Seq2Path: Generating Sentiment Tuples as Paths of a Tree

Yue Mao, Yi Shen, Jingchao Yang, Xiaoying Zhu, Longjun Cai
Alibaba Group
{maoyue.my, sy133447, yangjingchao.yjc, candy.zxy, longjun.clj}@alibaba-inc.com

Abstract

Aspect-based sentiment analysis (ABSA) tasks aim to extract sentiment tuples from a sentence. Recent generative methods such as Seq2Seq models have achieved good performance by formulating the output as a sequence of sentiment tuples. However, the orders between the sentiment tuples do not naturally exist and the generation of the current tuple should not condition on the previous ones. In this paper, we propose Seq2Path to generate sentiment tuples as paths of a tree. A tree can represent "1-to-n" relations (e.g., an aspect term may correspond to multiple opinion terms) and the paths of a tree are independent and do not have orders. For training, we treat each path as an independent target, and we calculate the average loss of the ordinary Seq2Seq model over paths. For inference, we apply beam search with constrained decoding. By introducing an additional discriminative token and applying a data augmentation technique, valid paths can be automatically selected. We conduct experiments on five tasks including AOPE, ASTE, TASD, UABSA, ACOS. We evaluate our method on four common benchmark datasets including Laptop14, Rest14, Rest15, Rest16. Our proposed method achieves state-of-the-art results in almost all cases.

1 Introduction

ABSA tasks. Aspect-based sentiment analysis (ABSA) is a classic research topic and has received continuous attention. The ABSA tasks aim to extract sentiment tuples of elements such as the aspect term (a), opinion term (o), aspect category (c), and sentiment polarity (s), respectively. Following the tasks definitions in (Zhang et al., 2021b), we consider various ABSA tasks including aspect opinion pair extraction (AOPE), aspect sentiment triplet extraction (ASTE), target aspect sentiment detection (TASD), unified aspect-based sentiment analysis (UABSA) and aspect category opinion sentiment (ACOS). The output formats are shown in Table 1. Throughout this paper, we assume the ASTE task is our default task to illustrate our ideas.

ABSA Task	Abbr	Output
Aspect Opinion Pair Extraction	AOPE	(a, o)
Aspect Sentiment Triplet Extraction	ASTE	(a, o, s)
Target Aspect Sentiment Detection	TASD	(c, a, s)
Unified Aspect-Based Sentiment Analysis	UABSA	(a, s)
Aspect Category Opinion Sentiment	ACOS	(c, a, o, s)

Table 1: The ABSA tasks with their output formats: AOPE (Zhao et al., 2020; Chen et al., 2020), ASTE (Peng et al., 2020), TASD (Wan et al., 2020), UABSA (Li et al., 2019; Chen et al., 2020), ACOS (Cai et al., 2021). Throughout this paper, the ASTE task is assumed to be our default task to illustrate our ideas.

Seq2Seq for ABSA. Instead of using separate models for each ABSA task, the recent trend is to design a unified framework to handle multiple ABSA tasks at the same time. Recently, the Seq2Seq models have been applied to the ABSA tasks (Yan et al., 2021; Zhang et al., 2021a,b) by formulating them as a text-to-text problem

$$\text{Input text} \Rightarrow \text{“}(a_1, o_1, s_1), (a_2, o_2, s_2), ...\text{”}$$

where the output is a sequence of sentiment tuples. Despite their success on performance, they still have two main drawbacks: (1) Orders, the orders between the tuples does not naturally exist. (2) Dependence, the generation of (a_2, o_2, s_2) should not condition on (a_1, o_1, s_1).

As a result, the fine-tuned model may be "confused" to make decisions. For example: Why does (a_1, o_1, s_1) have to be the first tuple instead of "(a_2, o_2, s_2)"? Why does (a_1, o_1, s_1) have to be followed by (a_2, o_2, s_2) instead of (a_3, o_3, s_3) or "<eos>"?

Seq2Path for ABSA. We claim that a tree is a better choice to represent the output. As we know, a

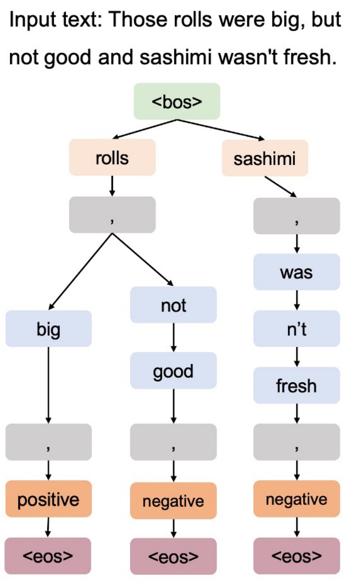


Figure 1: The generation process for the ASTE task can be represented by a tree where "<bos>", "<eos>" and "," stand for the start, end and separator tokens. Sentiment tuples are independent paths of the tree and do not have orders.

tree can represent "1-to-n" relations where a token can be followed by multiple valid tokens during generation. However, a sequence can only represent "1-to-1" relations where a token is followed by exactly one token during generation (greedy). Consider the example in Figure 1, the two sentiment tuples ("rolls", "big", "positive") and ("rolls", "not good", "negative") share the same aspect term "rolls". So it is a "1-to-n" relation because the token "big" and "not" are following the same token.

In this paper, we propose "Seq2Path" to by formulating the ABSA tasks as a "sequence to paths of a tree" problem: where each sentiment tuple can be viewed as a path of a tree and can be independently generated. As long as the input text is given, one can determine any of the valid sentiment tuples independently. For example, one can determine (a_2, o_2, s_2) is a valid sentiment tuple without knowing that (a_1, o_1, s_1) is also a valid one.

For training, we treat every sentiment tuple as an independent target. We use the ordinary Seq2Seq model to learn each target and calculate the average loss. For inference, we apply beam search to generate multiple paths along with their probabilities. The paths with high probabilities are more likely to be correct, but not always the case. We introduce a discriminative token to automatically select correct paths from beam search. We also augment the dataset to produce negative samples for the discriminative token.

Contributions. The main contributions in the paper are listed as follows:

- We propose Seq2Path, a parallel generative framework for ABSA. It generates sentiment tuples as paths of a tree. A discriminative token is introduced to automatically select valid paths from beam search.
- We also give some further motivations and show that Seq2Path is better in learning the precise conditional transition probability for token generation.
- Experimental results show that our model achieves state-of-the-art on four widely used datasets Laptop14, Rest14, Rest15, Rest16 on the AOPE, UABSA, ASTE, TASD, ACOS tasks. Our method outperforms the baseline models on F1 score in almost all cases.

2 Method

2.1 Overview of Seq2Path

We propose our Seq2Path as shown in Figure 2. The encoder-decoder architecture is an ordinary Seq2Seq architecture and their differences are described as follows. First, we treat each tuple as an independent target, train an ordinary Seq2Seq model and calculate the average loss. Second, the token generation process forms a tree, and we apply beam search to "parallelly" and "independently" generate paths. Third, the input is the text and the output is the set of all valid sentiment tuples with a binary discriminative token

$$v \in \{\text{“true”}, \text{“false”}\}$$

appended in the end:

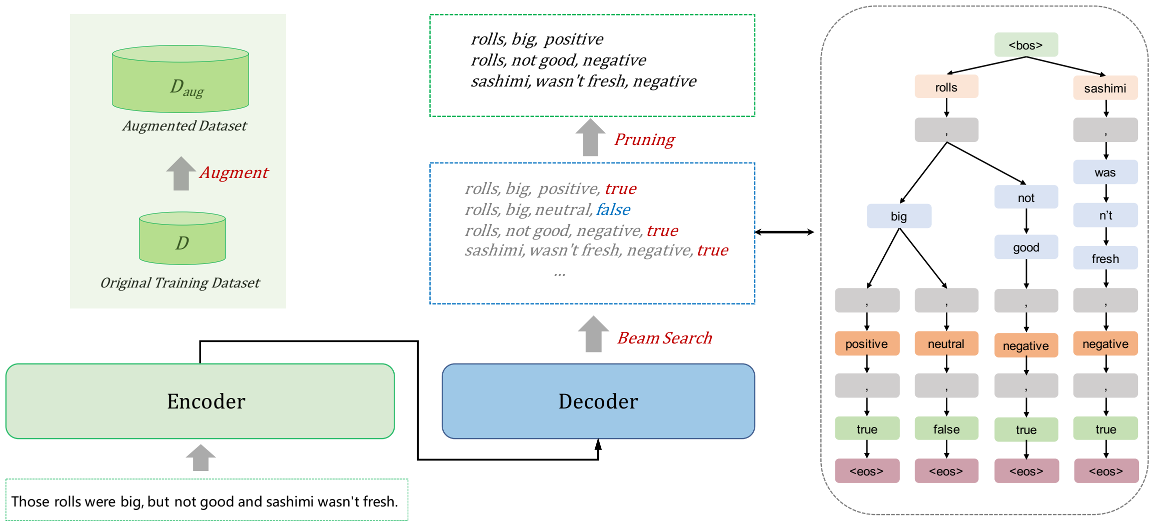
$$\begin{aligned}
\text{AOPE}: &\quad \text{Input text} \Rightarrow \text{“}a,\, o,\, v\text{”}\\
\text{ASTE}: &\quad \text{Input text} \Rightarrow \text{“}a,\, o,\, s,\, v\text{”}\\
\text{TASD}: &\quad \text{Input text} \Rightarrow \text{“}c,\, a,\, s,\, v\text{”}\\
\text{UABSA}: &\quad \text{Input text} \Rightarrow \text{“}a,\, s,\, v\text{”}\\
\text{ACOS}: &\quad \text{Input text} \Rightarrow \text{“}c,\, a,\, o,\, s,\, v\text{”}
\end{aligned}$$

where a, o, c, s denotes the aspect, opinion, category, sentiment, respectively. Since there are no negative samples for the discriminative token, we have to construct an augmented dataset for training.

2.2 Training

Loss averaged over paths. For an input sentence x, we want to output a set of tuples

$$Y = \{y^1, ..., y^k\} \tag{1}$$



D_{aug}
Augmented Dataset
Augment
D
Original Training Dataset
rolls, big, positive
rolls, not good, negative
sashimi, wasn't fresh, negative
Pruning
rolls, big, positive, true
rolls, big, neutral, false
rolls, not good, negative, true
sashimi, wasn't fresh, negative, true
...
Beam Search
Encoder
Decoder
Those rolls were big, but not good and sashimi wasn't fresh.
<bos>
rolls
sashimi
was
n't
fresh
big
not
good
positive
neutral
negative
negative
true
false
true
true
<eos>

Figure 2: The proposed Seq2Path framework. The ASTE task is used for illustrating.

The dataset D is a collection of (x, Y) pairs. As shown in Figure 1 and 2, the set Y can be represented as a tree. Then each[1] y corresponds to a path of the tree and k is the total number of paths. For the prediction $\hat{Y}$ from the input x, the loss can be defined as the average loss of the k paths

$$L(Y, \hat{Y}|x) \tag{2}$$

$$= \frac{1}{k} \sum_{y \in Y} L_{seq}(y, \hat{y}|x) \tag{3}$$

$$= \frac{1}{k} \sum_{y \in Y} \sum_{t} l(y_t, \hat{y}_t | x, y_{<t}) \tag{4}$$

where $L_{seq}(\cdot)$ is the ordinary Seq2Seq loss and $l(\cdot)$ is the loss for each time step t. More theoretical justification will be provided in Section 3.

2.3 Inference

Beam search. During the inference phase, we apply beam search (Srivastava et al., 2014) with constrained decoding. The beam search algorithm selects multiple alternatives for an input sequence at each step based on conditional probability. With beam search, we output the top-k paths with decreasing probabilities which represent how likely the paths are valid.

Constrained decoding is also applied during decoding. Instead of searching the whole vocabulary, we force beam search to search within only the allowed candidate tokens (inspired by (De Cao et al., 2020)). The candidate tokens are either from the input text or some extra task-specific tokens. For example, the ASTE task has the extra tokens including the sentiment polarities "positive", "negative", "neutral" and the separator token. Please refer to Appendix A.1 for more details on constrained decoding.

Pruning. We apply pruning to filter invalid paths. First, we remove some "overlapping" predictions. If beam search returns both "a, o, s, true" and "a, o, s, false", we prefer the one with the higher sequence probability. If beam search returns both "a_1, o, s, true" and "a_2, o, s, true" where a_1 and a_2 are overlapping, then we also prefer the one with the higher sequence probability. Then, we output the valid paths with a discriminative token v_i = "true" and filter the other invalid paths. Please refer to Appendix A.2 for the overlapping conditions for pruning.

2.4 Data augmentation

Since there are no negative samples for the discriminative token, the data augmentation step is necessary. In order to automatically select the valid paths, a discriminative token v = "false" is appended at the end of each negative sample. We generate negative samples

$$D_n = D_1 \bigcup D_2 \tag{5}$$

in the following two ways

- D_1: To improve the model's ability to match tuple elements, we randomly replace the tuple elements. For example, in Figure 1, we generate "rolls, wasn't fresh, positive, false", "sashimi, big, negative, false", etc.

[1]For notation simplicity, we write y^i as y from now on.

- D_2: To improve the model's ability to filter most of bad generations, we first train the model for small epochs then use beam search to generate negative samples. For example, in Figure 1, we generate "sashimi, n't fresh, negative, false", etc.

Then the augmented dataset is the union of the positive and negative samples

$$D = D_p \bigcup D_n \tag{6}$$

Loss mask for negative samples. We want the discriminative token v to be able to filter invalid paths. However, we do not want the model's generation to mimic the negative samples. We apply a tricky loss mask here. Suppose $y = (y_1, y_2, ...y_t, ...)$, the loss mask is defined as follows.

- If y is a negative sample, i.e., the validation token of y is "false", then the loss mask is

$$m(y_t) = \begin{cases} 1, & y_t = \text{“false”} \\ 1, & y_t = \text{“<eos>”} \\ 0, & o.w. \end{cases} \tag{7}$$

- If y is a positive sample, i.e., the validation token of y is "true", then the loss mask does not apply. In other words, we always have

$$m(y_t) = 1. \tag{8}$$

The loss mask means the token is skipped in loss calculating, see an example in Table 3. All tokens except the discriminative token and the "<eos>" token are masked. Let $L^m(\cdot)$ be the loss with the loss mask where only tokens with $m(t) = 1$ are involved in loss calculating

$$L^m(Y, \hat{Y}|x) = \frac{1}{k} \sum_{y \in Y} \sum_{m(t)=1} l(y_t, \hat{y}_t|x, y_{<t}) \tag{9}$$

and the loss for the augmented dataset is

$$Loss(D) = \sum_{(x,Y) \in D} L^m(Y, \hat{Y}|x). \tag{10}$$

2.5 Algorithm

The algorithm of Seq2Path are summarized as Algorithm 1 including training, inference and data augmentation.

Algorithm 1: Seq2Path.

```
Input: A training dataset $D_p$, beam size $k$.
Output: Valid sentiment tuples.
1 Train ordinary Seq2Seq on $D_p$ with loss
    averaged over paths for 5 epochs. Generate
    negative samples $D_1$ from beam search;
2 Generate more negative samples $D_2$ by
    randomly replacing tuple elements;
3 Let $D_n = D_1 \bigcup D_2$ be the negative samples.
    Construct the augmented dataset
    $D = D_p \bigcup D_n$ where each sample is
    appended with a discriminative token,
    either "true" or "false";
4 Train ordinary Seq2Seq on $D$ with loss
    averaged over paths for full epochs where
    the loss is masked on negative samples;
5 Apply beam search with constrained
    decoding for inference. Generate top $k$
    paths with decreasing probabilities $p_i$ and
    discriminative tokens $v_i$;
6 Apply pruning to select the valid paths
    based on $p_i$ and $v_i$. Return valid paths as
    valid sentiment tuples.
```

3 Why Seq2Path?

Conditional transition probability. In this section, we give more analysis on the motivation for Seq2Path. We claim that Seq2Path is better in learning the precise conditional transition probabilities for the token generation process:

$$P(y_t = v_i|x, y_{<t}) \tag{11}$$

where x is the input sentence and $y_{<t} = (y_1, y_2, ..., y_{t-1})$ represents the previous tokens and $V = \{v_1, v_2, ...\}$ is the vocabulary.

Intuitive case. Again, we take the example in Figure 1. Seq2Seq models formulate the output as sequence "(a_1, o_1, s_1), (a_2, o_2, s_2), ...", then the target probability distribution at each time step t is a one-hot vector in $\mathbb{R}^{|V|}$.

$$P_{\text{one-hot}}(y_4|x, y_{<4}) = \begin{cases} 1, & y_4 = \text{“big”} \\ 0, & o.w. \end{cases} \tag{12}$$

However, true target probability distribution is actually a multi-hot vector in $\mathbb{R}^{|V|}$. For $y_{<4} =$ ("<bos>", "rolls", ","),

$$P_{\text{multi-hot}}(y_4|x, y_{<4}) = \begin{cases} 0.5, & y_4 = \text{“big”} \\ 0.5, & y_4 = \text{“not”} \\ 0, & o.w. \end{cases} \tag{13}$$

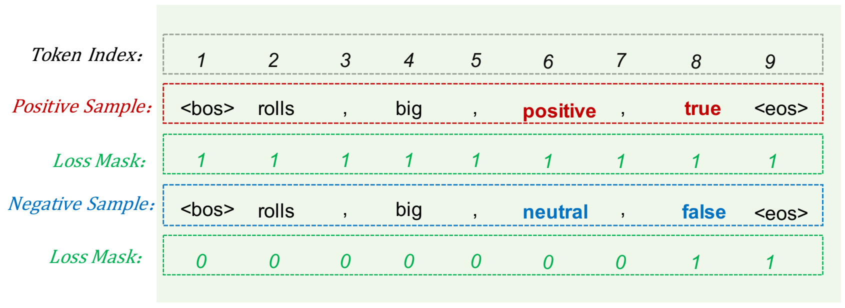

Token Index: 1 2 3 4 5 6 7 8 9
Positive Sample: <bos> rolls , big , positive , true <eos>
Loss Mask: 1 1 1 1 1 1 1 1 1
Negative Sample: <bos> rolls , big , neutral , false <eos>
Loss Mask: 0 0 0 0 0 0 0 1 1

Figure 3: An example to show the loss mask. The loss mask means the token is skipped in loss calculating. The loss mask does not apply to a positive sample. For a negative sample, the loss is calculated only upon the "false" and the "<eos>" token, and skips the other tokens because they form an invalid generation.

Why average loss over paths? Recall, during training, we treat each sentiment tuple as an independent target and calculate the average loss of ordinary Seq2Seq. Here we justify it. Formally, suppose the target contains k paths with the previous tokens $x, y_{<t}$, say

$$\begin{aligned} \text{path-1}: &\quad (x, y_1, y_2, ..., y_{t-1}, v_{j_1}, ...), \\ \text{path-2}: &\quad (x, y_1, y_2, ..., y_{t-1}, v_{j_2}, ...), \\ &\quad \cdots \\ \text{path-k}: &\quad (x, y_1, y_2, ..., y_{t-1}, v_{j_k}, ...). \end{aligned}$$

The next token could be $v_{j_1}, ..., v_{j_k}$, then the transition probability is a "multi-hot" vector $p \in \mathbb{R}^{|V|}$

$$p[i] = \begin{cases} \frac{1}{k}, & i = j_1, ..., j_k, \\ 0, & o.w. \end{cases} \tag{14}$$

On the other hand, each independent path is learned with ordinary Seq2Seq where the probability for i-th path is a "one-hot" vector $p'_i \in \mathbb{R}^{|V|}$

$$p'_i[\ell] = \begin{cases} 1, & \ell = j_i, \\ 0, & o.w. \end{cases} \tag{15}$$

The next lemma justifies why Seq2Path averages the ordinary Seq2Seq loss over paths. The proof is simple and can be found in Appendix A.2.

Lemma 1 *The average cross-entropy loss for the one-hot target (15) is equal to the cross-entropy loss for the multi-hot transition probability (14).*

4 Experiments

4.1 Experimental Settings

Datasets. We evaluate the proposed framework on four widely used benchmark datasets: Laptop14, Rest14, Rest15, and Rest16, originally provided by the SemEval shared challenges (Pontiki et al., 2014, 2015, 2016). We adopt the dataset provided by (Fan et al., 2019; Li et al., 2019; Xu et al., 2020; Wan et al., 2020) for AOPE, UABSA, ASTE, TASD, ACOS respectively. For a fair comparison, we keep the same data splits as previous works.

Baselines. In the following, we list the main baselines for each ABSA task. Several early baselines are skipped, especially those not encoded with BERT or T5.

- AOPE:
  - SpanMlt (Zhao et al., 2020) is an end-to-end method to jointly extract the aspect and opinion.
  - SDRN (Chen et al., 2020) proposes a synchronous dual-channel recursive network to simultaneously extract the opinion entities and relationships.
  - BMRC (Chen et al., 2021) proposes a unified model for ABSA tasks based on bidirectional MRC.
  - GAS-T5 (Zhang et al., 2021b) uses a unified generation method to solve various ABSA problems, and encodes natural language tags into the target output.
- ASTE:
  - Jet-BERT (Xu et al., 2020) uses two sets of BIO tags to annotate aspect and opinion terms in the same sequence in an end-to-end manner.
  - Dual-MRC(Mao et al., 2021) solves all ABSA tasks through a unified joint training framework of two MRCs.
  - ParaPhrase-T5 (Zhang et al., 2021a) treats quad prediction as a paraphrase generation.
  - BMRC (Chen et al., 2021) and GAS-T5 (Zhang et al., 2021b) were described previously.

- TASD:
  - TAS (Wan et al., 2020) is the first to introduce the target aspect sentiment detection (TASD) task.
  - GAS-T5 (Zhang et al., 2021b) and ParaPhrase-T5 (Zhang et al., 2021a) were described previously.
- UABSA:
  - RACL (Chen and Qian, 2020) is a BERT-based model with a relation propagation mechanism.
  - Dual-MRC (Mao et al., 2021), BMRC (Chen et al., 2021) and GAS-T5 (Zhang et al., 2021b) were described previously.
- ACOS:
  - ACOS-Baseline (Cai et al., 2021) is the first to introduce the aspect-category-opinion-sentiment (ACOS) quad prediction task.

Evaluation metrics. We use the F1 score as the evaluation metrics, when all elements of the prediction result are correct, the prediction result is considered correct. As a fair comparison, all F1 scores reported in this paper are averaged over 5 runs with different random seeds.

Implementation details. We use Google's T5-base model (Raffel et al., 2019) from Huggingface Transformer library[2]. The structure of the T5[3] encoder and decoder is similar to that of the Transformer (Vaswani et al., 2017). Since sentiment tuples are generated independently, the maximum output length $= 32$ can be very small comparing to the maximum sequence length $= 128$. It can reduce a lot of memory consumption.

For all ABSA tasks, we use a fixed batch size 8 and a fixed learning rate $1e^{-4}$ to train the model with a single Nvidia 1080Ti GPU. We first train the model with 5 epochs for augmentation. Then the final model is trained for $n = 20$ epochs. The best model is determined based on the loss on the validation set. For inference, the number of beams depends on the task and dataset and can be $k = 4, 6, 8, 10$. Typically, $k = 6$ can be used for most cases. In addition, the separator "," can be replaced with other separators[4] and the experimental results may improve slightly.

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

[3]Although T5-base and BERT-base are both named as "base" models, the T5-base should be more powerful because it has more parameters and trained on a larger corpus.

[4]The separator "|" seems to be slightly better than "," from our experiments. It may be related to the T5 tokenization and decoding mechanisms.

4.2 Main Results

The main results for the AOPE, UABSA, ASTE, TASD, ACOS tasks are reported in Table 2, 3, 4, 5, 6, respectively. Most baseline results are directly copied from (Zhang et al., 2021b). Our proposed method achieves the new state-of-the-art results in almost all F1 scores.

On the AOPE task, our proposed Seq2Path outperforms the previous best results by 4.74, 1.75, 3.91, 3.67 in percentage on Laptop14, Rest14, Rest15, Rest16 respectively. The main challenge for the AOPE task is to match the aspect a and the opinion o where there are many complex "1-to-n" relations. Our Seq2Path has a large performance gain because it can handle these complex relations very well.

	L14	R14	R15	R16
SpanMlt	68.66	75.60	64.68	71.78
SDRN	66.18	73.30	65.75	73.67
BMRC	67.45	76.23	68.60	76.52
GAS-T5	69.55	75.15	67.93	75.42
Seq2Path($k = 4$)	72.84	76.78	70.63	78.51
Seq2Path($k = 6$)	**74.29**	76.92	**71.84**	79.03
Seq2Path($k = 8$)	72.62	**77.35**	70.72	**79.09**
Seq2Path($k = 10$)	73.35	76.91	69.38	78.05

Table 2: Main results of the AOPE task with various beam sizes k. The best results are in bold and the second-best results are underlined.

On the ASTE task, our proposed Seq2Path outperforms the previous best results by 4.14, 3.36, 3.32, 1.97 in percentage on Laptop14, Rest14, Rest15, Rest16 respectively. The ASTE task is similar to the AOPE task, but ASTE is even harder as the sentiment s is also required. Again, our Seq2Path has a large performance gain.

	L14	R14	R15	R16
Jet-BERT	51.04	62.40	57.53	63.83
Dual-MRC	55.58	70.32	57.21	67.40
BMRC	59.27	70.69	61.05	68.13
GAS-T5	60.78	72.16	62.10	70.10
ParaPhrase-T5	61.13	72.03	62.56	71.70
Seq2Path($k = 4$)	64.09	74.29	65.42	**73.67**
Seq2Path($k = 6$)	**65.27**	73.00	**65.88**	71.62
Seq2Path($k = 8$)	64.20	74.88	64.89	72.67
Seq2Path($k = 10$)	64.82	**75.52**	65.88	72.87

Table 3: Main results of the ASTE task with various beam sizes k. The best results are in bold and the second-best results are underlined.

On the TASD task, our proposed Seq2Path outperforms the previous best results by 2.14, 0.13

in percentage on Rest15, Rest16 respectively. The challenge for the TASD task is that the aspect terms a can be "NULL", and an aspect a can have multiple categories c.

	R15	R16
TAS	58.09	65.89
GAS-T5	61.47	69.42
ParaPhrase-T5	63.06	$\underline{71.97}$
Seq2Path($k = 4$)	63.13	68.47
Seq2Path($k = 6$)	**65.20**	70.16
Seq2Path($k = 8$)	63.36	**72.10**
Seq2Path($k = 10$)	$\underline{63.89}$	69.23

Table 4: Main results of the TASD task with various beam sizes k. The best results are in bold and the second-best results are underlined.

On the UABSA task, our proposed Seq2Path outperforms the previous best results by 1.36, 1.67, 2.23 in percentage on Laptop14, Rest15, Rest16 respectively. The result on Rest14 is slightly lower (almost equal) than GAS-T5. The UABSA task is easier than other ABSA tasks since only two tuple elements (a, s) are extracted. One challenge is that the output can be a null set if there is no aspect in the input text. For such cases, it is most likely that all beams of Seq2Path will have a discriminative token $v =$ "false". Thus, our method is consistent with such a setting.

	L14	R14	R15	R16
RACL	63.40	75.42	66.05	-
Dual-MRC	65.94	75.95	65.08	-
BMRC	67.27	76.39	67.16	73.18
GAS-T5	68.64	**77.13**	66.78	73.64
Seq2Path($k = 4$)	**70.00**	77.01	$\underline{68.35}$	**75.87**
Seq2Path($k = 6$)	$\underline{69.94}$	76.07	67.71	$\underline{75.18}$
Seq2Path($k = 8$)	69.27	$\underline{77.10}$	68.33	74.96
Seq2Path($k = 10$)	69.08	76.01	**68.45**	73.73

Table 5: Main results of the UABSA task with various beam sizes k. The best results are in bold and the second-best results are underlined.

The ACOS task is newly published, and the original paper is the only baseline available. Our proposed Seq2Path outperforms the previous best results by 7.17, 13.80 in percentage on Laptop14, Rest16 respectively. This improvement is huge and should be partially from the power of T5.

4.3 Analysis

Analysis on the beam size. The main results in Table 2, 3, 4, 5, 6 use various beam sizes of $4, 6, 8, 10$. The beam size k is an important hyperparameter which affects both data augmentation and inference. The choice of the optimal k depends on the task and

	L14	R16
ACOS-Baseline	35.80	44.61
Seq2Path($k = 4$)	$\underline{42.60}$	57.72
Seq2Path($k = 6$)	41.45	$\underline{58.06}$
Seq2Path($k = 8$)	41.93	57.37
Seq2Path($k = 10$)	**42.97**	**58.41**

Table 6: Main results of the ACOS task with various beam sizes k. The best results are in bold and the second-best results are underlined.

the dataset. Roughly speaking, a smaller beam size will lead a worse recall while a larger beam size will lead a worse precision. Nevertheless, with our pruning process, our results are state-of-the-art regardless of the choice of k. Although beam search for inference will require a larger GPU memory, the Seq2Path can use a much shorter max output sequence length. Then, the memory consumption will be reduced by a lot.

Ablation study on data augmentation. The purpose of data augmentation is to generate negative samples for the discriminative token to automatically select the paths. The ablation results for data augmentation are shown in Table 7.

Task	Augment	L14	R14	R15	R16
	D_1	59.47	65.68	59.05	65.96
AOPE	D_2	73.05	**77.03**	69.91	77.83
	D_n	**74.29**	76.92	**71.84**	**79.03**
	D_1	54.98	65.68	57.69	64.90
ASTE	D_2	61.62	**73.29**	62.46	71.07
	D_n	**65.27**	73.00	**65.88**	**71.62**
	D_1	-	-	40.16	42.94
TASD	D_2	-	-	62.98	69.64
	D_n	-	-	**65.20**	**70.16**
	D_1	46.65	64.08	52.74	55.91
UABSA	D_2	69.26	**76.72**	**68.28**	72.79
	D_n	**69.94**	76.07	67.71	**75.18**
	D_1	29.08	-	-	37.39
ACOS	D_2	41.05	-	-	57.69
	D_n	**41.45**	-	-	**58.06**

Table 7: Ablation results for data augmentation. The beam size is fixed as $k = 6$. The datasets D_1, D_2, D_n are described in Section 2.2. All results are the F1 scores averaged over 5 runs with different random seeds. The best results are in bold.

The dataset D_1 has minor effects on the F1 scores. For most cases, adding D_1 can improve the F1 scores by up to 3%. It consists of negative samples by randomly replacing tuple elements and improves the model's ability to match tuple elements. The performance on Laptop14, Rest15 and Rest16 can benefit from D_1. However, D_1 seems to lead a slight performance drop on Rest14. One possible reason is that Rest14 has the biggest sample size and D_1 may not be necessary.

The dataset D_2 has major effects on the F1 scores. D_2 consists of negative samples generated by beam search from a fine-tuned model with small number of epochs. The F1 scores are significantly improved by adding D_2 because D_2 can guide the discriminative token to filter most of bad generations. For example, generating false repeated tokens is very common, and such bad cases can be handled here.

Case study. Figure 4 shows an example of beam search generation for the ASTE task with the beam size $k = 6$. The input sentence is "the staff was very nice and courteous and obviously chinese.". The probabilities are the sequence probabilities from beam search in decreasing order. The top 3 paths with $v =$ "true" are valid where they are marked as bold. The other 3 paths with $v =$ "false" are filtered. The tuple "staff, chinese, positive" is a valid one because the probability for the token "true" is larger than "false" $0.7114 > 0.4425$. The other paths such as "chinese stuff, nice, positive" are pruned for low probabilities.

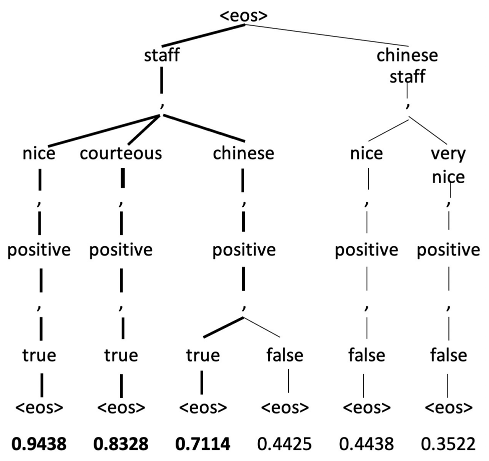


Figure 4: An example of beam search generation for the ASTE task with the beam size $k = 6$. The input sentence is "the staff was very nice and courteous and obviously chinese.". The probabilities at the bottom are the sequence probabilities returned from beam search. The paths with $v =$ "true" are marked as bold.

5 Related Work

There has been much work addressing technical solutions for ABSA. The main sentiment elements involved in ABSA include aspect term, opinion term, aspect category and sentiment polarity. In order to extract these sentiment elements, the main research direction of ABSA is to extract aspect terms (Liu et al., 2015; Yin et al., 2016; Li et al., 2018; Ma et al., 2019) and categorize the sentiment of a given aspect (Wang et al., 2016; Chen et al., 2017; Jiang et al., 2019; Zhang and Qian, 2020), and to jointly predict multiple elements simultaneously at the same time (Li et al., 2019; Wan et al., 2020; Peng et al., 2020; Zhao et al., 2020). Early ABSA problems were mostly expressed as sequence labeling or multi-classification problems (Li et al., 2019), which were predicted by designing task-specific classification networks and using the class index as labels for training (Huang and Carley, 2019; Wan et al., 2020). However, this approach requires the design of different classification models and ignores the label semantics. Recent works achieve good performance by converting the ABSA problems as a text generation problem (Yan et al., 2021; Zhang et al., 2021a,b).

The generative framework has been proven effective for some other natural language processing problems including dialogue state tracking (Feng et al., 2020), entity linking (De Cao et al., 2020), event extraction (Lu et al., 2021), information extraction (Sui et al., 2020), named entity recognition (Yan et al., 2021; Tan et al., 2021; Raffel et al., 2019; Athiwaratkun et al., 2020). In particular, (Paolini et al., 2021) solved various NLP tasks in a unified generative framework.

6 Conclusions

In this paper, we propose Seq2Path, a novel parallel generative framework for ABSA. The previous Seq2Seq based method formulates the output as a sequence that has two main drawbacks: the order and the dependence. Instead, our Seq2Path formulates the output as a tree and generates sentiment tuples as paths of the tree. Seq2Path can learn the precise conditional transition probability for token generation, by training with the loss of ordinary Seq2Seq averaged over paths. During inference, we apply beam search with constrained decoding. A discriminative token is also introduced to automatically select the valid paths. Experiments show that our model achieves state-of-the-art on AOPE, ASTE, TASD, UABSA, ACOS across common datasets including Laptop14, Rest14, Rest15, Rest16 in almost all cases. In the future, we plan to extend our method to other structure prediction tasks in NLP such as information extraction tasks, event extraction and nested named entity recognition.

References

Ben Athiwaratkun, Cicero Nogueira dos Santos, Jason Krone, and Bing Xiang. 2020. Augmented natural language for generative sequence labeling. *arXiv preprint arXiv:2009.13272*.

Hongjie Cai, Rui Xia, and Jianfei Yu. 2021. Aspect-category-opinion-sentiment quadruple extraction with implicit aspects and opinions. 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 340–350.

Peng Chen, Zhongqian Sun, Lidong Bing, and Wei Yang. 2017. Recurrent attention network on memory for aspect sentiment analysis. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 452–461.

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.

Shaowei Chen, Yu Wang, Jie Liu, and Yuelin Wang. 2021. Bidirectional machine reading comprehension for aspect sentiment triplet extraction. *arXiv preprint arXiv:2103.07665*.

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.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2020. Autoregressive entity retrieval. *arXiv preprint arXiv:2010.00904*.

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.

Yue Feng, Yang Wang, and Hang Li. 2020. A sequence-to-sequence approach to dialogue state tracking. *arXiv preprint arXiv:2011.09553*.

Binxuan Huang and Kathleen M Carley. 2019. Parameterized convolutional neural networks for aspect level sentiment classification. *arXiv preprint arXiv:1909.06276*.

Qingnan Jiang, Lei Chen, Ruifeng Xu, Xiang Ao, and Min Yang. 2019. A challenge dataset and effective models for aspect-based sentiment analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6280–6285.

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*.

Pengfei Liu, Shafiq Joty, and Helen Meng. 2015. Fine-grained opinion mining with recurrent neural networks and word embeddings. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1433–1443.

Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen. 2021. Text2event: Controllable sequence-to-structure generation for end-to-end event extraction. *arXiv preprint arXiv:2106.09232*.

Dehong Ma, Sujian Li, Fangzhao Wu, Xing Xie, and Houfeng Wang. 2019. Exploring sequence-to-sequence learning in aspect term extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3538–3547.

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*.

Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, Rishita Anubhai, Cicero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. 2021. Structured prediction as translation between augmented natural languages. *arXiv preprint arXiv:2101.05779*.

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.

M. Pontiki, D Galanis, J. Pavlopoulos, H. Papageorgiou, and S. Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. *Proceedings of International Workshop on Semantic Evaluation at*.

Maria Pontiki, Dimitrios Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad Al-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, et al. 2016. Semeval-2016 task 5: Aspect based sentiment analysis. In *International workshop on semantic evaluation*, pages 19–30.

Maria Pontiki, Dimitrios Galanis, Harris 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.

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*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Dianbo Sui, Yubo Chen, Kang Liu, Jun Zhao, Xiangrong Zeng, and Shengping Liu. 2020. Joint entity and relation extraction with set prediction networks. *arXiv preprint arXiv:2011.01675*.

Zeqi Tan, Yongliang Shen, Shuai Zhang, Weiming Lu, and Yueting Zhuang. 2021. A sequence-to-set network for nested named entity recognition. *arXiv preprint arXiv:2105.08901*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

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.

Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. 2016. Attention-based lstm for aspect-level sentiment classification. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 606–615.

Lu Xu, Hao Li, Wei Lu, and Lidong Bing. 2020. Position-aware tagging for aspect sentiment triplet extraction. *arXiv preprint arXiv:2010.02609*.

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*.

Yichun Yin, Furu Wei, Li Dong, Kaimeng Xu, Ming Zhang, and Ming Zhou. 2016. Unsupervised word and dependency path embeddings for aspect term extraction. *arXiv preprint arXiv:1605.07843*.

Mi Zhang and Tieyun Qian. 2020. Convolution over hierarchical syntactic and lexical graphs for aspect level sentiment analysis. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3540–3549.

Wenxuan Zhang, Yang Deng, Xin Li, Yifei Yuan, Lidong Bing, and Wai Lam. 2021a. Aspect sentiment quad prediction as paraphrase generation. *arXiv preprint arXiv:2110.00796*.

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 Constrained Decoding

A probability distribution is calculated over the whole vocabulary at each step of decoding. The candidate tokens can be restricted to a smaller set. For the input sentence x, the candidate tokens are the union of the original tokens in x and some extra task-specific candidate tokens

$$T(x, task) = T(x) \bigcup T(task) \tag{16}$$

where $T(x)$ stands for the token set for the input x and the task specific tokens $T(task)$ are defined in Table 8.

Task	$T(task)$
AOPE	sep
ASTE	sep, positive, negative, neutral
TASD	sep, positive, negative, neutral, all categories
UABSA	sep, positive, negative, neutral
ACOS	sep, positive, negative, neutral, all categories

Table 8: Task specific tokens for constrained decoding. The separator token is used to separate output tuples. The "positive", "negative" and "neutral" tokens are the sentiment polarities. For the TASD and ACOS tasks, categories should be included in the candidate tokens.

A.2 Pruning

We define the condition when two predictions are "overlapping" for a specific task in Table 9. If two predictions are overlapping, then we prefer the one with a higher probability.

Task	Prediction	Overlapping Condition
AOPE	(a, o)	$ovl(a_i, a_j), o_i = o_j$
		$ovl(o_i, o_j), a_i = a_j$
ASTE	(a, o, s)	$ovl(a_i, a_j), o_i = o_j$
		$ovl(o_i, o_j), a_i = a_j$
TASD	(c, a, s)	$ovl(a_i, a_j), c_i = c_j$
UABSA	(a, s)	$ovl(a_i, a_j)$
ACOS	(c, a, o, s)	$ovl(a_i, a_j), o_i = o_j, c_i = c_j$
		$ovl(o_i, o_j), a_i = a_j, c_i = c_j$

Table 9: The condition when two predictions are overlapping for various ABSA tasks. The letter a, o, c, s denotes the aspect, opinion, category, sentiment, respectively. The boolean function $ovl(\cdot)$ represents if two elements are overlapping.

A.3 Proof of Lemma 1

Proof: First, we consider the loss for the multi-hot vector. The cross-entropy loss for the target probability distribution $p \in \mathbb{R}^{|V|}$ and the predicted probability distribution $\hat{y}_t \in \mathbb{R}^{|V|}$ is

$$\begin{aligned} l(p, \hat{y}_t) &= -\sum_{i=1}^{n} p[i] \log \hat{y}_t[i] && (17) \\ &= -\sum_{i=1}^{k} \frac{1}{k} \log \hat{y}_t[j_i]. && (18) \end{aligned}$$

The equation holds because $p \in \mathbb{R}^{|V|}$ is "multi-hot" and $p[i] = 1$ if $i = j_i$ for $i = 1, 2, ..., k$. Now, we consider the loss average over paths. For i-th path $p'_i \in \mathbb{R}^{|V|}$,

$$\begin{aligned} l(p'_i, \hat{y}_t) &= -\sum_{\ell=1}^{n} p'_i[\ell] \log \hat{y}_t[\ell] && (19) \\ &= -\log \hat{y}_t[j_i]. && (20) \end{aligned}$$

The equation holds because $p'_i \in \mathbb{R}^{|V|}$ is "one-hot" and $p'_i[\ell] = 1$ if $\ell = j_i$. Therefore, it follows that

$$\frac{1}{k} \sum_{i=1}^{k} l(p'_i, \hat{y}_t) = l(p, \hat{y}_t) \tag{21}$$

and the proof is done. □