Aspect Fusion Graph Convolutional Networks for Aspect-Based Sentiment Analysis

Fuyao Zhang, Yijia Zhang, Shuo Hou, Fei Chen, and Mingyu Lu(✉)

School of Information Science and Technology, Dalian Maritime University, Dalian 116024, Liaoning, China
lumingyu@dlmu.edu.cn

Abstract. Aspect-based sentiment classification aims to distinguish the sentiment polarities over aspect terms in a sentence. Recent approaches to aspect-based sentiment classification use graph-based models to integrate the syntactic structure of sentences. While being practical, these methods ignore the close relationship between the topological structure of the dependency tree and the dependency distance. To solve this problem, we propose to build an Aspect Fusion Graph Convolutional Network (AFGCN) of sentences to take advantage of syntactic information and word dependencies. Specifically, we enhance the syntactic dependencies of each instance by introducing dependency tree and dependency-position graph. Then, we use two graph convolutional networks to fuse the dependency tree and the dependency-position graph to generate the interactive emotion features of aspects. Finally, we use a novel attention mechanism to fully integrate the significant features related to aspect semantics in the hidden state vectors of the convolution layer and the masking layer. Extensive experiments on five benchmark datasets show that our method achieves state-of-the-art performance.

Keywords: Aspect-based sentiment analysis · Graph convolutional network · Syntactic dependency

1 Introduction

Aspect-based sentiment analysis (ABSA) aims at fine-grained sentiment analysis of sentiment texts such as product reviews. More specifically, ABSA involves two tasks: (1) identifying various aspects of a sentence, (2) determining the sentiment polarity (for example, positive, negative, neutral) expressed in a particular aspect. This paper focuses on the second task: Aspect-based Sentiment Classification. For example, in a comment about a laptop saying, “From the speed to the multitouch gestures this operating system beats Windows easily.”, the sentiment polarities for two aspects of operating system and Windows are positive and negative, respectively. In the task of aspect sentiment analysis, we need to distinguish sentiment polarity according to different aspects.

In the early research of ABSA (Jiang et al., Mohammad et al.) [1, 2], machine learning algorithm is often used to construct sentiment classifiers. Dependency-based parse trees are used to provide more comprehensive syntax information. Therefore, the

H. Lin et al. (Eds.): CCIR 2021, LNCS 13026, pp. 74–87, 2021.
https://doi.org/10.1007/978-3-030-88189-4_6

whole dependency tree can be encoded from leaf to root by recursive neural network (RNN) (Dong et al., Nguyen et al., Wang et al.) [3–5]. Then various neural network models (Dong et al., Vo et al., Chen et al.) [3, 6, 7] are proposed, including long short-term memory network (LSTM) (Wang et al.) [8], convolutional neural network-based (CNN) (Huang et al., Li et al.) [9, 10], and memory-based (Tang et al.) [11] or hybrid methods (Xue et al.) [12], or the distance of the internal node can be calculated and used for attention weight decay (He et al.) [13]. These models represent a sentence as a word sequence, ignoring the syntactic relationship between words, making it difficult for them to find words far away from the expected words. In recent years, several studies have used graph-based models to combine sentence syntactic structure (Zhang et al., Sun et al., Huang et al., Liang et al., Chen et al.) [9, 14–17], which has better performance than the model without considering syntactic relationships. However, the above model only fully considers the topology structure of dependency tree, or the actual distance between words, but does not fully play to the advantages of dependency tree, and does not fully integrate the topology structure of dependency tree and dependency distance. The shortcomings of these approaches should not be overlooked.

To better capture opinion features for aspect sentiment classification, we propose the AFGCN model, which fully combines the topological structure and the dependency distance calculated from the dependency tree. Inspired by the position mechanism [18], this model aggregates valid features in an LSTM-based architecture and uses the syntactic proximity of a context word to the aspect, also known as proximity weight, to determine its importance in a sentence. At the same time, we apply GCN network on dependency tree and dependency-position graph, respectively. We can use long-range multiword relations and syntactic information through GCN to potentially draw syntactically related words to the target. The output is fed into a masking mechanism, which filters out non-aspect words to get focused aspect features. Aspect-specific features are fed into the LSTM output, and the aspect fusion attention mechanism is used to update the most relevant features. After all operations above, the representation of context and aspects concentrated on passing 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:

- We build a complex task-specific syntactic dependency module, which profoundly integrates dependency tree and dependency-position graph to enhance the syntactic dependency of each instance.
- An aspect fusion graph convolutional network model (AFGCN) was proposed, which combined attention mechanism to fully integrate prominent features related to aspect semantics in the hidden state vectors of the convolutional layer and the masking layer, to fully combine the topology structure and dependency distance of the dependency tree.
- Experimental results on five benchmark datasets show the effectiveness of our proposed model in capturing them in aspect-based sentiment classification.

2 Related Work

In aspect sentiment analysis, some early work focused on using machine learning algorithms to capture sentiment polarity based on rich features of content and syntactic structure (Jiang et al., Kiritchenko et al.) [1, 2]. The latest development of aspect-level sentiment classification (ASC) focuses on developing various types of deep learning models. The neural models without considering the syntactic models can be divided into several types: LSTM based (Tang et al., 2016a; Ma et al., 2017) [11, 19], CNN based (Huang et al., Li et al.) [9, 10], memory-based methods (Tang et al., Chen et al.) [7, 11], etc. In neural network approaches, some use RNN variants (such as LSTM and GRU) to model the sentence representation (Majumder et al.) [20].

Syntactic information allows dependency information to be kept in long sentences and helps to bridge the gap between aspects and opinion words. Tai et al. [21] proposed a tree-structured LSTM, which enables people to learn the dependency information between words and phrases. Mouetal et al. [22] utilizes the short path of dependency trees and uses convolutional neural networks to learn the representation of sentences. Recently, some studies use graph-based models to integrate syntactic structures. Zhang et al. [14] uses GCN to capture specific aspects of syntactic information and word dependency on the syntactic dependence tree. Liang et al. [23] proposes an Interactive Graph Convolutional Networks (InterGCN) model to extract both aspect-focused and inter-aspects sentiment features for the specific aspect. Zhang et al. [24] convolutes over hierarchical syntactic and lexical graphs and builds a concept hierarchy on both the syntactic and lexical graphs for differentiating dependency relations.

These observations enable us to build a neural model of dependency trees that fully integrates syntactic dependence and distance and makes accurate sentiment predictions about certain aspects. Specifically, we propose an Aspect Fusion Graph Convolutional Networks model (AFGCN).

3 The Proposed Model

The overall architecture of the proposed AFGCN model is shown in Fig. 1. We first assume a sentence with n words and m aspects from the SemEval-2014 dataset, i.e. $s = \{w_0, w_1, ..., w_a, w_{a+1}, ..., w_{a+m-1}, ..., w_{n-1}\}$, where w_i represents the i -th contextual word and w_a represents the start token of aspect words. Each word is embedded into a low-dimensional real-valued vector with a matrix $V \in R^{|N| \times d_i}$, where $|N|$ is the size of the dictionary while d_i is the dimension of a word vector. We use the pre-trained word embedding GloVe to initialize the word vectors, and the resulting word embeddings are adopted to a bidirectional LSTM to produce the sentence hidden state vectors h_t. Since the input representation already contains aspect information, the context representation specific to the aspect is obtained by linking the hidden state from both directions: $h_t = [\overrightarrow{h_t}; \overleftarrow{h_t}]$ where $\overrightarrow{h_t}$ is the hidden state from the forward LSTM and $\overleftarrow{h_t}$ is from the backward.

3.1 Producing Dependency Tree

We use spacy[1] to construct a given sentence into a directed dependency tree.

Then we construct the adjacency matrix based on the directed dependency tree, and we set all the diagonal elements of the matrix to 1. If there is a dependency between two words, we also write down the corresponding position in the matrix as 1.

And then, an adjacency matrix $M_{ij}^{T} \in R^{n \times n}$ is derived from the dependency tree of the input sentence.

3.2 Producing Dependency-Position Graph

To highlight the relationship between context and aspect, we compute the relative position weight of each element of the adjacency matrix according to aspect.

$$W_{i,j}^{F} = \begin{cases} 1 & if\ w_i \in \{a_i^s\}\ and\ w_j \in \{a_i^s\} \\ 1/(|j - p^b| + 1) & if\ w_i \in \{a_i^s\} \\ 1/(|i - p^b| + 1) & if\ w_j \in \{a_i^s\} \\ 0 & otherwise \end{cases} \tag{1}$$

where $|\cdot|$ is an absolute value function, p^b is the beginning position of the aspect, $\{a^s\}$ is the word set of the aspect.

To establish a closer dependency relationship between context words, we integrate ordinary dependency graph $D_{\mathrm{i,j}}^{G}$, which is obtained by the adjacency matrix of the dependency tree symmetrically along the diagonal, and relative position weight $W_{\mathrm{i,j}}^{G}$ to derive the adjacency matrix of the dependency-position graph.

$$M_{i,j}^{G} = \begin{cases} 1 + W_{i,j}^{G}\ if\ D_{i,j}^{G} = 1 \\ W_{i,j}^{G}\quad otherwise \end{cases} \tag{2}$$

3.3 Proximity-Weight Convolution

Previous dependency tree-based models mainly focus on the topology of the dependency tree or the distance of the dependency tree. However, few models apply them together, limiting the effectiveness of these models in identifying key context words used in representation. This syntactic dependency information is formalized as an adjacent weight in our proposed model, which describes the proximity between context and aspect. Recall the example of the dependency tree in Fig. 1: "But the staff was so horrible to us." The actual distance between the aspect "staff" and the sentiment word "horrible" is 3, but the dependency distance is 1. Intuitively, dependency distance is more beneficial to aspect-based sentiment classification than ordinary distance.

[1] In this work, we use spaCy toolkit for producing the dependency tree of the input sentence: https://spacy.io/.

We construct a dependency tree and then compute the dependency distance for the context words: the length of the shortest dependency path between the aspect and the sentiment words. If the aspect contains multiple words, we minimize the dependency distance between the context and all aspect words. The dependency proximity weights of the sentence are computed by the formula below:

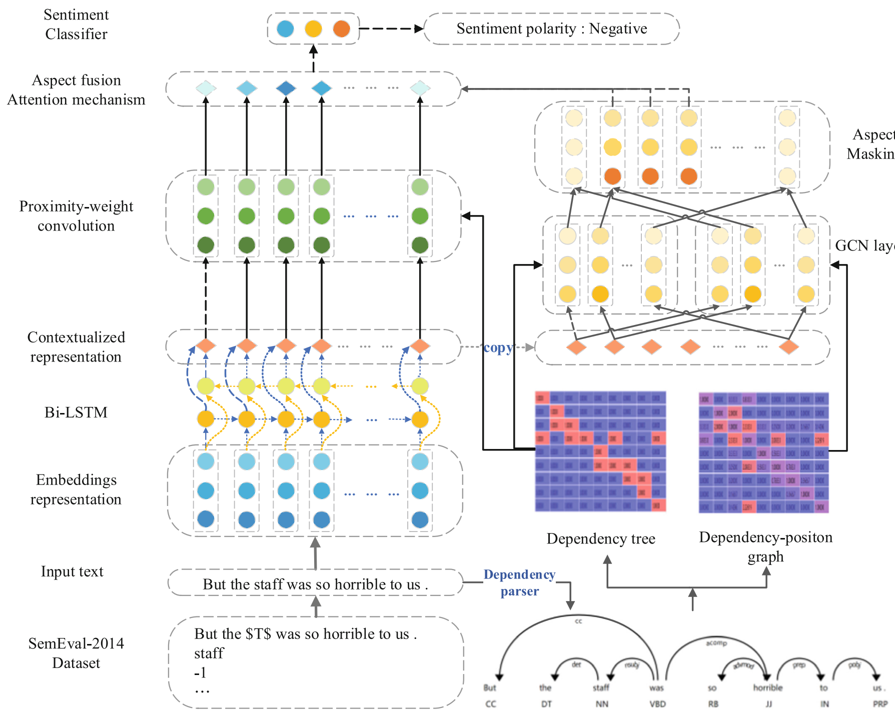
Sentiment Classifier
Sentiment polarity : Negative
Aspect fusion Attention mechanism
Aspect Masking
Proximity-weight convolution
GCN layers
Contextualized representation
copy
Bi-LSTM
Embeddings representation
Dependency tree
Dependency-positon graph
Input text
But the staff was so horrible to us .
Dependency parser
SemEval-2014 Dataset
But the T was so horrible to us .
staff
-1
…
But the staff was so horrible to us .
CC DT NN VBD RB JJ IN PRP

Fig. 1. Overview of aspect fusion graph convolutional network.

$$p_i = \begin{cases} 1 - \frac{d_i}{n} & 0 \le i < \tau or \tau + m \le i < n \\ 0 & \tau \le i < \tau + m \end{cases} \quad (3)$$

where proximity weight $p_i \in R$, d_i is the dependency distance from the word to aspect in the sentence.

Inspired by Zhang et al. [14], we introduce proximity-weight convolution. Unlike the original definition of convolution, proximity-weighted convolution allocates proximity weights before convolution calculation. It is essentially a one-dimensional convolution with a length-l kernel. The proximity-weight convolution process is then assigned as:

$$q_i \in max\left(W_c^T[r_{i-t} \oplus \cdots \oplus r_i \oplus \cdots \oplus r_{i+t}] + b_c, 0\right) \quad (4)$$

where $r_i = p_i h_i$ and $t = \left\lfloor \frac{l}{2} \right\rfloor$, $r_i \in R^{2d_h}$ represents the proximity-weighted representation of the i -th word in the sentence, $q_i \in R^{2d_h}$ represents the feature representation obtained from the convolution layer, and $W_c \in R^{l \cdot 2d_h \times 2d_h}$ and $b_c \in R^{2d_h}$ are weight and bias of the convolution kernel, respectively.

3.4 Aspect Fusion Graph Convolutional Network

Aiming to take advantage of syntactic dependency, we use two graph convolutional networks to fuse dependency tree and dependency-position graph, respectively, to generate interactive sentiment features for aspect. The representation of each node is calculated with graph convolution with normalization factor, and the representation of each node is updated according to the hidden representations of its neighborhood:

$$h_i^l = Relu((\sum\nolimits_{j=1}^{n} M_{ij} W^l g_j^{l-1})/(d_i + 1) + b^l) \tag{5}$$

$$g_j^l = P(h_i^l) \tag{6}$$

where $g_j^{l-1} \in R^{2d_h}$ is the representation of the j -th token evolved from the preceding GCN layer. P(·) is a PairNorm function that integrates position-aware transformation and has been used in previous GCN network (Xu et al., Zhao et al.) [25, 26]. M_{ij} includes M^G and M^T, we take these two matrices integrating different dependency relationships as the inputs of two groups of GCN, respectively. D_i is the degree of the i -th token in the tree. W^l and b^l are trainable parameters, respectively.

Then, we can capture the final representation of the GCN layers from different inputs, h^G and h^T, where h^G is the representation of M^G and h^T is the representation of M^T. And thus, inspired by Liang et al. [23], we combine these two final representations to extract the interactive relations between dependency-position feature and dependency feature:

$$\tilde{h} = h^G + \gamma h^T \tag{7}$$

where γ is the coefficient of dependency feature. The combination method takes into account both syntactical dependence and long-term multi-word relations. We use aspect masking to mask non-aspect representations to highlight the critical features of aspect words. In other words, we keep the final representation of the aspect words output by the GCN layer and set the final representation of the non-aspect words to 0.

3.5 Aspect Fusion Attention Mechanism

We intend to fuse the significant features related to aspect semantic in the hidden state vectors of the convolutional layer and the masking layer through a new way of Aspect fusion Attention mechanism, and setting accurate attention weight for each contextual word accordingly. The attention weights assigning process is formulated below:

$$u_{it} = tanh(W_w q_i h_i^M + b_w) \tag{8}$$

$$\alpha_t = \frac{exp(u_{it}^T u_w)}{\sum_{i=1}^{n} exp(u_{it}^T u_w)} \tag{9}$$

where h_i^M and q_i are the final hidden state vectors output by the Masking layer and the convolution layer respectively. W_w and U_w are weights that are randomly initialized. Then we use the formula $r = \sum_{t=1}^{n} \alpha_t q_i$ to get the corresponding attention weight.

Table 1. Statistics of the experimental datasets.

Dataset	Positive		Neural		Negative	
	Train	Test	Train	Test	Train	Test
Twitter	1561	173	3127	346	1560	173
Lap14	994	341	464	169	870	128
Rest14	2164	728	637	196	807	196
Rest15	912	326	36	34	265	182
Rest16	1240	469	69	30	439	117

3.6 Model Training

The aspect-based representation r is passed to a fully connected softmax layer whose output is a probability distribution over the different sentiment polarities.

$$p = \text{softmax}(W_p r + b_p) \tag{10}$$

where W_p and b_p are learnable parameters for the sentiment classifier layer.

The model is trained with the standard gradient descent algorithm by minimizing the cross-entropy loss on all training samples:

$$\zeta = -\sum_{i}^{J} p_i log \overset{\wedge}{p_i} + \lambda ||\Theta|| \tag{11}$$

where J is the number of training samples, p_i and $\overset{\wedge}{p_i}$ is the ground truth and predicted label for the i -th sample, Θ represents all trainable parameters, and λ is the coefficient of L2-regularization.

4 Experiments

4.1 Datasets and Experimental Settings

Datasets

Our experiments are conducted on five datasets: one is the twitter benchmark dataset constructed by dong et al. [3]. The other four are from the SemEval2014 (Pontiki et al., 2014) [27], SemEval2015 (Pontiki et al., 2015) [26], and SemEval2016 (Pontiki et al., 2016) [28] benchmark datasets, which are composed of two types of data: laptops and restaurants. Each sample is composed of comment sentences, aspects, and the sentiment polarity of the aspects. Building on previous work, we remove samples with conflicting polarity and undefined aspects in rest15 and rest16 sentences. The statistics for the datasets are shown in Table 1.

Settings

For the fairness of model comparison, we use similar parameters in the comparison model. In all experiments, we use 300-dimensional preprocessing GloVe vectors (Pennington et al.) [32] as initial word embeddings. The dimension of the hidden state vector is set to 300. To train the model, we use Adam as the optimizer with a learning rate of 0.001. The coefficient of L2-regularization is 10^{-5}, the coefficient γ is set to 0.2, and the batch size is 32. Besides, the number of GCN layers is set to 2, which is the best performing depth in the pilot study. We adopt Accuracy and Macro-Averaged F1 as the evaluation metrics.

Table 2. Comparison results for all methods in terms of accuracy and Fl (%). The best results on each dataset are in bold.

Model	Twitter		Lap14		Rest14		Rest15		Rest16	
	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1
SVM	63.40	63.30	70.49	–	80.16	–	–	–	–	–
ATAE-LSTM	69.65	67.40	69.14	63.18	77.32	66.57	75.43	56.34	83.25	63.85
Mem-Net	71.48	69.90	70.64	65.17	79.61	69.64	77.31	58.28	85.44	65.99
RAM	69.36	67.30	74.49	71.35	80.23	70.80	79.30	60.49	85.58	65.76
TNet-LF	72.98	71.43	74.61	70.14	80.42	71.03	78.47	59.47	89.07	70.43
TD-GAT	72.20	70.45	75.63	70.74	81.32	71.72	80.38	60.50	87.71	67.87
ASGCN	72.15	70.40	75.55	71.05	80.77	72.02	79.89	61.89	88.99	67.48
kumaGCN	72.45	70.07	76.12	72.42	81.43	73.64	80.69	65.99	89.39	**73.19**
BiGCN	74.16	**73.35**	74.59	71.84	81.97	73.48	**81.16**	64.79	88.96	70.84
AFGCN	**74.69**	73.23	**77.43**	**73.64**	**82.50**	**73.66**	79.89	**66.29**	**89.61**	72.02

4.2 Models for Comparison

A comprehensive comparison is carried out between our proposed model (AFGCN) against several state-of-the-art baseline models, as listed below:

- SVM (Kiritchenko et al.) [2] is the model which has won SemEval 2014 task 4 with conventional feature extraction methods.
- ATAE-LSTM (Wang et al.) [8] is a classic LSTM based model which explores the relationship between aspect and the content of an attention-based LSTM sentence.
- Mem-Net: (Tang et al.) [11] utilizes multi-hops attention to the context words used for sentence representation to illustrate the importance of each context word.

- RAM (Chen et al.) [7] uses multi-hops of attention layers and combines the outputs with a RNN for sentence representation.
- TNet-LF (Li et al.) [10] puts forward Context-Preserving Transformation (CPT) to preserve and strengthen the informative part of contexts.
- TD-GAT (Huang et al.) [9] proposes a graph attention network to explicitly utilize the dependency relationship among words.
- ASGCN (Zhang et al.) [13] employs a GCN over the dependency tree to exploit syntactical information and word dependencies.
- BiGCN (Zhang et al.) [23] convolutes over hierarchical syntactic and lexical graphs and build a concept hierarchy on both the syntactic and lexical graphs.
- kumaGCN (Chen et al.) [16] propose gating mechanisms to dynamically combine information from word dependency graphs and latent graphs.

Among the baselines, the first five methods are classic models with typical neural structures. The bottom five methods are graph-based and syntax-integrated ones.

We reproduce the results for baselines if the authors provide the source code. For the methods (TD-GAT) with no released code, we implement them by ourselves using the optimal hyperparameters settings reported in their papers. In our experiments, since we report the results over three runs with the random initialization, we stop training when the F1 score does not increase for a certain number (5) of rounds at one run.

4.3 Performance Comparison

The comparison results for all methods are shown in Table 2. From these results, we make the following observations.

Our proposed model AFGCN shows significant improvements on the five datasets. Table 2. shows the performance comparisons. Our method outperforms SVM by 2.34 and 6.94 Acc. score on Rest14 and Lap14, respectively. This indicates that our neural approach extracts more practical features than hardcoded feature engineering. Our model achieves the best performance on three datasets (Lap14, Rest14 and, Rest15) and is only 0.12% lower than the F1. score of the best-performing model on the Twitter dataset. However, our model performed poorly on the Rest16 dataset, with a 1.17% difference in F1. score from the best-performing kumaGCN. We speculate that the reason for its poor performance may be caused by the different distribution of positive, neutral and negative sentiment between the train set and the test set, as shown in Table 1.

The methods based on the combination of graph and syntax (TD-GAT, ASGCN, kumaGCN, and BiGCN) are significantly better than the first five methods without considering syntax, indicating that the dependency relationship is beneficial to the recognition of sentiment polarity, which is consistent with previous studies. However, they are worse than the AFGCN model we proposed because our proposed model fully integrates topology structure and dependent distance. The result proves that our AFGCN model, which combines dependency tree and dependency distance, is helpful to improve performance.

The large performance gaps between our model and baseline models confirm the effectiveness of our proposed architecture. We believe that using context and dependency information from the sentence, we can encode aspect vectors through proximity-weight convolution and GCN layers. Proximity-weight convolution and GCN layers can be considered messaging networks that propagate information along word sequence chains or syntactic dependency paths. Since relevant information is transmitted to aspect, we only need a simple Attention mechanism to encode the weighted information in significant words, thus preserving information relevant to the categorization task.

4.4 Ablation Study

We conduct an ablation study further to analyze the impact of different components of AFGCN. The results are shown in Table 3.

Table 3. Ablation study results (%). Acc. represents accuracy, F1 represents Macro-F1 score.

Model	Twitter		Lap14		Rest14		Rest15		Rest16	
	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1
AFGCN w/o P-w	72.98	71.46	**77.58**	**73.70**	80.71	72.01	79.73	63.35	89.11	69.18
AFGCN w/o GCN	73.12	70.89	75.70	71.96	81.33	73.33	78.22	60.67	88.96	71.26
AFGCN w/o Att	73.84	72.19	76.95	72.84	80.62	72.38	79.33	63.28	88.79	**73.24**
AFGCN w/o tree	73.98	72.28	76.95	73.32	81.25	72.04	80.07	65.08	88.96	71.19
AFGCN w/o graph	73.55	71.98	75.70	71.82	82.31	73.23	79.52	64.35	88.14	71.15
AFGCN	**74.69**	**73.23**	77.43	73.64	**82.50**	**73.56**	**79.89**	**66.29**	**89.61**	72.02

First, removal of proximity-weight convolution (i.e., AFGCN w/o P-w.) degrades the performance of four datasets but improves the performance for about 0.1% of Lap14 datasets. We argue that if the syntax is not essential to the data, then the integration of adjacent weights does not help reduce the noise of user-generated content.

Second, the removal of GCN layers is generally an evident performance degradation. Thus, it can be seen that GCN layers promote the development of AFGCN to a great extent because GCN captures both syntactic lexical dependencies and long-range lexical relationships. We can also observe that removing "aspect fusion Attention mechanism" (i.e., AFGCN w/o Att.) slightly degrades performance, indicating that our Attention mechanism helps integrate significant features related to aspect semantics in sentences and is an integral part of AFGCN.

Then we investigate the impacts of dependency tree (i.e., AFGCN w/o tree.) and dependency-position graph (i.e., AFGCN w/o graph.) Compared with the complete AFGCN, the performance of both is degraded, indicating that the effect of one graph (tree) is not as good as that of two fused graphs. We also found that the two compete on Rest datasets, with each having their contribution from a lexical and syntactic perspective.

4.5 Impact of GCN Layers

We investigate the effect of the number of layers on the performance of our proposed AFGCN. We vary the layer number from 1 to 8 and report the results in Fig. 2. It can be seen that our model achieves the best results with two layers, and thus we set the number of GCN layers as 2 in our experiments. Using only one layer of AFGCN is not enough to obtain specific syntactic dependencies of the context on aspect. However, the performance does not constantly get improved with the increasing number of layers. The performance of AFGCN fluctuates with the increase of the number of GCN layers and basically decreases when the model depth is greater than 2. Analysis implies that a larger model introduces more parameters, resulting in a less generic model and challenging to train.

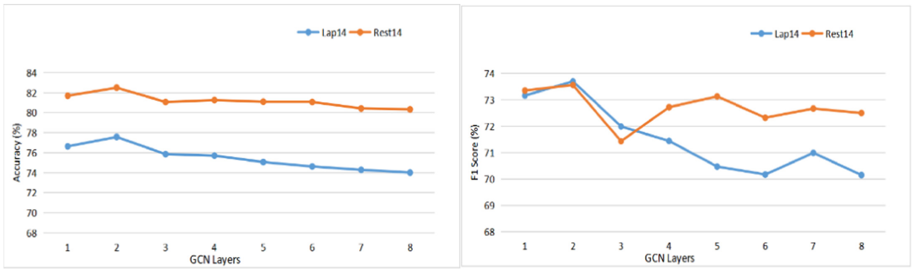

Fig. 2. Impacts of GCN layers.

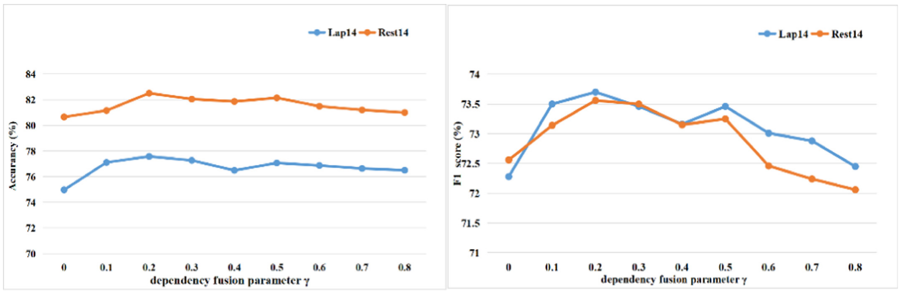

Fig. 3. Impacts of the dependency fusion parameter γ.

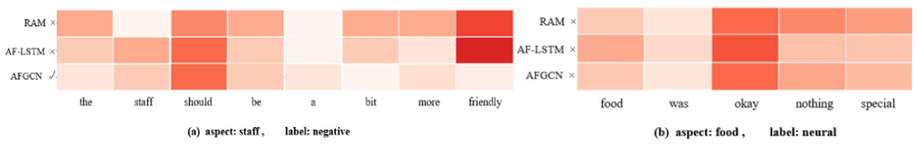

Fig. 4. Visualization results for RAM, AF-LSTM and AFGCN, where $\surd$ and $\times$ denotes the correct and wrong prediction, respectively.

4.6 Impact of the Dependency Fusion Parameter γ

To investigate how the trade-off between using dependency-position graph and dependency tree affects AFGCN performance, we use a step size of 0.1 to vary γ from 0 to 0.8.

Figure 3 shows the F1. scores obtained by AFGCN on Lap14 and Rest14 with different γ. When $\gamma = 0$, the model degenerates to GCN of fused dependency-position graph only. It can be observed that the performance significantly improves with the increase of γ value from 0 to 0.2, indicating that the fusion of graph and tree is beneficial to focusing the aspect related features. the curve reaches its maximum value when $\gamma = 0.2$, indicating that the dependency-position graph and dependency tree structure in this model are complementary. When γ is greater than 0.2, the curve shows a fluctuating downward trend. Thus, we set $\gamma = 0.2$.

4.7 Case Study and Error Analysis

To gain more insights into our model's behavior, we show two case studies in Fig. 4. We visualize the attention scores, the predicted and the ground truth labels for these examples.

As can be seen from Fig. 4(a), the aspect for the given example is "staff" with negative sentiment, and only our model predicts correct sentiment. This example uses the subjunctive word "should", which makes it extra difficult to detect grammar. Due to the lack of syntax information, RAM and AF-LSTM cannot make the right decision for the two examples. Both models assign the highest weight to the word "friendly", which is an irrelevant sentiment word to this target, leading to an incorrect prediction. In contrast, our model assigns the largest weight to the sentiment keyword "should" and correctly predict the negative polarity of the aspect "staff" in the first sentence.

Figure 4(b) shows the examples for error analysis. RAM gives relatively high attention weight to the words "nothing" and "special", but it still predicts the wrong sentiment polarity. Although AF-LSTM calculates the relationship between the context and the aspect, the short distance between "food" and "okay" causes the LSTM to assign the most significant attention scores to "okay". On the other hand, since "good" and "food" are closely related in the dependency tree, the solid positive polarity of "good" also prejudices the AFGCN decision. This type of error frequently appears in neutral cases. If negative expression (e.g., "nothing", "shouldn't") is related to aspect, the neural model does not differentiate well.

5 Conclusion

Previous methods for aspect-based sentiment classification depended on the syntactic relationship between aspect and context often ignore the dependency distance relationship between context. In this paper, we have built a framework that leverages graph-based approaches and syntactic dependencies between contextual terms and aspect to construct an applicable model. In addition to dependency tree, we built a dependency-position graph to enhance the syntactic dependencies of each instance. And we propose an aspect fusion graph convolutional network model to fully combine the topology structure and dependency distance of dependency tree. Finally, we design the aspect fusion attention module to fully integrate the significant features related to aspect semantics in the hidden state vectors of the convolution layer and the masking layer. Experimental results demonstrate the effectiveness of our proposed model and suggest that dependency distance and syntactic dependency are more beneficial to aspect-based sentiment classification.

Acknowledgment. This work is supported by grant from the Natural Science Foundation of China (No. 61976124) and Social and Science Foundation of Liaoning Province (No. L20BTQ008).

References

1. Jiang, L., Yu, M., Zhou, M., Liu, X., Zhao, T.: Target-dependent twitter sentiment classification. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pp. 151–160 (2011)
2. Kiritchenko, S., Zhu, X., Cherry, C., Mohammad, S.: NRC-Canada-2014: detecting aspects and sentiment in customer reviews. In: Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), pp. 437–442 (2014)
3. Dong, L., Wei, F., Tan, C., Tang, D., Zhou, M., Xu, K.: 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), pp. 49–54 (2014)
4. Nguyen, T.H., Shirai, K.: Phrase RNN: phrase recursive neural network for aspect-based sentiment analysis. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 2509–2514 (2015)
5. Wang, W., Pan, S.J., Dahlmeier, D., Xiao, X.: Recursive neural conditional random fields for aspect-based sentiment analysis. arXiv preprint arXiv:1603.06679 (2016)
6. Vo, D.T., Zhang, Y.: Target-dependent twitter sentiment classification with rich automatic features. In: Twenty-Fourth International Joint Conference on Artificial Intelligence (2015)
7. Chen, P., Sun, Z., Bing, L., Yang, W.: Recurrent attention network on memory for aspect sentiment analysis. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 452–461 (2017)
8. Wang, Y., Huang, M., Zhu, X., Zhao, L.: Attention-based LSTM for aspect-level sentiment classification. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 606–615 (2016)
9. Huang, B., Carley, K.M.: Parameterized convolutional neural networks for aspect level sentiment classification. arXiv preprint arXiv:1909.06276 (2019)
10. Xin, L., Bing, L., Lam, W., Bei, S.: Transformation networks for target-oriented sentiment classification. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (2018)
11. Tang, D., Qin, B., Feng, X., Liu, T.: Effective LSTMs for target-dependent sentiment classification. arXiv preprint arXiv:1512.01100 (2015)
12. Xue, W., Li, T.: Aspect based sentiment analysis with gated convolutional networks. arXiv preprint arXiv:1805.07043 (2018)
13. He, R., Lee, W.S., Ng, H.T., Dahlmeier, D.: Effective attention modeling for aspect-level sentiment classification. In: Proceedings of the 27th International Conference on Computational Linguistics, pp. 1121–1131 (2018)
14. Zhang, C., Li, Q., Song, D.: Aspect-based sentiment classification with aspect-specific graph convolutional networks. arXiv preprint arXiv:1909.03477 (2019)
15. Sun, K., Zhang, R., Mensah, S., Mao, Y., Liu, X.: 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), pp. 5683–5692 (2019)

16. Liang, B., Du, J., Xu, R., Li, B., Huang, H.: Context-aware embedding for targeted aspect-based sentiment analysis. arXiv preprint arXiv:1906.06945 (2019)
17. Chen, C., Teng, Z., Zhang, Y.: Inducing target-specific latent structures for aspect sentiment classification. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 5596–5607 (2020)
18. Fan, C., Gao, Q., Du, J., Gui, L., Xu, R., Wong, K.: Convolution-based memory network for aspect-based sentiment analysis. The 41st International ACM SIGIR Conference on Research Development in Information Retrieval (2018)
19. Ma, D., Li, S., Zhang, X., Wang, H.: Interactive attention networks for aspect-level sentiment classification. arXiv preprint arXiv:1709.00893 (2017)
20. Majumder, N., Poria, S., Gelbukh, A., Akhtar, M.S., Cambria, E., Ekbal, A.: IARM: inter-aspect relation modeling with memory networks in aspect-based sentiment analysis. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 3402–3411 (2018)
21. Tai, K.S., Socher, R., Manning, C.D.: Improved semantic representations from tree-structured long short-term memory networks. arXiv preprint arXiv:1503.00075 (2015)
22. Liang, B., Yin, R., Gui, L., Du, J., Xu, R.: Jointly learning aspect-focused and inter-aspect relations with graph convolutional networks for aspect sentiment analysis. In: Proceedings of the 28th International Conference on Computational Linguistics, pp. 150–161 (2020)
23. Zhang, M., Qian, T.: Convolution over hierarchical syntactic and lexical graphs for aspect level sentiment analysis. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 3540–3549 (2020)
24. Xu, L., Bing, L., Lu, W., Huang, F.: Aspect sentiment classification with aspect-specific opinion spans. arXiv preprint arXiv:2010.02696 (2020)
25. Zhao, L., Akoglu, L.: PairNorm: tackling oversmoothing in GNNs. arXiv preprint arXiv:1909.12223 (2019)
26. Pontiki, M., Galanis, D., Pavlopoulos, J., Papageorgiou, H., Androutsopoulos, I., Manandhar, S.: Semeval-2014 task 4: aspect based sentiment analysis. In: COLING 2014 (2014)
27. Pontiki, M., Galanis, D., Papageorgiou, H., Manandhar, S., Androutsopoulos, I.: Semeval-2015 task 12: aspect based sentiment analysis. In: Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval2015), pp. 486–495 (2015)
28. Pennington, J., Socher, R., Manning, C.D.: GloVe: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543 (2014)