A Multi-Task Dual-Tree Network for Aspect Sentiment Triplet Extraction

Yichun Zhao[1], **Kui Meng**[1], **Gongshen Liu**[1*]
Jintao Du[2], **Huijia Zhu**[2]
[1]Shanghai Jiao Tong University
[2]Ant Group Co., Ltd.
[1]{zhaoyichun, mengkui, lgshen}@sjtu.edu.cn
[2]{lingke.djt, huijia.zhj}@antgroup.com

Abstract

Aspect Sentiment Triplet Extraction (ASTE) aims at extracting triplets from a given sentence, where each triplet includes an aspect, its sentiment polarity, and a corresponding opinion explaining the polarity. Existing methods are poor at detecting complicated relations between aspects and opinions as well as classifying multiple sentiment polarities in a sentence. Detecting unclear boundaries of multi-word aspects and opinions is also a challenge. In this paper, we propose a Multi-Task Dual-Tree Network (MTDTN) to address these issues. We employ a constituency tree and a modified dependency tree in two sub-tasks of Aspect Opinion Co-Extraction (AOCE) and ASTE, respectively. To enhance the information interaction between the two sub-tasks, we further design a Transition-Based Inference Strategy (TBIS) that transfers the boundary information from tags of AOCE to ASTE through a transition matrix. Extensive experiments are conducted on four popular datasets, and the results show the effectiveness of our model.

1 Introduction

Aspect Based Sentiment Analysis (ABSA), also known as Target Based Sentiment Analysis, has received widespread attention in both academia and industry in recent years. ABSA allows a sentiment analysis of different aspects in a given sentence, which can be applied in many areas, such as social media and E-commerce reviews. Compared with sentence sentiment analysis, ABSA is more fine-grained and more in line with reality. ABSA contains many sub-tasks, such as Aspect Term Extraction (ATE) (Xu et al., 2018; Yang et al., 2020), Opinion Term Extraction (OTE) (Fan et al., 2019; Wu et al., 2020b), Aspect Level Sentiment Classification (ALSC) (Xiao et al., 2021; Li et al., 2021), Aspect Sentiment Pair Extraction (ASPE) (Li et al., 2019; Ji et al., 2020; Chen and Qian, 2020; Luo et al., 2020), Aspect Opinion Co-Extraction (AOCE) (Dai and Song, 2019), Aspect Opinion Pair Extraction (AOPE) (Chen et al., 2020; Zhao et al., 2020) and Aspect Sentiment Triplet Extraction (ASTE) (Wu et al., 2020a; Chen et al., 2021a).

Table 1: An example of different ABSA sub-tasks. Aspects, opinions and sentiment polarities are in blue, red and green respectively.

Sentence:	Good service but poor taste
Aspect Term Extration:	{service, taste}
Opinion Term Extraction:	{Good, poor}
Aspect Sentiment Pair Extraction:	{(service, pos), (taste, neg)}
Aspect Opinion Co-Extraction:	{service, Good, taste, poor}
Aspect Opinion Pair Extraction:	{(service, Good), (taste, poor)}
Aspect Sentiment Triplet Extraction:	{(service, Good, pos), (taste, poor, neg)}

Table 1 gives an example of different ABSA sub-tasks for the sentence 'Good service but poor taste'. This paper mainly concentrate on ASTE, which extracts triplets of all aspects in a sentence with the corresponding opinion and the sentiment polarity for each aspect simultaneously.

Although researches have been conducted in the area, ASTE still faces many challenges:

- **Complicated relations.** The corresponding relations between aspects and opinions can be one-to-one, one-to-many, many-to-one, and even many-to-many. It is hard to detect these relations accurately and unambiguously.
- **Multiple sentiment polarities.** Each sentence may contain multiple sentiment polarities, which are usually influenced by corresponding relations between aspects and opinions. Therefore, relations need to be integrated into the sentiment classification task in a proper way.
- **Unclear boundaries.** Aspects and opinions often contain multiple successive words, making their boundaries difficult to be detected.

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 7065–7074
October 12–17, 2022.

To address the above challenges, we propose a Multi-Task Dual-Tree Network for ASTE, namely **MTDTN**. The constituency tree and dependency tree are two parsing methods of a sentence in Natural Language Processing (NLP), and the latter has been widely used in ABSA tasks (Wang et al., 2020; Pereg et al., 2020). Although the two trees of one sentence can be transformed into each other, it may require hops over the structure in graph neural networks or self-attentions. Thus our model employs both types of trees for AOCE and ASTE, respectively. The constituency tree is applied in the co-extraction module to detect constituent boundaries, and the dependency tree is applied in the triplet extraction module to capture relations between aspects and opinions. Moreover, for the reason that different layers of BERT (Devlin et al., 2019) capture hierarchical features, with surface features in lower layers, syntactic features in middle layers and semantic features in higher layers (Jawahar et al., 2019), we employ self-attention weights of the middle layer to modify the dependency graph. The modified graph can reduce inevitable parsing errors and imply more accurate relations between words. Finally, we use a similar tagging scheme as the Grid Tagging Scheme (Wu et al., 2020a) for triplet extraction and design a Transition-Based Inference Strategy (TBIS) to transfer the boundary information from the co-extraction module to the triplet extraction module.

The contributions of our work can be summarized as follows:

- We propose a Multi-Task Dual-Tree Network for ASTE, employing a constituency tree and a modified dependency tree in two sub-tasks of AOCE and ASTE, respectively.
- We design a Transition-Based Inference Strategy that transfers the boundary information from tags of AOCE to ASTE through a transition matrix.
- We conduct extensive experiments on four popular datasets, and the results show that our model outperforms state-of-the-art models.

2 Related work

Aspect-Opinion Co-Extraction (AOCE) has been focused on in recent years, aiming to explore the interactions between Aspect Term Extraction (ATE) and Opinion Term Extraction (OTE). Initially, models have been proposed to co-extract aspects and opinions in a sentence, treating the task as a sequence labeling problem (Wang et al., 2017; Dai and Song, 2019; He et al., 2019). However, they do not consider the relations between corresponding aspects and opinions. Then (Zhao et al., 2020) define the Aspect Opinion Pair Extraction (AOPE) task and propose a span-based multi-task learning framework. (Chen et al., 2020) propose a synchronous double-channel recurrent network to obtain aspect-opinion pairs and achieve great performance. To further explore the interactions between paired terms and sentiment polarity, (Peng et al., 2020) first define the task of Aspect-Sentiment Triplet Extraction (ASTE) and propose a two-stage model to address it. Following this work, a position-aware tagging scheme (Xu et al., 2020) and a grid tagging scheme (Wu et al., 2020a) are designed to jointly extract the triplets in an end-to-end manner. (Chen et al., 2021b) further represent the semantic and syntactic relations between word pairs by a graph to enhance the vanilla grid tagging scheme. Interactions between aspect spans and opinion spans are also studied to not only consider word-to-word interactions (Xu et al., 2021). (Chen et al., 2021a) transform the triplet extraction task into a machine reading comprehension (MRC) task with well-designed queries.

3 Task Definition

Given an input sentence $X = \{x_1, x_2, \cdots, x_n\}$ of length n, we then formulate two sub-tasks as two different sequence labeling problems.

3.1 Aspect-Opinion Co-Extraction

AOCE aims to extract all aspect terms and opinion terms appearing in a sentence. We use 5 tags in $\mathcal{Y} = \{BA, IA, BO, IO, OT\}$ to label each word x_i. BA and BO denote the beginning of an aspect term or an opinion term, IA and IO denote the inside of an aspect term or an opinion term, OT denotes the outside of both kinds of terms.

3.2 Aspect-Sentiment Triplet Extraction

ASTE aims to extract triplets of all aspect terms in a sentence with the corresponding opinion term and the sentiment polarity for each aspect term simultaneously. We employ the Grid Tagging Schema (Wu et al., 2020a), which uses 6 tags in $\mathcal{G} = \{A, O, NEG, NEU, POS, N\}$ to label the relation between two words x_i and x_j. A and O denote x_i and x_j are in the same aspect term or

opinion term, NEG, NEU and POS denote x_i and x_j are separately in an aspect term and another opinion term with the corresponding sentiment polarity, N denotes x_i and x_j have no above relations.

4 Proposed Model

4.1 Model Overview

The overview of our model is shown in Figure 1. It first accepts a sentence X as the input into a shared BERT encoder, then different layers of BERT are sent to different downstream modules. For the co-extraction module, we employ the consistency tree to construct a heterogeneous graph and apply multi-layers of Graph Convolution Networks over it to generate the final representation. For the triplet extraction module, we propose a Dep-Enhanced Transformer Decoder (DETD), which receives a modified dependency graph constructed from the dependency tree to incorporate the syntactic information. Finally, a Transition-Based Inference Strategy (TBIS) is designed, transferring the boundary information from the co-extraction module to the triplet extraction module through a transition matrix.

4.2 Shared BERT Encoder

Since pre-trained models show powerful performance in Natural Language Understanding (NLU) tasks, we choose BERT (Devlin et al., 2019) as the text encoder of our model. For a given sentence X, the following representations can be generated on the pre-trained BERT:

$$H^{[1:L]} = BERT(X) \tag{1}$$

where $H^{[1:L]}$ denote hidden states of all layers of BERT and L is the max layer.

For the reason that BERT is proven to capture a rich hierarchy of linguistic information, different layers are selected for two sub-tasks:

$$\begin{aligned} H_{ce} &= H^{L} \\ H_{te} &= H^{l} \end{aligned} \tag{2}$$

where H_{ce} and H_{te} are inputs for the co-extraction module and the triplet extraction module respectively, H^L denotes hidden states of the highest BERT layer which contains more semantic information, H^l denotes hidden states of the lth BERT layer which contains more syntactic information. We assume that the co-extraction pays more attention to semantic features and the triplet extraction pays more attention to syntactic features because the latter needs to describe word-to-word relations.

4.3 Co-Extraction Module

The constituency tree is based on the formalism of context-free grammars. In this type of tree, a sentence is divided into constituents which are sub-phrases that belong to specific categories in the grammar. For instance, a verb phrase (VP) can be formed of a verb (V) and a noun phrase (NP).

For a given sentence, we employ CoreNLP to generate a constituency tree and then construct an undirected heterogeneous graph based on the tree. The graph contains $n + m$ nodes, where n leaf nodes are tokens in the sentence, and m internal nodes are constituents in the tree. There are two types of edges in the graph: self-loop edges of leaf nodes and edges between each node and its parent node in the tree. In the forward process, leaf nodes are initialized with H_{ce} and internal nodes are randomly initialized embeddings that can be updated among training.

Then we apply Graph Convolution Networks (GCN) (Kipf and Welling, 2016) over the generated graph, concatenating the representation of leaf nodes and internal nodes as the initial input:

$$H^0 = [H_{ce}; e(c)] \tag{3}$$

where c denotes the list of constituents in the tree and e denotes the lookup table of constituent embeddings.

The GCN operation can be written as:

$$h_i^{k+1} = ReLU(\sum_{j=1}^{n+m}(A_{ij}W^{k+1}h_j^k)) \tag{4}$$

where k is the number of the current layer, $A \in \mathbb{R}^{(n+m)\times(n+m)}$ denotes the adjacency matrix of the graph, $W \in \mathbb{R}^{d\times d}$ is trainable weight, d denotes the hidden size of BERT.

After K layers of GCNs, the final representation of each token is then fed into a fully-connected layer followed by a softmax layer to yield a probability distribution over $\mathcal{Y}$:

$$p_i^{ce} = softmax(W_c h_i^K + b_c) \tag{5}$$

where $W_c \in \mathbb{R}^{d\times|\mathcal{Y}|}$ and $b_c \in \mathbb{R}^{|\mathcal{Y}|}$ are trainable weight and bias.

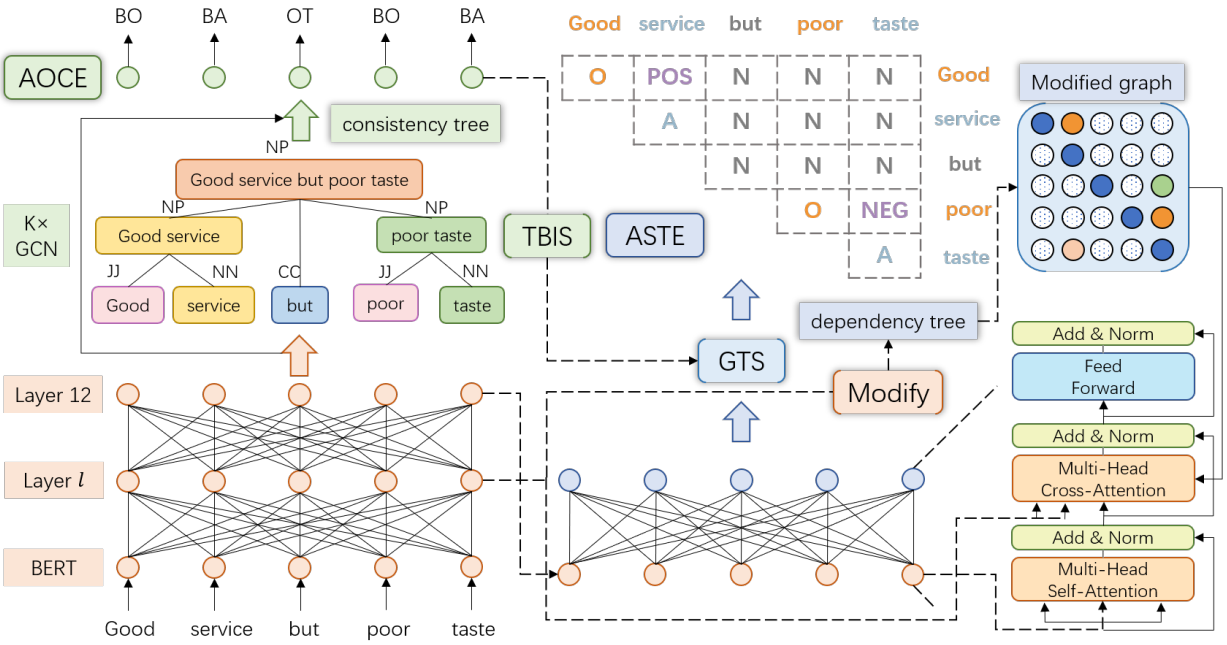


Figure 1: The overall architecture of our proposed MTDTN.

4.4 Triplet Extraction Module

The dependency tree of a sentence is a directed acyclic graph with words as nodes and relations as edges. The relation between any two words in the tree can be described as a "head-dependent" pair. For the same sentence as in the co-extraction module, we employ CoreNLP to generate a dependency tree and then construct an undirected isomorphic dependency graph based on the tree without relations .

The dependency graph generated by tools may have inevitable parsing errors. Different from (Xiao et al., 2021) which employs self-attention weights of layers all over BERT to supply the dependency graph, we only make use of the middle layer which contains more syntactic information to modify it:

$$A^{att} = softmax(\frac{Q_{att}W^{Q}_{att}(K_{att}W^{K}_{att})^{T}}{\sqrt{d}})$$

$$A^{modi}_{i,j} = \begin{cases} 1, & \alpha \leq A^{att}_{i,j} \\ A^{dep}_{i,j}, & \beta < A^{att}_{i,j} < \alpha \\ 0, & A^{att}_{i,j} \leq \beta \end{cases} \quad (6)$$

where Q_{att} and K_{att} are both equal to H_{te}, W^{Q}_{att} and W^{K}_{att} denote trainable weights, $A^{dep} \in \mathbb{R}^{n \times n}$ and $A^{modi} \in \mathbb{R}^{n \times n}$ denote adjacency matrices of the original dependency graph and the modified graph respectively, α and β are hyper-parameters.

In order to receive the modified dependency graph, we design a Dep-Enhanced Transformer Decoder (DETD) as shown in Figure 2, which uses H_{ce} as input to the first sub-layer and H_{te} as key and value to the second sub-layer:

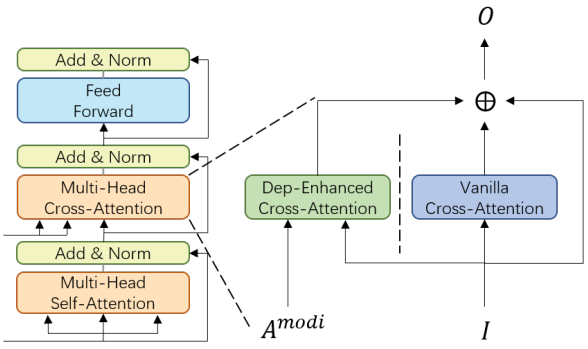


Figure 2: Dep-Enhanced Transformer Decoder.

$$T = \text{DETD}(H_{te}, H_{ce}, A^{modi}) \quad (7)$$

where $T = \{t_1, t_2, \cdots, t_n\}$ denotes the output of DETD.

Unlike the vanilla transformer decoder, we use multi-head attention instead of masked multi-head attention in the first sub-layer. Since the vanilla transformer does not explicitly encode syntactic features, in the second sub-layer, we incorporate the modified dependency graph into multi-head cross-attention by changing the calculation method

of the attention coefficients:

$$
\begin{aligned}
A =& softmax(\frac{(QW^Q(KW^K)^T) * A^{modi}}{\sqrt{d}}) \\
CA_{dep} =& FF(AV) \\
O =& LN(I + CA(I) + CA_{dep}(I))
\end{aligned} \tag{8}
$$

where I denotes the input of and O denotes the output of multi-head cross-attention, Q is equal to I, K and V are both equal to H_{te}, W^Q and W^K denote trainable weights, $*$ denotes an element-wise multiplication between A^{modi} and the dot product of Q and K, FF and LN are feed-forward network and layer normalization in transformer, CA and CA_{dep} denote the vanilla and the dep-enhanced cross-attention respectively.

Finally, we concatenate the DETD representations of word x_i and x_j to represent the word-pair (x_i, x_j), i.e., $r_{ij} = [t_i; t_j]$, where $[;]$ is the concatenation operation. Then r_{ij} is sent to a fully-connected layer to calculate the temporary triplet tag probability:

$$
z_{ij} = W^s r_{ij} + b^s \tag{9}
$$

where $W_s \in \mathbb{R}^{d \times |\mathcal{G}|}$ and $b_s \in \mathbb{R}^{|\mathcal{G}|}$ are trainable weight and bias.

4.5 Transition-Based Inference Strategy

The inference strategy of the original Grid Tagging Schema (GTS) (Wu et al., 2020a) requires indefinite iterations to capture word-to-word relations, which will increase the time complexity. Inspired by boundary guidance in E2E-ABSA (Li et al., 2019), we further propose a Transition-Based Inference Strategy (TBIS) to accelerate the convergence.

Firstly, we use a similar approach to the original GTS, leveraging features of distributions of the temporary triplet tag probability and capturing the associated features between x_i/x_j and others to obtain more accurate results. The new probability q_{ij} can be calculated as follows:

$$
\begin{aligned}
z_i &= maxpooling(z_{i,:}) \\
z_j &= maxpooling(z_{j,:}) \\
\widetilde{r}_{ij} &= [r_{ij}; z_i; z_j; z_{ij}] \\
o_{ij} &= W^o \widetilde{r}_{ij} + b^o \\
q_{ij} &= W^s o_{ij} + b^s
\end{aligned} \tag{10}
$$

where $z_{i,:} = (z_{1:i,i}, z_{i,i:n})$ according to the upper triangular grid in GTS, W^o and b^o are trainable weight and bias, W^s and b^s share the same parameters as above.

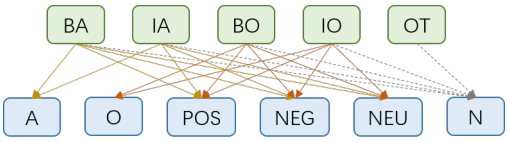


Figure 3: Constraints between co-extraction tags and triplet extraction tags.

Secondly, we encode the constraints between co-extraction tags and triplet extraction tags into a transition matrix $W^g \in \mathbb{R}^{|\mathcal{Y}| \times |\mathcal{G}|}$ as shown in Figure 3. The matrix is initialized as follows and updated as a linear layer during training:

$$
W^g_{ij} = \begin{cases} \frac{1}{|\mathcal{T}_i|} & \text{if} \quad j \in \mathcal{T}_i \\ 0, & \text{Otherwise} \end{cases} \tag{11}
$$

where $\mathcal{T}_i$ is the set of valid triplet extraction tags in $\mathcal{G}$ corresponding to the ith co-extraction tag in $\mathcal{Y}$. We transfer boundary information of aspects and opinions by mapping the probability scores of the co-extraction tag space to the triplet tag space. The transition-based score of x_i is calculated as follows:

$$
g_i = (W^g)^T p^{ce}_i \tag{12}
$$

A gating mechanism is applied to fuse the transition-based score with the triplet extraction tag probability. We calculate a gating score $\alpha_i \in \mathbb{R}$ based on the confidence score c_i:

$$
\begin{aligned}
c_i &= (p^{ce}_i)^T p^{ce}_i \\
\alpha_i &= \epsilon c_i
\end{aligned} \tag{13}
$$

where c_i represents co-extraction module's confidence in the predicted result p^{ce}_i, ϵ is a hyper-parameter to control the contribution of the transition-based score g_i in the final result.

Finally, q_{ij} is fused with g_i by the gating score α_i and the result is fed into a softmax layer to yield a probability distribution over $\mathcal{G}$:

$$
p^{te}_{ij} = softmax(\alpha_i g_i + (1 - \alpha_i) q_{ij}) \tag{14}
$$

4.6 Joint Training Loss

Training losses for two sub-tasks are both defined as cross-entropy loss:

$$
\mathcal{L}_{ce} = -\sum_{i=1}^{n} \sum_{k \in \mathcal{Y}} I(y^{ce}_i = k) log(p^{ce}_{i|k}) \tag{15}
$$

$$\mathcal{L}_{te} = -\sum_{i=1}^{n}\sum_{j=i}^{n}\sum_{k\in\mathcal{G}} I(y_{ij}^{te}=k)log(p_{ij|k}^{te}) \quad (16)$$

where y_i^{ce} denotes the ground truth tag of word x_i in co-extraction, y_{ij}^{te} denotes the ground truth tag of the relation between word x_i and x_j in triplet extraction, p_i^{ce} and $p_{i,j}^{te}$ denote predicted tagging distributions, $I(\cdot)$ is the indicator function, $\mathcal{Y}$ and $\mathcal{G}$ denote two label sets.

To jointly train two sub-tasks and make them mutually beneficial, we combine the above loss functions to form the final objective, where the hyper-parameter γ denotes their ratio.

$$\mathcal{L} = \gamma\mathcal{L}_{ce} + \mathcal{L}_{te} \quad (17)$$

5 Experiments

5.1 Datasets and Metrics

Experiments are conducted on datasets created by Wu (Wu et al., 2020a). There are four datasets, among which 14res, 15res and 16res are in the restaurant domain, and 14lap is in the laptop domain. The statistics of all datasets are listed in Table 2.

Following previous works, we employ precision, recall and micro F1-score as metrics. During training, we use the model that performed best on the development set for testing. For reproducibility, on each dataset we train the model 5 times with different random seeds and report the averaged results.

Table 2: Statistics of datasets (#S, #P, #neg, #neu, and #pos denote the number of sentences, pairs, negative triplets, neutral triplets, and positive triplets, respectively.)

Datasets		#S	#P	#neg	#neu	#pos
14res	Train	1,259	2,356	491	172	1693
	Dev	315	580	107	46	427
	Test	493	1008	156	427	784
14lap	Train	899	1452	533	111	808
	Dev	225	383	136	48	199
	Test	332	547	116	67	364
15res	Train	603	1038	210	29	799
	Dev	151	239	49	9	181
	Test	325	493	144	25	324
16res	Train	863	1421	330	55	1036
	Dev	216	348	77	8	263
	Test	328	525	79	30	416

5.2 Baselines

We compare our model with the following baselines to evaluate the performance of MTDTN, where part of them are pipeline models and others are end-to-end models.

- **Peng-unified-R+PD** (Peng et al., 2020) propose a two-stage framework to address the ASTE task. In the first stage, Peng-unified-R extracts aspects with sentiment and opinions by utilizing mutual influence between aspects and opinions. In the second stage, an MLP-based classifier (PD) is applied to all possible triplets to determine whether each triplet is valid or not.

- **Li-unified-R+PD** is a pipeline model combined by (Peng et al., 2020), which first employs a modified model Li-unified-R (Li et al., 2019) to extract aspects with sentiment and opinions and then applies PD to obtain all the valid triplets.

- **Peng-unified-R+IOG** is a pipeline model combined by (Wu et al., 2020a), which first uses Peng-unified-R (Peng et al., 2020) to extract aspects with sentiment and then employ IOG (Fan et al., 2019) to generate triplets. IOG can effectively encode aspect information to extract the corresponding opinion.

- **IMN+IOG** is another pipeline model combined by (Wu et al., 2020a), which first uses IMN (He et al., 2019) to extract aspects with sentiment and then employ IOG (Fan et al., 2019) to generate triplets.

- **GTS** (Wu et al., 2020a) propose a unified grid tagging scheme to address the ASTE task and design an inference strategy to exploit mutual indications between different opinion factors.

- **S^3E^2** (Chen et al., 2021b) further represent the semantic and syntactic relations between word pairs by a graph neural network to enhance the vanilla GTS.

- **BMRC** (Chen et al., 2021a) convert the ASTE task into a multi-turn machine reading comprehension (MRC) task with well-designed queries.

Table 3: Main results of triplet extraction (%). All methods' results are from original papers or the paper of GTS. The mark '-' means that the paper of BMRC does not release the precision and the recall on each dataset.

Methods	14res			14lap			15res			16res		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Li-unified-R+PD	41.44	68.79	51.68	42.25	42.78	42.47	43.34	50.73	46.69	38.19	53.47	44.51
Peng-unified-R+PD	44.18	62.99	51.89	40.40	47.24	43.50	40.97	54.68	46.79	46.76	62.97	53.62
Peng-unified-R+IOG	58.89	60.41	59.64	48.62	45.52	47.02	51.70	46.04	48.71	59.25	58.09	58.67
IMN+IOG	59.57	63.88	61.65	49.21	46.23	47.68	55.24	52.33	53.75	-	-	-
GTS-CNN	70.79	61.71	65.94	55.93	47.52	51.38	60.09	53.57	56.64	62.63	66.98	64.73
GTS-BiLSTM	67.28	61.91	64.49	59.42	45.13	51.30	63.26	50.71	56.29	66.07	65.05	65.56
S3E2	69.08	64.55	66.74	59.43	46.23	52.01	61.06	56.44	58.66	71.08	63.13	66.87
GTS-BERT	**70.92**	69.49	70.20	57.52	51.92	54.58	**59.29**	58.07	58.67	68.58	66.60	67.58
BMRC-BERT	-	-	70.01	-	-	57.83	-	-	58.74	-	-	67.49
Ours	70.00	**71.78**	**70.88**	**61.98**	**54.71**	**58.12**	59.03	**62.68**	**60.80**	**69.04**	**69.98**	**69.51**

5.3 Implementation Details

For the shared BERT encoder, we choose the uncased version of **BERT-base** (Devlin et al., 2019) with 12 stacked Transformer blocks, 12 attention heads and the hidden size of 768, which is implemented based on HuggingFace's **Transformers** (Wolf et al., 2020) library. While training the joint model, we employ AdamW (Loshchilov and Hutter, 2018) as the optimizer with the weight decay of 0.01 and the warmup rate of 0.1. The learning rate is set to 2e-5 for the BERT parameter group and 1e-3 for other parameter groups. The batch size is set to 32 with a max sequence length of 128. When constructing graphs for constituency trees and dependency trees, we only keep edges associated with the first sub-word of each word tokenized by BERT. We set the middle layer l to 8 for 14res, 14lap and 9 for 15res, 16res respectively. We set the thresholds α and β to 0.1 and 0.9 to generate the modified graph. For the joint training loss, the ratio γ is set to 1. All experiments are conducted on two Nvidia RTX 3080 GPUs.

5.4 Main Results

The main results of baselines and our MTDTN model are shown in Table 3. According to the results, MTDTN outperforms all baselines and achieves state-of-the-art performances on four popular datasets, which proving our model's effectiveness.

In general, due to the strong text expression ability of pre-trained models, the BERT-based models like GTS-BERT, BMRC-BERT and MTDTN surpass other models which do not employ BERT as the text encoder layer significantly.

More importantly, MTDTN achieves 0.68%, 3.54%, 2.13% and 1.93% absolute F1 scores gain over GTS-BERT, which is the state-of-the-art method we followed on four datasets. We think it is because our model can accurately locate aspects and opinions and capture the relation between them by introducing syntactic information and internal interaction of multiple tasks.

Then, compared with BMRC-BERT, which is another state-of-the-art model, MTDTN achieves an absolute F1 score increase of 0.87%, 0.29%, 2.06%, 2.02% on four datasets. BMRC-BERT converts the ASTE task into a machine reading comprehension task, while the restrictive query may not correctly capture the relation between aspects and opinions. This may be the actual cause of the performance difference.

5.5 Ablation Study

To verify the validity of different modules in our MTDTN, we further carry out an ablation study as shown in Table 4.

Table 4: Results of ablation study for ASTE task (F1%).

Methods	14res	14lap	15res	16res
MTDTN	**70.88**	**58.12**	**60.80**	**69.51**
MTDTN w/o CE	70.13	56.98	58.61	67.28
MTDTN w/o TBIS	70.33	57.39	59.51	67.69
MTDTN w/o DETD	68.81	54.56	58.11	67.03
MTDTN w/o MG	69.80	56.32	59.83	68.94

Firstly, we verify the effectiveness of the multi-task framework by removing the co-extraction module, which is 'MTDTN w/o CE' in the table. It can be observed that there is a certain decline in performance, which shows that the auxiliary task is fully effective in extracting triplets from sentences.

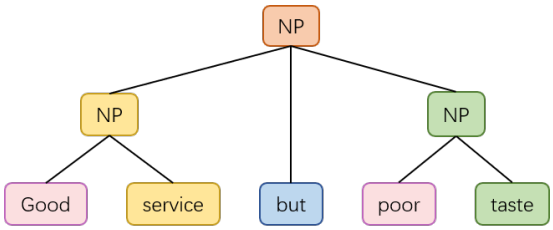
After that, we replace TBIS with the original inference strategy in GTS, which refers to 'MTDTN

w/o TBIS'. We can see the performance drops, which shows that the designed inference strategy can utilize the boundary information of aspects and opinions in the AOCE task to promote the ASTE task.

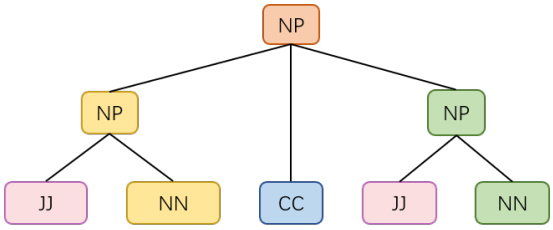
Then, we remove DETD and replace it with a vanilla transformer decoder without masked multi-head attention. It can be seen from 'MTDTN w/o DETD' that the model's performance is significantly declined, showing that the syntactic information can help the model better capture the relations between words.

More detailed, 'MTDTN w/o MG' means directly replacing the modified graph with the dependency graph generated by the parser. The dropping in performance shows that the modified graph can reduce parsing errors and is more suitable for the specific task.

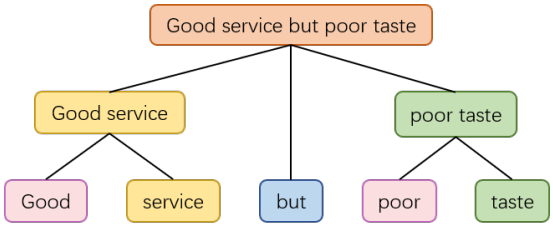
5.6 Analysis of Graph Type



(a) Heterogeneous graph with nodes of constituent and word.



(b) Isomorphic graph with nodes of constituent.



(c) Isomorphic graph with nodes of word.

Figure 4: Different types of graph.

In this section, we compare three graphs as in Figure 4 on four datasets, the results are shown in Table 5, where 'POS sequence' denotes simple addition of POS embeddings to H_{ce}, 'None' denotes H_{ce} directly being sent to the tag decoder. In 'Isomorphic graph (constituent)' and 'POS sequence', embeddings of constituents and POS are randomly initialized and updated during training. In 'Isomorphic graph (word)', embeddings of phrases are calculated from the average of word embeddings it contains.

Table 5: Results of graph analysis for AOCE task (F1%).

Methods	14res	14lap	15res	16res
Heterogeneous graph	**87.37**	**86.07**	**81.67**	81.79
Isomorphic graph (constituent)	86.22	85.07	81.39	81.42
Isomorphic graph (word)	85.06	84.23	80.28	82.15
POS sequence	85.14	83.72	80.12	**82.69**
None	84.77	84.02	79.22	82.44

We observe that the model using a heterogeneous graph obtains better AOCE performance than other methods on all datasets except 16res. This may be because the fact that the node interaction of heterogeneous graphs is more explainable compared to isomorphic graphs. On the one hand, embeddings of constituents can obtain information from the fully trained hidden states of words. On the other hand, the word representation can also get boundary information and constituent information from the graph of the constituency tree.

5.7 Analysis of BERT Layer

To investigate the effect of different BERT layers modifying the dependency graph and being the key and value of DETD, we evaluate our MTDTN model with each layer of BERT on four datasets. As shown in Figure 5, MTDTN with the 8th or 9th layer of BERT performs the best. The results are

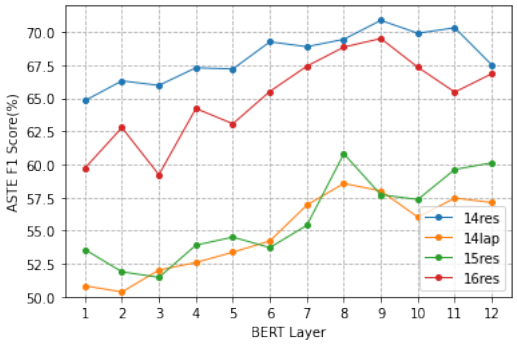


Figure 5: Impact of different BERT layers

consistent with the hierarchical characteristics of BERT (Jawahar et al., 2019) that middle layers capture rich syntactic features. Therefore, employing self-attention weights of the middle layer to modify the dependency graph can reduce parsing errors and make it suit the specific task better.

6 Conclusions

In this paper, we propose a multi-task framework for Aspect Sentiment Triplet Extraction (ASTE) with Aspect Opinion Co-Extraction (AOCE) as an auxiliary task. The two sub-tasks utilize two types of trees to capture different information of the text. For a given sentence, a constituency tree is employed by a graph convolution network for AOCE, and a modified dependency tree is employed by a special transformer decoder for ASTE. We further designed a Transition-Based Inference Strategy (TBIS) to enhance information interaction between sub-tasks by transferring the boundary information from AOCE to ASTE through a transition matrix. The whole model is called Multi-Task Dual-Tree Network (MTDTN) and extensive experiments demonstrate that our model achieves state-of-the-art performance on four popular datasets.

Acknowledge

This research work has been sponsored by Ant Group Security and Risk Management Fund, and the Joint Funds of the National Natural Science Foundation of China (Grant No. U21B2020).

References

Shaowei Chen, Jie Liu, Yu Wang, Wenzheng Zhang, and Ziming Chi. 2020. Synchronous double-channel recurrent network for aspect-opinion pair extraction. In *Proc. of ACL*.

Shaowei Chen, Yu Wang, Jie Liu, and Yuelin Wang. 2021a. Bidirectional machine reading comprehension for aspect sentiment triplet extraction. In *Proc. of AAAI*.

Zhexue Chen, Hong Huang, Bang Liu, Xuanhua Shi, and Hai Jin. 2021b. Semantic and syntactic enhanced aspect sentiment triplet extraction. In *Proc. of ACL*.

Zhuang Chen and Tieyun Qian. 2020. Relation-aware collaborative learning for unified aspect-based sentiment analysis. In *Proc. of ACL*.

Hongliang Dai and Yangqiu Song. 2019. Neural aspect and opinion term extraction with mined rules as weak supervision. In *Proc. of ACL*.

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

Zhifang Fan, Zhen Wu, Xinyu Dai, Shujian Huang, and Jiajun Chen. 2019. Target-oriented opinion words extraction with target-fused neural sequence labeling. In *Proc. of NAACL*.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2019. An interactive multi-task learning network for end-to-end aspect-based sentiment analysis. In *Proc. of ACL*.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does bert learn about the structure of language? In *Proc. of ACL*.

Qian Ji, Xiang Lin, Yinghua Ma, Gongshen Liu, and Shilin Wang. 2020. A unified labeling model for open-domain aspect-based sentiment analysis. In *Proc. of DSC*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Ruifan Li, Hao Chen, Fangxiang Feng, Zhanyu Ma, Xiaojie Wang, and Eduard Hovy. 2021. Dual graph convolutional networks for aspect-based sentiment analysis. In *Proc. of ACL*.

Xin Li, Lidong Bing, Piji Li, and Wai Lam. 2019. A unified model for opinion target extraction and target sentiment prediction. In *Proc. of AAAI*.

Ilya Loshchilov and Frank Hutter. 2018. Fixing weight decay regularization in adam. *arXiv preprint arXiv:1711.05101*.

Huaishao Luo, Lei Ji, Tianrui Li, Daxin Jiang, and Nan Duan. 2020. Grace: Gradient harmonized and cascaded labeling for aspect-based sentiment analysis. In *Proc. of EMNLP*.

Haiyun Peng, Lu Xu, Lidong Bing, Fei Huang, Wei Lu, and Luo Si. 2020. Knowing what, how and why: A near complete solution for aspect-based sentiment analysis. In *Proc. of AAAI*.

Oren Pereg, Daniel Korat, and Moshe Wasserblat. 2020. Syntactically aware cross-domain aspect and opinion terms extraction. In *Proc. of COLING*.

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

Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2017. Coupled multi-layer attentions for co-extraction of aspect and opinion terms. In *Proc. of AAAI*.

Thomas Wolf, Julien Chaumond, Lysandre Debut, Victor Sanh, Clement Delangue, Anthony Moi, Pierric Cistac, Morgan Funtowicz, Joe Davison, Sam Shleifer, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proc. of EMNLP*.

Zhen Wu, Chengcan Ying, Fei Zhao, Zhifang Fan, Xinyu Dai, and Rui Xia. 2020a. Grid tagging scheme for end-to-end fine-grained opinion extraction. In *Proc. of EMNLP*.

Zhen Wu, Fei Zhao, Xin-Yu Dai, Shujian Huang, and Jiajun Chen. 2020b. Latent opinions transfer network for target-oriented opinion words extraction. In *Proc. of AAAI*.

Zeguan Xiao, Jiarun Wu, Qingliang Chen, and Congjian Deng. 2021. Bert4gcn: Using bert intermediate layers to augment gcn for aspect-based sentiment classification. In *Proc. of EMNLP*.

Hu Xu, Bing Liu, Lei Shu, and S Yu Philip. 2018. Double embeddings and cnn-based sequence labeling for aspect extraction. In *Proc. of ACL*.

Lu Xu, Yew Ken Chia, and Lidong Bing. 2021. Learning span-level interactions for aspect sentiment triplet extraction. In *Proc. of ACL*.

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

Yunyi Yang, Kun Li, Xiaojun Quan, Weizhou Shen, and Qinliang Su. 2020. Constituency lattice encoding for aspect term extraction. In *Proc. of COLING*.

He Zhao, Longtao Huang, Rong Zhang, Quan Lu, and Hui Xue. 2020. Spanmlt: A span-based multi-task learning framework for pair-wise aspect and opinion terms extraction. In *Proc. of ACL*.