Convolution over Hierarchical Syntactic and Lexical Graphs for Aspect Level Sentiment Analysis

Mi Zhang, Tieyun Qian*
School of Computer Science, Wuhan University, China
{mizhanggd, qty}@whu.edu.cn

Abstract

The state-of-the-art methods in aspect-level sentiment classification have leveraged the graph based models to incorporate the syntactic structure of a sentence. While being effective, these methods *ignore the corpus level word co-occurrence information*, which reflect the collocations in linguistics like "nothing special". Moreover, they *do not distinguish the different types of syntactic dependency*, e.g., a nominal subject relation "food-was" is treated equally as an adjectival complement relation "was-okay" in "food was okay".

To tackle the above two limitations, we propose a novel architecture which convolutes over hierarchical syntactic and lexical graphs. Specifically, we *employ a global lexical graph* to encode the corpus level word co-occurrence information. Moreover, we *build a concept hierarchy* on both the syntactic and lexical graphs for differentiating various types of dependency relations or lexical word pairs. Finally, we *design a bi-level interactive graph convolution network* to fully exploit these two graphs. Extensive experiments on five benchmark datasets show that our method achieves the state-of-the-art performance.

1 Introduction

Aspect-level sentiment classification (ASC) (Hu and Liu, 2004) aims to determine the sentiment polarity (i.e., positive, negative, neutral) of the aspect(s) in a sentence. Take the review "*great food but the service was dreadful*" as an example. Given two aspect terms "food" and "service", the goal is to infer the sentiment polarities for the aspect terms: positive for food and negative for service. ASC can provide fine-grained analysis of the users' opinion towards the specific aspect and is fundamental to many natural language processing tasks. Consequently, it has aroused much research attention in recent years.

Early studies on ASC (Mohammad et al., 2013; Jiang et al., 2011) mostly use machine learning algorithms to build sentiment classifier. Later, various neural network models (Dong et al., 2014; Vo and Zhang, 2015; Chen et al., 2017) are proposed for this task, including long short-term memory (LSTM) based (Wang et al., 2016), convolutional neural networks (CNN) based (Huang and Carley, 2018; Li et al., 2018), and memory based (Tang et al., 2016b) or hybrid methods (Xue and Li, 2018). These models represent the sentence as a word sequence and neglect the syntactic relations between words, and thus it is hard for them to find the opinion words which are far away from the aspect term. To solve this problem, several recent researches (Zhang et al., 2019; Huang and Carley, 2019; Sun et al., 2019) leverage the graph based models to incorporate the syntactic structure of a sentence, and have shown better performance than those without considering syntactic relations.

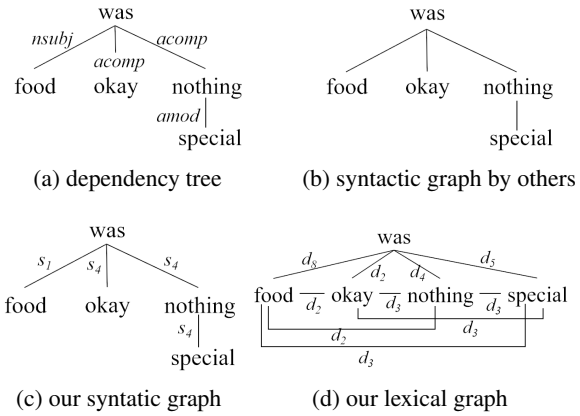
Despite of their effectiveness, the seminal syntax based methods ignore the corpus level word co-occurrence information. Moreover, they do not distinguish the different types of syntactic dependency. We argue that both will incur information loss. (1) The frequently co-occurred words represent the collocations in linguistics. For example, in the sentence "*food was okay, nothing special*", the word pair "nothing special" occurs five times in the SemEval training set, denoting a negative polarity. Without such global information to counteract the positive polarity of "okay", syntax based methods will make wrong prediction on "food". (2) Each type of syntactic dependency denotes a specific relation. For example, in "*i like hamburgers*", "i-like" is a nsubj relation, and "like-hamburgers" is a dobj relation. If the nsubj relation and dobjs relation are

*Corresponding author.

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 3540–3549,
November 16–20, 2020. ©2020 Association for Computational Linguistics

treated equally, we are unable to differentiate the subject and the object of the action "like".

To tackle the above limitations, we propose a novel architecture which convolutes over hierarchical syntactic and lexical graphs. We first *employ a global lexical graph* to encode the corpus level word co-occurrence information, where nodes are words and the edge denotes the frequency between two word nodes in the training corpus. We then *build a concept hierarchy* on each of the syntactic and lexical graphs to distinguish different types of dependency relations or word co-occurrence relations. For example, the acomp relation "was-nothing" and the amod relation "nothing-special" are grouped into an adjective relation type, while the nsubj relation "food-was" will form another noun relation type. For illustration, we show in Figure 1 a sample sentence with its dependency tree and the corresponding lexical and syntactic graphs in our and other works (Zhang et al., 2019; Huang and Carley, 2019; Sun et al., 2019).



(a) dependency tree (b) syntactic graph by others

(c) our syntatic graph (d) our lexical graph

Figure 1: A sample of depedency tree and different graphs in our and other papers

It is clear from Fig.1 (b) that existing syntax integrated methods do not differentiate various types of dependency relations, as an edge simply represents that there is a relation between two nodes. In contrast, each edge in our syntactic graph (Fig.1 (c)) is attached with a label denoting the relation type. In addition, we construct a lexical graph (Fig.1 (d)) which also has a concept hierarchy to capture the various word co-occurrence relations. Finally, in order to let the syntactic and lexical graphs cooperate with each other, we design a bi-level interactive graph convolution network to fully exploit these two graphs.

We conduct extensive experiments on five SemEval datasets. Results demonstrate that our model achieves the state-of-the-art performance.

2 Related Work

Recent advances in aspect-level sentiment classification (ASC) focus on developing various types of deep learning models. We briefly review the neural models without considering syntax, and then go to the syntax based ones.

The neural models without considering syntax models can be mainly categorized into several types: LSTM based (Tang et al., 2016a; Wang et al., 2016; Ma et al., 2017), CNN based (Huang and Carley, 2018; Li et al., 2018), memory based (Tang et al., 2016b; Chen et al., 2017), and other hybrid methods (Weston et al., 2015; Xue and Li, 2018). For example, Zhang et al. (2016) use the gated neural network structures to model the interaction between the surrounding contexts and the target. Li et al. (2018) employ a CNN instead of attention to extract important features from the transformed word representations. Xue and Li (2018) combine the CNN and gating structure to extract aspect-specific information from contexts.

The syntactic information enables dependency information to be preserved in lengthy sentences, and helps shorten the distance between aspect and opinion words. There has long been research on incorporating syntactic information in document-level sentiment classification (Matsumoto et al., 2005; Ng et al., 2006; Nakagawa et al., 2010). Later, Dong et al. (2014); Nguyen and Shirai (2015); He et al. (2018); Salwa et al. (2018) also take the syntax structure of a sentence and or POS tags into account for aspect based sentiment analysis. Nevertheless, the effect of syntactical structure has not been fully exploited without the proper utilization of the dependencies along the syntactic paths.

More recently, several studies (Sun et al., 2019; Huang and Carley, 2019; Zhang et al., 2019) employ graph based models to integrate the syntactic structure. The basic idea is to transform the dependency tree into a graph, and then impose the graph convolutional networks (GCN) or graph attention networks (GAT) to propagate information from syntax neighbourhood opinion words to aspect words. There are also attempts (Tay et al., 2018; Yao et al., 2019) at exploiting the word co-occurrence information for sentiment analysis.

Unlike all the aforementioned methods, our model exploits both the syntactic and lexical graphs for capturing the dependency relations in a sentence and the word co-occurrence relations in the train-

ing corpus. Moreover, we construct the concept hierarchy for each graph, which can group relations with similar uses or meanings together and reduce noises. As we will show in our experiments, the introduction of hierarchy greatly boosts the performance.

3 Preliminary

Problem definition (ASC) Given a review sentence $S = [w_1, ..., w_{a+1}, ..., w_{a+m}, ..., w_n]$ consisting of *n* words and a corresponding *m*-length aspect starting from the $(a+1)^{th}$ position, the aspect-level sentiment classification task ASC aims at identifying the sentiment polarity of the given aspect(s) in a sentence.

Hierarchical syntactic graph construction A syntactic graph (SG) has a node set V_s and an edge set E_s. Each node v in V_s is a word in the sentence and each edge e in E_s between two words denotes that they are syntactically related.

Existing syntax integrated methods for ASC (Sun et al., 2019; Huang and Carley, 2019; Zhang et al., 2019) do not utilize various types of dependency relations and an edge in their syntactic graph as shown in Fig.1 (b) simply denotes there exists a dependency relation between two words. As we pointed out in the introduction, each dependency relation represents a specific grammatical function that a word plays in a sentence and should be used in its own manner. However, since there are a good number of relations in the parsed tree, directly using one dependency relation as a type of edge in the graph may incur noises like a parsing error.

To solve this problem, we add a *syntactic concept hierarchy* R_s over the dependency relations. Specifically, we group 36 dependency relations into 5 relation types, including "noun", "verb", "adverb", "adjective", and "others", denoted as s_1 .. s_5 in R_s, respectively.

In particular, since most aspect and opinion words are noun and adjective, respectively, they become two main types. Verb expresses an action, an event, or a state, and adverb modifies verbs and adjectives, thus they also become two types. All the remaining constitutes the "others" type.

We then construct a *hierarchical syntactic graph* HSG based on the syntactic concept hierarchy. Specifically, HSG is denoted as $\{V_s, E_s, R_s\}$, where V_s, E_s, and R_s is a node set, an edge set, and a syntactic relation type set, respectively. Note that each edge in E_s is now attached with a label denoting the dependency relation type in R_s.

Hierarchical lexical graph construction *A global lexical graph* LG^T has a node set V^T and a edge set E^T. Each node v in V^T represents a word and each edge e in E^T denotes the co-occurrence frequency between two words in the training corpus whose vocabulary size is *N*.

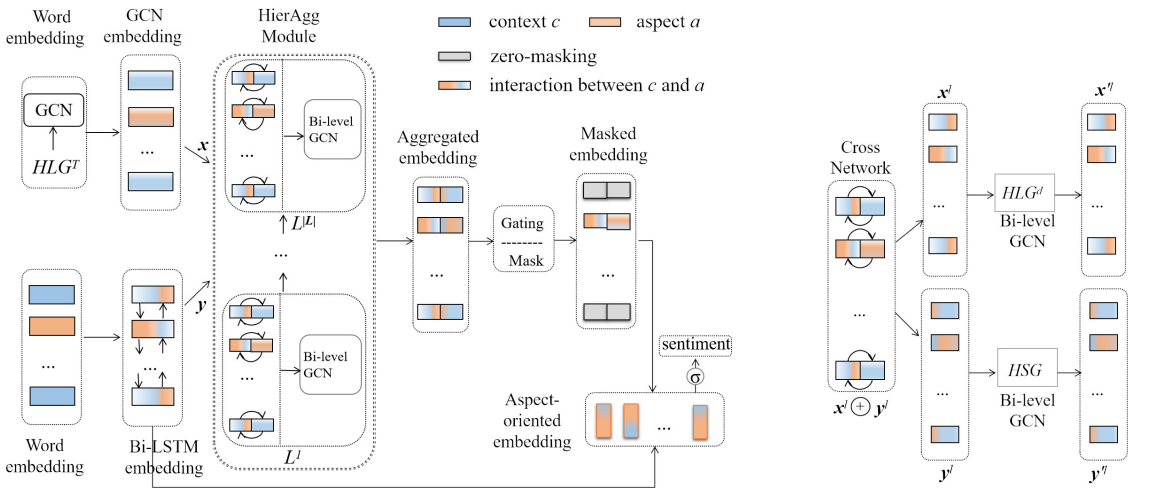
We then construct *a local lexical graph* LG^d for each sentence, where each node represents a word in the sentence and each edge denotes two words co-occur in the sentence. However, the edge is attached a same weight as that of the edge between two same words in LG^T. The rationale is to transfer the global word distribution information in LG^T into the local lexical graph LG^d.

The frequency of word co-occurrence in the corpus is highly skewed, where most word pairs occur one or two times, and a few of them have a large frequency. Clearly, the frequent word pairs should be treated differently from the rare ones. Hence we add a *lexical concept hierarchy* R_d over the word co-occurrence relations. To this end, we group the frequency of word pairs according to the log-normal distribution (Bhagat et al., 2018). Specifically, we use d_1 and d_2 to denote the word pair relation with the frequency of 2^0 and 2^1, and d_3, ..., d_7 to denote the word pair relation whose frequency falls in the interval of [2^k+1, 2^{k+1}] ($1 \leq k \leq 5$). The last one d_8 denotes the lexical relation for all the word pairs whose frequency is larger than 2^6.

Finally we can construct a *hierarchical global lexical graph* HLG^T based on the lexical concept hierarchy, denoted as $\{V_d^T, E_d^T, R_d\}$, where V_d^T, E_d^T, and R_d is a node set, an edge set, and a lexical relation type set, respectively. Similarly, we have a *hierarchical local lexical graph* HLG^d = $\{V_d^d, E_d^d, R_d\}$, where V_d^d is identical to V_s.

4 Proposed Model

In this section, we present our proposed BiGCN model. We first show its architecture in Figure 2. As can be seen from Fig. 2 (a), BiGCN takes the global lexical graph and the word sequence as the input to get the initial sentence representation. It then introduces a HiarAgg module where the local lexical graph and syntactic graph interact with each other to refine the sentence representation. Finally, BiGCN obtains the aspect-oriented representation via the mask and gating mechanism for better predicting the sentiment polarity towards a specific



(a) Overall Framework

(b) The l^{th} layer in HierAgg module

Figure 2: Architecture of BiGCN model.

aspect in the sentence.

4.1 Getting Initial Sentence Representation

Let $\mathbf{E}_w \in \mathbb{R}^{|Vo| \times da}$ be the pre-trained word embedding, where $|Vo|$ is the vocabulary size and da is the dimension of word embedding. $\mathbf{E}_w$ is used to map the review sequence S with n words to word vectors $[\boldsymbol{e_1}, ..., \boldsymbol{e_{a+1}}, ..., \boldsymbol{e_{a+m}}, ..., \boldsymbol{e_n}] \in \mathbb{R}^{n \times da}$. We then propose two types of text representations for improving the sentence embedding. One is the GCN embedding based on our global lexical graph. The other is the Bi-LSTM embedding based on the bi-directional LSTM.

GCN Embedding Firstly, we wish to encode the corpus-specific lexical information into the review representation. For this target, we first build an embedding matrix $\mathbf{E}_{wt} \in \mathbb{R}^{N \times da}$ as the feature matrix for training corpus, where N is the vocabulary size for the training corpus. We then perform a standard GCN (Kipf and Welling, 2017) on the hierarchical global lexical graph HLG^T, and get a new embedding matrix $\mathbf{E}_{gcn} \in \mathbb{R}^{N \times dx}$. $\mathbf{E}_{gcn}$ is then used to form the GCN embedding of the review sequence S, i.e., $[\boldsymbol{x_1}, ..., \boldsymbol{x_{a+1}}, ..., \boldsymbol{x_{a+m}}, ..., \boldsymbol{x_n}] \in \mathbb{R}^{n \times dx}$ via a look-up table, denoted as $\boldsymbol{x}$ in Figure 2 (a).

Bi-LSTM Embedding Secondly, we encode the sequential information into the review representation following most of previous studies (Wang et al., 2016; Sun et al., 2019; Zhang et al., 2019). In addition, since the token closer to aspect may contribute more in judging the sentiment of the aspect (Gu et al., 2018), we calculate the absolute distance from each context word w_t to the corresponding aspect, and get a position sequence for S. Let $\mathbf{E}_p \in \mathbb{R}^{n \times dp}$ be the position embedding lookup table with random initialization, the position lookup layer maps the position sequence to a list of position embedding $[\boldsymbol{p_1}, ..., \boldsymbol{p_{a+1}}, ..., \boldsymbol{p_{a+m}}, ..., \boldsymbol{p_n}]$.

For each word w_t in S, its embedding is calculated as $\boldsymbol{e_t^p} = \boldsymbol{e_t} \oplus \boldsymbol{p_t} \in \mathbb{R}^{da+dp}$, where $\oplus$ denotes concatenation, $\boldsymbol{e_t}$ and $\boldsymbol{p_t}$ is pre-trained word embedding and the position embedding of the t^{th} word in S. The sentence S with the above representation is sent to a Bi-LSTM layer (Wang et al., 2016; Zhang et al., 2019). We omit the detail due to the space limitation. S is then transformed into a Bi-LSTM embedding $[\boldsymbol{y_1}, ..., \boldsymbol{y_{a+1}}, ..., \boldsymbol{y_{a+m}}, ..., \boldsymbol{y_n}] \in \mathbb{R}^{n \times dy}$, denoted as $\boldsymbol{y}$ in Figure 2 (a).

4.2 Refining Sentence Representation

With the the GCN embedding $\boldsymbol{x}$ and the Bi-LSTM embedding $\boldsymbol{y}$ as the initial sentence representation, we further leverage the local lexical graph and syntactic graph to get better representation for the sentence S. The basic idea is to let these two graphs interact with each other in a carefully designed HierAgg module. Briefly, HierAgg is a multi-layer structure, where each layer includes a cross network to fuse GCN and Bi-LSTM embedding and a Bi-level GCN to convolute on hierarchical syntactic and lexical graphs. The multi-layer structure ensures the collaboration of different types of information to be performed at different levels. This section gives the detail for one layer in HierAgg, as shown in Fig. 2 (b).

Cross Network To deeply fuse the GCN embedding $\boldsymbol{x}$ and Bi-LSTM embedding $\boldsymbol{y}$, we adopt the cross network structure (Wang et al., 2017), which is simple yet effective. In particular, we first

concatenate $\boldsymbol{x}$ and $\boldsymbol{y}$ to form a fixed combination $\boldsymbol{f^0} \in \mathbb{R}^{dh}$, i.e., $\boldsymbol{f^0} = \boldsymbol{x} \oplus \boldsymbol{y}$. Then in each layer of the cross network, we use the following formula to update the fused embedding.

$$\boldsymbol{f^l} = \boldsymbol{f^0} \boldsymbol{f^{l-1}}^\top \boldsymbol{w^l} + \boldsymbol{b^l} + \boldsymbol{f^{l-1}}, \quad (1)$$

where l denotes the layer number (l=1,2,...,$|L|$), and $\boldsymbol{w^l}, \boldsymbol{b^l} \in \mathbb{R}^{dh}$ are the weight and bias parameters. The fused embedding $\boldsymbol{f}^l$ in the l^{th} layer is then detached into $\boldsymbol{x}^l$ and $\boldsymbol{y}^l$ from the original concatenation position, which will serve as the input node representation for two graphs in Bi-level GCN.

Bi-level GCN Since our syntactic and lexical graphs contain a concept hierarchy, a vanilla GCN cannot convolute over the graph with a labelled edge. To address this problem, we propose a bi-level GCN for aggregating different relation types. Given a sentence with its two graphs, we will perform a bi-level convolution using two aggregating operations.

The first aggregation (low-level): it aggregates the nodes with the same relation type to a virtual node, and then uses the same normalized hidden feature sum in the vanilla GCN (Kipf and Welling, 2017) as the aggregation function to obtain the embedding for the virtual node. Hence each relation type r has a representation $\tilde{\boldsymbol{h}}_{\boldsymbol{t}}^{\boldsymbol{l},\boldsymbol{r}}$, where l is the layer number and t is the target node for aggregation. For example, in Fig. 1 (c), "okay" and "nothing" have the same label and thus are aggregated into a virtual node "s_4" for the target node "was". Similarly, "food" itself is aggregated into a virtual node "s_1" for "was".

The second aggregation (high-level): it aggregates all virtual nodes together with their specific relation. The representation of the target word t is updated using the mean aggregation function over different relation types (virtual nodes):

$$\boldsymbol{h}_{\boldsymbol{t}}^{\boldsymbol{l}} = ReLU(\mathbf{W}_l \cdot (\oplus_r \tilde{\boldsymbol{h}}_{\boldsymbol{t}}^{\boldsymbol{l},\boldsymbol{r}})), \quad (2)$$

where $\oplus_r$ denotes the concatenation of the representations of different relation types, and $\mathbf{W}_l$ is the weight matrix in the l^{th}-layer.

We then get the refined sentence representation $\boldsymbol{x}'^l = [\boldsymbol{h}_{\boldsymbol{1}}^{\boldsymbol{l},\boldsymbol{d}}, ..., \boldsymbol{h}_{\boldsymbol{a+1}}^{\boldsymbol{l},\boldsymbol{d}}, ..., \boldsymbol{h}_{\boldsymbol{a+m}}^{\boldsymbol{l},\boldsymbol{d}}, ..., \boldsymbol{h}_{\boldsymbol{n}}^{\boldsymbol{l},\boldsymbol{d}}]$ and $\boldsymbol{y}'^l = [\boldsymbol{h}_{\boldsymbol{1}}^{\boldsymbol{l},\boldsymbol{s}}, ..., \boldsymbol{h}_{\boldsymbol{a+1}}^{\boldsymbol{l},\boldsymbol{s}}, ..., \boldsymbol{h}_{\boldsymbol{a+m}}^{\boldsymbol{l},\boldsymbol{s}}, ..., \boldsymbol{h}_{\boldsymbol{n}}^{\boldsymbol{l},\boldsymbol{s}}]$ after the first and second aggregations on lexical and syntactic graph, respectively, which will be used as the input of the next layer. Note that in the last layer in Hier-Agg module, we combine $\boldsymbol{x}'^L$ and $\boldsymbol{y}'^L$ to form an aggregated embedding $\boldsymbol{h^L} = \boldsymbol{x}'^L \oplus \boldsymbol{y}'^L$.

4.3 Generating Aspect-oriented Representation

For better predicting the sentiment polarity of an aspect, we propose to use a gating mechanism (Dauphin et al., 2017) to control the flow of sentiment information towards the given aspect:

$$\alpha_t = tanh(\boldsymbol{h^L} + \boldsymbol{h_a^l} \boldsymbol{W}_{g\alpha} + \boldsymbol{b}_{g\alpha}), \boldsymbol{h}'^L = \boldsymbol{h^L} * \alpha_t, \quad (3)$$

where $\boldsymbol{h_a^l}$ is the aspect in $\boldsymbol{h^L}$, $\boldsymbol{W}_{g\alpha}$, $\boldsymbol{b}_{g\alpha}$ are weights and bias, respectively, and $*$ is the element-wise product. We then mask non-aspect words and keep aspect words unchanged in the gated embedding $\boldsymbol{h}'^L$, and we get a zero-masked embedding $[0,...,\boldsymbol{h}_{\boldsymbol{a+1}}^{\boldsymbol{l'}},...,\boldsymbol{h}_{\boldsymbol{a+m}}^{\boldsymbol{l'}},...,0] \in \mathbb{R}^{dh}$.

Finally, we retrieve the important features that are semantically related to aspect words, and set the retrieval-based attention weights (Zhang et al., 2019) for each context word. The final representation $\boldsymbol{z}$ for the sentence is formulated as:

$$\theta_t = \sum_{i=1}^{n} \boldsymbol{y_t}^\top \boldsymbol{h_i^{l''}}, \quad \gamma_t = \frac{exp(\theta_t)}{\sum_{i=1}^{n} exp(\theta_i)}, \quad (4)$$

$$\boldsymbol{z} = \sum_{t=1}^{n} \gamma_t \boldsymbol{y_t}, \quad (5)$$

where $\boldsymbol{y_t} \in \mathbb{R}^{dy}$ is the Bi-LSTM embedding, $\boldsymbol{h_i^{l''}}$ is transformed from the zero-masked embedding $\boldsymbol{h_i^{l'}}$ via a fully connected layer to keep the same dimensionality as that of $\boldsymbol{y_t}$.

4.4 Model Training

After obtaining the aspect-oriented representation $\boldsymbol{z}$, we feed it into a fully connected layer and a softmax layer to project it into the prediction space:

$$\boldsymbol{u} = softmax(\boldsymbol{W_u z} + \boldsymbol{b_u}), \quad (6)$$

where $\boldsymbol{u}$ is a probability distribution of the prediction, $\boldsymbol{W_u}$ and $\boldsymbol{b_u}$ are the weight matrix and bias, respectively. Then the label of the highest probability is set as the final prediction $\hat{u}$.

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

$$\zeta = -\sum_{i}^{J} u_i log\hat{u}_i + \lambda \|\Theta\|, \quad (7)$$

where J is the number of training samples, u_i and $\hat{u}_i$ is the ground truth and predicted label for the i^{th} sample, Θ represents all trainable parameters, and λ is the co-efficient of L2-regularization.

5 Experiments

5.1 Datasets and Settings

Datasets We evaluated our proposed model on five benchmark datasets. One is the Twitter dataset constructed by Dong et al. (2014). It consists of twitter posts. The other four datasets (Lap14, Rest14, Rest15, Rest16) are all from SemEval (Pontiki et al., 2014, 2015, 2016) tasks, which contain reviews on laptop and restaurant. Following previous studies (Tang et al., 2016b; Zhang et al., 2019), we remove the samples with conflicting polarities and those without explicit aspects in the sentences. The statistics for five datasets are shown in Table 1.

Table 1: Dataset statistics

Dataset		#Pos.	#Neu.	#Neg.
Twitter	Train	1561	3127	1560
	Test	173	346	173
Lap14	Train	994	464	870
	Test	341	169	128
Rest14	Train	2164	637	807
	Test	728	196	196
Rest15	Train	912	36	256
	Test	326	34	182
Rest16	Train	1240	69	439
	Test	469	30	117

Settings We initialize word embeddings using the 300-dimension GloVe vectors provided by Pennington et al. (2014). This is a standard setting commonly used in Huang and Carley (2019); Zhang et al. (2019); Sun et al. (2019). Moreover, since we use the position information, we use the same dimensionality 30 as that in Sun et al. (2019) for the position embedding for a fair comparison. We use spaCy toolkit to get dependency relations.

We use Adam as the optimizer with a learning rate of 0.001. The coefficient λ of L2-regularization is 10^5 and batch size is 32. Moreover, the layer number in our BiGCN module is set to 2, and we will examine its impacts later. The experimental results are obtained by averaging three runs with random initialization, where Accuracy and Macro-F1 are used as the evaluation metrics[1].

Baselines We compare our model with the following eight baselines.

(1) ATAE-LSTM (Wang et al., 2016) is a classic LSTM based model which explores the connection between an aspect and the content of a sentence with an attention-based LSTM.

[1]Our code and data are available at https://github.com/NLPWM-WHU/BiGCN.

(2) GCAE (Xue and Li, 2018) is a CNN based model which has two convolutional layers and their outputs are combined by the gating units.

(3) MemNet (Tang et al., 2016b) is a memory based method combining a neural attention model with an external memory to calculate the importance of each context word towards an aspect.

(4) RAM (Chen et al., 2017) uses multi-hops of attention layers and combines the outputs with a RNN for sentence representation.

(5) AF-LSTM (Tay et al., 2018) is an aspect fusion LSTM model learning the associative relationships between sentence words and the aspect.

(6) TD-GAT (Huang and Carley, 2019) proposes a graph attention network to explicitly utilize the dependency relationship among words.

(7) ASGCN (Zhang et al., 2019) employs a GCN over the dependency tree to exploit syntactical information and word dependencies.

(8) CDT (Sun et al., 2019) uses a GCN to model the structure of a sentence through its dependency tree. It also utilizes position information.

Among the baselines, the first four methods are classic models with typical neural structures like attention, LSTM, CNN, memory, and RNN. The middle one (AF-LSTM) exploits the word co-occurrence information. The bottom three methods are graph based and syntax integrated ones. We do not take TextGCN (Yao et al., 2019) as a baseline since it is developed for text or document level sentiment classification.

We re-produce the results for baselines if the authors provide the source code. For three methods (TD-GAT, AF-LSTM, and GCAE) with no released code, we implement them by ourselves using the optimal hyper-parameters settings reported in their papers. Since we do not use validation sets, the results for TD-GAT are higher than those in Huang and Carley (2019). The results for CDT (Sun et al., 2019) are lower than those in the original paper. CDT reports the best results among a certain number (100) of rounds. 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. This stopping criterion is used for all methods for a fair comparison.

5.2 Results and Analysis

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

Table 2: Comparison results for all methods in terms of accuracy and F1 (%). The best results on each dataset are in bold. The second best ones are underlined.

Model	Twitter		Lap14		Rest14		Rest15		Rest16	
	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1
ATAE-LSTM	69.65	67.40	69.14	63.18	77.32	66.57	75.43	56.34	83.25	63.85
GCAE	71.64	69.88	69.90	65.71	78.57	68.06	77.85	59.63	86.29	65.87
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
AF-LSTM	69.21	68.24	69.97	63.49	77.46	65.18	76.12	56.29	85.61	66.15
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
CDT	73.29	72.02	**75.63**	**72.01**	**83.10**	73.01	79.42	61.68	86.24	67.62
BiGCN	**74.16**	**73.35**	74.59	71.84	81.97	**73.48**	**81.16**	**64.79**	88.96	**70.84**

ing observations.

(1) Our proposed BiGCN model achieves the best results in terms of macro-F1 scores on all datasets. In particular, it gets an improvements of 3.12, 2.77, and 1.36 F1 score over the second best one on Rest16, Rest15, and Twitter dataset, respectively. Its accuracy scores are also among the best ones, and are only slightly worse than the baselines on Lap14 and Rest14, where the difference is tiny, i.e., 0.15 and 0.06.

(2) The graph based and syntax integrated methods (TD-GAT, ASGCN, and CDT) are much better than the upper five methods without considering syntax, showing that the dependency relations are beneficial to identify the sentiment polarity. This is consistent with the previous studies (Huang and Carley, 2019; Zhang et al., 2019; Sun et al., 2019). However, they are worse than our proposed BiGCN model. This proves that the lexical graph in our BiGCN also helps improve the performance.

(3) The AF-LSTM method exploits the word co-occurrences between the aspect and contexts by calculating their circular correlation or circular convolution and then inputting them into an attention layer. However, its performance does not always show improvements over other classic methods. This infers that a direct integration of word association information via an attention layer is insufficient to exploit the lexical relations.

5.3 Ablation Study

To examine the influence of each component in our BiGCN model, we conduct an ablation study and show the results in Table 3.

We first investigate the impacts of hierarchical lexical (M1) and syntactic graph (M2). Compared with the complete BiGCN, the performance of M1 and M2 both decrease, showing that one single graph is not as good as two interactive graphs. We also find that M1 and M2 have competitive results, indicating that they have their own contributions from the point view of lexicon and syntax.

We then show the effects of concept hierarchy by further removing the relation types from M1 and M2, resulting a basic lexical (M3) and syntactic graph (M4). We can see that the results on these basic graphs without the concept hierarchy are both worse than their counterparts (M1-M3, M2-M4). This clearly reveals the positive influence of our proposed concept hierarchy.

5.4 Case Study

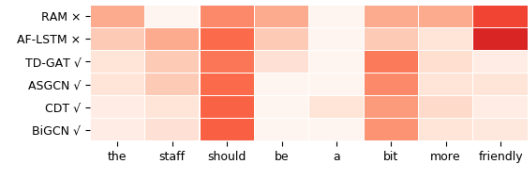
To better understand how our BiGCN works, we present the case study on three testing examples. We visualize the attention scores, the predicted and the ground truth labels for these example. Due to the space limitation, we only present the results for RAM, AF-LSTM, TD-GAT, ASGCN, CDT, and BiGCN in Figure 3, where RAM is the top-performed classic neural model. AF-LSTM leverages the word co-occurrence information. TD-GAT, ASGCN, and CDT are three graph based models considering syntax information.

RAM is unable to make correct decision for all three examples due to the lack of syntax information. For the same reason, AF-LSTM also makes wrong prediction in the first sentence either. As can be seen from Fig. 3 (a), RAM and AF-LSTM emphasize "friendly". Our model and three syntax integrated methods TD-GAT, ASGCN, and CDT can identify the dummy word "*should*" in the first sentence, and thus correctly predict the negative polarity for the aspect "*staff*".

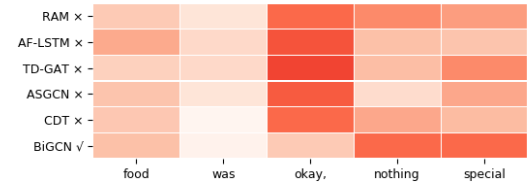
In the second sentence, Although AF-LSTM calculates the relations between the aspect and its context, the short distance between "*food*" and "*okay*" causes LSTM to assign the largest attention score to "*okay*". On the other hand, since "*okay*" and "*food*" are closely connected in the dependency tree, the strong positive polarity of "*okay*" prejudices the de-

Table 3: Results for ablation study (%). ↓ denotes the drop of performance compared with the BiGCN model. M1: hierarchical lexical graph, M2: hierarchical syntactic graph, M3: basic lexical graph, M4: basic syntactic graph.

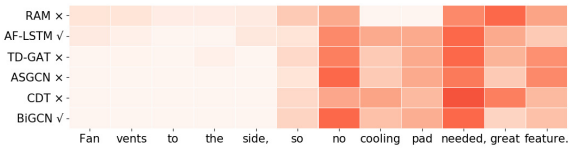
Model	Twitter		Lap14		Rest14		Rest15		Rest16	
	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1
BiGCN	**74.16**	**73.35**	**74.59**	**71.84**	**81.97**	**73.48**	**81.16**	**64.79**	**88.96**	**70.84**
M1	73.18 ↓ 0.98	71.29 ↓2.06	74.05 ↓ 0.54	70.29 ↓ 1.55	81.26 ↓ 0.71	72.62 ↓ 0.86	80.34 ↓0.82	62.20 ↓2.59	88.28 ↓ 0.68	68.94 ↓ 1.90
M2	73.14 ↓ 1.02	71.36 ↓1.99	74.11 ↓ 0.48	70.34 ↓ 1.50	81.56 ↓ 0.41	72.73 ↓0.75	80.47 ↓0.69	62.53 ↓2.26	88.56 ↓ 0.40	69.07 ↓ 1.77
M3	72.14 ↓ 2.02	70.84 ↓ 2.51	73.19 ↓ 1.40	69.86 ↓1.98	80.29 ↓ 1.68	71.85 ↓ 1.63	80.06 ↓ 1.10	61.17 ↓ 3.62	87.49 ↓ 1.47	68.17 ↓ 2.67
M4	72.83 ↓1.33	70.62 ↓ 2.73	74.23 ↓0.36	69.46 ↓ 2.38	80.87 ↓ 1.10	72.31 ↓1.17	80.22 ↓ 0.94	61.09 ↓3.70	87.95 ↓ 1.01	68.46 ↓ 2.38



(a) aspect: *staff*, label: *negative*



(b) aspect: *food*, label: *neutral*



(c) aspect: *cooling pad*, label: *neutral*

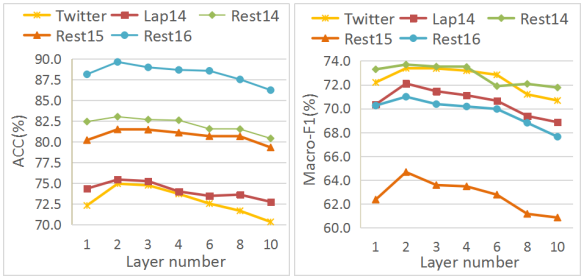
Figure 3: Visualization results for RAM, AF-LSTM, TD-GAT, ASGCN, CDT, and BiGCN, where a ✓and × denotes the correct and wrong prediction, respectively.

cision of TD-GAT, ASGCN, and CDT. In contrast, with the help of global lexical information, our BiGCN model focuses on "*nothing special*" and correctly predict the neutral polarity for "*food*".

In the third sentence, the output from the parser connects "*no*" and "*cooling pad*" together. However, it also connects "*great*" with "*pad*", and "*needed*" with "*feature*", which results in the wrong prediction of TD-GAT, ASGCN and CDT. We notice that AF-LSTM can predict the polarity correctly. This is because AF-LSTM exploits the word association between "*no*" and "*needed*" which co-occur eight times in the training corpus. Similarly, with the help of such lexical information, our BiGCN model also highlights on "*no*" and "*needed*", and assigns the neutral polarity to "*cooling pad*". Note that this sentence has two aspects: fan and cooling pad. Since almost all models can make correct prediction for fan, we only present detailed analysis for cooling pad.

5.5 Impacts of Layer Number

One of the key contributions of our model is that the syntactic graph and lexical graph can interact on each other. The layer number in the HierAgg module denotes the number of interactions between two graphs. In this section, we examine the impacts of layer number l by varying it in [1, 2, 3, 4, 6, 8, 10]. The results are shown in Figure 4.



(a) (b)

Figure 4: Impacts of the layer number l.

It can be seen that our model achieves the best results with 2 or 3 layers. If only using 1 layer, the interaction between two graphs is not sufficient to produce good results. However, the performance does not always get improved with the increasing number of layers. This is because a large l value makes it hard to train the model. Moreover, a larger l introduces more parameters and results in a less generalizable model.

5.6 Analysis on Computational Cost

In this section, we compare the averaged training time over three runs of our BiGCN model with that of three typical baselines which are all graph based. The results are shown in Table 4.

It can be seen that the time cost of our model

Table 4: Running time of four methods.

Model	Twitter	Lap14	Rest14	Rest15	Rest16
ASGCN	600.43	52.34	110.04	40.67	59.42
CDT	584.93	49.96	100.43	37.56	62.14
TD-GAT	621.94	62.11	122.36	47.39	66.74
BiGCN	642.28	68.75	120.44	46.25	79.60

does not change much though we use two types of graphs. For example, on Rest14 and Rest15, the computational cost of our proposed BiGCN is less than that of TD-GAT. Even on the largest Twitter dataset, the ratio of increased time cost of our BiGCN to the most efficient CDT method is less than 10%.

6 Conclusions

In this paper, we propose a novel framework BiGCN to leverage the graph based methods for aspect level sentiment classification tasks. Besides the ordinary syntactic graph, we employ a lexical graph to capture the global word co-occurrence information in the training corpus. Furthermore, we build a concept hierarchy on each of the lexical and syntactic graphs, such that the functionally different types of relations in the graph can be treated separately. Finally, we design a HierAgg module to let the lexical and syntactic graphs work in a cooperative way. We conduct a set of experiments on five real world datasets. The results prove that our model achieves the state-of-the-art performance.

Acknowledgments

The work described in this paper is supported by the NSFC projects (61572376, 91646206), and the 111 project (B07037).

References

Rahul Bhagat, Srevatsan Muralidharan, Alex Lobzhanidze, and Shankar Vishwanath. 2018. Buy it again: Modeling repeat purchase recommendations. In *KDD*, pages 62–70.

Peng Chen, Zhongqian Sun, Lidong Bing, and Wei Yang. 2017. Recurrent attention network on memory for aspect sentiment analysis. In *EMNLP*, pages 452–461.

Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. 2017. Language modeling with gated convolutional networks. In *ICML*, pages 933–941.

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 *ACL*, pages 49–54.

Shuqin Gu, Lipeng Zhang, Yuexian Hou, and Yin Song. 2018. A position-aware bidirectional attention network for aspect-level sentiment analysis. In *COLING*, pages 774–784.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2018. Effective attention modeling for aspect-level sentiment classification. In *COLING*, pages 1121–1131.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *KDD*, pages 168–177.

Binxuan Huang and Kathleen M. Carley. 2018. Parameterized convolutional neural networks for aspect level sentiment classification. In *EMNLP*, pages 1091–1096.

Binxuan Huang and Kathleen M. Carley. 2019. Syntax-aware aspect level sentiment classification with graph attention networks. In *EMNLP-IJCNLP*, pages 5468–5476.

Long Jiang, Mo Yu, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. 2011. Target-dependent twitter sentiment classification. In *ACL*, pages 151–160.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Xin Li, Lidong Bing, Wai Lam, and Bei Shi. 2018. Transformation networks for target-oriented sentiment classification. In *ACL*, pages 946–956.

Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2017. Interactive attention networks for aspect-level sentiment classification. In *IJCAI*, pages 4068–4074.

Shotaro Matsumoto, Hiroya Takamura, and Manabu Okumura. 2005. Sentiment classification using word sub-sequences and dependency sub-trees. In *PAKDD*, pages 301–311.

Saif M. Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013. Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets. *CoRR*, abs/1308.6242.

Tetsuji Nakagawa, Kentaro Inui, and Sadao Kurohashi. 2010. Dependency tree-based sentiment classification using crfs with hidden variables. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 2-4, 2010, Los Angeles, California, USA*, pages 786–794.

Vincent Ng, Sajib Dasgupta, and S. M. Niaz Arifin. 2006. Examining the role of linguistic knowledge sources in the automatic identification and classification of reviews. In *ACL*.

Thien Hai Nguyen and Kiyoaki Shirai. 2015. Phrasernn: Phrase recursive neural network for aspect-based sentiment analysis. In *EMNLP*, pages 2509–2514.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad Al-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, Véronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia V. Loukachevitch, Evgeniy V. Kotelnikov, Núria Bel, Salud María Jiménez Zafra, and Gülsen Eryigit. 2016. Semeval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2016, San Diego, CA, USA, June 16-17, 2016*, pages 19–30.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. Semeval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2015, Denver, Colorado, USA, June 4-5, 2015*, pages 486–495.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation, SemEval@COLING 2014, Dublin, Ireland, August 23-24, 2014*, pages 27–35.

Ana Salwa, Nurfadhlina Sharef, Masrah Murad, and Azreen Azman. 2018. Aspect extraction performance with pos tag pattern of dependency relation in aspect-based sentiment analysis. In *CAMP*, pages 1–6.

Kai Sun, Richong Zhang, Samuel Mensah, Yongyi Mao, and Xudong Liu. 2019. Aspect-level sentiment analysis via convolution over dependency tree. In *EMNLP-IJCNLP*, pages 5678–5687.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2016a. Effective lstms for target-dependent sentiment classification. In *COLING*, pages 3298–3307.

Duyu Tang, Bing Qin, and Ting Liu. 2016b. Aspect level sentiment classification with deep memory network. In *EMNLP*, pages 214–224.

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Learning to attend via word-aspect associative fusion for aspect-based sentiment analysis. In *AAAI*, pages 5956–5963.

Duy-Tin Vo and Yue Zhang. 2015. Target-dependent twitter sentiment classification with rich automatic features. In *IJCAI*, pages 1347–1353.

Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep & cross network for ad click predictions. In *Proceedings of the ADKDD'17, Halifax, NS, Canada, August 13 - 17, 2017*, pages 12:1–12:7.

Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. 2016. Attention-based LSTM for aspect-level sentiment classification. In *EMNLP*, pages 606–615.

Jason Weston, Sumit Chopra, and Antoine Bordes. 2015. Memory networks. In *ICLR*.

Wei Xue and Tao Li. 2018. Aspect based sentiment analysis with gated convolutional networks. In *ACL*, pages 2514–2523.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *AAAI*, pages 7370–7377.

Chen Zhang, Qiuchi Li, and Dawei Song. 2019. Aspect-based sentiment classification with aspect-specific graph convolutional networks. In *EMNLP-IJCNLP*, pages 4567–4577.

Meishan Zhang, Yue Zhang, and Duy-Tin Vo. 2016. Gated neural networks for targeted sentiment analysis. In *AAAI*, pages 3087–3093.