Aspect-based Sentiment Analysis with Type-aware Graph Convolutional Networks and Layer Ensemble

Yuanhe Tian[♥*], **Guimin Chen**[♡*], **Yan Song**[♠♡†]

[♥]University of Washington [♡]Shenzhen Research Institute of Big Data

[♠]The Chinese University of Hong Kong (Shenzhen)

[♥]yhtian@uw.edu [♡]chenguimin@foxmail.com [♠]songyan@cuhk.edu.cn

Abstract

It is popular that neural graph-based models are applied in existing aspect-based sentiment analysis (ABSA) studies for utilizing word relations through dependency parses to facilitate the task with better semantic guidance for analyzing context and aspect words. However, most of these studies only leverage dependency relations without considering their dependency types, and are limited in lacking efficient mechanisms to distinguish the important relations as well as learn from different layers of graph based models. To address such limitations, in this paper, we propose an approach to explicitly utilize dependency types for ABSA with type-aware graph convolutional networks (T-GCN), where attention is used in T-GCN to distinguish different edges (relations) in the graph and attentive layer ensemble is proposed to comprehensively learn from different layers of T-GCN. The validity and effectiveness of our approach are demonstrated in the experimental results, where state-of-the-art performance is achieved on six English benchmark datasets. Further experiments are conducted to analyze the contributions of each component in our approach and illustrate how different layers in T-GCN help ABSA with quantitative and qualitative analysis.[1]

1 Introduction

Aspect-based sentiment analysis (ABSA) processes fine-grained sentiment polarities towards specific aspects, where in many cases, it is required to identify different sentiments for multiple aspects in the same context. For example, in the sentence "*The drink menu is limited but the wines are excellent.*", the sentiment polarity towards "*drink menu*" is negative while that towards "*wines*" is positive; an ABSA system may predict wrong if it fails to capture the important contextual information for each aspects. Therefore, to model such contextual information, neural models (e.g., Bi-LSTM and Transformer (Vaswani et al., 2017)) have been widely used for ABSA and demonstrated to be useful for this task (Wang et al., 2016; Tang et al., 2016a; Chen et al., 2017; Ma et al., 2017; Fan et al., 2018).

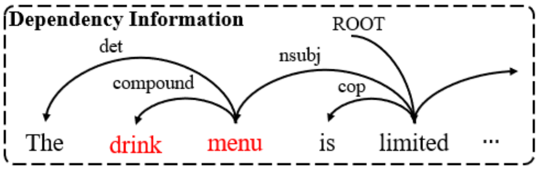


Figure 1: An example sentence (including the aspect term "*drink menu*") with its dependency parsing result.

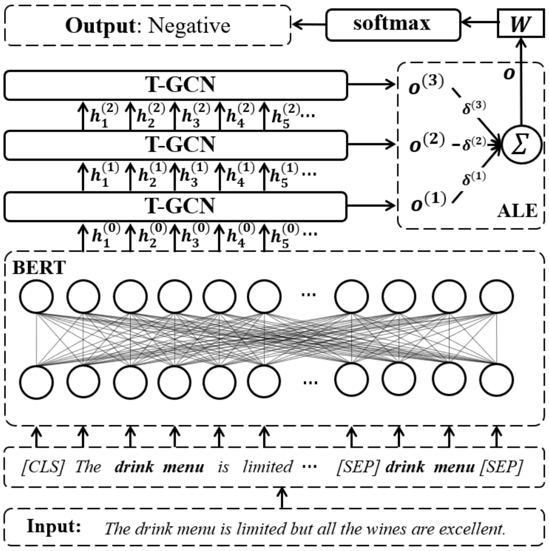
As a further enhancement of encoding contextual information for ABSA, there are studies (Sun et al., 2019; Huang and Carley, 2019; Zhang et al., 2019a) using graph convolutional networks (GCN) to learn from a graph that is often built over the dependency parsing results of the input texts. As a result, the GCN models are able to learn from distant word-word relations that are more helpful to ABSA. However, GCN models used in these studies are limited by omitting the information carried in dependency types and treating all word-word relations in the graph equally, therefore unimportant relations may not be distinguished and mislead ABSA accordingly. For example, Figure 1 illustrates an example sentence with an aspect highlighted in red, where the aspect word "*menu*" is connected with three others words, i.e., "*the*", "*drink*", and "*limited*". The connection between "*menu*" and "*limited*" could be the most important one since its dependency type, i.e., "*nsubj*", suggests that "*menu*" is the nominal subject of "*limited*", which strongly guides sentiment analysis towards "*menu*". In this case, if the dependency type is not modeled, one may not be able to leverage such beneficial information. In addition, although previous GCN models learn such

[*] Equal contribution.

[†] Corresponding author.

[1] The code and models involved in this paper are released at https://github.com/cuhksz-nlp/ASA-TGCN.

Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 2910–2922
June 6–11, 2021. ©2021 Association for Computational Linguistics



Output: Negative
softmax
W
T-GCN
T-GCN
T-GCN
$h_1^{(2)}$ $h_2^{(2)}$ $h_3^{(2)}$ $h_4^{(2)}$ $h_5^{(2)}$ ···
$h_1^{(1)}$ $h_2^{(1)}$ $h_3^{(1)}$ $h_4^{(1)}$ $h_5^{(1)}$ ···
$h_1^{(0)}$ $h_2^{(0)}$ $h_3^{(0)}$ $h_4^{(0)}$ $h_5^{(0)}$ ···
o
$o^{(3)}$
$\delta^{(3)}$
$o^{(2)}$
$\delta^{(2)}$
Σ
$\delta^{(1)}$
$o^{(1)}$
ALE
BERT
[CLS] The drink menu is limited ··· [SEP] drink menu [SEP]
Input: The drink menu is limited but all the wines are excellent.

Figure 2: The overall architecture of our approach with an example sentence-aspect pair input (the aspect words "*dink menu*" are in boldface) from a sentence. Our T-GCN and ALE are marked on top of the figure.

word-word relations by multiple GCN layers, they only use the output from the last layer for ABSA, where the encodings from intermediate layers are omitted and some essential information may be lost because different context information are modeled across layers. Thus an appropriate approach is required to enhance current GCN models for ABSA.

In this paper, we propose a type-aware graph convolutional networks (T-GCN) with multiple layers to enhance ABSA by incorporating both word relations and their dependency types to comprehensively learn from dependency parsing results. Specifically, we firstly obtain the dependency parsing results of the input texts through off-the-shelf toolkits, then build the graph over the dependency tree with each edge labeled by the corresponding dependency type between the two connected words, later apply an attention mechanism to the graph to weight all edges according to their contributions to the task, and finally use attentive layer ensemble to weight and combine the contextual information learned from different GCN layers. In doing so, our proposed T-GCN model can not only model word-word relations and their dependency types, but also distinguish the important contextual information from such relations to enhance ABSA. Experiments on six English benchmark datasets are conducted to evaluate the proposed model, where the results illustrate its effectiveness and state-of-the-art performance is observed over previous studies on all datasets. We also perform further analysis to investigate the contribution of each component (i.e., type-aware graph, attention for edges, and attentive layer ensemble) in our approach, and illustrate how different layers in T-GCN helps ABSA with quantitative and qualitative studies.

2 The Approach

Given an input sentence $\mathcal{X} = x_1, x_2, \cdots, x_n$ and the aspect terms $\mathcal{A} \subset \mathcal{X}$ ($\mathcal{A}$ is usually a sub-string of $\mathcal{X}$), the conventional ABSA approaches often take the sentence-aspect pair as the input and predicts $\mathcal{A}$'s sentiment polarity $\widehat{y}$ (Tang et al., 2016b; Ma et al., 2017; Xue and Li, 2018; Hazarika et al., 2018; Fan et al., 2018; Huang and Carley, 2018; Tang et al., 2019; Chen and Qian, 2019; Tan et al., 2019; Tang et al., 2020). We follow this paradigm and the overview of our approach is illustrated in Figure 2, with a contextual encoder (i.e., BERT), the proposed T-GCN and the attentive layer ensemble (ALE). The overall conceptual formalism of our approach can be written as

$$\widehat{y} = \arg\max_{y \in \mathcal{T}} p\left(y | ALE\left(T\text{-}GCN\left(\mathcal{X}, \mathcal{A}\right)\right)\right) \quad (1)$$

where $\mathcal{T}$ denotes the set of all sentiment labels for y (i.e., *positive*, *neutral*, and *negative*) and p computes the probability of predicting $y \in \mathcal{T}$ given $\mathcal{X}$ and $\mathcal{A}$ through T-GCN and ALE. In the following texts, we firstly describe the construction of the graph with dependency types, then elaborate the details of our T-GCN model, and the ALE to incorporate contextual information from different T-GCN layers, and finally illustrate incorporating T-GCN to ABSA.

2.1 Type-aware Graph Construction

Contextual features such as n-grams and syntactic information have been demonstrated to be useful to enhance text representation and thus improve model performance for many NLP tasks (Sun and Xu, 2011; Song and Xia, 2012; Gong et al., 2012; Song et al., 2012; Xu et al., 2015; Chen et al., 2017; Zhang et al., 2019b; Tang et al., 2020). In addition, it is demonstrated by many recent studies that GCN models are effective in capturing contextual features that are represented in graph-like signals, i.e., dependencies among words, of an input sentence (Sun et al., 2019; Huang and Carley, 2019; Zhang et al., 2019a; Tian et al., 2020c; Chen et al., 2020). In the graph for conventional GCN models, each edge between any two words x_i and x_j in the input sentence is added to the graph if there is a

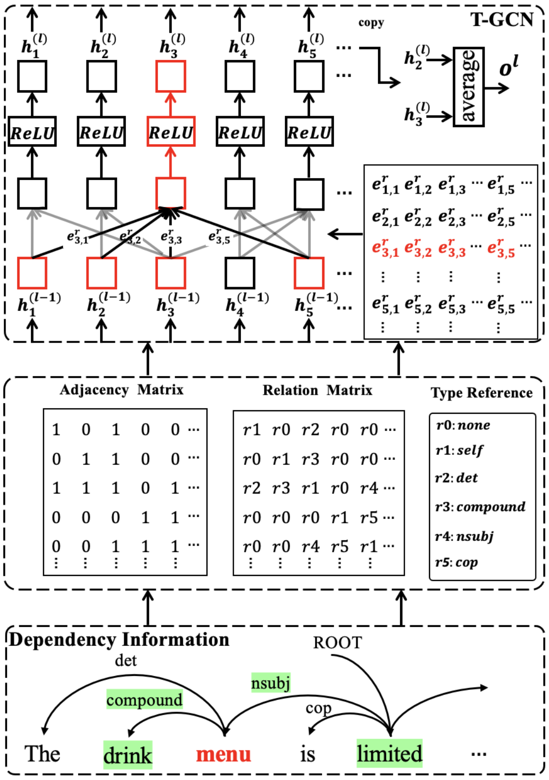

T-GCN
copy
$h_1^{(l)}$ $h_2^{(l)}$ $h_3^{(l)}$ $h_4^{(l)}$ $h_5^{(l)}$ ···
$h_2^{(l)}$
$h_3^{(l)}$
average
o^l
ReLU
$e^r_{3,1}$ $e^r_{3,2}$ $e^r_{3,3}$ $e^r_{3,5}$
$h_1^{(l-1)}$ $h_2^{(l-1)}$ $h_3^{(l-1)}$ $h_4^{(l-1)}$ $h_5^{(l-1)}$ ···
$e^r_{1,1}$ $e^r_{1,2}$ $e^r_{1,3}$ ··· $e^r_{1,5}$ ···
$e^r_{2,1}$ $e^r_{2,2}$ $e^r_{2,3}$ ··· $e^r_{2,5}$ ···
$e^r_{3,1}$ $e^r_{3,2}$ $e^r_{3,3}$ ··· $e^r_{3,5}$ ···
$e^r_{5,1}$ $e^r_{5,2}$ $e^r_{5,3}$ ··· $e^r_{5,5}$ ···
Adjacency Matrix
1 0 1 0 0 ···
0 1 1 0 0 ···
1 1 1 0 1 ···
0 0 0 1 1 ···
0 0 1 1 1 ···
Relation Matrix
r1 r0 r2 r0 r0 ···
r0 r1 r3 r0 r0 ···
r2 r3 r1 r0 r4 ···
r0 r0 r0 r1 r5 ···
r0 r0 r4 r5 r1 ···
Type Reference
r0: none
r1: self
r2: det
r3: compound
r4: nsubj
r5: cop
Dependency Information
ROOT
det
compound
nsubj
cop
The drink menu is limited ···

Figure 3: An illustration of how we build the type-aware graph from dependency parsing results and the detail of a T-GCN layer that consumes the graph. Edges and their dependency types are illustrated in the adjacency matrix and the relation matrix, respectively.

dependency relation on them. Therefore, they fail to comprehensively use the dependency parsing results because dependency types are always omitted in the graph. To leverage the such type information, we propose the type-aware graph for feeding our T-GCN via the following steps.

First, we use off-the-shelf toolkits to obtain the dependency results, which can be represented by a list of dependency tuples $(x_i, x_j, r_{i,j})$ with $r_{i,j}$ denoting the dependency type between x_i and x_j. Second, we use an adjacency matrix $\mathbf{A} = \{a_{i,j}\}_{n \times n}$ to present the graph by recording word relations in all tuples and a relation type matrix $\mathbf{R} = \{r_{i,j}\}_{n \times n}$ to represent the edges with their dependency types. Therefore, $\mathbf{A}$ is a 0-1 matrix where $a_{i,j} = 1$ if there is an edge between x_i and x_j, and $a_{i,j} = 0$ otherwise. For $\mathbf{R}$, each element $r_{i,j}$ in it uses a mark to denote the dependency type between x_i and x_j. Figure 3 illustrates the dependency parsing results of an example sentence as well as its type-aware graph represented by $\mathbf{A}$ and $\mathbf{R}$, with the marks for $r_{i,j}$ listed in the "Type Reference". Finally, to leverage the relation types, we use a transition matrix to map all $r_{i,j}$ to their embeddings $\mathbf{e}^r_{i,j}$.

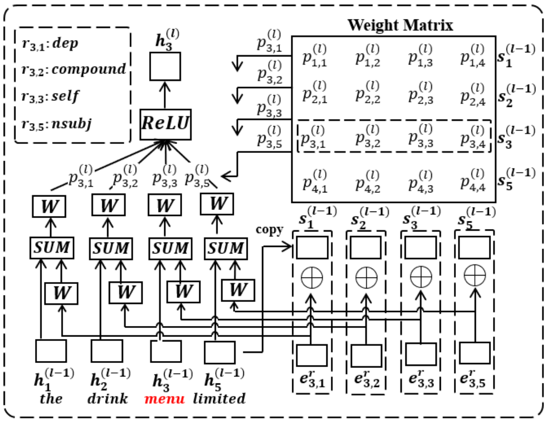

$r_{3,1}$: dep
$r_{3,2}$: compound
$r_{3,3}$: self
$r_{3,5}$: nsubj
$h_3^{(l)}$
ReLU
$p_{3,1}^{(l)}$ $p_{3,2}^{(l)}$ $p_{3,3}^{(l)}$ $p_{3,5}^{(l)}$
Weight Matrix
$p_{1,1}^{(l)}$ $p_{1,2}^{(l)}$ $p_{1,3}^{(l)}$ $p_{1,4}^{(l)}$ $s_1^{(l-1)}$
$p_{2,1}^{(l)}$ $p_{2,2}^{(l)}$ $p_{2,3}^{(l)}$ $p_{2,4}^{(l)}$ $s_2^{(l-1)}$
$p_{3,1}^{(l)}$ $p_{3,2}^{(l)}$ $p_{3,3}^{(l)}$ $p_{3,4}^{(l)}$ $s_3^{(l-1)}$
$p_{4,1}^{(l)}$ $p_{4,2}^{(l)}$ $p_{4,3}^{(l)}$ $p_{4,4}^{(l)}$ $s_5^{(l-1)}$
W
SUM
copy
$s_1^{(l-1)}$ $s_2^{(l-1)}$ $s_3^{(l-1)}$ $s_5^{(l-1)}$
⊕
$h_1^{(l-1)}$ $h_2^{(l-1)}$ $h_3^{(l-1)}$ $h_5^{(l-1)}$
the drink menu limited
$e^r_{3,1}$ $e^r_{3,2}$ $e^r_{3,3}$ $e^r_{3,5}$

Figure 4: The illustration of how we compute $\mathbf{h}_i^{(l)}$ for x_3 ="*menu*" through a T-GCN layer. All words x_j connected to "*menu*" with their dependency types (in embeddings $\mathbf{e}^r_{i,j}$) are shown at the bottom part.

2.2 T-GCN

With the type-aware graph, we propose an L-layer T-GCN and for each layer we apply attention to the edges in the graph to weight them by their contributions to the ABSA task. Figure 4 illustrates the processes of doing so for the aspect word "*menu*" in the sentence "*The drink menu is limited but all the wines are excellent.*". In detail, for a each edge between x_i and x_j, the l-th GCN layer takes the hidden vectors $\mathbf{h}_i^{(l-1)}$ and $\mathbf{h}_j^{(l-1)}$ of x_i and x_j from the $(l-1)$-th GCN layer ($\mathbf{h}_i^{(0)}$ and $\mathbf{h}_i^{(0)}$ are from the context encoder) and concatenate them with the embeddings of their dependency types $\mathbf{e}^r_{i,j}$ by

$$\mathbf{s}_i^{(l)} = \mathbf{h}_i^{(l-1)} \oplus \mathbf{e}^r_{i,j} \tag{2}$$

and

$$\mathbf{s}_j^{(l)} = \mathbf{h}_j^{(l-1)} \oplus \mathbf{e}^r_{i,j} \tag{3}$$

Then, we compute the weight $p_{i,j}^{(l)}$ for this edge by

$$p_{i,j}^{(l)} = \frac{a_{i,j} \cdot exp\left(\mathbf{s}_i^{(l)} \cdot \mathbf{s}_j^{(l)}\right)}{\sum_{j=1}^{n} a_{i,j} \cdot exp\left(\mathbf{s}_i^{(l)} \cdot \mathbf{s}_j^{(l)}\right)} \tag{4}$$

and align the dimension of $\mathbf{e}^r_{i,j}$ to $\mathbf{h}_j^{(l-1)}$ by a trainable matrix $\mathbf{W}_R^{(l)}$ of the l-th GCN layer by

$$\mathbf{h}_j^{(l-1)'} = \mathbf{h}_j^{(l-1)} + \mathbf{W}_R^{(l)} \cdot \mathbf{e}^r_{i,j} \tag{5}$$

Finally, we apply $p_{i,j}^{(l)}$ to this edge and compute the output for x_i at l-th layer following a similar process in the conventional GCN by

$$\mathbf{h}_i^{(l)} = \sigma\left(\sum_{j=1}^{n} p_{ij}\left(\mathbf{W}^{(l)} \cdot \mathbf{h}_j^{(l-1)'} + \mathbf{b}^{(l)}\right)\right) \tag{6}$$

where $\mathbf{W}^{(l)}$ and $\mathbf{b}^{(l)}$ denote trainable parameters in the l-th GCN layer and σ refers to the *ReLU* activation function. The above process is conducted for every x_i and throughout all GCN layers, thus the information of dependency types are incorporated into the GCN to enhance ABSA accordingly.

2.3 Attentive Layer Ensemble

For each word x_i, since every T-GCN layer incorporates information from the words that directly connect to it, so that multiple T-GCN layers could learn indirect word relations from long distance. Thus it is assumed that different layers have their unique capabilities to encode contextual information. To utilize such capabilities, we propose to comprehensively learn from all T-GCN layers with attentive layer ensemble.

In doing so, we firstly obtain the output $\mathbf{o}^{(l)}$ from each T-GCN layer by averaging the output hidden vectors of all aspect terms $x_k \in \mathcal{A}$:

$$\mathbf{o}^{(l)} = \frac{1}{|\mathcal{A}|} \cdot \sum_{x_k \in \mathcal{A}} \mathbf{h}_k^{(l)} \tag{7}$$

where $|\mathcal{A}|$ is the number of words in the aspect terms $\mathcal{A}$. Then we attentively ensemble the output of all T-GCN layers through a weighted average:

$$\mathbf{o} = \sum_{l=1}^{L} \delta^{(l)} \cdot \mathbf{o}^{(l)} \tag{8}$$

where $\mathbf{o}$ is the final vector output for ABSA and $\delta^{(l)}$ is a trainable weight assigned to $\mathbf{o}^{(l)}$ to balance its contribution and satisfying $\sum_{l=1}^{L} \delta^{(l)} = 1$.

2.4 Encoding and Decoding with T-GCN

To support applying T-GCN for ABSA, there are necessary encoding and decoding processes. For encoding, there are two ways in doing so. The first is to take the sentence $\mathcal{X}$ as the input and obtain the hidden vectors $\mathbf{h}_i^{(0)}$ for all x_i by

$$\mathbf{H}^{\mathcal{X}} = BERT(\mathcal{X}) \tag{9}$$

where $\mathbf{H}^{\mathcal{X}}$ is the hidden vectors of all words in $\mathcal{X}$, and we use BERT as the encoder (same below). The second is to take the sentence-aspect pair as the input, which can be formalized by

$$[\mathbf{H}^{\mathcal{X}}, \mathbf{H}^{\mathcal{A}}] = BERT(\mathcal{X}, \mathcal{A}) \tag{10}$$

where $\mathbf{H}^{\mathcal{A}}$ is the hidden vectors of all aspect words. Then, the hidden vectors from $\mathbf{H}^{\mathcal{X}}$ or $\mathbf{H}^{\mathcal{A}}$ are feed into the T-GCN model as that described in §2.2. For decoding, after we obtain $\mathbf{o}$ from ALE, we firstly map $\mathbf{o}$ to the label space by a fully connected layer, $\mathbf{u} = \mathbf{W} \cdot \mathbf{o} + \mathbf{b}$, where $\mathbf{W}$ and $\mathbf{b}$ are the trainable matrix and the bias, respectively, and each dimension of $\mathbf{u}$ corresponds to a sentiment type. Thus, we employ a *softmax* function to $\mathbf{u}$ and predict the output sentiment $\hat{y}$ for the aspect $\mathcal{A}$ in $\mathcal{X}$ by:

$$\hat{y} = \arg\max \frac{exp(u^t)}{\sum_{t=1}^{|\mathcal{T}|} exp(u^t)} \tag{11}$$

where u^t is the value at dimension t in $\mathbf{u}$.

Datasets		Pos. #	Neu. #	Neg. #
LAP14	Train	994	464	870
	Test	341	169	128
REST14	Train	2,164	637	807
	Test	728	196	182
REST15	Train	907	36	254
	Test	326	34	207
REST16	Train	1,229	69	437
	Test	469	30	114
TWITTER	Train	1,561	3,127	1,560
	Test	173	346	173
MAMS (ATSA)	Train	3,380	5,042	2,764
	Dev	403	604	325
	Test	400	607	329

Table 1: The number of aspects with *positive*, *neutral*, and *negative* sentiment polarities in all datasets.

3 Experimental Settings

3.1 Datasets

In the experiments, we employ five widely used English benchmark datasets: LAP14 and REST14 from Pontiki et al. (2014), REST15 from Pontiki et al. (2015), REST16 from Pontiki et al. (2016), and TWITTER from Dong et al. (2014), with their official train/test splits. In addition, we try another recently released English dataset, named MAMS[2] (Jiang et al., 2019), with the official train/dev/test splits for ABSA, which is much larger than the aforementioned five datasets. It is worth noting that, in addition to the *positive*, *neutral*, and *negative* sentiment labels, LAP14, REST14, and REST16

[2] We use the ATSA part of MAMS obtained from `https://github.com/siat-nlp/MAMS-for-ABSA`.

Models	**Lap14**		**Rest14**		**Rest15**		**Rest16**		**Twitter**		**MAMS**	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
BERT-base (S)	77.74	73.30	82.68	73.54	81.34	63.57	88.89	68.19	73.70	71.50	78.94	79.42
+ GCN	79.52	76.01	84.79	77.93	83.60	65.71	90.76	72.79	75.16	72.96	80.69	80.27
+ T-GCN	80.25	76.92	85.54	78.86	85.07	**72.50**	91.83	76.86	76.16	74.44	81.73	81.12
BERT-base (P)	78.68	74.64	84.55	77.34	83.40	65.28	89.54	70.47	75.00	72.53	80.11	80.34
+ GCN	79.94	76.72	85.09	78.81	84.14	65.75	91.01	73.38	75.29	73.68	81.96	81.31
+ T-GCN	**80.88**	**77.03**	**86.16**	**79.95**	**85.26**	71.69	**92.32**	**77.29**	**76.45**	**75.25**	**83.38**	**82.77**
BERT-large (S)	78.06	74.67	83.04	73.27	83.02	68.34	90.20	73.64	73.12	72.08	79.33	79.87
+ GCN	80.09	76.84	86.07	80.35	84.69	70.31	91.48	74.96	75.21	73.69	81.36	81.04
+ T-GCN	81.50	78.48	86.88	81.03	85.07	70.30	92.32	75.83	75.43	73.71	82.70	82.16
BERT-large (P)	79.62	75.77	85.53	77.64	84.14	69.67	91.34	74.35	75.43	73.55	80.62	80.77
+ GCN	80.68	77.85	86.48	80.63	85.42	70.42	91.69	75.24	75.26	73.41	82.56	82.14
+ T-GCN	**81.97**	**78.71**	**87.41**	**82.23**	**86.00**	**72.81**	**92.97**	**80.07**	**78.03**	**77.31**	**83.68**	**83.07**

Table 2: Experimental results (accuracy and F1 scores) of using two encoders i.e., BERT-base and BERT-large, with different configurations on six benchmark datasets. "GCN" refers to the normal GCN model without using type-aware graph, attention mechanism as well as ALE. "S" and "P" refer to the settings that the input is a single sentence and a sentence-aspect pair, respectively.

contain another *conflict* label, which identifies the aspects that have conflict sentiment polarities. For example, the aspect "*sushi*" is assigned by a *conflict* label in "*Certainly not the best sushi in New York, however, it is always fresh.*" from Rest14. Therefore, we follow Tang et al. (2016b) to clean the datasets by removing all aspects with the aforementioned *conflict* label, as well as sentences without an aspect. The statistics (number of aspects with *positive*, *neutral*, and *negative* labels) of the processed six datasets are reported in Table 1.

3.2 Implementation Details

To build the graph for T-GCN, we firstly use the current best performing constituency parser, i.e., SAPar[3] (Tian et al., 2020d), to parse all input text into constituency trees, then convert the trees into dependency trees by Stanford Converter[4], and finally build the graph over the dependency relations and types from the trees.[5] Since high quality text representations can improve the performance of NLP models (Mikolov et al., 2013; Song et al., 2017; Bojanowski et al., 2017; Song and Shi, 2018; Song et al., 2018), we employ BERT (Devlin et al., 2019) as the context encoder, which and whose variants (Diao et al., 2020; Dai et al., 2019; Joshi et al., 2020) have demonstrated their effectiveness to encode context information and achieved state-of-the-art performance in many NLP tasks (Huang and Carley, 2019; Tian et al., 2020a,b; Tang et al., 2020; Nie et al., 2020; Wang et al., 2020). Specifically, we use the uncased BERT-base and BERT-large[6] with their default settings, i.e., 12 layers of self-attention with 768 dimensional hidden vectors for BERT-base and 24 layers of self-attention with 1024 dimensional hidden vectors for BERT-large, and use three T-GCN layers. We try two ways to encode the input, where the first encodes the single sentence and the second encodes the sentence-aspect pair. For all models, we use the pre-trained parameters of BERT and initialize all other trainable parameters by Xavier (Glorot and Bengio, 2010). Moreover, we use the cross-entropy loss function for our models and follow previous studies (Tang et al., 2016a; Chen et al., 2017; He et al., 2018a; Sun et al., 2019; Zhang et al., 2019a) to evaluate them via accuracy and macro-averaged F1 scores over all sentiment polarities. For datasets without the official development set, we randomly sample 10% instances from the training set and regard them as the development set to find the best hyper-parameter setting which is then used to train different models on the entire training set.[7]

[3] https://github.com/cuhksz-nlp/SAPar

[4] We use the converter of version 3.3.0 from https://stanfordnlp.github.io/CoreNLP/index.html.

[5] We also try Stanford CoreNLP Toolkits (https://stanfordnlp.github.io/CoreNLP/) (Manning et al., 2014) and spaCy (https://spacy.io/) dependency parsers with similar results obtained.

[6] We obtain the BERT models from https://github.com/huggingface/pytorch-pretrained-BERT.

[7] We report the hyper-parameter settings of different models with their size and running speed in Appendix A and B.

Models	Lap14		Rest14		Rest15		Rest16		Twitter		MAMS	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
†Chen et al. (2017)	74.49	71.35	80.23	70.80	-	-	-	-	69.36	67.30	-	-
Ma et al. (2017)	72.10	-	78.60	-	-	-	-	-	-	-	-	-
Fan et al. (2018)	75.39	72.47	81.25	71.94	-	-	-	-	72.54	70.81	-	-
Gu et al. (2018)	74.12	-	81.16	-	-	-	-	-	-	-	-	-
†He et al. (2018a)	72.57	69.13	80.63	71.32	81.67	66.05	64.61	67.45	-	-	-	-
He et al. (2018b)	71.15	67.46	79.11	69.73	81.30	68.74	85.58	69.76	-	-	-	-
Huang and Carley (2018)	70.06	-	79.20	-	-	-	-	-	-	-	-	-
Li et al. (2018)	76.54	71.75	80.69	71.27	-	-	-	-	74.97	73.60	-	-
Chen and Qian (2019)	73.87	70.10	79.55	71.41	-	-	-	-	-	-	-	-
Du et al. (2019)	76.80	73.29	81.79	73.40	-	-	-	-	75.01	73.81	-	-
Hu et al. (2019)	-	-	84.28	74.45	78.58	54.72	-	-	-	-	-	-
*Mao et al. (2019)	75.84	72.49	82.49	72.10	-	-	-	-	72.35	69.45	-	-
*Song et al. (2019)	79.93	76.31	83.12	73.76	-	-	-	-	74.71	73.13	-	-
*Xu et al. (2019)	78.07	75.08	84.95	76.96	-	-	-	-	-	-	-	-
*Jiang et al. (2019)	-	-	85.93	-	-	-	-	-	-	-	83.39	-
†Sun et al. (2019)	77.19	72.99	82.30	74.02	-	-	85.58	69.93	74.66	73.66	-	-
†Zhang et al. (2019a)	75.55	71.05	81.22	72.94	79.89	61.89	88.99	67.48	72.69	70.59	-	-
*†Huang and Carley (2019)	80.10	-	83.00	-	-	-	-	-	-	-	-	-
*†Wang et al. (2020)	78.21	74.07	86.60	81.35	-	-	-	-	76.15	74.88	-	-
*†Tang et al. (2020)	79.8	75.6	86.3	80.0	84.0	71.0	91.9	79.0	77.9	75.4	-	-
*†Our Best Model	**81.97**	**78.71**	**87.41**	**82.23**	**86.00**	**72.81**	**92.97**	**80.07**	**78.03**	**77.31**	**83.68**	**83.07**

Table 3: Performance (accuracy and F1 scores) comparison of our best model (i.e., T-GCN and ALE on large BERT with sentence-aspect pair input) with previous studies on all six benchmark datasets. Models using BERT-large and dependency information are marked by "*" and "†", respectively.

4 Experimental Results

4.1 Effect of T-GCN

In the main experiments, for each encoder (i.e., BERT base and large), we run two baselines: 1, only using BERT and 2, BERT with normal GCN where all edges are equally treated and the ABSA result is predicted based on the output of the last GCN layer. Table 2 reports the experimental results from all baselines and our models.[8]

There are several observations. First, for both BERT-base and BERT-large encoders, although the models with normal GCN are able to enhance the BERT baselines, our models can further improve the performance in both accuracy and F1 socres on all datasets. This observation clearly illustrate the effectiveness of incorporating dependency type information into GCN and thus improves ABSA accordingly. Second, in most cases, our models that encode the sentence-aspect pair achieve higher results than the ones encoding the single sentence, which is not surprising because the aspect is therefore emphasized in the input and provide more contextual information to be modeled for ABSA.

4.2 Comparison with Previous Studies

To further demonstrate the effective of our approach, we compare the performance of our best model (i.e., T-GCN using BERT-large encoder with sentence-aspect pair input), with previous studies on all datasets. The results are reported in Table 3, where our model outperforms previous studies, including the ones (Huang and Carley, 2019; Wang et al., 2020; Tang et al., 2020) using BERT-large (marked by "*") and dependency information (marked by "†"), on all datasets in terms of both accuracy and F1 scores. In particular, compared with our approach, Huang and Carley (2019) use a variant of graph attention networks (GAT), while they do not use dependency types; Wang et al. (2020) also use a variant of GAT and they use the relation type as well, but they do not assign different weight to separate word-word relations; Tang et al. (2020) use a variant of GCN but they do not use the dependency type information. Our model shows its superiority to the aforementioned studies since we not only assign different weights to dependencies, but also comprehensively leverage the dependency parsing results with both word relations and their dependency type information, as well as fined-grained encoding results from multiple T-GCN layers.

5 Analyses

5.1 Ablation Study

To explore the effectiveness of different components in our model, i.e., type-aware graph (TG),

[8] We report the mean and the standard deviation of the results of the same group of models in Appendix C.

	Setting			LAP14		REST14		REST15		REST16		TWITTER		MAMS	
	TG	Att	ALE	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
1	√	√	√	**81.97**	**78.71**	**87.41**	**82.23**	**86.00**	**72.81**	**92.97**	**80.07**	**78.03**	**77.31**	**83.68**	**83.07**
2	×	√	√	81.46	78.69	87.21	81.35	85.93	72.06	92.14	77.80	77.31	76.42	83.01	82.53
3	√	×	√	80.96	77.66	86.68	81.09	85.66	71.03	91.89	76.40	76.73	75.90	82.78	82.47
4	√	√	×	80.88	77.54	86.85	81.20	85.89	71.32	91.75	76.65	77.16	76.23	83.31	82.70
5	√	×	×	80.79	77.42	86.50	80.42	85.65	70.50	91.45	75.64	76.15	75.28	83.16	82.76
6	×	√	×	81.10	78.12	86.88	81.02	85.89	71.10	91.99	77.50	77.31	76.18	83.53	82.90
7	×	×	√	80.85	77.56	86.45	80.21	85.79	70.91	91.66	75.92	76.73	74.97	82.86	82.38
8	×	×	×	80.68	77.85	86.48	80.63	85.42	70.42	91.69	75.24	75.26	73.41	82.56	82.14

Table 4: Experimental results of ablation study on the six datasets, with different configurations applied to our best model. 'TG" refers to the type-aware graph; "ATT" denotes the attention mechanism in T-GCN; "ALE" stands for the attentive layer ensemble. "√" and "×" represent if a corresponding component is used or not.

attention (Att), and ALE, we conduct an ablation study based on our best model (i.e., T-GCN on BERT-large encoder with sentence-aspect pair input). The experimental results on all datasets with respect to using different combinations of such components are reported in Table 4, with the results of the full model and the baseline with normal GCN illustrated on the first (ID: 1) and last row (ID: 8), respectively. Herein, models without ALE (ID: 4-6) use the output of the last T-GCN layer (i.e., the third layer) to predict the sentiment polarity.[9]

Here are some observations. First, it is clearly indicated in results that, the model performance drops on all datasets if any component is excluded from the full model. This observation indicates that all three components play important roles in our approach to enhance ABSA; each one has its unique contribution to the full model. Second, for each single components, compared with the results from GCN baseline (ID: 8), the results from models with a particular module (ID: 5-7) demonstrate that the attention mechanism is the most important one to improve model performance, where on all datasets, the model (ID: 6) with attention outperforms the others. This observation complies with our intuition because the attention directly guides the model to distinguish the contextual information to the aspect words, so that informative words are highlighted so as to improve ABSA accordingly.

5.2 Impact of Different T-GCN Layers

Besides those components, we also investigate the effect of each layer when our model is trained on different datasets. In doing so, we perform experiments on all datasets using our best performing model and use the weight ($\delta^{(l)}$ in Eq. (8)) assigned to each T-GCN layer to identify the contribution of them. The results are illustrated in Figure 5, with the weights for the 1st, 2nd, and 3rd T-GCN layers drawn in blue, green, and orange bars, respectively.

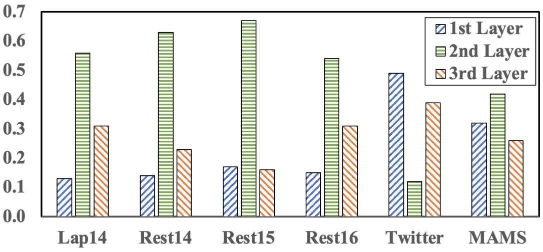


Figure 5: The histograms of weights assigned to different T-GCN layers (blue, green, and orange bars refer to the weights for the 1st, 2nd, and 3rd layer, respectively) in ALE with respect to each dataset.

We have following observations. First, all layers contribute to the final prediction for ABSA, which complies with our expectation and confirms the validity of leveraging the information from all layers of GCN. Therefore, the model is able to provide comprehensive contextual information comparing to that only uses the output from the last layer. Second, interestingly, as shown in the histograms, for most datasets (i.e. LAP14, REST14, REST15, REST16, and MAMS), the second layer of T-GCN contributes the most among all three layers. A possible reason behind is that (1) the second layer is able to encode contextual information from a larger range (because the edges in the first layer only cover words with direct relations, while the second and third layer provide indirect relations, i.e., second and third order dependencies in practice); (2) comparing to the third layer, the second layer may introduce less irrelevant information from multi-word relations. Third, we also notice that for TWITTER, the weight distribution among three layers is rather different from the other

[9] We obtain similar results when using the output of intermediate layers. The details are reported in Appendix D.

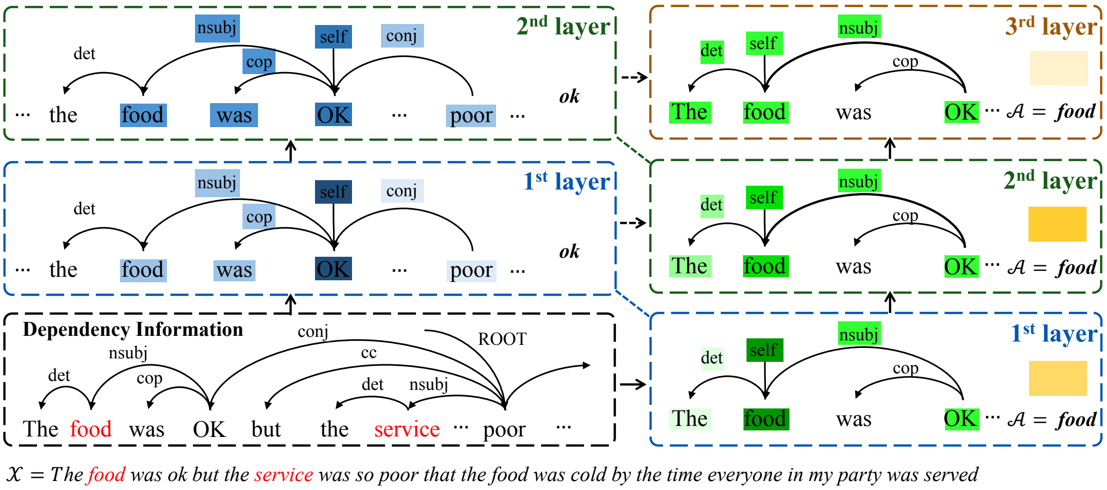

2nd layer
nsubj
self
conj
det
cop
··· the food was OK ··· poor ···
ok
1st layer
Dependency Information
conj
ROOT
nsubj
cc
det
cop
det
nsubj
The food was OK but the service ··· poor ···
3rd layer
The food was OK ··· $\mathcal{A}$ = food
$\mathcal{X}$ = The food was ok but the service was so poor that the food was cold by the time everyone in my party was served

Figure 6: Visualization of the weights assigned to different edges and dependency types in each T-GCN layer for an example sentence with two aspects (in red) in conflict sentiment polarities. The edge and type weights (in blue) for "*OK*" in the first and second layer are illustrated on the left, while such weights (in green) for "*food*" and ALE weights (in yellow) for each layer are illustrated on the right. Deeper color refers to the higher weight.

datasets, where the first and last layer contributes more to ABSA. This observation can be explained by that, TWITTER is social medial data, where, in general, sentences in such data are short and less organized, so that our model may require the information from either local context or the entire sentence for ABSA.

5.3 Case Study

To further illustrate the effectiveness of T-GCN on leveraging the information of dependency types and weighting salient word relations for improving ABSA, we conduct a case study on using our model to process the sentence "*The food was OK but the service was so poor that the food was cold by the time everyone in my party was served*" from REST16. In this sentence, there are two aspects with contrast sentiment polarities, i.e., "*food*" and "*service*" have *positive* and *negative* sentiment suggested by "*OK*" and "*limited*", respectively.

To demonstrate the effectiveness of our model to process such sentence with conflict sentiments, on the right part of Figure 6, we visualize weights (in green) assigned to the edges connected to '*food*" from the attention in all T-GCN layers, and the ALE weights (in yellow) for each layer, where deeper color refers to higher weight. For those edges, except for its self-connection, the edge between "*food*" and "*OK*" receives the highest weight in every layer, and the second layer receives the highest weight in ALE. Note that in this case, the reason why T-GCN works can be explained by that, when there are more than two layers are used in a GCN model, the edges connecting to "*OK*" also influence the ABSA results because indirect relations are introduced across layers. As a result, the noisy connection between "*OK*" and "*poor*" may contribute to the prediction and the normal GCN could possibly fail on this case because of lacking a mechanism to distinguish it from other edges. Therefore, as shown in the left part of Figure 6, we also visualize the weights for edges connecting to "*OK*" from the first and second T-GCN layers,[10] where the informative word relations and their dependency types receive much heavier weights than that for noisy ones. Moreover, it is noticed that the dependency type for the edge between "*OK*" and "*poor*" is "*conj*" (conjunction), which suggests that "*poor*" is syntactically parallel with "*OK*" and is thus less likely to provide essential sentiment guidance for "*OK*". Overall, this case study illustrates that our model successfully identifies that "*OK*" is the most important contextual information to determine the sentiment for "*food*", with the help of dependency type and attention used in T-GCN, and also shows that the final prediction relies on the contributions from different T-GCN layers.

6 Related Work

ABSA is in the line of research on sentiment analysis in a fine-grained level focusing on categoriz-

[10] Note that we do not visualize the weights for "*OK*" in the third layer because its resulting hidden vector does not contribute to the final sentiment prediction.

ing sentiment polarities for a specific aspect (e.g., "*chicken*") or category (e.g., "*food*") in a sentence. Conventionally, this task is formulated as to classify a sentence-aspect pair and most of studies try to explore the contextual information between aspect and the entire sentence to facilitate the analysis of sentiment (Dong et al., 2014; Wang et al., 2016; Tang et al., 2016a; Ma et al., 2017; Chen et al., 2017; Xue and Li, 2018; Li et al., 2018; Xu et al., 2019; Wang et al., 2020; Tang et al., 2020). To further enhancing the modeling of contextual information, dependency parses were leveraged by many studies, where adaptive recursive neural networks (Dong et al., 2014), attention mechanism (He et al., 2018a), and key-value memory networks (Tian et al., 2021) are used. Later, Huang and Carley (2019); Sun et al. (2019); Zhang et al. (2019a); Wang et al. (2020); Tang et al. (2020) leveraged graph neural models (e.g., GCN) for ABSA with their graph built upon the dependency tree obtained from off-the-self dependency parsers, and demonstrated promising results. The models in their studies normally focus on building the graph with the dependency structure without considering dependency types, meanwhile treating the edges in the graph equally. In addition, they usually use the output of the last layer to predict sentiment labels although their models consist multiple layers. Thus, our approach differs from previous graph-based ones on several aspects, including the integration of depdendency type information, applying attention to edges, and ensemble of multiple layers to comprehensively learn from the graph model.

7 Conclusion

In this paper, we propose a neural approach for ABSA with T-GCN, where the input graph is built on the dependency tree of the input sentence. Specifically, the edges in the graph are constructed on top of both dependency relations and types for the input sentence; for each word, we use attention to weight all such type-aware edges associated to it in the T-GCN; we also apply attentive layer ensemble to comprehensively learn contextual information from different T-GCN layers. Experimental results on six widely used English benchmark datasets demonstrate the effectiveness of our approach, where state-of-the-art performance are achieved on all datasets. Further analyses illustrate the validity of incorporating type information into our model as well as applying attentive ensemble to learning from its multiple layers.

Acknowledgements

This work is supported by Chinese Key-Area Research and Development Program of Guangdong Province (2020B0101350001) and also partially supported by NSFC under the project "The Essential Algorithms and Technologies for Standardized Analytics of Clinical Texts" (12026610).

References

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Guimin Chen, Yuanhe Tian, and Yan Song. 2020. Joint Aspect Extraction and Sentiment Analysis with Directional Graph Convolutional Networks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 272–279.

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.

Zhuang Chen and Tieyun Qian. 2019. Transfer Capsule Network for Aspect Level Sentiment Classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 547–556.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Shizhe Diao, Jiaxin Bai, Yan Song, Tong Zhang, and Yonggang Wang. 2020. ZEN: Pre-training Chinese Text Encoder Enhanced by N-gram Representations. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4729–4740.

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.

Chunning Du, Haifeng Sun, Jingyu Wang, Qi Qi, Jianxin Liao, Tong Xu, and Ming Liu. 2019. Capsule Network with Interactive Attention for Aspect-level Sentiment Classification. 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 5492–5501.

Feifan Fan, Yansong Feng, and Dongyan Zhao. 2018. Multi-grained Attention Network for Aspect-level Sentiment Classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3433–3442.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the Difficulty of Training Deep Feedforward Neural Networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256.

Zhengxian Gong, Min Zhang, Chew Lim Tan, and Guodong Zhou. 2012. N-gram-based Tense Models for Statistical Machine Translation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 276–285.

Shuqin Gu, Lipeng Zhang, Yuexian Hou, and Yin Song. 2018. A Position-aware Bidirectional Attention Network for Aspect-level Sentiment Analysis. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 774–784.

Devamanyu Hazarika, Soujanya Poria, Prateek Vij, Gangeshwar Krishnamurthy, Erik Cambria, and Roger Zimmermann. 2018. Modeling Inter-Aspect Dependencies for Aspect-Based Sentiment Analysis. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 266–270.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2018a. Effective Attention Modeling for Aspect-level Sentiment Classification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1121–1131.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2018b. Exploiting Document Knowledge for Aspect-level Sentiment Classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 579–585.

Mengting Hu, Shiwan Zhao, Li Zhang, Keke Cai, Zhong Su, Renhong Cheng, and Xiaowei Shen. 2019. CAN: Constrained Attention Networks for Multi-Aspect Sentiment Analysis. 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 4593–4602.

Binxuan Huang and Kathleen M Carley. 2018. Parameterized Convolutional Neural Networks for Aspect Level Sentiment Classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1091–1096.

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.

Qingnan Jiang, Lei Chen, Ruifeng Xu, Xiang Ao, and Min Yang. 2019. A Challenge Dataset and Effective Models for Aspect-Based Sentiment Analysis. 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 6280–6285, Hong Kong, China.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving Pre-training by Representing and Predicting Spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

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.

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.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.

Qianren Mao, Jianxin Li, Senzhang Wang, Yuanning Zhang, Hao Peng, Min He, and Lihong Wang. 2019. Aspect-Based Sentiment Classification with Attentive Neural Turing Machines. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 5139–5145.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *arXiv preprint arXiv:1301.3781*.

Yuyang Nie, Yuanhe Tian, Yan Song, Xiang Ao, and Xiang Wan. 2020. Improving Named Entity Recognition with Attentive Ensemble of Syntactic Information. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4231–4245.

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 Loukachevitch, Evgeniy Kotelnikov, Nuria Bel, Salud María Jiménez-Zafra, and Gülşen Eryiğit. 2016. SemEval-2016 Task 5: Aspect Based Bentiment Analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-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 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 2014)*, pages 27–35.

Yan Song, Prescott Klassen, Fei Xia, and Chunyu Kit. 2012. Entropy-based Training Data Selection for Domain Adaptation. In *Proceedings of COLING 2012: Posters*, pages 1191–1200.

Yan Song, Chia-Jung Lee, and Fei Xia. 2017. Learning Word Representations with Regularization from Prior Knowledge. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 143–152.

Yan Song and Shuming Shi. 2018. Complementary Learning of Word Embeddings. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4368–4374.

Yan Song, Shuming Shi, Jing Li, and Haisong Zhang. 2018. Directional Skip-Gram: Explicitly Distinguishing Left and Right Context for Word Embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 175–180.

Yan Song and Fei Xia. 2012. Using a Goodness Measurement for Domain Adaptation: A Case Study on Chinese Word Segmentation. In *LREC*, pages 3853–3860.

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.

Weiwei Sun and Jia Xu. 2011. Enhancing Chinese Word Segmentation Using Unlabeled Data. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 970–979.

Xingwei Tan, Yi Cai, and Changxi Zhu. 2019. Recognizing Conflict Opinions in Aspect-level Sentiment Classification with Dual 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 3417–3422.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2016a. Effective LSTMs for Target-Dependent Sentiment Classification. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3298–3307.

Duyu Tang, Bing Qin, and Ting Liu. 2016b. Aspect Level Sentiment Classification with Deep Memory Network. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 214–224.

Hao Tang, Donghong Ji, Chenliang Li, and Qiji Zhou. 2020. Dependency Graph Enhanced Dual-transformer Structure for Aspect-based Sentiment Classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6578–6588.

Jialong Tang, Ziyao Lu, Jinsong Su, Yubin Ge, Linfeng Song, Le Sun, and Jiebo Luo. 2019. Progressive Self-Supervised Attention Learning for Aspect-Level Sentiment Analysis. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 557–566.

Yuanhe Tian, Guimin Chen, and Yan Song. 2021. Enhancing Aspect-level Sentiment Analysis with Word Dependencies. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*.

Yuanhe Tian, Yan Song, Xiang Ao, Fei Xia, Xiaojun Quan, Tong Zhang, and Yonggang Wang. 2020a. Joint Chinese Word Segmentation and Part-of-speech Tagging via Two-way Attentions of Auto-analyzed Knowledge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8286–8296.

Yuanhe Tian, Yan Song, and Fei Xia. 2020b. Joint Chinese Word Segmentation and Part-of-speech Tagging via Multi-channel Attention of Character N-grams. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2073–2084.

Yuanhe Tian, Yan Song, and Fei Xia. 2020c. Supertagging Combinatory Categorial Grammar with Attentive Graph Convolutional Networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6037–6044.

Yuanhe Tian, Yan Song, Fei Xia, and Tong Zhang. 2020d. Improving Constituency Parsing with Span Attention. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1691–1703.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30*, pages 5998–6008.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020. Relational Graph Attention Network for Aspect-based Sentiment Analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3229–3238.

Yequan Wang, Minlie Huang, Li Zhao, et al. 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.

Hu Xu, Bing Liu, Lei Shu, and S Yu Philip. 2019. BERT Post-Training for Review Reading Comprehension and Aspect-based Sentiment Analysis. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2324–2335.

Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. 2015. Classifying Relations via Long Short Term Memory Networks Along Shortest Dependency Paths. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1785–1794.

Wei Xue and Tao Li. 2018. Aspect Based Sentiment Analysis with Gated Convolutional Networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2514–2523.

Chen Zhang, Qiuchi Li, and Dawei Song. 2019a. Aspect-based Sentiment Classification with Aspect-specific Graph Convolutional 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 4560–4570.

Hongming Zhang, Yan Song, and Yangqiu Song. 2019b. Incorporating Context and External Knowledge for Pronoun Coreference Resolution. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 872–881.

Appendix

A. Hyper-parameter Settings

Table 5 reports the hyper-parameters tested in training our models. We test all combinations of them for each model and use the one achieving the highest accuracy score in our final experiments.

Hyper-parameters	**Values**
Learning Rate	$5e-6, 1e-5, \mathbf{2e-5}, 3e-5$
Warmup Rate	$0.06, \mathbf{0.1}$
Dropout Rate	$\mathbf{0.1}$
Batch Size	$8, \mathbf{16}, 32$
Max Input Length	$\mathbf{100}$

Table 5: The hyper-parameters tested in tuning our models, where the best ones used in our final experiments are highlighted in boldface.

B. Model Size and Running Speed

Table 6 reports the number of trainable parameters and the inference speed (sentences per second) of the baseline models (BERT) and our best performing models (i.e., T-GCN and ALE using BERT-large encoder with sentence-aspect pair input) on all datasets. All models are performed on an Nvidia Quadro RTX 6000 GPU.

C. Mean and Deviation of the Results

In our experiments, we run models using BERT-base and BERT-large encoders with different configurations, where models using single sentence input (S) or sentence-aspect pair input (P) as well as models using normal GCN (+ GCN) or T-GCN (+ T-GCN) are tested. For each model, we train it with the best hyper-parameter setting using five different random seeds. We report the mean (μ) and standard deviation (σ) of the experimental results (accuracy and F1 scores) on all datasets in Table 7.

D. Effect of T-GCN layer

In our ablation study, we run models with different configurations of type-aware graph (TG), attention (Att), and ALE, where three T-GCN layers are used. For settings without ALE (ID: 4-6 and 8), we also try different number of layers and report the results in Table 8, where similar trend is observed.

Models	Lap14		Rest14		Rest15		Rest16		Twitter		MAMS	
	Para.	Speed	Para.	Speed	Para.	Speed	Para.	Speed	Para.	Speed	Para.	Speed
BERT-Base	109.5M	37.1	109.5M	38.1	109.5M	37.3	109.5M	38.5	109.5M	38.2	109.5M	38.0
Full Model	114.8M	31.4	114.8M	30.9	114.8M	30.6	114.8M	29.6	114.8M	30.7	114.8M	30.2
BERT-Large	335.1M	20.0	335.1M	20.1	335.1M	20.5	335.1M	20.5	335.1M	19.6	335.1M	20.0
Full Model	344.6M	16.4	344.6M	17.4	344.6M	16.8	344.6M	17.1	344.6M	17.6	344.6M	17.3

Table 6: Numbers of trainable parameters (Para.) in different models and the inference speed (sentences per second) of these models on the test sets of all datasets.

Models	Lap14				Rest14				Rest15				Rest16				Twitter				MAMS			
	Acc		F1		Acc		F1		Acc		F1		Acc		F1		Acc		F1		Acc		F1	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
BERT-base (S)	77.61	0.74	72.85	0.74	82.45	0.41	72.62	0.74	81.48	0.61	64.00	0.64	89.05	0.46	69.04	1.94	73.89	0.24	72.20	0.63	78.85	0.15	79.29	0.20
+ GCN	78.95	0.37	74.71	0.62	85.39	0.36	78.72	0.64	83.65	0.75	69.66	1.93	91.37	0.63	74.99	1.14	75.05	0.71	73.39	0.90	80.51	0.33	80.22	0.31
+ T-GCN	**79.33**	0.68	**75.29**	1.00	85.86	0.30	79.19	0.61	**84.44**	0.75	69.61	2.34	91.61	0.53	76.63	1.03	75.48	0.75	74.00	0.89	81.80	0.23	81.20	0.31
BERT-base (P)	78.97	0.49	74,75	0.30	85.49	0.75	78.54	1.23	83.40	0.73	69.12	2.83	91.04	0.87	74.93	2.41	74.88	0.16	73.43	0.46	80.02	0.32	80.16	0.27
+ GCN	79.00	0.82	74.90	1.14	85.76	0.13	**79.62**	0.73	83.69	0.79	69.86	1.59	91.60	0.57	76.87	0.61	75.31	0.95	73.66	1.08	81.76	0.44	81.12	0.37
+ T-GCN	79.10	0.87	75.16	0.95	**86.19**	0.24	79.56	0.72	84.16	0.81	**69.95**	2.03	**92.36**	0.33	**77.70**	1.69	**75.65**	0.91	**74.40**	0.98	**83.09**	0.33	**82.41**	0.42
BERT-large (S)	78.11	0.30	74.00	0.67	82.35	0.87	71.75	1.77	82.03	0.51	68.04	0.39	89.30	0.90	70.24	1.39	74.47	0.87	73.16	0.72	79.17	0.26	79.71	0.32
+ GCN	80.77	0.58	77.56	0.69	86.87	0.51	81.00	0.50	84.93	0.58	69.83	1.49	92.18	0.76	76.84	1.90	75.39	0.47	73.95	0.42	81.30	0.13	80.89	0.18
+ T-GCN	81.00	0.98	77.89	0.96	87.02	0.10	81.26	0.16	85.57	0.35	69.73	1.07	92.23	0.18	76.78	1.48	75.52	0.34	74.16	0.45	82.51	0.22	82.05	0.18
BERT-large (P)	80.06	0.48	76.50	0.67	86.17	0.33	79.10	0.76	81.59	1.87	59.23	7.40	89.29	2.04	65.32	9.02	75.34	0.31	74.21	0.53	80.45	0.35	80.62	0.30
+ GCN	81.22	0.50	77.15	0.75	86.83	0.22	80.18	0.32	85.26	0.56	68.37	1.50	92.21	0.65	77.86	1.68	75.65	0.30	74.25	0.75	82.60	0.11	82.19	0.19
+ T-GCN	**81.37**	0.68	**77.94**	0.92	**87.11**	0.25	**81.33**	0.85	**85.89**	0.72	**70.06**	2.53	**92.74**	0.28	**78.60**	1.33	**77.73**	0.41	**77.01**	0.48	**83.56**	0.25	**82.91**	0.21

Table 7: The mean μ and standard deviation σ of accuracy and F1 scores of all models on six benchmark datasets. "GCN" refers to the normal GCN model without using type-aware graph, attention mechanism and ALE. "S" and "P" refer to the settings that the input is a single sentence and a sentence-aspect pair, respectively.

	Setting			Lap14		Rest14		Rest15		Rest16		Twitter		MAMS	
	TG	Att	ALE	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
4	√	√	×	81.50	78.42	87.35	82.17	85.44	72.32	92.48	78.26	76.58	75.53	83.21	82.53
5	√	×	×	81.21	77.79	87.26	81.64	86.13	73.96	92.42	75.69	74.88	73.89	83.18	82.39
6	×	√	×	81.32	77.75	87.21	81.43	86.19	73.78	92.43	75.87	75.17	73.91	83.33	82.56
8	×	×	×	80.88	77.63	86.84	81.07	85.85	72.11	91.89	75.18	75.04	73.99	82.26	82.13

(a) 1 T-GCN layer

	Setting			Lap14		Rest14		Rest15		Rest16		Twitter		MAMS	
	TG	Att	ALE	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
4	√	√	×	81.34	77.84	87.31	81.57	86.19	73.99	92.48	75.64	75.00	73.96	83.18	82.47
5	√	×	×	81.13	77.68	87.16	81.42	86.08	73.86	92.38	75.61	74.90	73.82	83.08	82.32
6	×	√	×	81.38	77.85	87.35	81.58	86.23	73.98	92.58	75.89	75.08	73.99	83.28	82.52
8	×	×	×	80.83	77.57	86.68	81.02	85.88	72.03	91.92	75.24	75.00	73.96	82.18	82.07

(b) 2 T-GCN layers

Table 8: Experimental results of ablation study on the six datasets, with different configurations applied to our best model without attentive layer ensemble (ALE). 'TG" refers to the type-aware graph; "Att" denotes the attention mechanism in T-GCN. "√" and "×" represent if a corresponding component is used or not. (a) reports the results where only 1 T-GCN layer is used; (b) reports the results where 2 T-GCN layers are used.