# Efficient Hybrid Generation Framework for Aspect-Based Sentiment Analysis

**Haoran Lv[1], Junyi Liu[1], Henan Wang[1], Yaoming Wang[2], Jixiang Luo[2], Yaxiao Liu[1]**
[1]Amazon Web Services, China
[2]Shanghai Jiao Tong University, China
{lvhaoran, liujunyi, henanwan, liuyaxia}@amazon.com
{wang_yaoming, ljx123456}@sjtu.edu.cn

## Abstract

Aspect-based sentiment analysis (ABSA) has attracted broad attention due to its commercial value. Natural Language Generation-based (NLG) approaches dominate the recent advance in ABSA tasks. However, current NLG practices are inefficient because most of them directly employ an autoregressive generation framework that cannot efficiently generate location information and semantic representations of ABSA targets. In this paper, we propose a novel framework, namely Efficient Hybrid Generation (EHG) to revolutionize traditions. Specifically, we leverage an Efficient Hybrid Transformer to generate the location and semantic information of ABSA targets in parallel. Besides, we design a novel global hybrid loss function in combination with bipartite matching to achieve end-to-end model training. Extensive experiments demonstrate that our proposed EHG framework greatly improves the efficiency of NLG-based methods and outperforms the competitive baselines in almost all cases.

## 1 Introduction

Aspect-based sentiment analysis (ABSA) enjoys broad commercial value, e.g., analyzing customer's opinions through review data to improve products. ABSA consists of four basic sentiment elements, including aspect term, opinion term, aspect category and sentiment polarity. Illustrated in Fig. 1, the aspect terms $a_1, a_2$ in the sentence belong to the aspect categories $c_1, c_2$ , respectively, and their corresponding opinion terms are $o_1, o_2$ , with sentiment polarity $s_1, s_2$. For simplicity, we name the aspect term and opinion term as **sentiment entities** (SE), and the aspect category and sentiment polarity as **sentiment abstractions** (SA). The majority of ABSA subtasks utilize sentences as input and the combination of SE and SA as output, so we define the output set of all ABSA subtasks as **sentiment tuples** (ST). Fig. 1 shows all the subtasks studied in this work.

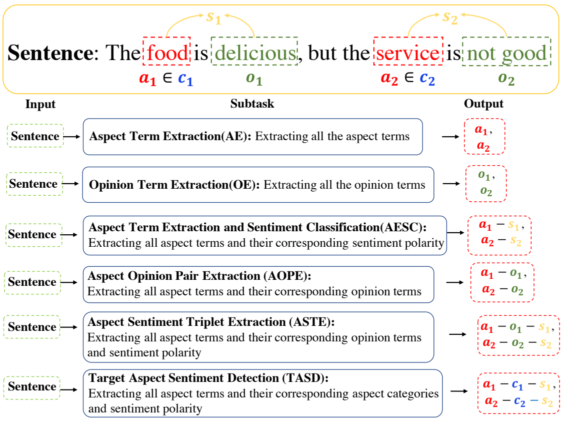
Sentence: The food is delicious, but the service is not good
$s_1$
$s_2$
$a_1 \in c_1$
$o_1$
$a_2 \in c_2$
$o_2$
Input
Subtask
Output
Sentence
Aspect Term Extraction(AE): Extracting all the aspect terms
$a_1$, $a_2$
Sentence
Opinion Term Extraction(OE): Extracting all the opinion terms
$o_1$, $o_2$
Sentence
Aspect Term Extraction and Sentiment Classification(AESC): Extracting all aspect terms and their corresponding sentiment polarity
$a_1 - s_1$, $a_2 - s_2$
Sentence
Aspect Opinion Pair Extraction (AOPE): Extracting all aspect terms and their corresponding opinion terms
$a_1 - o_1$, $a_2 - o_2$
Sentence
Aspect Sentiment Triplet Extraction (ASTE): Extracting all aspect terms and their corresponding opinion terms and sentiment polarity
$a_1 - o_1 - s_1$, $a_2 - o_2 - s_2$
Sentence
Target Aspect Sentiment Detection (TASD): Extracting all aspect terms and their corresponding aspect categories and sentiment polarity
$a_1 - c_1 - s_1$, $a_2 - c_2 - s_2$

Figure 1: The six ABSA subtasks studied in this work, where $a_1$=*food*, $a_2$=*service*, $c_1$=*food quality*, $c_2$=*service general*, $o_1$=*delicious*, $o_2$=*not good*, $s_1$=*positive*, $s_2$=*negative*.

Natural Language Understanding (NLU) based approaches were utilized in early research for modeling ABSA problems. The Two stage pipeline, i.e., using a sequence tagging model to extract SE, then using a classification model to distinguish SA (Li and Lam, 2017; Ma et al., 2017; Xue and Li, 2018; Fan et al., 2019), ignored the semantics of the labels and propagated error in a not end-to-end manner. Besides, unique end-to-end NLU variants were proposed to enhance the information interaction between elements within ST, however, these approaches suffered from complex model design and were not generic for different ABSA subtasks.

Natural Language Generation (NLG) based approaches model the ABSA tasks uniformly as sequence-to-sequence generation tasks (Zhang et al., 2021b; Yan et al., 2021; Zhang et al., 2021a; Mao et al., 2022) and have dominated the recent advance in ABSA tasks. Specifically, the sequence-to-sequence generation framework can adapt to multiple ABSA subtasks simultaneously without additional architectural design, and the autoregressive generation mode (Neal, 1992) can naturally

*Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1007–1018
May 2-6, 2023 ©2023 Association for Computational Linguistics

include information interaction of elements within ST. Despite the superior performance of NLG-based approaches, the autoregressive generation also renders NLG-based methods time inefficient in generating multiple sets of STs. Besides, existing NLG-based methods fail to adequately coordinate the semantic representation of STs and the positioning of SEs when generating results.

In this paper, we propose an Efficient Hybrid Generation framework EHG to generate semantic representations of multiple STs in parallel and simultaneously output the location information of SEs corresponding to each ST. EHG is composed of two modules: Efficient Hybrid Transformer (EHT) and Entity Correction Module (ECM). For EHT, inspired by DETR (Carion et al., 2020), we use a fixed number of *object queries* to detect STs in sentences in parallel and locate the corresponding SE. Further, we pass *object queries* that detect ST (obtained by *Filter* or *Matcher*), through the *Semi-Autoregressive Generator* to generate semantic representations of the corresponding ST. In addition we propose a novel global hybrid loss function in combination with bipartite matching named hybrid Hungarian loss, to achieve end-to-end model training. For ECM, we normalize the location information of the corresponding SEs based on the input text and the semantic representation of each ST generated. The contributions of this paper are summarized as below.

- We propose a novel and Efficient Hybrid Generation framework that generates semantic representations of STs and simultaneously locates the SEs within them.
- We introduce a fixed number of *object queries* to detect and decode multiple sentiment tuples in parallel, which greatly improves the efficiency of current generation methods.
- We propose a novel hybrid Hungarian loss function that allows our Efficient Hybrid Transformer to be trained end-to-end.

Experimental results demonstrate that our proposed EHG achieves the state-of-art performance in almost all datasets on the six ABSA subtasks and outperforms existing NLG-based methods in terms of inference efficiency.

## 2 Related Work

In this section, we summarize the existing NLU-based and NLG-based approaches for aspect-based sentiment analysis.

### 2.1 NLU-based ABSA

For NLU-based methods, early work was often devoted to solving simple problems. For example, extracting SEs or classifying the sentiment polarity for a given aspect (Tang et al., 2015; Xu et al., 2018; Li et al., 2018; Wang et al., 2016). Among them, the vast majority defined the extraction of SEs as a sequence tagging problem (Huang et al., 2015) and the determination SAs as a classification problem. However, this approach ignores the information interaction of the elements within ST and makes the errors cumulative. Recently, much work has been devoted to solving complex multi-objective problems (ASTE (Zhang et al., 2015; Li et al., 2019; Mitchell et al., 2013), TASD(Wan et al., 2020), AOPE (Zhao et al., 2020; Chen et al., 2020)), such as predicting multiple elements simultaneously. Some of these works use the idea of pipeline to solve the extraction of SEs and the classification of SAs in stages (Peng et al., 2020; Mao et al., 2021). There are also some recent approaches that attempt to introduce interactions between multiple elements in the ST during inference to achieve end-to-end multi-objective modeling (Xu et al., 2021, 2020; Ma et al., 2018). However, this approach usually requires complex model design and is not generic for different ABSA subtasks.

### 2.2 NLG-based ABSA

Generation frameworks are increasingly used in NLP due to their powerful generality and rich pretrained models (PTMs) (Sutskever et al., 2014; Cho et al., 2014; Radford et al., 2018; Shao et al., 2021). There has been some recent work using NLG-based approaches to solve ABSA problems. Yan et al. (2021) use BART (Lewis et al., 2019) to receive sentence input and generate STs directly autoregressively, where SEs are represented using location indexes. More directly, Zhang et al. (2021b,a); Mao et al. (2022) used T5 to generate semantic representations of STs, and in addition, to locate the location of SEs, Zhang et al. (2021b) additionally proposed an *Annotation* pattern to insert STs to the original sentences. Although the NLG-based approach can model multiple ABSA subtasks in a unified framework, the above two methods share the common problem that the efficiency of the generation will be greatly reduced when there are multiple sets of STs on the input in the autoregressive framework. In addition, for the direct output of the SE index, although it can locate entity locations,

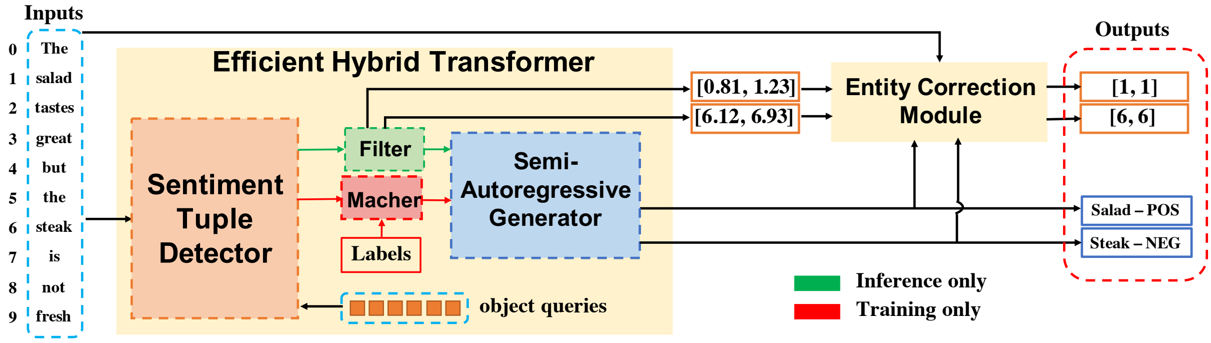

Figure 2: Efficient Hybrid Generation Framework. 1) the input sentence is directly passed through the *Efficient Hybrid Transformer* to generate a set of STs in parallel and locate the corresponding SEs. Specifically, a fixed number of learnable embeddings, called *object queries*, will be fed into the *Sentiment Tuple Detector* along with the input, and each *object queries* will focus on capturing an ST and locating the SE in it. The output of the *Sentiment Tuple Detector* will then be passed through a *Filter* (for training, it's *Matcher*) to remove the part of the ST that is not detected. Finally, the output of the *Filter* will be fed to the *Semi-Autoregressive Generator* to generate multiple STs in parallel. 2) the SE location information will be combined with the input and the corresponding ST to get the corrected SE index by *Entity Correction Module*.

this approach is not intuitive enough compared to directly generating semantic representations, and still requires additional design for scenarios that require the extraction of aspect categories, such as TASD. For the direct generation of semantic representations, it is impossible to locate the location of SEs accurately (the same SE may appear multiple times in the input), and although the *Annotation* pattern is proposed, the operation of inserting sentiment tuples to the input makes the generated results very redundant, which amplifies the problem of autoregressive inefficiency.

## 3 Methodology

Our proposed Efficient Hybrid Generation Framework is depicted in detail in Fig. 2. Since the output of all six ABSA subtasks contains the part of SEs, they can be described under a unified framework. To facilitate the description, this section expands on AESC as an example, first introducing the task definition, and then Efficient Hybrid Transformer and Entity Correction Module, are described in detail.

### 3.1 Problem Formulation

As discussed above, the proposed framework contains two parts of output, which are the semantic representation of multiple STs and the location of the corresponding SEs. Specifically, the proposed framework supports a fixed maximum number $M$ of outputs, where $M$ is set to be much larger than the number of STs generally present in the sentence. Here we denote aspect term, opinion term, aspect category, sentiment polarity, sentiment entity, sentiment abstraction and sentiment tuple as $a$, $o$, $c$, $s$, $se$, $sa$ and $st$ respectively, and define $X = \{x_i\}_{i=1}^{N}$ to represent an input text of length $N$, $L_{hybrid}(\cdot)$ represents the proposed hybrid Hungarian loss function. We use hat " $*, \widehat{*}$ " to distinguish between model output and ground truth, then the problem is defined as follows,

$$argmin \sum_{j=1}^{M} L_{hybrid}[(d_j^{se}, y_j^{st}), (\widehat{d_j^{se}}, \widehat{y_j^{st}})] \quad (1)$$

where $d_j^{se}$ and $y_j^{st}$ represent the detection result and semantic representation of the $j$-th ST, respectively. Specifically, $d_j^{se} = (c_j^{st}, idx_j)$ where $c_j^{st}$ is the class label of the ST and there are only two classes representing the presence or absence of the ST respectively. The $idx_j$ is defined as the index range of the SE in the $j$-th ST normalized in the sentence, and we set $idx_j = [-1, -1]$ when $c_j^{st} = \oslash$ represents the absence of the ST.

Furthermore, since the outputs of different ABSA subtasks may contain different SEs, we describe the $idx_j$ corresponding to the different subtasks as follows:

$$idx_j = \begin{cases} (idx_j^a), & [AE, AESC, TASD], \\ (idx_j^o), & [OE], \\ (idx_j^a, idx_j^o), & [AOPE, ASTE], \end{cases}$$

where $idx_j^a = [0, 1]^2$ and $idx_j^o = [0, 1]^2$ represent the position indices of $a_j$ and $o_j$ respectively.

### 3.2 Efficient Hybrid Transformer

Efficient Hybrid Transformer contains two main components, *Sentiment Tuple Detector*, which detects the STs in $X$, and *Semi-Autoregressive Generator*, which generates a semantic representation for

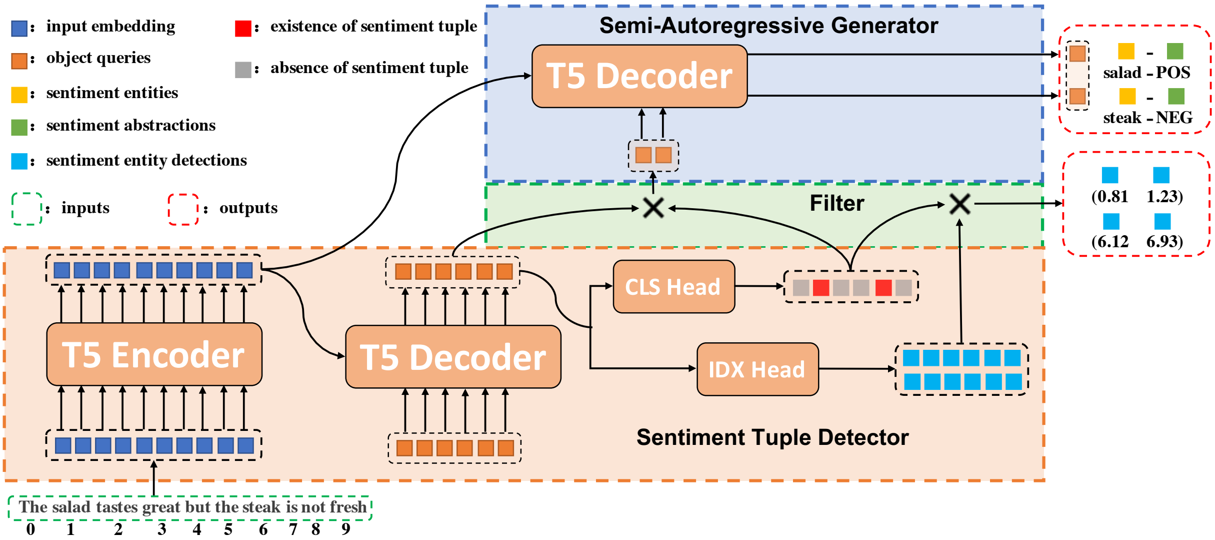

Figure 3: Inference details of Efficient Hybrid Transformer under AESC.

each detected ST, the details are shown in Fig. 3. It is worth mentioning that our EHT uses an asymmetric inference and training process, the difference between the two is that *Filter* is used for inference and *Matcher* is used for training.

### 3.2.1 Sentiment Tuple Detector

Inspired by DETR, we define mining STs in sentences as an object detection problem. Specifically, we detect the presence of STs and then locate the corresponding SEs in $X$. The *Sentiment Tuple Detector* utilizes the conventional transformer encoder-decoder architecture (Vaswani et al., 2017) , a setup that is convenient to extend to any similar pre-trained model (Qiu et al., 2020; Peters et al., 2018; Devlin et al., 2018; Raffel et al., 2019; Song et al., 2019), and in this paper we directly use the encoder and decoder of the pre-trained T5 model.

The input text $X$ is vectorized and passed through the encoder to obtain the encoded $D$-dimensional hidden variable $H \in \mathbb{R}^{N\times D}$. This process can be described in the following form:

$$H = T5Encoder(Embed(X)), \qquad (2)$$

where $Embed(X) \in \mathbb{R}^{N\times D}$, represents the vectorized $X$.

Different from the original autoregressive decoder, the decoder of the *Sentiment Tuple Detector* decodes $M$ objects in parallel. Similar to DETR, we use $M$ learnable embeddings $q^{st}_{in} \in \mathbb{R}^{M\times D}$, called *object queries*, as input to the decoder to generate output embeddings $q^{st}_{out} \in \mathbb{R}^{M\times D}$ through global self-attention and cross-attention with $H$.

$$q^{st}_{out} = T5Decoder(q^{st}_{in}, H), \qquad (3)$$

The output embedding $q^{st}_{out}$ finally outputs the class labels of the STs and the location coordinates of the corresponding SEs independently through the linear projection layer and the feed-forward neural network, respectively.

$$\begin{gathered}\widehat{c}^{st} = Softmax(MLP(q^{st}_{out})),\\ \widehat{idx}^{a} = FFN_a(q^{st}_{out}), \widehat{idx}^{o} = FFN_o(q^{st}_{out}),\end{gathered} \qquad (4)$$

where $\widehat{c}^{st} = \{\widehat{c}^{st}_j\}^M_{j=1}, \in \mathbb{R}^{M\times 2}$ is the class vector of the ST (here only two classes are present and absent). $\widehat{idx}^{a} = \{\widehat{idx}^{a}_j\}^M_{j=1}, \in \mathbb{R}^{M\times 2}$ and $\widehat{idx}^{o} = \{\widehat{idx}^{o}_j\}^M_{j=1}, \in \mathbb{R}^{M\times 2}$ are generated by two 3-layer feed-forward neural networks ($FFN$) with ReLU activation function and hidden dimension $D$, representing the normalized position coordinates of $a$ and $o$, respectively. Since different ABSA subtasks contain different SEs in their outputs, $FFN_a$ and $FFN_o$ are set according to the actual needs, and in the case of ASEC, only $FFN_a$ is used to generate $\widehat{idx}^{a}$.

### 3.2.2 Filter

The role of the *Filter* in inference is 2-fold, firstly filtering out the output of the *Sentiment Tuple Detector*; secondly the corresponding $q^{st}_{out}$ of detected ST need to be fed into the *Semi-Autoregressive Generator* to decode the semantic representation of STs. The process can be expressed as follows.

$$\widehat{d}^{se}_{\widehat{c}^{st}\neq\emptyset}, q^{st}_{out,\widehat{c}^{st}\neq\emptyset} = Filter(\widehat{D}^{se}, q^{st}_{out}), \qquad (5)$$

### 3.2.3 Semi-Autoregressive Generator

The semi-autoregressive decoder aims to decode the detected STs in parallel. Different from traditional autoregressive decoding, our semi-autoregressive decoder uses $q^{st}_{out,\widehat{c}^{st}\neq\oslash}$ as the embedding of the starting token for each ST and then performs autoregressive decoding separately. For the $s$-th ST, we use $q^{st}_{out,s,\widehat{c}^{st}\neq\oslash}$ as the starting embedding, then the autoregressive decoding process can be expressed as:

$$\begin{aligned} h^d_{s,0} &= q^{st}_{out,s,\widehat{c}^{st}\neq\oslash} \\ h^d_{s,t} &= T5Decoder(H^e, h^d_{s,t-1}) \\ st_{s,t} &= Softmax(MLP(h^d_{s,t})) \end{aligned} \tag{6}$$

where $h^d_{s,t} \in \mathbb{R}^D$ is the decoder output of step $t$ of the $s$-th ST, and $st_{s,t}$ is the corresponding token representation, which is generated by a linear projection layer that immediately follows softmax.

Besides Decoder in the *Semi-Autoregressive Generator* shares parameters with the Decoder in the *Sentiment Tuple Detector*, which allows our model to maintain almost the same number of parameters as the original T5.

### 3.2.4 Matcher

For model training, we need to know the correspondence between a fixed number of $M$ outputs and the ground truth in order to calculate the loss. Thus we use the Hungarian algorithm (Kuhn, 1955) to obtain the optimal bipartite matching between model outputs and the ground truth, and then calculate the loss for the completed matched outputs and the ground truth to train the model, this part focuses on the matching process.

We define $\phi \in \Phi_M$ as all possible mapping connections between the ground truth and the model outputs, and $L_{match}$ as the matching loss function under the given mapping condition, then the objective function for finding the optimal mapping $\phi_{optimal}$ is as follows.

$$\phi_{optimal} = \underset{\phi\in\Phi_M}{argmin} \sum_{j=1}^{M} L_{match}(d^{se}_j, \widehat{d}^{se}_{\phi(j)}) \tag{7}$$

Since we assume that $M$ is greater than the number of STs in the actual input, we use $\oslash$ to padding each ground truth $d^{se}$ that is less than $M$. In Eq. 7, the index of $\widehat{d}^{se}$ is mapped by $\phi_{optimal}$ such that $L_{match}$ is minimized. For $L_{match}$, we consider two components which are class cost $L_{cls}$ and location cost $L_{idx}$, as follows.

$$L_{match} = \lambda_{cls}L_{cls} + \lambda_{idx}L_{idx} \tag{8}$$

$$L_{cls} = -\sum_{j=1}^{M}(\widehat{p}_{\phi(j)}(c^{st}_j))_{c^{st}_j\neq\oslash} \tag{9}$$

$$L_{idx} = \sum_{j=1}^{M}(\|idx_j - \widehat{idx}_{\phi(j)}\|_1)_{c^{st}_j\neq\oslash} \tag{10}$$

where $\lambda_{cls}$ and $\lambda_{idx}$ are hyperparameters, $\widehat{p}_{\phi(j)}(c^{st}_j)$ and $idx_{\phi(j)}$ represents the probability of class $c_j$ and normalized index of SEs with index $\phi(i)$, respectively. At this point, we have completed the matching between the model output and the ground truth without any gradient computation, but the accurate matching is very important for the subsequent loss calculation, as discussed later.

### 3.2.5 Loss

To enable end-to-end training of the EHT, we propose a hybrid Hungarian loss $L_{hybrid}$, which considers both the detection loss $L_{det}$ and the generation loss $L_{gen}$ under a given optimal matching $\phi_{optimal}$.

$$L_{hybrid} = L_{det} + \mu_{gen}L_{gen}$$

$$L_{det} = \mu_{cls}\sum_{j=1}^{M}[-log(\widehat{p}_{\phi_{optimal}(j)}(c^{st}_j)) + \mu_{idx}(\|idx_j - \widehat{idx}_{\phi_{optimal}(j)}\|_1)_{c^{st}_j\neq\oslash}]$$

$$L_{gen} = \sum_{j=1}^{M}(\sum_{t=1} -log(\widehat{p}_{\phi_{optimal}(j)}(st_{j,t})))_{c^{st}_j\neq\oslash}$$

where $\phi_{optimal}$ is obtained in *Matcher*, $\mu_{cls}$, $\mu_{idx}$ and $\mu_{gen}$ are hyperparameters. For $L_{det}$, the difference with Eq. 9 is that we use the standard cross-entropy loss and consider the part of $c^{st}_j \neq \oslash$ into the classification loss as well, which reduces the redundant detections, as will be shown in the experimental section. For $L_{gen}$, we simply computed the negative log-likelihood loss of the ST generated by the matched $q^{st}_{out,\widehat{c}^{st}\neq\oslash}$ with the ground truth ST. Thus we jointly optimize $L_{det}$ and $L_{gen}$ for end-to-end training.

## 3.3 Entity Correction Module

In computer vision, object detection is not required to output the exact pixel coordinates because the images have redundant information, but for ABSA, inaccurate detection of SEs is unacceptable. To solve this problem, we propose an *Entity Correction Module* that corrects the detections of SEs by combining the input text and the output of the *Semi-Autoregressive Generator*.

First, we define SEs in the $s$-th ST of the model output as $\widehat{se}_s \in \widehat{st}_s$ and the corresponding detection result as $\widehat{idx}_s$, then we have the following

**Algorithm 1** Entity Correction Module

```
$N = len(X); res = \widehat{idx}_s$
$Mean_{idx} = Int(mean(\widehat{idx}_s))$
$s, e = \widehat{st}_s[0], \widehat{st}_s[-1]$
$L, R = Mean_{idx}; w = len(\widehat{st}_s))$
**while** True **do**
    **if** $x[L]$==$s$ and $x[L : L$+$w]$==$\widehat{st}_s$ **then**
        $res \leftarrow (L, L$+$w)$; berak
    **else if** $x[R]$==$e$ and $x[R$-$w$+1 : $R$+1]==$\widehat{st}_s$ **then**
        $res \leftarrow (R$-$w$+1$, R)$; berak
    **else if** $L$==0 and $R$==$N$-1 **then**
        berak
    **else**
        $L = max(0, L$-1$)$;
        $R = min(N$-1$, R$+1$)$;
    **end if**
**end while**
**return** $res$
```

algorithm.

In Algorithm 1, $Mean_{idx}$ represents the midpoint of $\widehat{idx}_s$ rounded down. The essence of the whole algorithm is to consider the midpoint $Mean_{idx}$ as a fuzzy detection, and then use $Mean_{idx}$ to initialize two pointers in the input $X$ to search for the nearest SE forward and backward respectively, and finally return the original detection $\widehat{idx}_s$ if there is no match.

## 4 Experiments

### 4.1 Datasets

We evaluate our EHG framework on four benchmark datasets Laptop14, Rest14, Rest15, and Rest16 that are publicly available in SemEval 2014, 2015 and 2016 (Pontiki et al., 2014, 2015, 2016). We use multiple publicly available datasets derived from the original dataset to respond to different ABSA subtasks, and these derived datasets are often additionally annotated. Specifically, we used the dataset $D_{17}$ provided by Wang et al. (2017) to evaluate the AE and OE; the dataset $D_{19}$ provided by Peng et al. (2020) to evaluate the AOPE and AESC; the dataset $D_{20a}$ provided by Xu et al. (2020) to evaluate the ASTE task; and the dataset $D_{20b}$ provided by Wan et al. (2020) to evaluate the TASD task.

| Model | Rest14 | | Rest15 | | Laptop14 | |
|---|---|---|---|---|---|---|
| | AE | OE | AE | OE | AE | OE |
| IMN-BERT | 84.06 | 85.10 | 69.9 | 73.29 | 77.55 | 81.0 |
| RACL-BERT | 86.38 | 87.18 | 73.99 | 76.0 | 81.79 | 79.72 |
| Dual-MRC | 86.6 | - | 75.08 | - | 82.51 | - |
| SPAN-BERT | 86.71 | - | 74.63 | - | 82.34 | - |
| Gen-idx | 87.07 | 87.29 | 75.48 | 76.49 | 83.52 | 77.86 |
| EHG(Ours) | **87.43** | **88.87** | **79.41** | **82.72** | **85.32** | **84.42** |

Table 1: Comparison of F1 scores for AE and OE. All comparison results are from Yan et al. (2021). Bolded fonts for the best performance and underlined fonts for the second-best performance.

### 4.2 Metrics

We use precision(P), recall (R) and F1 scores to evaluate for all experiments, where a correct prediction is considered when and only when the ST is exactly the same as the ground truth.

### 4.3 Experiment Setup

In order to maintain a similar number of parameters as existing methods, the encoder and decoders in the proposed EHT are derived from the encoder and decoder of T5 base model. Since all T5 decoders in EHT are parameter shared, we have almost the same number of parameters as the original T5[1]. It is worth mentioning that our EHT is based on the T5 model implementation of the huggingface Transformer library[2].

For the different ABSA subtasks, we used a similar experimental setup. We use the AdamW (Loshchilov and Hutter, 2018) optimizer to train the EHT with a learning rate of 3e-4 and a batch size of 2 and a cumulative gradient every two batches, and all Transformers are initialized using a pre-trained T5-base. We train all datasets with 16 object queries ($M$=16), a setting larger than the maximum number of sentiment tuples in all datasets for a single sentence. We set $\lambda_{cls}$=2, $\lambda_{idx}$=1, $\mu_{cls}$=0.75 and $\mu_{idx}$=1.5 for all ABSA subtasks. In addition, for $\mu_{gen}$, we set 1.5 for AE, OE and AOPE, 1.25 for ASTE, 1 for AESC, and 0.75 TASD. All training processes are performed on an ml.p3.2xlarge instance of Amazon Elastic Compute Cloud, which includes an NVIDIA Tesla V100 (16G) GPU.

### 4.4 Competitive Baselines

To demonstrate the superiority of our proposed EHG framework, we compare multiple SOTA

[1] Compared to T5-base, which has about 222M parameters, our EHT has about 223M parameters.

[2] `https://github.com/huggingface/transformers`

| Models | R14 | R15 | R16 | L14 |
|---|---|---|---|---|
| Li-unified+[†] | 51.0 | 47.82 | 44.31 | 42.34 |
| Two-stage[†] | 51.46 | 52.32 | 54.21 | 42.87 |
| JET-BERT[†] | 62.40 | 57.53 | 63.83 | 51.04 |
| Gen-idx[†] | 65.25 | 59.26 | 67.62 | 58.69 |
| GAS-R* | 70.52 | 60.23 | 69.05 | 58.19 |
| GAS-Para* | **72.03** | 62.56 | 71.70 | 61.13 |
| EHG(Ours) | 70.13 | 62.71 | 70.17 | 60.8 |
| EHG-Para(Ours) | 71.82 | **63.58** | **72.35** | **61.53** |

Table 2: F1 results for ASTE task. The comparison results with "†" are retrieved from Yan et al. (2021), and result with "$*$" is from Zhang et al. (2021a). Bolded fonts for the best performance and underlined fonts for the second-best performance.

methods for six ABSA subtasks, including NLG-based methods Gen-idx (Yan et al., 2021), GAS-R (Zhang et al., 2021b) and GAS-Para (Zhang et al., 2021a). In addition, for fairness, all NLG-based methods do not use prediction normalization in GAS-R (Zhang et al., 2021b). For NLU-based methods, IMN-BERT (He et al., 2019), RACL-BERT (Chen and Qian, 2020), SPAN-BERT (Hu et al., 2019), Jet+BERT (Xu et al., 2020), Peng-two-stage (Peng et al., 2020), Li-unified-R (Li et al., 2019), Dual-MRC (Mao et al., 2021), SDRN (Chen et al., 2020), TAS (Wan et al., 2020) and Baseline (Brun and Nikoulina, 2018) are selected for comparison.

**The performance**: Tables 1, 2, 3, 4 show the main comparison results for AE, OE, ASTE, AOPE, AESC, and TASD, respectively. It is worth mentioning that the proposed method achieves the best F1 score in all cases except for AOPE and ASTE on the Rest14 dataset, and TASD on the Rest15 dataset. In addition, for ASTE and TASD, we also combine EHG with GAS-Para to further improve the performance while maintaining the advantages of the original EHG inference efficiency (EHG-Para), which fully demonstrates that the proposed method can be easily combined with existing NLG-based methods, each taking advantage of the other. Finally, we found that all NLG-based methods achieved stronger performance on ASTE, AOPE, AESC, and TASD tasks compared to NLU-based methods, demonstrating that the semantic interactions within ST tend to be more effective.

**Inference efficiency**: To demonstrate the speed advantage of the proposed method, we compared the generation speed and generation quality of the ASTE task on the Rest16 dataset for different NLG-based methods. As shown in Fig. 4, where the horizontal coordinate represents the number of STs generated in a single inference, the left vertical coordinate represents the average time to generate different numbers of STs and the right vertical coordinate represents the F1 corresponding to different NLG-based methods for different number of STs in ground truth. It can be clearly found that the inference time of the proposed method is independent of the number of generated STs, but depends only on the longest ST generated in parallel, and that the quality of generation does not decrease for different numbers of STs, but is relatively better among the three methods (highest in 1, 3, and 5).

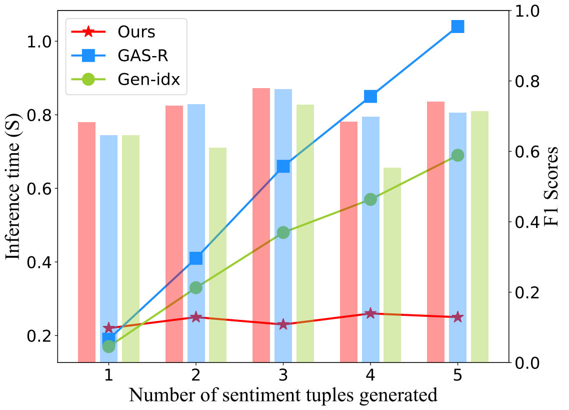

Figure 4: Comparison of the generation speed and generation quality of different NLG-based methods, where the bar chart corresponds to the F1 scores on the right and the line chart corresponds to the inference time on the left.

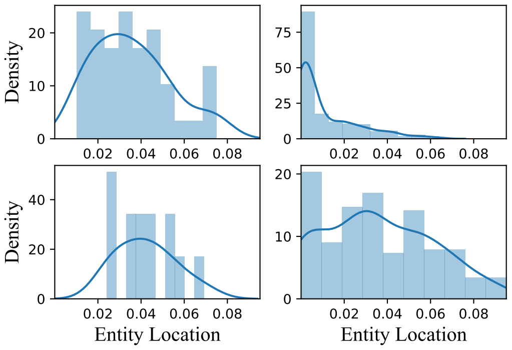

Figure 5: Visualizing the location distribution of sentiment entities detected by the first four object queries on the rest16 test set under the TASD task.

## 4.5 Analysis and Ablation Studies

To better understand the proposed framework, we visualize the location distribution of SEs (aspect

| Model | Rest14 | | Rest15 | | Rest16 | | Laptop14 | |
|---|---|---|---|---|---|---|---|---|
| | AOPE | AESC | AOPE | AESC | AOPE | AESC | AOPE | AESC |
| Li-unified+† | 55.34 | 73.79 | 56.85 | 64.95 | 53.75 | 70.20 | 52.56 | 63.38 |
| Two-stage† | 56.10 | 74.19 | 56.23 | 65.79 | 60.04 | 71.73 | 53.85 | 62.34 |
| Dual-MRC† | 74.93 | 76.57 | 64.97 | 65.14 | 75.71 | 70.84 | 63.37 | 64.59 |
| CMLA+† | 48.95 | 70.62 | 44.60 | 53.60 | 50.00 | 61.20 | 44.10 | 56.90 |
| Gen-idx‡ | 77.68 | 78.47 | 67.98 | 69.95 | 77.38 | 75.69 | 66.11 | 68.17 |
| GAS-R | **77.75** | 79.06 | 68.46 | 68.82 | 76.63 | 75.73 | 66.25 | 65.87 |
| EHG(Ours) | 77.17 | **79.32** | **69.11** | **70.04** | **78.19** | **77.12** | **69.05** | **68.48** |

Table 3: Comparison of F1 scores for AOPE and AESC. The comparison results with "†" are retrieved from Mao et al. (2021), and result with "‡" is from Yan et al. (2021). Bolded fonts for the best performance and underlined fonts for the second-best performance.

| Model | Rest15 | Rest16 |
|---|---|---|
| Baseline | - | 38.10 |
| TAS-LPM | 54.76 | 64.66 |
| TAS-SW-CRF | 57.51 | 65.89 |
| TAS-SW-TO | 58.09 | 65.44 |
| GAS-R | 60.63 | 68.31 |
| GAS-Para | **63.06** | 71.97 |
| EHG(Ours) | 60.56 | 68.56 |
| EHG-Para(Ours) | 62.83 | **72.09** |

Table 4: Comparison of F1 scores for TASD. All comparison results are from Zhang et al. (2021a). Bolded fonts for the best performance and underlined fonts for the second-best performance.

terms) generated from the first four *object queries* in the Rest16 test set under the TASD task. As shown in Figure 5, it can be found that different *object queries* have different location preferences and focus on capturing SEs in different locations.

| $M$ | $\mu_{gen}$ | TASD | AESC | ASTE | AOPE |
|---|---|---|---|---|---|
| 16 | 0.75 | **68.56** | 76.33 | 67.87 | 75.73 |
| 16 | 1 | 66.13 | **77.12** | 66.22 | 74.97 |
| 16 | 1.25 | 67.32 | 75.03 | **70.02** | 75.50 |
| 16 | 1.5 | 67.15 | 74.33 | 67.17 | **78.19** |
| 8 | 1 | **66.87** | 76.16 | **68.09** | **75.86** |
| 16 | 1 | 66.13 | **77.12** | 66.22 | 74.97 |
| 32 | 1 | 63.59 | 74.64 | 66.79 | 66.91 |

Table 5: Comparing the effects of different $\mu_{gen}$ and $M$ for different ABSA subtasks.

To evaluate the impact of different $\mu_{gen}$ and $M$ settings on different ABSA subtasks, we trained multiple models for the ablation study. All experiments occurred on the Rest16 dataset, as shown in Table 5. It can be found that the effect of different $\mu_{gen}$ is different for different subtasks, where TASD performs best at $\mu_{gen}$=0.75, AESC performs best at $\mu_{gen}$=1, ASTE performs best at $\mu_{gen}$=1 and AOPE performs best at $\mu_{gen}$=1.5. From this we conclude that as the proportion of SA in ST increases, $\mu_{gen}$ needs to be smaller to obtain better performance. The reason for this phenomenon we believe is that since SAs are usually composed of a small fixed number of words, it is relatively simple for the model to learn the mapping of different SEs to SAs, while the diversity of SEs is greatly increased, which again makes it relatively difficult to generate SEs. For $M$, we found that all ABSA subtasks performed best when $M$ was set to 8 or 16, and the performance decreased when M was set to 32. Moreover, through data analysis, we found that the average number of STs per sentence in all datasets is about 1.6, and thus we believe that setting $M$ far beyond the actual number of STs is not conducive to model training.

### 4.6 Error Analysis

To explore the behavior of EHG more intuitively, we performed an error analysis. Specifically, we verified the actual performance of the ASTE task in the test set of Rest16.

First, we find that the largest percentage of prediction errors is caused by the prediction errors of the opinion term, which accounts for about 50 percent of all errors. As in **Example 2** in Figure 6., in many cases we find that EHG has a mislocalization of the opinion term, which often occurs after "and", which we believe is related to the semi-autoregressive decoding. In **Example 2**, the aspect terms before and after "and" point to a same opinion term, but since the decoding of "server" and

**Example 1**
**Sentence**: Portions was just enough for me, but may not be for a big eater.
**Gold Lable**: (Portions , enough , neutral)
**Prediction**: (Portions , enough , positive)

**Example 2**
**Sentence**: While I could have done without the youth who shared the evening with us, our wonderful server and food made the experience a very positive one.
**Gold Lable**: (server , wonderful , positive); (food , wonderful , positive)
**Prediction**: (server , wonderful , positive); (food , positive , positive)

**Example 3**
**Sentence**: My friend enjoyed the grilled Alaskan King Salmon with delectable creamed Washington russet potatoes and crisp green beans.
**Gold Lable**: (grilled Alaskan King Salmon , enjoyed , positive); (creamed Washington russet potatoes , delectable , positive); (green beans , crisp , positive)
**Prediction**: (grilled Alaskan King Salmon , enjoyed , positive); (creamed Washington russet potatoes , delectable ,positive); (green beans , crisp , positive)

Figure 6: Examples containing the input sentence, gold label and predicted.

"food" by EHG is independent, the relationship between them is difficult to be captured. This is also a part worth optimizing in the future.

Second, we found that EHG is error-prone for neutral sentiment mining, which is of course a common problem of many current ABSA methods. We believe that this is a contradictory point in the ABSA problem, as positive or negative sentiment polarity is often highly correlated with the opinion term, while neutral sentiment polarity often requires the understanding of the whole sentence. However, the model tends to learn the correspondence between opinion term and sentiment polarity while ignoring the semantics of the whole sentence during training, which is also worthy of deeper investigation by researchers.

Finally, since EHG uses different *object queries* to capture different sentiment tuples, it is better at gaining advantages in the extraction of multiple sentiment tuples, as demonstrated by **Example 3** in Figure 6 as well as Figure 4.

## 5 Conclusion

In this paper, we propose a novel Efficient Hybrid Generation framework to model various ABSA subtasks. Specifically, we propose an Efficient Hybrid Transformer to generate sentiment tuples in parallel and simultaneously locate the corresponding sentiment entities. Moreover, we propose a hybrid Hungarian loss to achieve end-to-end model training. Finally, we propose an entity correction module to normalize the location information of sentiment entities. Extensive experiments demonstrate that our proposed EHG framework achieves competitive results while maintaining efficient inference.

## 6 Limitations

Although our proposed framework can generate semantic representations of sentiment tuples and the corresponding sentiment entity detection in parallel, there are still some limitations.

First, for the case where the input sentence contains only one set of sentiment tuples, our framework does not have a speed advantage over other NLG-based approaches, due to the fact that the efficiency of the proposed framework is based on the parallel generation of multiple sets of sentiment tuples.

In addition, the convergence speed of the proposed framework is slower than other NLG-based methods, and in all experiments we find that the convergence speed is positively related to the number of *object queries*, specifically, the average number of convergence epochs is 35 when the number of *object queries* is set to 8 and 45 when the number of *object queries* is set to 16.

## Acknowledgements

We would like to thank all authors for their contributions to this work and all reviewers for their comments.

## References

Caroline Brun and Vassilina Nikoulina. 2018. Aspect based sentiment analysis into the wild. In *Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 116–122.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer.

Shaowei Chen, Jie Liu, Yu Wang, Wenzheng Zhang, and Ziming Chi. 2020. Synchronous double-channel recurrent network for aspect-opinion pair extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6515–6524.

Zhuang Chen and Tieyun Qian. 2020. Relation-aware collaborative learning for unified aspect-based sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3685–3694.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Zhifang Fan, Zhen Wu, Xinyu Dai, Shujian Huang, and Jiajun Chen. 2019. Target-oriented opinion words extraction with target-fused neural sequence labeling. 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 2509–2518.

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 *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 504–515.

Minghao Hu, Yuxing Peng, Zhen Huang, Dongsheng Li, and Yiwei Lv. 2019. Open-domain targeted sentiment analysis via span-based extraction and classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 537–546.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Harold W Kuhn. 1955. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Xin Li, Lidong Bing, Piji Li, and Wai Lam. 2019. A unified model for opinion target extraction and target sentiment prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 6714–6721.

Xin Li, Lidong Bing, Piji Li, Wai Lam, and Zhimou Yang. 2018. Aspect term extraction with history attention and selective transformation. *arXiv preprint arXiv:1805.00760*.

Xin Li and Wai Lam. 2017. Deep multi-task learning for aspect term extraction with memory interaction. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 2886–2892.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Dehong Ma, Sujian Li, and Houfeng Wang. 2018. Joint learning for targeted sentiment analysis. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4737–4742.

Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2017. Interactive attention networks for aspect-level sentiment classification. *arXiv preprint arXiv:1709.00893*.

Yue Mao, Yi Shen, Jingchao Yang, Xiaoying Zhu, and Longjun Cai. 2022. Seq2path: Generating sentiment tuples as paths of a tree. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2215–2225.

Yue Mao, Yi Shen, Chao Yu, and Longjun Cai. 2021. A joint training dual-mrc framework for aspect based sentiment analysis. *arXiv preprint arXiv:2101.00816*.

Margaret Mitchell, Jacqui Aguilar, Theresa Wilson, and Benjamin Van Durme. 2013. Open domain targeted sentiment. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1643–1654.

Radford M Neal. 1992. Connectionist learning of belief networks. *Artificial intelligence*, 56(1):71–113.

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 *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8600–8607.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

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 sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 19–30, San Diego, California. Association for Computational Linguistics.

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, Denver, Colorado. Association for Computational Linguistics.

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, Dublin, Ireland. Association for Computational Linguistics.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10):1872–1897.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Yunfan Shao, Zhichao Geng, Yitao Liu, Junqi Dai, Fei Yang, Li Zhe, Hujun Bao, and Xipeng Qiu. 2021. Cpt: A pre-trained unbalanced transformer for both chinese language understanding and generation. *arXiv preprint arXiv:2109.05729*.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. Mass: Masked sequence to sequence pre-training for language generation. In *ICML*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2015. Effective lstms for target-dependent sentiment classification. *arXiv preprint arXiv:1512.01100*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Hai Wan, Yufei Yang, Jianfeng Du, Yanan Liu, Kunxun Qi, and Jeff Z Pan. 2020. Target-aspect-sentiment joint detection for aspect-based sentiment analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9122–9129.

Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2016. Recursive neural conditional random fields for aspect-based sentiment analysis. *arXiv preprint arXiv:1603.06679*.

Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2017. Coupled multi-layer attentions for co-extraction of aspect and opinion terms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Hu Xu, Bing Liu, Lei Shu, and Philip S Yu. 2018. Double embeddings and cnn-based sequence labeling for aspect extraction. *arXiv preprint arXiv:1805.04601*.

Lu Xu, Yew Ken Chia, and Lidong Bing. 2021. Learning span-level interactions for aspect sentiment triplet extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4755–4766, Online. Association for Computational Linguistics.

Lu Xu, Hao Li, Wei Lu, and Lidong Bing. 2020. Position-aware tagging for aspect sentiment triplet extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2339–2349.

Wei Xue and Tao Li. 2018. Aspect based sentiment analysis with gated convolutional networks. *arXiv preprint arXiv:1805.07043*.

Hang Yan, Junqi Dai, Xipeng Qiu, Zheng Zhang, et al. 2021. A unified generative framework for aspect-based sentiment analysis. *arXiv preprint arXiv:2106.04300*.

Meishan Zhang, Yue Zhang, and Duy Tin Vo. 2015. Neural networks for open domain targeted sentiment. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 612–621.

Wenxuan Zhang, Yang Deng, Xin Li, Yifei Yuan, Lidong Bing, and Wai Lam. 2021a. Aspect sentiment quad prediction as paraphrase generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9209–9219.

Wenxuan Zhang, Xin Li, Yang Deng, Lidong Bing, and Wai Lam. 2021b. Towards generative aspect-based sentiment analysis. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 504–510.

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 *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3239–3248.

# A Appendix

## A.1 Hyperparametric ablation studies

As described in the previous section, the proposed framework has five hyperparameters $\lambda_{cls}$, $\lambda_{idx}$, $\mu_{cls}$, $\mu_{idx}$ and $\mu_{gen}$. Among them, $\mu_{gen}$ has been discussed in 4.5, and the ablation experiments of the remaining hyperparameters are released here.

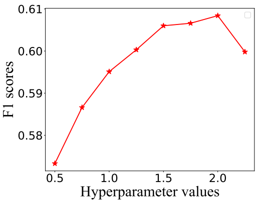

Figure 7: F1 scores under different $\lambda_{cls}$ settings.

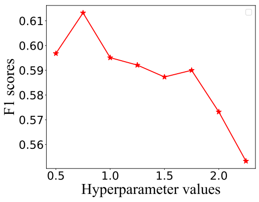

Figure 8: F1 scores under different $\mu_{cls}$ settings.

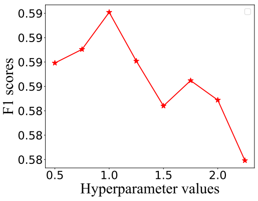

Figure 9: F1 scores under different $\lambda_{idx}$ settings.

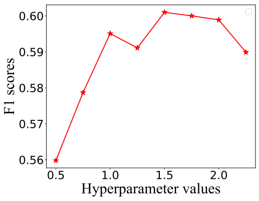

Figure 10: F1 scores under different $\mu_{idx}$ settings.

To facilitate the adjustment, we fixed all the remaining hyperparameters to 1 when adjusting a particular hyperparameter, and we set $M$=8 for all experiments in order to obtain results quickly. All experiments occurred in the ASTE task on the Rest15 dataset. Figures 7, 8, 9, 10 show the curves of F1 variation with $\lambda_{cls}$, $\mu_{cls}$, $\lambda_{idx}$ and $\mu_{idx}$, respectively, from which we use $\lambda_{cls}$=2, $\lambda_{idx}$=1, $\mu_{cls}$=0.75 and $\mu_{idx}$=1.5 for all experiments.

## A.2 Training details

| Subtask | Rest14 | Rest15 | Rest16 | Laptop14 |
|---|---|---|---|---|
| AE | 40 | 40 | - | 40 |
| OE | 40 | 40 | - | 40 |
| ASTE | 50 | 45 | 50 | 40 |
| AOPE | 50 | 55 | 55 | 50 |
| AESC | 45 | 45 | 45 | 45 |
| TASD | - | 40 | 55 | - |

Table 6: Training epochs of EHT under different ABSA subtasks and different datasets.

Since our comparison experiments consist of multiple versions of the same dataset, we give the number of training rounds for all ABSA subtasks under different datasets, which can be found in Table 6.