To be Closer: Learning to Link up Aspects with Opinions

Yuxiang Zhou[1*], **Lejian Liao**[1], **Yang Gao**[1†], **Zhanming Jie**[2,3] and **Wei Lu**[2]
[1]School of Computer Science and Technology, Beijing Institute of Technology
[2]StatNLP Research Group, Singapore University of Technology and Design
[3]ByteDance AI Lab
yxzhou@bit.edu.cn, liaolj@bit.edu.cn, gyang@bit.edu.cn
allan@bytedance.com, luwei@sutd.edu.sg

Abstract

Dependency parse trees are helpful for discovering the opinion words in aspect-based sentiment analysis (ABSA) (Huang and Carley, 2019). However, the trees obtained from off-the-shelf dependency parsers are static, and could be sub-optimal in ABSA. This is because the syntactic trees are not designed for capturing the interactions between opinion words and aspect words. In this work, we aim to shorten the distance between aspects and corresponding opinion words by learning an aspect-centric tree structure. The aspect and opinion words are expected to be closer along such tree structure compared to the standard dependency parse tree. The learning process allows the tree structure to adaptively correlate the aspect and opinion words, enabling us to better identify the polarity in the ABSA task. We conduct experiments on five aspect-based sentiment datasets, and the proposed model significantly outperforms recent strong baselines. Furthermore, our thorough analysis demonstrates the average distance between aspect and opinion words are shortened by at least 19% on the standard SemEval Restaurant14 (Pontiki et al., 2014) dataset[1].

1 Introduction

Aspect-based sentiment analysis (ABSA) (Pang et al., 2008; Liu, 2012) aims at determining the sentiment polarity expressed towards a particular target in a sentence. For example, in the sentence *"The **battery life** of this laptop is very long, but the **price** is too high"*, the sentiment expressed towards the aspect term *"battery life"* is positive, whereas the sentiment towards the aspect term *"price"* is negative. Early research efforts (Wang et al., 2016; Chen et al., 2017; Liu and Zhang, 2017; Li and Lu, 2019; Xu et al., 2020a) focus on using an attention mechanism (Bahdanau et al., 2015) to model interactions between aspect and context words. However, such attention-based models may suffer from overly focusing on the frequent words that express sentiment polarity while ignoring low-frequency ones (Tang et al., 2019; Sun and Lu, 2020). Recent efforts show that the syntactic structures of sentences can facilitate the identification of sentiment features related to aspect words (Zhang et al., 2019; Sun et al., 2019b; Huang and Carley, 2019). Nonetheless, these methods unfortunately suffer from two shortcomings. First, the trees obtained from off-the-shelf dependency parsers are static, and thus cannot adaptively model the complex relationship between multiple aspects and opinion words. Second, an inaccurate parse tree could lead to error propagation downstream in the pipeline. Several research groups have explored the above issues with a more refined parse tree. For example, Chen et al. (2020) constructed task-specific structures by developing a gate mechanism to dynamically combine the parse tree information and a stochastic graph sampled from the HardKuma distribution (Bastings et al., 2019). On the other hand, Wang et al. (2020) greedily reshaped the dependency parse tree by using manual rules to

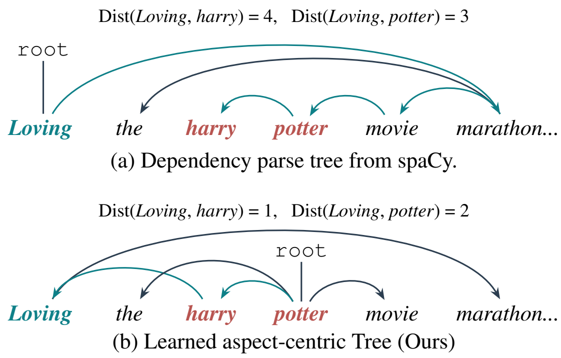

Figure 1: An example with different tree representations in the Twitter dataset. "Dist" returns the number of hops between two words in the tree. Words marked in red and blue are aspect and opinion, respectively.

[*]Work done when visiting SUTD.
[†]Corresponding author.
[1]Our code is available at https://github.com/zyxnlp/ACLT.

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 3899–3909
November 7–11, 2021. ©2021 Association for Computational Linguistics

obtain the aspect-related syntactic structures.

Despite being able to effectively alleviate the tree representation problem, existing methods still depend on external parse trees, leading to one potential problem. The dependency parse trees are not designed for the purpose of ABSA but to express syntactic relations. Specifically, the aspect term is usually a noun or a noun phrase, while the root of the dependency tree is often a verb or an adverb. According to statistics, for almost 90%[2] of the sentences, the roots of their dependency trees are not aspect words. Such a *root inconsistency* issue may prevent the model from effectively capturing the relationships between opinion words and aspect words. For example, Figure 1(a) shows the dependency tree obtained by the toolkit spaCy[3]. The root is the gerund verb "*Loving*" while the aspect term is the noun phrase "*harry potter*". The distance between the aspect words "*harry*" and "*potter*" and the critical opinion word "*Loving*" under a dependency tree are four hops and three hops, respectively. However, their relative distances in the sequential order are two and three, respectively. Intuitively, closer distance enables us to identify the polarity in the ABSA task better. Figure 1(b) shows an aspect-centric tree where the tree is rooted by the aspect words. The distances between aspect and opinion words are one hop and two hops, which is closer than the distance in the standard dependency parse tree.

In this paper, we propose a model that learns *Aspect-Centric Latent Trees* (we name it as the ACLT model) which are specifically tailored for the ABSA task. We assume that inducing tree structures whose roots are within aspect term enables the model to correlate the aspect and opinion words better. We built our model based on the structure attention mechanism (Kim et al., 2017; Liu and Lapata, 2018) and a variant of the Matrix-Tree Theorem (MTT) (Tutte, 1984; Koo et al., 2007). Additionally, we proposed to impose a soft constraint to encourage the aspect words to serve as the root of the tree structure induced by MTT. As a result, the search space of inferring the root is reduced during the training process.

Our contributions are summarized as follows:

- We propose to use *Aspect-Centric Latent Trees* (ACLT) which are specifically tailored for the ABSA task to link up aspects with opinion words in an end-to-end fashion.
- Our ACLT model is able to learn an aspect-centric latent tree with a root refinement strategy to better correlate the aspect and opinion words than the standard parse tree.
- Experiments show that our model outperforms the existing approaches, and also yields new state-of-the-art results on four ABSA benchmark datasets. Quantitative and qualitative experiments further justify the effectiveness of the learned aspect-centric trees. The analysis demonstrates that our ACLT is capable of shortening the average distances between aspect and opinion words by at least 19% on the standard SemEval Restaurant14 dataset. To the best of our knowledge, we are the first to link up aspects with opinions through the specifically designed latent tree that imposes root constraints.

2 Model

In this section, we present the proposed Aspect-Centric Latent Tree (ACLT) model (Figure 2) for the ABSA task. We first obtain the contextualized representations from the sentence encoder. Next, we use a tree inducer to produce the distribution over all the possible latent trees. The underlying tree inducer is a latent-variable model which treats tree structures as the latent variable. Once we have the distribution over the latent trees, we adopt the root refinement procedure to obtain aspect-centric latent trees. Then, we can encode the probabilistic latent trees with a graph or tree encoder. Finally, we use the structured representation from the tree encoder for sentiment classification.

2.1 Sentence Encoder

Given a sentence $\boldsymbol{s} = [w_1, ..., w_n]$ and the corresponding aspect term $\boldsymbol{a} = [w_i, ..., w_j]$ $(1 \leq i \leq j \leq n)$, we adopt the pre-trained language model BERT (Devlin et al., 2019) to obtain the contextualized representation for each word. We concatenate the words in the sentence and explicitly present the aspect term in the input representation: $\mathbf{x} = (\texttt{[CLS]}\ w_1, ..., w_n\ \texttt{[SEP]}\ w_i, ..., w_j\ \texttt{[SEP]})$. The contextualized representation $\boldsymbol{H}$ can be obtained via $\text{BERT}(\mathbf{x})$, where $\boldsymbol{H} = [\boldsymbol{h}_1, ..., \boldsymbol{h}_n]$, $\boldsymbol{h}_i \in \boldsymbol{H}$ represents the contextualized representation of the i-th token.

[2]The detail statistic can be found in Appendix A.

[3]https://spacy.io/api/dependencyparser

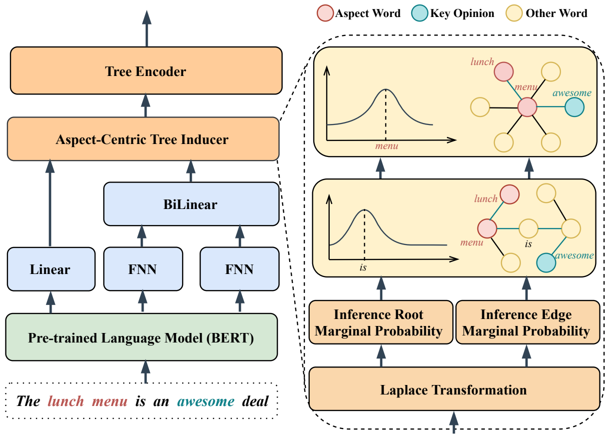

Figure 2: ACLT architecture.

2.2 Aspect-Centric Tree Inducer

While prior efforts (Wang et al., 2020; Chen et al., 2020) on learning latent (or explicit) trees for the ABSA task exist, one of the major contributions of our work is that we link up aspects and opinion words by addressing the root inconsistency issue. Inspired by recent work (Liu and Lapata, 2018; Nan et al., 2020), we use a variant of Kirchhoff's Matrix-Tree Theorem (Tutte, 1984; Koo et al., 2007) to induce the latent dependency structure.

Given the contextualized representation $\boldsymbol{h} \in \mathbb{R}^d$ of each node (token) in the sentence, where d is the dimension of the node representations. We first calculate pair-wise unnormalized edge scores $\boldsymbol{e}_{ij}$ between the i-th and the j-th node with the node representation $\boldsymbol{h}_i$ and $\boldsymbol{h}_j$ by way of a two feed-forward neural network (FNN) and a bilinear function:

$$\boldsymbol{e}_{ij} = \left(\tanh(\boldsymbol{W}_p \boldsymbol{h}_i)\right)^T \boldsymbol{W}_b (\tanh(\boldsymbol{W}_c \boldsymbol{h}_j)), \quad (1)$$

where $\boldsymbol{W}_p \in \mathbb{R}^{d\times d}$ and $\boldsymbol{W}_c \in \mathbb{R}^{d\times d}$ are weights for two feedforward neural networks, tanh is applied as the activation function. $\boldsymbol{W}_b \in \mathbb{R}^{d\times d}$ is the weight for the bilinear transformation. $\boldsymbol{e}_{ij} \in \mathbb{R}^{d\times d}$ can be viewed as a weighted adjacency matrix for a graph G with n nodes where each node corresponds to a word in the sentence.

Next, we calculate the root score $\boldsymbol{r}_i$, representing the unnormalized probability of the i-th node to be selected as the root of the structure:

$$\boldsymbol{r}_i = \boldsymbol{W}_r \boldsymbol{h}_i, \quad (2)$$

where $\boldsymbol{W}_r \in \mathbb{R}^{1\times d}$ is the weight for the linear transformation. Following Koo et al. (2007), we calculate the marginal probability of the dependency edge of the latent structure:

$$\boldsymbol{A}_{ij} = \begin{cases} 0 & \text{if } i = j \\ \exp\left(\boldsymbol{e}_{ij}\right) & \text{otherwise} \end{cases} \quad (3)$$

$$\boldsymbol{L}_{ij} = \begin{cases} \sum_{i'=1}^{n} \boldsymbol{A}_{i'j} & \text{if } i = j \\ -\boldsymbol{A}_{ij} & \text{otherwise} \end{cases} \quad (4)$$

$$\bar{\boldsymbol{L}}_{ij} = \begin{cases} \boldsymbol{L}_{ij} + \exp\left(\boldsymbol{r}_i\right) & \text{if } i = j \\ \boldsymbol{L}_{ij} & \text{otherwise,} \end{cases} \quad (5)$$

where we first assign non-negative weights $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ to the edges, $\boldsymbol{A}_{ij}$ is the weight of the edge between the i-th and the j-th node. Then, we build the Laplacian matrix $\boldsymbol{L} \in \mathbb{R}^{n\times n}$ for graph G and its variant $\bar{\boldsymbol{L}}$ which takes the root node into consideration for further computation (Koo et al., 2007). We use $\boldsymbol{P}_{ij}$ to denote the marginal probability of the dependency edge between the i-th and the j-th node, and $\boldsymbol{P}_i^r$ is defined as the marginal probability of the i-th word headed by the root of the tree. Then, $\boldsymbol{P}_{ij}$ and $\boldsymbol{P}_i^r$ can be derived:

$$\begin{aligned} \boldsymbol{P}_{ij} &= \left(1 - \delta_{1,j}\right) \boldsymbol{A}_{ij} \left[\bar{\boldsymbol{L}}^{-1}\right]_{jj} \\ &\quad - \left(1 - \delta_{i,1}\right) \boldsymbol{A}_{ij} \left[\bar{\boldsymbol{L}}^{-1}\right]_{ji} \end{aligned} \quad (6)$$

$$\boldsymbol{P}_i^r = \exp\left(\boldsymbol{r}_i\right) \left[\bar{\boldsymbol{L}}^{-1}\right]_{i1}, \quad (7)$$

where δ is the Kronecker delta. Here, $\boldsymbol{P} \in \mathbb{R}^{n\times n}$ can be interpreted as a weighted adjacency matrix of the word-level graph. We refer the interested reader to Koo et al. (2007) for more details.

Root Refinement Despite the successful application of tree information induced by MTT in previous works (Liu and Lapata, 2018; Guo et al., 2020; Nan et al., 2020), unfortunately, the MTT would still produce arbitrary trees which inappropriate for the specific task if there is no structure supervision. Under the assumption that inducing tree structures whose roots are within aspect term enables the model to better correlate the aspect and opinion words than the standard parse tree, we proposed to impose a soft constraint to encourage the aspect words $w \in \boldsymbol{a}$ to serve as the root of tree structure induced by MTT.

Specifically, we introduce a cross-entropy loss for this assumption:

$$\begin{aligned} \mathcal{L}_a = -\sum_{i=1}^{L} \big(& t_i \log(\boldsymbol{P}_i^r) \\ &+ (1 - t_i) \log(1 - \boldsymbol{P}_i^r)\big), \end{aligned} \quad (8)$$

where $t_i \in \{0, 1\}$ indicates whether the i-th token is the aspect word, $\boldsymbol{P}_i^r$ is the probability of the i-th token being the root from Equation 7. The nice property of this loss is that minimizing the loss is essentially adjusting the aspect words to be the root in the latent trees. On the other hand, this supervision reduce the search space of inferring root for MTT in the training process.

Intuitively, the tree inducer module produces a random structure at early iterations during training since information propagates mostly between neighboring nodes. As the roots are adjusted to the aspect words and the structure gets more refined when the loss becomes smaller, the tree inducer is more likely to generate an aspect-centric latent structure. Our experiment in Section 3.4 shows that the root refinement loss (Equation 8) is able to successfully guide the inducing of latent trees, in which the aspect word is consistent with its root.

2.3 Tree Encoder

Given contextualized representation $\boldsymbol{h}$ and the corresponding aspect-centric graph $\boldsymbol{P}$, we follow Kim et al. (2017) and Liu and Lapata (2018) to encode the tree information by structure attention mechanism:

$$\begin{aligned} \boldsymbol{s}_i^p &= \sum_{k=1}^{n} \boldsymbol{P}_{ki}\boldsymbol{h}_k + \boldsymbol{P}_i^r \boldsymbol{h}_a \\ \boldsymbol{s}_i^c &= \sum_{k=1}^{n} \boldsymbol{P}_{ik}\boldsymbol{h}_i \\ \boldsymbol{s}_i &= \tanh\left(\boldsymbol{W}_s\left[\boldsymbol{s}_i^p, \boldsymbol{s}_i^c, \boldsymbol{h}_i\right]\right), \end{aligned} \tag{9}$$

where $\boldsymbol{s}_i^p \in \mathbb{R}^d$ is the context representation gathered from possible parents of $\boldsymbol{h}_i$, $\boldsymbol{s}_i^c \in \mathbb{R}^d$ is the context representation gathered from possible children, and $\boldsymbol{h}_a$ is the representation for the root node. We concatenate $\boldsymbol{s}_i^p$, $\boldsymbol{s}_i^c$ with $\boldsymbol{h}_i$ and transform with weights $\boldsymbol{W}_s \in \mathbb{R}^{d\times 3d}$ to obtain the structured representation of the i-th word $\boldsymbol{s}_i$.

2.4 Classifier

Following Xu et al. (2019) and Sun et al. (2019a), we leverage $\boldsymbol{s}_0$, which is the structured aspect-aware representation of each sentence, to compute the probability over the different sentiment polarities as:

$$y_p = \mathrm{softmax}\left(\boldsymbol{W}_p \boldsymbol{s}_0 + \boldsymbol{b}_p\right), \tag{10}$$

where $\boldsymbol{W}_p$ and $\boldsymbol{b}_p$ are model parameters for the classifier, and y_p is the predicted sentiment probability distribution.

Dataset	Train			Dev			Test		
	#Pos.	#Neu.	#Neg.	#Pos.	#Neu.	#Neg.	#Pos.	#Neu.	#Neg.
Lap14	895	418	783	99	46	87	341	169	128
Rest14	1,948	573	726	216	64	81	728	196	196
Rest15	821	32	230	91	4	26	326	34	182
Rest16	1,116	62	395	124	7	44	469	30	117
Twitter	1,405	2,814	1,404	156	313	156	173	346	173

Table 1: Statistics of datasets.

The objective of the classifier is to minimize the cross-entropy loss for an instance $(\mathbf{x}, y)$:

$$\mathcal{L}_s = -\log P(y|\mathbf{x}) \tag{11}$$

where $y \in \{positive, negative, neutral\}$. Our final objective function is a multi-task learning objective, defined as weighted sum of the loss on root refinement and classification:

$$\mathcal{L} = \alpha\mathcal{L}_a + (1-\alpha)\mathcal{L}_s, \tag{12}$$

where $\alpha \in (0, 1)$ is a coefficient that balances the contribution of each component in the training process. The hyper-parameter α is selected based on the performance on the validation set.

3 Experiments

3.1 Experimental Setup

We evaluate our proposed ACLT model on five benchmark datasets: the Laptop (Lap14) and Restaurant (Rest14) review datasets from SemEval 2014 Task4 (Pontiki et al., 2014), the Restaurant15 (Rest15) review dataset from SemEval 2015 Task12 (Pontiki et al., 2015), the Restaurant16 (Rest16) review dataset from SemEval 2016 Task5 (Pontiki et al., 2016), and Twitter posts from (Dong et al., 2014). Following the previous works (Tang et al., 2016; Chen et al., 2017; Wang and Lu, 2018), we remove a few examples that have conflicting labels. We randomly split 10% of data from the training dataset as the development dataset, and the model is only trained with the remaining data. Detailed statistics of the datasets can be found in Table 1. All hyper-parameters are tuned based on the development set[4]. We employed the uncased version of the BERT-base (Devlin et al., 2019) model in PyTorch (Wolf et al., 2020)[5]. Following previous conventions, we repeat each experiment three times and average the results, reporting accuracy (Acc.) and macro-f1 (F_1).

[4]We list some of the important hyper-parameters in Appendix B.

[5]https://github.com/huggingface/transformers

3.2 Baselines

The state-of-the-art baselines selected for comparison fall into three main categories: Syntax information free models, dependency parse tree based models, and latent tree based models. Syntax information free models include:

- **TNet-AS** Li et al. (2018) implements a context-preserving mechanism to get the aspect-specific representations.
- **BERT-PT** Xu et al. (2019) explores a novel post-training approach on BERT to enhance the performance of BERT which has been fine-tuned for ABSA and RRC.
- **BERT-PAIR** Sun et al. (2019a) constructs auxiliary sentences from the aspect and converts the ABSA task to a sentence-pair classification task.
- **BERT-SRC** (Devlin et al., 2019) is the vanilla BERT model which directly uses the last layer's `[CLS]` representation of the model as a classification feature.

The dependency parse tree based models are:

- **ASGCN** Zhang et al. (2019) uses GCNs to capture the long-range dependencies between words.
- **CDT** Sun et al. (2019b) uses GCNs to integrate dependency parse tree information.
- **BiGCN** Zhang and Qian (2020) uses syntactic graph and lexical graph to capture the global word co-occurrence information.
- **ASGCN+BERT** is a baseline that uses BERT instead of BiLSTM as the context encoder of ASGCN (Zhang et al., 2019).
- **R-GAT+BERT** (Wang et al., 2020) is a dependency tree based model that greedily reshapes the dependency parse tree using manually defined rules.

A latent tree based model:

- **KumaGCN+BERT** Chen et al. (2020) constructs syntactic information by developing a gate mechanism to combine HardKuma structure and dependency parse tree.

We reproduce the results for baselines whenever the authors provide the source code. For ASGCN+BERT and KumaGCN+BERT models where the code is not made available as of this writing, we implement them by ourselves using the optimal hyper-parameter setting reported in their paper. Since we randomly split 10% of data from the training dataset as the development dataset, and the model is only trained with the remaining data, the results of R-GAT+BERT (Wang et al., 2020) and KumaGCN+BERT (Chen et al., 2020) are lower than which reported in the original paper. In our experiments, we report the average result and the mean absolute deviations over three runs with the random initialization. We stop training when iterations reached the maximum of 30 epochs.

3.3 Main Results

As shown in Table 2, dependency tree based models and latent tree based models generally achieve better results than syntax information free models, suggesting that syntactic information indeed benefits the ABSA task and enables it to achieve promising results.

Our model consistently outperforms the models which do not use any syntactic information. For example, ACLT improves upon the BERT-SRC model by 3.56 points in terms of F_1 on the Lap14 dataset, which suggests that our proposed model is able to induce an effective latent tree for ABSA in an end-to-end fashion. In particular, with the exception of R-GAT+BERT on the Rest14 dataset in terms of F_1, our model surpassed all compared models by a significant margin. For example, our model achieves 72.08 and 78.64 F_1 on the Rest15 and Rest16 datasets, which significantly outperform the current state-of-the-art model KumaGCN+BERT, under the same setting. The statistics empirically show that compared to the models that use syntactic information, ACLT can induce a more informative latent task-specific structure to establish effective connections between aspect words and context. Our ACLT model also shows its superiority over all baselines in terms of accuracy.

Does ACLT shorten the distances between aspect and opinion words?

To gain further insight on the relationship between aspect and opinion words in the text, we inspect the distance just between aspect words and selected opinion words. Specifically, we first selected the top five most frequent positive and negative opinion words, respectively, in the Rest14 dataset. We define the distance between the aspect and opinion words to be the number of interaction hops between them. Thus we can calculate the distance between these opinion words and aspect words in a parse tree[6] and an aspect-centric tree, respectively.

[6]We use Chu-Liu-Edmonds' algorithm to extract the aspect-centric trees. More detail can be found in section 3.5.

Models	Tree	Lap14		Rest14		Rest15		Rest16		Twitter	
		Acc	F_1	Acc	F_1	Acc	F_1	Acc	F_1	Acc	F_1
TNet-AS♮	None	76.54	71.75	80.69	71.27	-	-	-	-	74.97	73.60
BERT-PT♮	None	78.07	75.08	84.95	76.96	-	-	-	-	-	-
BERT-PAIR♮	None	78.99	75.03	84.46	76.98	-	-	-	-	-	-
BERT-SRC	None	77.59±0.18	72.27±0.02	85.27±0.28	77.61±0.38	81.73±0.45	66.22±0.43	90.91±0.07	76.29±0.76	73.12±0.29	72.29±0.25
ASGCN♮	Dependency	75.55	71.05	80.77	72.02	79.89	61.89	88.99	67.48	72.15	70.40
CDT♮	Dependency	77.19	72.99	82.30	74.02	-	-	85.58	69.93	74.66	73.66
BiGCN♮	Dependency	74.59	71.84	81.97	73.48	81.16	64.79	88.96	70.84	74.16	73.35
ASGCN+BERT	Dependency	77.90±0.10	73.01±0.14	83.78±0.22	75.02±0.51	80.69±045	62.02±0.39	89.99±0.58	74.46±0.16	72.78±0.71	71.76±0.64
R-GAT+BERT*	Dependency	78.53±0.31	74.63±0.35	85.63±0.24	**78.82±0.54**	81.61±0.78	65.30±0.22	90.96±0.18	75.26±0.39	73.80±0.61	72.63±0.46
KumaGCN+BERT‡	Latent	79.57±0.28	75.61±0.28	84.91±0.30	77.22±0.37	82.10±0.62	65.56±0.61	90.80±0.47	74.93±0.97	74.33±0.32	73.42±0.31
ACLT	Latent	**79.68±0.38**	**75.83±0.03**	**85.71±0.06**	78.44±0.09	**84.44±0.08**†	**72.08±0.08**†	**92.15±0.14**†	**78.64±0.19**†	**75.48±0.16**†	**74.51±0.32**†

Table 2: Main Results (%). The results of model with the symbol ♮ are retrieved from the original paper, and those with the * symbol are computed based on their open implementations. ‡ denotes the model using both the dependency parse tree and the latent tree. The best results on each dataset are in bold. The second-best ones are underlined. The † marker refers to p-value < 0.05 in comparison with the second-best results.

	Positive opinion words				
Tree	*great*	*good*	*excellent*	*fresh*	*delicious*
Parser	4.38	4.50	5.11	8.02	6.91
MTT	3.84	4.47	5.05	6.61	4.43
ACLT	**2.81**	**3.40**	**4.08**	**4.84**	**3.57**
	Negative opinion words				
Tree	*rude*	*small*	*bad*	*awful*	*worst*
Parser	6.67	11.27	9.44	4.00	3.88
MTT	5.87	10.18	8.56	4.00	3.75
ACLT	**3.27**	**6.45**	**3.89**	**2.89**	**3.13**

Table 3: The average distances (lower is better) between the top five opinion words and aspect words.

Table 3 presents various statistics for the average distance of aspect and opinion words in the trees produced by spaCy dependency parser (**Parser**), the Matrix Tree Theory without specific root refinements (**MTT**), and our model (**ACLT**). As can be seen, in our aspect-centric latent tree, the average distance between opinion words and aspect words is shorter than those in dependency parse tree and MTT. We also observe that without the root refinement, the average distance between opinion words and aspect words in MTT is roughly equivalent to the parse tree. These results confirm our hypothesis that inducing tree structures whose roots are within aspect term enables the model to better correlate the aspect and opinion words than the standard parse tree.

3.4 Model Analysis

Effect of different tree representations

Our proposed aspect-centric latent tree, the latent Matrix tree, and the standard dependency parse tree all represent the structure of a sentence. Nevertheless, the differences between them and how they directly affect the aspect-based results need to be further investigated. In this section, we first use BERT-base as a contextual encoder, then use GCN to encode dependency parse tree information (Parser+GCN), latent Matrix tree information (MTT+GCN), latent Kuma structure (Kuma+GCN[7]) and our aspect-centric tree information (ACLT+GCN). Table 4 summarizes the results.

We observe that models incorporated with syntactic information generally outperform the vanilla BERT-SRC, indicating that syntactic information benefits the ABSA task. Such a phenomenon can also be observed in other fundamental NLP tasks (Jie and Lu, 2019; Xu et al., 2021). Moreover, we also found that both our ACLT and ACLT+GCN model consistently outperform models equipped with other dependency trees by a significant margin. These results demonstrate that the aspect-centric tree induced by our model is indeed capable of effectively building relationships between aspect and context words for the ABSA task. Under the same setting, ACLT+GCN outperforms Parser+GCN, MTT+GCN, and Kuma+GCN on all the datasets. In particular, our ACLT+GCN obtains 1.8, 2.6, and 7 points improvement over Parser+GCN, MTT+GCN, and Kuma+GCN on Rest15 in terms of F_1. Moreover, ACLT+GCN outperforms ACLT on the Rest14 and the Twitter datasets, indicating using a GCN as a tree encoder can boost the model performance to a certain extent.

We also have similar observations for our ACLT model under the setting of accuracy. These experimental results demonstrate that our proposed aspect-centric latent tree is a more effective structure for ABSA, compared to the parse tree. Interestingly, we observe that BERT cannot achieve

[7]For a fair comparison, we only use the Kuma structure rather than combining the dependency tree and the Kuma structure in this experiment.

Models	Lap14		Rest14		Rest15		Rest16		Twitter	
	Acc.	F_1	Acc.	F_1	Acc.	F_1	Acc.	F_1	Acc.	F_1
BERT-SRC	77.6	72.3	85.3	77.6	81.7	66.2	90.9	76.3	73.1	72.3
Parser+GCN	78.0	73.6	85.3	77.6	83.0	68.4	91.0	74.9	74.0	73.3
MTT+GCN	78.9	74.7	84.7	76.3	81.9	67.6	91.2	74.8	75.3	74.3
Kuma+GCN	78.1	73.5	85.3	77.9	80.3	63.2	90.4	75.2	74.6	73.7
ACLT+GCN	78.6	74.3	**86.3**	**79.4**	83.1	70.2	91.8	76.7	**75.6**	**74.7**
ACLT	**79.7**	**75.8**	85.7	78.4	**84.4**	**72.1**	**92.2**	**78.6**	75.5	74.5

Table 4: The performance of BERT with our aspect-centric latent tree vs. BERT with other tree structures.

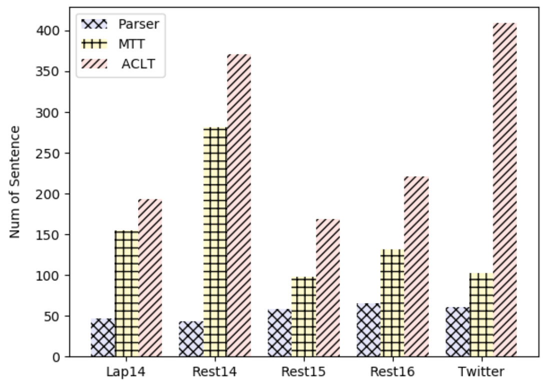

Figure 3: The number of sentences where the aspect words are roots under three different types of trees.

Models	Rest14		Twitter	
	Acc.	F_1	Acc.	F_1
BERT-SRC	85.27	77.61	73.12	72.29
ACLT (Entire Tree)	85.71	78.44	75.48	74.51
Pruned Tree ($k = 1$)	85.27	77.37	74.56	73.75
Pruned Tree ($k = 2$)	84.91	77.05	73.84	72.72
R-GAT+BERT (Entire Tree)	85.63	78.82	73.80	72.63
Pruned Tree ($k = 1$)	85.71	79.14	74.71	73.85
Pruned Tree ($k = 2$)	84.73	78.67	73.99	73.16
KumaGCN+BERT (Entire Tree)	84.91	77.22	74.33	73.42
Pruned Tree ($k = 1$)	84.91	76.73	75.14	73.90
Pruned Tree ($k = 2$)	85.09	77.35	75.43	74.42

Table 5: The results of ACLT, R-GAT+BERT and KumaGCN+BERT with different tree pruning. k=1: only keep the first-order edges to the aspect. k=2: keep both the first-order and second-order edges.

a promising result on all datasets when introduced with the parse tree structure. For example, Parser+GCN drops 1.4 points in F_1 on the Rest16 dataset in comparison with vanilla BERT-SRC. This suggests that a dependency parse tree structure may not be able to capture the complicated interactions between aspect and opinion words effectively.

Did root refinement work?

We quantify the effectiveness of root refinement that adjusts the aspect words to be the root. We experiment with three different structures, including the dependency parse tree obtained by spaCy (Parser), the tree directly induced by MTT without specific root refinements (MTT), and the aspect-centric tree induced by our model (ACLT). Figure 3 shows the number of sentences where the aspect word is consistent with its root under three different tree structures in each dataset. Compared to the other two tree structures, we observe that the roots of our learned trees are consistent with the aspect words in most sentences. For example, in the Rest16 test dataset, there are 421 sentences in which the aspect words are consistent with the root words using the ACLT model. These results demonstrate that the problem of inconsistency between root and aspect has come close to being solved with our ACLT model.

Effect of tree pruning

To further investigate the effect of different tree structures on model performance, we examine ACLT, R-GAT+BERT, and KumaGCN+BERT with different tree pruning. More specifically, for R-GAT+BERT using the standard prase tree, we discard the dependency relation beyond first-order (k=1) and second-order (k=2) to aspects, respectively. Following Guo et al. (2020), we mask the information of the adjacency matrix $\boldsymbol{P}$ (Equation 6) that is beyond first-order (k=1) and second-order (k=2) with respect to the aspect for KumaGCN+BERT and our ACLT model. As shown in Table 5, on the Twitter dataset, our ACLT yields the best performance with the entire tree, outperforming the first-order pruned tree and second-order pruned tree by 0.76 and 1.79 points in terms of F_1, respectively. This indicates it is necessary to induce

Models	Lap14		Rest14		Rest15		Rest16		Twitter	
	Acc.	F_1	Acc.	F_1	Acc.	F_1	Acc.	F_1	Acc.	F_1
Full Model	79.7	75.8	85.7	78.4	84.4	72.1	92.2	78.6	75.5	74.5
w/o Root Refinement	77.8	73.2	84.5	76.4	81.5	66.9	89.8	73.8	73.4	72.6
w/o Latent Tree	77.8	73.4	84.7	76.7	83.9	69.1	91.5	77.6	74.0	73.3
w Fixed Root	79.2	75.0	84.6	76.3	83.2	68.9	91.1	75.0	75.0	74.1

Table 6: Ablation study of ACLT on various datasets. w/o and w indicate without and with, resepectively. Fixed Root means the tree's root is fixed on the first word of aspect term (Wang et al., 2020).

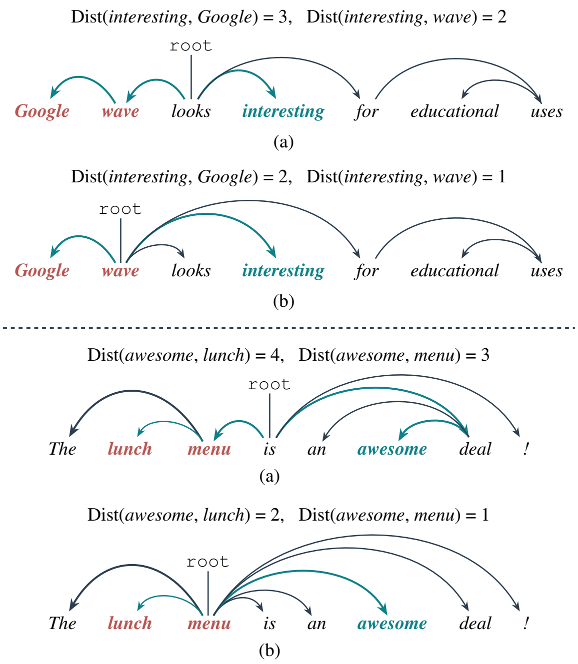

Figure 4: Two examples from the Twitter and Rest16 dataset to illustrate the difference between a dependency parse tree (a) and an aspect-centric tree (b). Red words indicate the aspect words of the sentence.

an entire aspect-centric latent tree rather than its pruned subtree in our model. Interestingly, we observe that R-GAT+BERT and KumaGCN+BERT achieve the best results in cases of Pruned Tree ($k = 1$) and Pruned Tree ($k = 2$), respectively. It is likely because that both R-GAT+BERT and KumaGCN+BERT rely on the parse tree. Nevertheless, only a small part of the standard parse tree is related to the ABSA task. Introducing the entire tree may prevent the model from effectively capturing the relationships between opinion words and aspect words.

Ablation Study

We conducted experiments to examine the effectiveness of the major components of our ACLT model, and Table 6 shows the ablation results on the five datasets we used. We observe that both latent tree and root refinement component contribute to the main model. Specifically, with removal of the root refinement module, performance of ACLT drops considerably, leading to a 5.2 and 4.8 decrease, in terms of F_1, on the Rest15 dataset and the Rest16 dataset, respectively. This result illustrates that refining root to aspect words plays a crucial role in learning a task-specific latent structure for ABSA. The performance drop on fixed root indicates that computing each aspect word's probability to become the root is essential for achieving good performance.

3.5 Case Study

To gain further insight on our induced aspect-centric tree, we use Chu-Liu-Edmonds' algorithm (Edmonds, 1967) to extract the aspect-centric trees, where each tree is expressed by a weighted adjacency matrix as shown in Equation (7). We selected two examples from the Twitter and Rest16 datasets, whose sentiments can be correctly predicted by our ACLT model. Overall we observe that aspect-centric trees differ from the standard dependency trees in the types of dependencies they create which tend to be of shorter length.

Specifically, as shown in Figure 4 top (a), the root of the dependency parse tree is the word "*looks*" which is inconsistent with the aspect word "*google*" or "*wave*", and the key opinion word "*interesting*" requires three-hop and two-hop interactions to establish a connection with each of the two aspect words respectively. However, as shown in Figure 4 top (b), our aspect-centric tree is rooted in the aspect word "*wave*"[8]. In addition, we observe that the opinion words and aspect words can be connected by two-hop and one-hop interactions through our aspect-centric tree, which is more effective than the number of interaction hops needed in the dependency parse tree.

We also have similar observations for the second case. Illustrated in Figure 4 bottom (a), the distance between the aspect words "*lunch*" and "*menu*" and the critical opinion word "*awesome*" is four-hops and three-hops, respectively, in the parse tree. In contrast, Figure 4 bottom (b) shows that in the aspect-centric tree extracted by our model, the distances between aspect and opinion words are one-hop and two-hops, which is closer than the distance in the standard dependency parse tree.

[8]Here "wave" is chosen as the root because it has the highest probability (i.e., $\boldsymbol{P}_i^r$ in Equation 7).

4 Related Work

Aspect-based sentiment analysis. Early efforts on aspect-based sentiment focused on predicting polarity by employing attention mechanism (Bahdanau et al., 2015) to model interactions between aspect and context words (Wang et al., 2016; Chen et al., 2017; Liu and Zhang, 2017; Li et al., 2018; Wang et al., 2018). More recently, neural pre-trained language models, for instance, BERT (Devlin et al., 2019) enabled ABSA to achieve promising results. For example, Sun et al. (2019a) manually constructed auxiliary sentences using the aspect word to convert ABSA into a sentence-pair classification task. Huang and Carley (2019) propagated opinion features from syntax neighborhood words to the aspect words, in a BERT-based model. Another line of work in ABSA focused on leveraging the explicit dependency parse trees to model the relationships between context and aspect words. Zhang et al. (2019) and Sun et al. (2019b) used GCNs to integrate dependency tree information to capture structural and contextual information simultaneously for aspect-based sentiment analysis. Wang et al. (2020) greedily reshaped the dependency parse trees by using manual rules to obtain the task-specific syntactic structures.

Latent variable induction. Latent variable models (Maillard et al., 2017; Kim et al., 2017; Niculae et al., 2018; Mensch and Blondel, 2018; Liu and Lapata, 2018; Zou and Lu, 2019) have gained much popularity in building Natural Language Processing (NLP) pipelines and discovering task-specific linguistic structures (Kim et al., 2018; Martins et al., 2019). The crucial obstacle of designing structured latent variable models is that they typically involve computing an "argmax" (i.e., searching the highest-scoring discrete structure, such as a parse tree) in the middle of a computation graph. End-to-end approaches directly replace the "argmax" approach by introducing a continuous relaxation for which the exact gradient can be computed and back-propagated normally. For example, Nan et al. (2020) and Guo et al. (2020) used marginal inference to construct latent structures to improve information aggregation in the relation extraction task. More in line with our work, Chen et al. (2020) constructed task-specific structures by developing a gate mechanism to dynamically combine the parse tree information and HardKuma structure. Our work differs from this prior work in three main aspects. First, we construct the aspect-specific tree for inference without relying on an external parser. Second, we facilitate the interactions between target and opinion by introducing an explicit supervision to adaptively adjust the aspect to be the root in an end-to-end fashion. Third, we compute each aspect word's probability to become the root which enables our model to reduce the search space of inferring root for MTT in the training process.

5 Conclusion and Future Work

In this paper, we propose to use *Aspect-Centric Latent Trees* (ACLT) which are specifically tailored for the ABSA task to link up aspects with opinion words in an end-to-end fashion. Experiments on five benchmark datasets show the effectiveness of our model. The qualitative and quantitative analysis illustrate that our model is able to improve the probability of aspect words becoming the root of the sentence by imposing root constraints. Moreover, thorough analysis demonstrates our model shortens the average distances between aspect and opinions by at least 19% on the SemEval Restaurant14 dataset. To the best of our knowledge, we are the first to link up aspects with opinions through the specifically designed latent tree that imposes root constraints. One possible future direction is to apply the proposed approach to other sentiment analysis tasks, such as aspect triplet extraction (Xu et al., 2020b).

Acknowledgements

We thank Xiaochi Wei, Qian Liu and Ankit Garg and the anonymous reviewers for their comments. This work was supported by the National Key R&D Program of China under Grant No. 2020AAA0106600, supported by the National Natural Science Foundation of China (Grant No. U19B2020), sponsored by CCF-Tencent Open Research Fund (CCF-Tencent IAGR20200103) and the funding from China Scholarship Council No. 201906030188. This research is also supported by Ministry of Education, Singapore, under its Academic Research Fund (AcRF) Tier 2 Programme (MOE AcRF Tier 2 Award No: MOE2017-T2-1-156). Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of the Ministry of Education, Singapore.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR*.

Jasmijn Bastings, Wilker Aziz, and Ivan Titov. 2019. Interpretable neural predictions with differentiable binary variables. In *Proceedings of ACL*.

Chenhua Chen, Zhiyang Teng, and Yue Zhang. 2020. Inducing target-specific latent structures for aspect sentiment classification. In *Proceedings of EMNLP*.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL*.

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 ACL*.

Jack Edmonds. 1967. Optimum branchings. *Journal of Research of the national Bureau of Standards B*.

Zhijiang Guo, Guoshun Nan, Wei Lu, and Shay B. Cohen. 2020. Learning latent forests for medical relation extraction. In *Proceedings of IJCAI*.

Binxuan Huang and Kathleen Carley. 2019. Syntax-aware aspect level sentiment classification with graph attention networks. In *Proceedings of EMNLP*.

Zhanming Jie and Wei Lu. 2019. Dependency-guided lstm-crf for named entity recognition. In *Proceedings of EMNLP*.

Yoon Kim, Carl Denton, Luong Hoang, and Alexander M. Rush. 2017. Structured attention networks. In *Proceedings of ICLR*.

Yoon Kim, Sam Wiseman, and Alexander M Rush. 2018. A tutorial on deep latent variable models of natural language. *In Proceedings of ACL*.

Terry Koo, Amir Globerson, Xavier Carreras, and Michael Collins. 2007. Structured prediction models via the matrix-tree theorem. In *Proceedings of EMNLP*.

Hao Li and Wei Lu. 2019. Learning explicit and implicit structures for targeted sentiment analysis. In *Proceedings of EMNLP*.

Xin Li, Lidong Bing, Wai Lam, and Bei Shi. 2018. Transformation networks for target-oriented sentiment classification. In *Proceedings of ACL*.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*.

Jiangming Liu and Yue Zhang. 2017. Attention modeling for targeted sentiment. In *Proceedings of EACL*.

Yang Liu and Mirella Lapata. 2018. Learning structured text representations. *Transactions of the Association for Computational Linguistics*, pages 63–75.

Jean Maillard, Stephen Clark, and Dani Yogatama. 2017. Jointly learning sentence embeddings and syntax with unsupervised tree-lstms. *arXiv preprint arXiv:1705.09189*.

André F. T. Martins, Tsvetomila Mihaylova, Nikita Nangia, and Vlad Niculae. 2019. Latent structure models for natural language processing. In *Proceedings of ACL*.

Arthur Mensch and Mathieu Blondel. 2018. Differentiable dynamic programming for structured prediction and attention. In *Proceedings of ICML*.

Guoshun Nan, Zhijiang Guo, Ivan Sekulic, and Wei Lu. 2020. Reasoning with latent structure refinement for document-level relation extraction. In *Proceedings of ACL*.

Vlad Niculae, André F. T. Martins, and Claire Cardie. 2018. Towards dynamic computation graphs via sparse latent structure. In *Proceedings of EMNLP*.

Bo Pang, Lillian Lee, et al. 2008. Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval*.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, S. Manandhar, and Ion Androutsopoulos. 2015. Semeval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of Workshop on SemEval*.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, S. Manandhar, and Ion Androutsopoulos. 2016. Semeval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of Workshop on SemEval*.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Haris Papageorgiou, Ion Androutsopoulos, and S. Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of Workshop on SemEval*.

Chi Sun, Luyao Huang, and Xipeng Qiu. 2019a. Utilizing BERT for aspect-based sentiment analysis via constructing auxiliary sentence. In *Proceedings of NAACL*.

Kai Sun, Richong Zhang, Samuel Mensah, Yongyi Mao, and Xudong Liu. 2019b. Aspect-level sentiment analysis via convolution over dependency tree. In *Proceedings of EMNLP*.

Xiaobing Sun and Wei Lu. 2020. Understanding attention for text classification. In *Proceedings of ACL*, pages 3418–3428, Online. Association for Computational Linguistics.

Duyu Tang, Bing Qin, and Ting Liu. 2016. Aspect level sentiment classification with deep memory network. In *Proceedings of EMNLP*.

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 ACL*.

William Thomas Tutte. 1984. *Graph theory*. Clarendon Press.

Bailin Wang and Wei Lu. 2018. Learning latent opinions for aspect-level sentiment classification. In *Proceedings of AAAI*.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020. Relational graph attention network for aspect-based sentiment analysis. In *Proceedings of ACL*.

Shuai Wang, Sahisnu Mazumder, Bing Liu, Mianwei Zhou, and Yi Chang. 2018. Target-sensitive memory networks for aspect sentiment classification. In *Proceedings of ACL*.

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

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of EMNLP*.

Hu Xu, Bing Liu, Lei Shu, and Philip Yu. 2019. BERT post-training for review reading comprehension and aspect-based sentiment analysis. In *Proceedings of NAACL*.

Lu Xu, Lidong Bing, Wei Lu, and Fei Huang. 2020a. Aspect sentiment classification with aspect-specific opinion spans. In *Proceedings of EMNLP*.

Lu Xu, Zhanming Jie, Wei Lu, and Lidong Bing. 2021. Better feature integration for named entity recognition. In *Proceedings of NAACL*.

Lu Xu, Hao Li, Wei Lu, and Lidong Bing. 2020b. Position-aware tagging for aspect sentiment triplet extraction. In *Proceedings of EMNLP*.

Chen Zhang, Qiuchi Li, and Dawei Song. 2019. Aspect-based sentiment classification with aspect-specific graph convolutional networks. In *Proceedings of EMNLP*.

Mi Zhang and Tieyun Qian. 2020. Convolution over hierarchical syntactic and lexical graphs for aspect level sentiment analysis. In *Proceedings of EMNLP*.

Yanyan Zou and Wei Lu. 2019. Quantity tagger: A latent-variable sequence labeling approach tosolving addition-subtraction word problems. In *Proceedings of ACL*.

A Statistics of root inconsistency

We count the number of sentences where the aspect word is inconsistent with the roots of its three different structures for all five datasets. Table 7 shows the details.

Tree	Lap14	Rest14	Rest15	Rest16	Twitter
Parser	591 (93%)	1,077 (96%)	484 (89%)	551 (89%)	631 (91%)
MTT	542 (85%)	1,018 (91%)	463 (85%)	533 (87%)	514 (74%)
ACLT	**213 (33%)**	**882 (79%)**	**202 (37%)**	**195 (32%)**	**353 (51%)**
Total	638	1,120	542	616	692

Table 7: Statistics of sentences where the aspect word is inconsistent with the roots of its three different structures.

B Hyper-parameters of ACLT

All hyper-parameters are tuned based on the development set. The important hyper-parameters are listed in Table 8. We employed the uncased version of the BERT model in PyTorch. Following previous conventions, we repeat each experiment three times and average the results, reporting accuracy (Acc.) and macro-f1 (F_1).

Batch size	64
Learning rate	5.00E-05
Optimizer	Adam
Max Sequence Length	96
Hidden Size	798
Hidden Layer	12
Dropout probability	0.1

Table 8: Hyper-parameters of ACLT.