint arXiv:1902.09314*.

Kai Sun, Richong Zhang, Samuel Mensah, Yongyi Mao, and Xudong Liu. 2019. Aspect-level sentiment analysis via convolution over dependency tree. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5683–5692.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2015a. Effective lstms for target-dependent sentiment classification. *arXiv preprint arXiv:1512.01100*.

Duyu Tang, Bing Qin, and Ting Liu. 2015b. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1422–1432.

Duyu Tang, Bing Qin, and Ting Liu. 2016. Aspect level sentiment classification with deep memory network. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 214–224.

Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. 2016. Attention-based lstm for aspect-level sentiment classification. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 606–615.

Paul J Werbos. 1990. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560.

Yue Zhang, Qi Liu, and Linfeng Song. 2018. Sentence-state lstm for text representation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 317–327.

Chen Zhang, Qiuchi Li, and Dawei Song. 2019. Syntax-aware aspect-level sentiment classification with proximity-weighted convolution network. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1145–1148.

Zhu Zhang. 2008. Weighing stars: Aggregating online product reviews for intelligent e-commerce applications. *IEEE Intelligent Systems*, 23(5):42–49.

Pinlong Zhao, Linlin Hou, and Ou Wu. 2019. Modeling sentiment dependencies with graph convolutional networks for aspect-level sentiment classification. *Knowledge-Based Systems*, page 105443.