PASTE: A Tagging-Free Decoding Framework Using Pointer Networks for Aspect Sentiment Triplet Extraction

Rajdeep Mukherjee*, **Tapas Nayak***, **Yash Butala**, **Sourangshu Bhattacharya** and **Pawan Goyal**
Department of Computer Science and Engineering, IIT Kharagpur, India
{rajdeep1989, yashbutala}@iitkgp.ac.in, tnk02.05@gmail.com,
{sourangshu, pawang}@cse.iitkgp.ac.in

Abstract

Aspect Sentiment Triplet Extraction (ASTE) deals with extracting *opinion triplets*, consisting of an opinion target or aspect, its associated sentiment, and the corresponding opinion term/span explaining the rationale behind the sentiment. Existing research efforts are majorly *tagging*-based. Among the methods taking a *sequence tagging* approach, some fail to capture the strong interdependence between the three *opinion factors*, whereas others fall short of identifying triplets with overlapping aspect/opinion spans. A recent *grid tagging* approach on the other hand fails to capture the span-level semantics while predicting the sentiment between an aspect-opinion pair. Different from these, we present a *tagging-free* solution for the task, while addressing the limitations of the existing works. We adapt an encoder-decoder architecture with a *Pointer Network*-based decoding framework that generates an entire opinion triplet at each time step thereby making our solution end-to-end. Interactions between the aspects and opinions are effectively captured by the decoder by considering their entire detected spans while predicting their connecting sentiment. Extensive experiments on several benchmark datasets establish the better efficacy of our proposed approach, especially in *recall*, and in predicting multiple and aspect/opinion-overlapped triplets from the same review sentence. We report our results both with and without BERT and also demonstrate the utility of domain-specific BERT *post-training* for the task.

1 Introduction

Aspect-based Sentiment Analysis (ABSA) is a broad umbrella of several fine-grained sentiment analysis tasks, and has been extensively studied since its humble beginning in *SemEval 2014* (Pontiki et al., 2014a). Overall, the task revolves around automatically extracting the opinion targets or aspects being discussed in review sentences, along with the sentiments expressed towards them. Early efforts on *Aspect-level Sentiment Classification* (Tay et al., 2018; Li et al., 2018a; Xue and Li, 2018) focus on predicting the sentiment polarities for given aspects. However, in a real-world scenario, aspects may not be known a-priori. Works on *End-to-End ABSA* (Li et al., 2019; He et al., 2019; Chen and Qian, 2020) thus focus on extracting the aspects as well as the corresponding sentiments simultaneously. These methods do not however capture the reasons behind the expressed sentiments, which could otherwise provide valuable clues for more effective extraction of aspect-sentiment pairs.

Sent 1:	The film was good , but could have been better .
Triplets	[Aspect ; Opinion ; Sentiment] (1) film ; good ; positive (2) film ; could have been better ; negative
Sent 2:	The weather was gloomy , but the food was tasty .
Triplets	(1) weather ; gloomy ; negative (2) food ; tasty ; positive

Table 1: Examples of Aspect-Opinion-Sentiment triplets (*opinion triplets*) present in review sentences.

Consider the two examples shown in Table 1. For the first sentence, the sentiment associated with the aspect *film*, changes depending on the connecting opinion phrases; *good* suggesting a positive sentiment, and *could have been better* indicating a negative sentiment. Hence, simply extracting the pairs *film-positive*, and *film-negative* without additionally capturing the reasoning phrases may confuse the reader. For the second sentence, the opinion term *gloomy* has a higher probability of being associated with *weather*, than with *food*. We therefore observe that the three elements or *opinion factors* of an *opinion triplet* are strongly interdependent. In order to offer a complete picture of *what* is being discussed, *how* is the sentiment, and *why* is it so, (Peng et al., 2020) defined the task

*Equal contribution

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 9279–9291
November 7–11, 2021. ©2021 Association for Computational Linguistics

of **A**spect **S**entiment **T**riplet **E**xtraction (**ASTE**). Given an opinionated sentence, it deals with extracting all three elements: the aspect term/span, the opinion term/span, and the connecting sentiment in the form of *opinion triplets* as shown in Table 1. It is to be noted here that a given sentence might contain multiple triplets, which may further share aspect or opinion spans (For e.g., the two triplets for Sent. 1 in Table 1 share the aspect *film*). An efficient solution for the task must therefore be able to handle such challenging data points.

Peng et al. (2020) propose a two-stage pipeline framework. In the first stage, they extract aspect-sentiment pairs and opinion spans using two separate sequence-tagging tasks, the former leveraging a *unified tagging* scheme proposed by (Li et al., 2019), and the later based on *BIEOS*[1] tagging scheme. In the second stage, they pair up the extracted aspect and opinion spans, and use an MLP-based classifier to determine the validity of each generated triplet. Zhang et al. (2020) propose a multi-task framework to jointly detect aspects, opinions, and sentiment dependencies. Although they decouple the sentiment prediction task from aspect extraction, they use two separate sequence taggers (*BIEOS*-based) to detect the aspect and opinion spans in isolation before predicting the connecting sentiment. Both these methods however break the interaction between aspects and opinions during the extraction process. While the former additionally suffers from error propagation problem, the latter, relying on word-level sentiment dependencies, cannot guarantee sentiment consistency over multi-word aspect/opinion spans.

Xu et al. (2020b) propose a novel position-aware tagging scheme (extending *BIEOS* tags) to better capture the interactions among the three *opinion factors*. One of their model variants however cannot detect aspect-overlapped triplets, while the other cannot identify opinion-overlapped triplets. Hence, they need an ensemble of two variants to be trained for handling all cases. Wu et al. (2020) try to address this limitation by proposing a novel grid tagging scheme-based approach. However, they end up predicting the relationship between every possible word pair, irrespective of how they are syntactically connected, thereby impacting the span-level sentiment consistency guarantees.

Different from all these tagging-based methods, we propose to investigate the utility of a **tagging-free** scheme for the task. Our innovation lies in formulating ASTE as a structured prediction problem. Taking motivation from similar sequence-to-sequence approaches proposed for *joint entity-relation extraction* (Nayak and Ng, 2020; Chen et al., 2021), *semantic role labeling* (Fei et al., 2021) etc., we propose **PASTE**, a Pointer Network-based encoder-decoder architecture for the task of ASTE. The pointer network effectively captures the aspect-opinion interdependence while detecting their respective spans. The decoder then learns to model the span-level interactions while predicting the connecting sentiment. An entire *opinion triplet* is thus decoded at every time step, thereby making our solution end-to-end. We note here however, that the aspect and opinion spans can be of varying lengths, which makes the triplet decoding process challenging. For ensuring uniformity, we also propose a position-based representation scheme to be suitably exploited by our proposed architecture. Here, each *opinion triplet* is represented as a 5-point tuple, consisting of the start and end positions of the aspect and opinion spans, and the sentiment (POS/NEG/NEU) expressed towards the aspect. To summarize our contributions:

- We present an end-to-end *tagging-free* solution for the task of ASTE that addresses the limitations of previous tagging-based methods. Our proposed architecture, **PASTE**, not only exploits the aspect-opinion interdependence during the span detection process, but also models the span-level interactions for sentiment prediction, thereby truly capturing the inter-relatedness between all three elements of an *opinion triplet*.

- We propose a position-based scheme to uniformly represent an opinion triplet, irrespective of varying lengths of aspect and opinion spans.

- Extensive experiments on the ASTE-Data-V2 dataset (Xu et al., 2020b) establish the overall superiority of *PASTE* over strong state-of-the-art baselines, especially in predicting multiple and/or overlapping triplets. We also achieve significant (15.6%) recall gains in the process.

2 Our Approach

Given the task of ASTE, our objective is to jointly extract the three elements of an opinion triplet, i.e., the aspect span, its associated sentiment, and the

[1]BIOES is a commonly used tagging scheme for sequence labeling tasks, and denotes “begin, inside, outside, end and single” respectively.

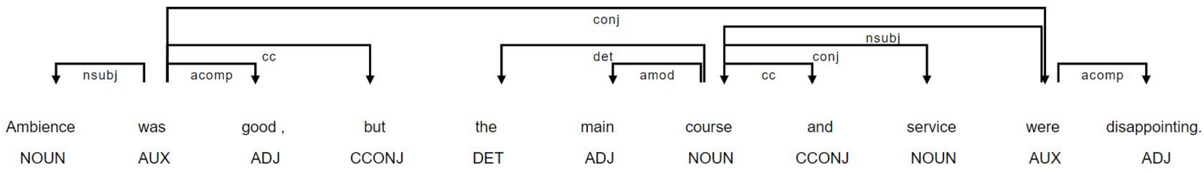

Figure 1: Dependency Parse Tree for the example review sentence in Table 2

Sentence	Ambience was good , but the main course and service were disappointing .
Target Triplets	(0 0 2 2 POS) (6 7 11 11 NEG) (9 9 11 11 NEG)
Overlapping Triplets	(6 7 11 11 NEG) (9 9 11 11 NEG)

Table 2: Triplet representation for Pointer-network based decoding

corresponding opinion span, while modeling their interdependence. Towards this goal, we first introduce our triplet representation scheme, followed by our problem formulation. We then present our Pointer Network-based decoding framework, **PASTE**, and finally discuss a few model variants. Through exhaustive experiments, we investigate the utility of our approach and present a performance comparison with strong state-of-the-art baselines.

2.1 Triplet Representation

In order to address the limitations of *BIEOS* tagging-based approaches and to facilitate joint extraction of all three elements of an *opinion triplet*, we represent each triplet as a 5-point tuple, consisting of the start and end positions of the aspect span, the start and end positions of the opinion span, and the sentiment (POS/NEG/NEU) expressed towards the aspect. This allows us to model the relative context between an aspect-opinion pair which is not possible if they were extracted in isolation. It further helps to jointly extract the sentiment associated with such a pair. An example sentence with triplets represented under the proposed scheme is shown in Table 2. As may be noted, such a scheme can easily represent triplets with overlapping aspect or opinion spans, possibly with varying lengths.

2.2 Problem Formulation

To formally define the ASTE task, given a review sentence $s = \{w_1, w_2, ..., w_n\}$ with n words, our goal is to extract a set of opinion triplets $T = \{t_i \mid t_i = [(s_i^{ap}, e_i^{ap}), (s_i^{op}, e_i^{op}), senti_i]\}_{i=1}^{|T|}$, where t_i represents the i^{th} triplet and $|T|$ represents the length of the triplet set. For the i^{th} triplet, s_i^{ap} and e_i^{ap} respectively denote the start and end positions of its constituent aspect span, s_i^{op} and e_i^{op} respectively denote the start and end positions of its constituent opinion span, and $senti_i$ represents the sentiment polarity associated between them. Here, $senti_i \in \{POS, NEU, NEU\}$, where POS, NEG, and NEU respectively represent the *positive*, *negative*, and *neutral* sentiments.

2.3 The PASTE Framework

We now present **PASTE**, our **P**ointer network-based decoding framework for the task of **A**spect **S**entiment **T**riplet **E**xtraction. Figure 2 gives an overview of our proposed architecture.

2.3.1 Sentence Encoder

As previously motivated, the association between an aspect, an opinion, and their connecting sentiment is highly contextual. This factor is more noteworthy in sentences containing multiple triplets with/without varying sentiment polarities and/or overlapping aspect/opinion spans. Long Short Memory Networks (or LSTMs) (Hochreiter and Schmidhuber, 1997) are known for their context modeling capabilities. Similar to (Nayak and Ng, 2020; Chen et al., 2021), we employ a Bi-directional LSTM (Bi-LSTM) to encode our input sentences. We use pre-trained word vectors of dimension d_w to obtain the word-level features. We then note from Figure 1 that aspect spans are often characterized by *noun phrases*, whereas opinion spans are often composed of *adjective phrases*. Referring to the dependency tree in the same figure, the aspect and the opinion spans belonging to the same opinion triplet are often connected by the same *head word*. These observations motivate us to use both part-of-speech (POS) and dependency-based (DEP) features for each word.

More specifically, we use two embedding layers, $E_{pos} \in \mathbb{R}^{|\text{POS}| \times d_{pos}}$, and $E_{dep} \in \mathbb{R}^{|\text{DEP}| \times d_{dep}}$ to obtain the POS and DEP-features of dimensions d_{pos} and d_{dep} respectively, with $|\text{POS}|$ and $|\text{DEP}|$ representing the length of POS-tag and DEP-tag

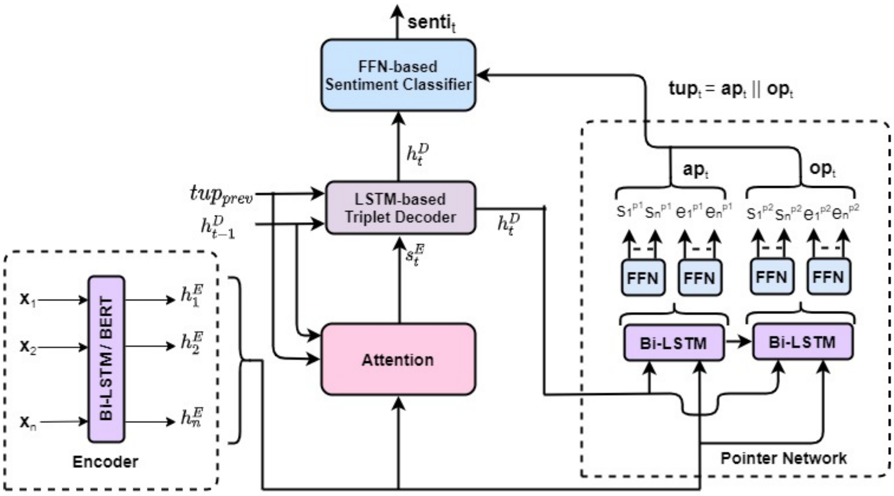

Figure 2: Model architecture of PASTE, a Pointer Network-based decoding framework for ASTE.

sets over all input sentences. All three features are concatenated to obtain the input vector representation $\mathbf{x_i} \in \mathbb{R}^{d_w+d_{pos}+d_{dep}}$ corresponding to the i^{th} word in the given sentence $S = \{w_1, w_2, ..., w_n\}$. The vectors are passed through the Bi-LSTM to obain the contextualized representations $\mathbf{h_i^E} \in \mathbb{R}^{d_h}$. Here, d_h represents the hidden state dimension of the triplet generating LSTM decoder as detailed in the next section. Accordingly, the hidden state dimension of both the forward and backward LSTM of the Bi-LSTM encoder are set to $d_h/2$.

For the BERT-based variant of our model, Bi-LSTM gets replaced by BERT (Devlin et al., 2019) as the sentence encoder. The pre-trained word vectors are accordingly replaced by BERT token embeddings. We now append the POS and DEP features vectors to the 768-dim. token-level outputs from the final layer of BERT.

2.3.2 Pointer Network-based Decoder

Referring to Figure 2, opinion triplets are decoded using an LSTM-based *Triplet Decoder*, that takes into account the history of previously generated pairs/tuples of aspect and opinion spans, in order to avoid repetition. At each time step t, it generates a hidden representation $\mathbf{h_t^D} \in \mathbb{R}^{d_h}$ that is used by the two Bi-LSTM + FFN-based *Pointer Networks* to respectively predict the aspect and opinion spans, while exploiting their interdependence. The tuple representation tup_t thus obtained is concatenated with $\mathbf{h_t^D}$ and passed through an FFN-based *Sentiment Classifier* to predict the connecting sentiment, thereby decoding an entire opinion triplet at the t^{th} time step. We now elaborate each component of our proposed decoder framework in greater depth:

Span Detection with Pointer Networks

Our pointer network consists of a Bi-LSTM, with hidden dimension d_p, followed by two feed-forward layers (FFN) on top to respectively predict the start and end locations of an entity span. We use two such pointer networks to produce a tuple of hidden vectors corresponding to the aspect and opinion spans of the triplet to be decoded at time step t. We concatenate $\mathbf{h_t^D}$ with each of the encoder hidden state vectors $\mathbf{h_i^E}$ and pass them as input to the first Bi-LSTM. The output hidden state vector corresponding to the i^{th} token of the sentence thus obtained is simultaneously fed to the two FFNs with *sigmoid* to generate a pair of scores in the range of 0 to 1. After repeating the process for all tokens, the normalized probabilities of the i^{th} token to be the start and end positions of an *aspect span* ($s_i^{p_1}$ and $e_i^{p_1}$ respectively) are obtained using *softmax* operations over the two sets of scores thus generated. Here p_1 refers to the first pointer network. Similar scores corresponding to the *opinion span* are obtained using the second pointer network, p_2; difference being that apart from $\mathbf{h_t^D}$, we also concatenate the output vectors from the first Bi-LSTM with encoder hidden states $\mathbf{h_i^E}$ and pass them as input to the second Bi-LSTM. This helps us to model the interdependence between an aspect-opinion pair. These scores are used to obtain the hidden state representations $ap_t \in \mathbb{R}^{2d_p}$ and $op_t \in \mathbb{R}^{2d_p}$ corresponding to the pair of aspect and opinion spans thus predicted at time step t. We request our readers to kindly refer to the appendix for more elaborate implementation details.

Here we introduce the term *generation direc-*

tion which refers to the order in which we generate the hidden representations for the two entities, i.e. aspect and opinion spans. This allows us to define two **variants** of our model. The variant discussed so far uses p_1 to detect the aspect span before predicting the opinion span using p_2, and is henceforth referred to as **PASTE-AF** (AF stands for *aspect first*). Similarly, we obtain the second variant **PASTE-OF** (*opinion first*) by reversing the *generation direction*. The other two components of our model remain the same for both the variants.

Triplet Decoder and Attention Modeling

The decoder consists of an LSTM with hidden dimension d_h whose goal is to generate the sequence of opinion triplets, T, as defined in Section 2.2. Let $tup_t = \mathrm{ap_t} \parallel \mathrm{op_t}$; $\mathrm{tup_t} \in \mathbb{R}^{4d_p}$ denote the tuple (aspect, opinion) representation obtained from the pointer networks at time step t. Then, $tup_{prev} = \sum_{j<t} tup_j$; $tup_0 = \vec{0} \in \mathbb{R}^{4d_p}$ represents the cumulative information about all tuples predicted before the current time step. We obtain an attention-weighted context representation of the input sentence at time step t ($s_t^E \in \mathbb{R}^{d_h}$) using Bahdanau et al. (2015) *Attention*[2]. In order to prevent the decoder from generating the same tuple again, we pass tup_{prev} as input to the LSTM along with s_t^E to generate $\mathbf{h_t^D} \in \mathbb{R}^{d_h}$, the hidden representation for predicting the triplet at time step t:

$$\mathbf{h_t^D} = \mathbf{LSTM}(s_t^E \parallel tup_{prev} \, , \, \mathbf{h_{t-1}^D})$$

Sentiment Classifier

Finally, we concatenate tup_t, with $\mathbf{h_t^D}$ and pass it through a feed-forward network with softmax to generate the normalized probabilities over $\{POS, NEG, NEU\} \cup \{NONE\}$, thereby predicting the sentiment label $senti_t$ for the current triplet. Interaction between the entire predicted spans of aspect and opinion is thus captured for sentiment identification. Here POS, NEG, NEU respectively represent the positive, negative, and neutral sentiments. $NONE$ is a dummy sentiment that acts as an implicit stopping criteria for the decoder. During training, once a triplet with sentiment $NONE$ is predicted, we ignore all subsequent predictions, and none of them contribute to the loss. Similarly, during inference, we ignore any triplet predicted with the $NONE$ sentiment.

[2]Please refer to the appendix for implementation details.

2.3.3 Training

For training our model, we minimize the sum of negative log-likelihood loss for classifying the sentiment and the four pointer locations corresponding to the aspect and opinion spans:

$$\mathcal{L} = -\frac{1}{M \times J} \sum_{m=1}^{M} \sum_{j=1}^{J} [log(s_{ap,j}^m \cdot e_{ap,j}^m) + log(s_{op,j}^m \cdot e_{op,j}^m) + log(sen_j^m)]$$

Here, m represents the m^{th} training instance with M being the batch size, j represents the j^{th} decoding time step with J being the length of the longest target sequence among all instances in the current batch. s_p, e_p; $p \in \{ap, op\}$ and sen respectively represent the softmax scores corresponding to the true start and end positions of the aspect and opinion spans and their associated true sentiment label.

2.3.4 Inferring The Triplets

Let $s_i^{ap}, e_i^{ap}, s_i^{op}, e_i^{op}$; $i \in [1, n]$ represent the obtained pointer probabilities for the i^{th} token in the given sentence (of length n) to be the start and end positions of an aspect span and opinion span respectively. First, we choose the start (j) and end (k) positions of the aspect span with the constraint $1 \le j \le k \le n$ such that $s_j^{ap} \times e_k^{ap}$ is maximized. We then choose the start and end positions of the opinion span similarly such that they do not overlap with the aspect span. Thus, we obtain one set of four pointer probabilities. We repeat the process to obtain the second set, this time by choosing the opinion span before the aspect span. Finally, we choose the set (of aspect and opinion spans) that gives the higher product of the four probabilities.

3 Experiments

3.1 Datasets and Evaluation Metrics

We conduct our experiments on the **ASTE-Data-V2** dataset created by Xu et al. (2020b). It is derived from **ASTE-Data-V1** (Peng et al., 2020) and presents a more challenging scenario with **27.68%** of all sentences containing triplets with overlapping aspect or opinion spans. The dataset contains triplet-annotated sentences from two domains: laptop and restaurant, corresponding to the original datasets released by the SemEval Challenge (Pontiki et al., 2014a,b,c). It is to be noted here that the opinion term annotations were originally derived from (Fan et al., 2019). *14Lap* belongs to

Dataset	14Lap			14Rest			15Rest			16Rest			Restaurant (All)		
	# Pos.	# Neg.	# Neu.	# Pos.	# Neg.	# Neu.	# Pos.	# Neg.	# Neu.	# Pos.	# Neg.	# Neu.	# Pos.	# Neg.	# Neu.
Train	817	517	126	1692	480	166	783	205	25	1015	329	50	3490	1014	241
Dev	169	141	36	404	119	54	185	53	11	252	76	11	841	248	76
Test	364	116	63	773	155	66	317	143	25	407	78	29	1497	376	120

Table 3: ASTE-Data-V2 Statistics: # Triplets with various sentiment polarities

Dataset	Laptop					Restaurant				
	Single	Multi	MultiPol	Overlap	# Sent.	Single	Multi	MultiPol	Overlap	# Sent.
Train	545	361	47	257	906	1447	1281	205	731	2728
Dev	133	86	10	59	219	347	321	45	197	668
Test	184	144	18	97	328	608	532	71	317	1140
Total	**862**	**591**	**75**	**413**	**1453**	**2402**	**2134**	**321**	**1245**	**4536**

Table 4: Statistics of *Laptop* and *Restaurant* datasets from ASTE-Data-V2: *Single* and *Multi* respectively represent # sentences with single and multiple triplets. *MultiPol* and *Overlap* are subsets of *Multi*. *MultiPol* representing # sentences containing at least two triplets with different sentiment polarities. *Overlap* represents # sentences with aspect/opinion overlapped triplets. # Sent. represents the total no. of sentences overall.

the laptop domain and is henceforth referred to as the *Laptop*. *14Rest*, *15Rest*, and *16Rest* belong to the restaurant domain. Each dataset comes with its pre-defined split of training, development, and test sets. Similar to prior works, we report our results on the individual datasets. Additionally, we also conduct experiments on the combined restaurant dataset, henceforth referred to as the *Restaurant*. Tables 3 and 4 present the dataset statistics.

We consider *precision*, *recall*, and *micro-F1* as our evaluation metrics for the triplet extraction task. A predicted triplet is considered a true positive only if all three predicted elements exactly match with those of a ground-truth opinion triplet.

3.2 Experimental Setup

For our non-BERT experiments, word embeddings are initialized (and kept trainable) using pre-trained 300-dim. Glove vectors (Pennington et al., 2014), and accordingly d_w is set to 300. d_{pos} and d_{dep} are set to 50 each. d_h is set to 300, and accordingly the hidden state dimensions of both the LSTMs (backward and forward) of the Bi-LSTM-based encoder are set to 150 each. d_p is set to 300. For our BERT experiments, *uncased* version of pre-trained BERT-base (Devlin et al., 2019) is fine-tuned to encode each sentence. All our model variants are trained end-to-end on Tesla P100-PCIE 16GB GPU with *Adam* optimizer (learning rate: 10^{-3}, weight decay: 10^{-5}). A *dropout* rate of 0.5 is applied on the embeddings to avoid overfitting[3]. We make our codes and datasets publicly available[4].

[3]Please refer to the appendix for more details.
[4]https://github.com/rajdeep345/PASTE/

3.3 Baselines

- Wang et al. (2017) (**CMLA**) and Dai and Song (2019) (**RINANTE**) propose different methods to co-extract aspects and opinion terms from review sentences. Li et al. (2019) propose a unified tagging scheme-based method for extracting opinion target-sentiment pairs. Peng et al. (2020) modifies these methods to jointly extract targets with sentiment, and opinion spans. It then applies an MLP-based classifier to determine the validity of all possible generated triplets. These modified versions are referred to as **CMLA+**, **RINANTE+**, and **Li-unified-R**, respectively.

- Peng et al. (2020) propose a BiLSTM+GCN-based approach to co-extract aspect-sentiment pairs, and opinion spans. They then use the same inference strategy as above to confirm the correctness of the generated triplets.

- **OTE-MTL** (Zhang et al., 2020) uses a multi-task learning framework to jointly detect aspects, opinions, and sentiment dependencies.

- **JET** (Xu et al., 2020b) is the first end-to-end approach for the task of ASTE that leverages a novel position-aware tagging scheme. One of their variants, **JET$^{\mathbf{t}}$**, however cannot handle aspect-overlapped triplets. Similarly, **JET$^{\mathbf{o}}$**, cannot handle opinion-overlapped triplets.

- **GTS** (Wu et al., 2020) models ASTE as a novel grid-tagging task. However, given that it predicts the sentiment relation between all possible word pairs, it uses a relaxed (majority-based) matching criteria to determine the final triplets.

Model	Laptop				Restaurant			
	P.	R.	F_1	Dev F_1	P.	R.	F_1	Dev F_1
CMLA$^+$	0.301	0.369	0.332	-	-	-	-	-
RINANTE$^+$	0.217	0.187	0.201	-	-	-	-	-
Li-unified-R	0.406	0.443	0.423	-	-	-	-	-
(Peng et al., 2020)	0.374	0.504	0.429	-	-	-	-	-
JET^o (M = 4)	0.546	0.354	0.429	0.457	0.770	0.520	0.621	0.641
JET^o (M = 5)	0.560	0.354	0.433	0.458	-	-	-	-
OTE-MTL	0.492	0.405	0.451	0.458	0.710	0.579	0.637	0.729
GTS-BiLSTM w/o DE	0.597	0.348	0.439	0.465	0.768	0.629	0.692	0.748
PASTE-AF	0.537	0.486	**0.510**	0.496	0.707	0.701	0.704	0.741
PASTE-OF	0.521	0.481	0.500	0.482	0.707	0.706	**0.707**	0.740
With BERT								
JET^o (M = 4)	0.570	0.389	0.462	0.475	0.727	0.549	0.626	0.645
JET^o (M = 6)	0.554	0.473	0.510	0.488	-	-	-	-
GTS-BERT	0.549	0.521	0.535	0.579	0.748	0.732	**0.740**	0.767
PASTE-AF	0.550	0.516	0.532	0.514	0.710	0.704	0.707	0.744
w/ BERT-PT	0.612	0.536	0.571	0.576	0.747	0.718	0.732	0.759
PASTE-OF	0.554	0.519	**0.536**	0.503	0.705	0.705	0.705	0.744
w/ BERT-PT	0.597	0.553	**0.574**	0.547	0.737	0.737	0.737	0.759

Table 5: Comparative results on the *Laptop* (14Lap) and *Restaurant* datasets from ASTE-Data-V2. **Bolded** values represent the best F_1 scores. Underlined scores are obtained with Post-trained BERT.

Model	14Rest				15Rest				16Rest			
	P.	R.	F_1	Dev F_1	P.	R.	F_1	Dev F_1	P.	R.	F_1	Dev F_1
CMLA$^+$	0.392	0.471	0.428	-	0.346	0.398	0.370	-	0.413	0.421	0.417	-
RINANTE$^+$	0.314	0.394	0.350	-	0.299	0.301	0.300	-	0.257	0.223	0.239	-
Li-unified-R	0.410	0.674	0.510	-	0.447	0.514	0.478	-	0.373	0.545	0.443	-
(Peng et al., 2020)	0.432	0.637	0.515	-	0.481	0.575	0.523	-	0.470	0.642	0.542	-
OTE-MTL	0.630	0.551	0.587	0.547	0.579	0.427	0.489	0.569	0.603	0.534	0.565	0.597
JET^o $(M = 6)$	0.615	0.551	0.581	0.535	0.644	0.443	0.525	0.610	0.709	0.570	**0.632**	0.609
GTS-BiLSTM w/o DE	0.686	0.528	0.597	0.556	0.654	0.443	0.528	0.606	0.686	0.515	0.588	0.625
PASTE-AF	0.624	0.618	0.621	0.568	0.548	0.534	**0.541**	0.649	0.622	0.628	0.625	0.667
PASTE-OF	0.634	0.619	**0.626**	0.566	0.548	0.526	0.537	0.650	0.623	0.636	0.629	0.659
With BERT												
JET^o (M = 6)	0.706	0.559	0.624	0.569	0.645	0.520	0.575	0.648	0.704	0.584	0.638	0.638
GTS-BERT	0.674	0.673	**0.674**	0.651	0.637	0.551	**0.591**	0.720	0.654	0.680	**0.667**	0.715
PASTE-AF	0.648	0.638	0.643	0.570	0.583	0.567	0.575	0.626	0.655	0.644	0.650	0.660
w/ BERT-PT	0.667	0.665	0.666	0.585	0.617	0.608	0.613	0.673	0.661	0.698	0.679	0.690
PASTE-OF	0.667	0.608	0.636	0.573	0.585	0.565	0.575	0.645	0.619	0.667	0.642	0.670
w/ BERT-PT	0.687	0.638	0.661	0.592	0.636	0.598	0.616	0.660	0.680	0.677	0.678	0.695

Table 6: Comparative results on the individual restaurant datasets from ASTE-Data-V2

3.4 Experimental Results

While training our model variants, the best weights are selected based on F_1 scores on the development set. We report our median scores over 5 runs of the experiment. Performance comparisons on the *Laptop* (14Lap) and combined *Restaurant* datasets are reported in Table 5, whereas the same on individual restaurant datasets are reported in Table 6. Both the tables are divided into two sections; the former comparing the results without BERT, and the latter comparing those with BERT. The scores for **CMLA$^+$**, **RINANTE$^+$**, **Li-unified-R**, and (Peng et al., 2020) are taken from Xu et al. (2020b). We replicate the results for **OTE-MTL** on ASTE-Data-V2 and report their average scores over 10 runs of the experiment. For **JET**, we compare with their best reported results on the individual datasets; i.e. JET^o (M = 5) for 14Lap (w/o BERT), JET^o (M = 6) for 14Lap (w/ BERT), and JET^o (M = 6) for 14Rest, 15Rest, and 16Rest (both w/ and w/o BERT). However, owing to resource constraints and known optimization issues with their codes, we could not replicate their results on the *Restaurant* dataset beyond M = 4 (for both w/ and w/o BERT). **GTS** uses double embeddings (Xu et al., 2018) (general Glove vectors + domain-specific embeddings trained with *fastText*). For fair comparison, we replicate their results without using the domain-specific embeddings (DE). For both w/ and w/o BERT, we report their median scores over 5 runs of the experiment. We also report the F_1 scores on the development set corresponding to the test set results.

Model	Laptop				Restaurant			
	Single	Multi	MultiPol	Overlap	Single	Multi	MultiPol	Overlap
JET^o (M = 4)	0.453	0.406	0.219	0.363	0.654	0.602	0.558	0.518
OTE-MTL	0.485	0.277	0.172	0.380	0.716	0.656	0.506	0.646
GTS-BiLSTM w/o DE	0.418	0.452	**0.237**	0.403	**0.726**	0.675	**0.588**	0.660
PASTE-AF	**0.506**	**0.512**	0.216	0.507	0.702	**0.705**	0.567	0.688
PASTE-OF	0.495	0.502	0.205	**0.511**	0.711	0.704	0.582	**0.693**
With BERT								
JET^o (M = 4)	0.514	0.430	0.229	0.400	0.655	0.609	0.509	0.536
GTS-BERT	0.533	**0.536**	**0.338**	**0.540**	**0.739**	**0.740**	**0.648**	**0.722**
PASTE-AF	0.555	0.519	0.265	0.526	0.704	0.709	0.601	0.699
PASTE-OF	**0.593**	0.502	0.282	0.511	0.699	0.708	0.571	0.697

Table 7: Comparison of F1 scores on different splits of *Laptop* and *Restaurant* datasets from ASTE-Data-V2

From Table 5, both our variants, PASTE-AF and PASTE-OF, perform comparably as we substantially outperform all the non-BERT baselines. On *Laptop*, we achieve **13.1%** F_1 gains over OTE-MTL, whereas on *Restaurant*, we obtain **2.2%** F_1 gains over GTS-BiLSTM. We draw similar conclusions from Table 6, except that we are narrowly outperformed by JET^o (M = 6) on 16Rest. Our better performance may be attributed to our better *Recall* scores with around **15.6% recall gains** (averaged across both our variants) over the respective strongest baselines (in terms of F_1) on the *Laptop* and *Restaurant* datasets. Such an observation establishes the better efficacy of PASTE in modeling the interactions between the three *opinion factors* as we are able to identify more ground-truth triplets from the data, compared to our baselines.

With BERT, we comfortably outperform JET on all the datasets. Although we narrowly beat GTS-BERT on *Laptop*, it outperforms us on all the restaurant datasets. This is owing to the fact that GTS-BERT obtains a substantial improvement in scores over GTS since its grid-tag prediction task and both the pre-training tasks of BERT are all discriminative in nature. We on the other hand, do not observe such huge jumps (F_1 gains of 5.1%, 2.7%, 6.3%, and 3.3% on the *Laptop*, Rest14, Rest15, and Rest16 datasets respectively, noticeably more improvement on datasets with lesser training data; no gains on *Restaurant*) since BERT is known to be unsuitable for generative tasks. We envisage to improve our model by replacing BERT with BART (Lewis et al., 2020), a strong sequence-to-sequence pretrained model for NLG tasks.

Finally, motivated by Xu et al. (2019, 2020a), we also demonstrate the utility of leveraging domain-specific language understanding for the task by reporting our results with BERT-PT (task-agnostic post-training of pre-trained BERT on domain-specific data) in both the tables. While we achieve substantial performance improvement, we do not use these scores to draw our conclusions in order to ensure fair comparison with the baselines.

4 Analysis & Discussion

4.1 Robustness Analysis

In order to better understand the relative advantage of our proposed approach when compared to our baselines for the opinion triplet extraction task, and to further investigate the reason behind our better recall scores, in Table 7 we compare the F_1 scores on various splits of the test sets as defined in Table 4. We observe that with our core architecture (w/o BERT), PASTE consistently outperforms the baselines on both *Laptop* and *Restaurant* datasets when it comes to handling sentences with multiple triplets, especially those with overlapping aspect/opinion spans. This establishes the fact that PASTE is better than previous tagging-based approaches in terms of modeling aspect-opinion span-level interdependence during the extraction process. This is an important observation considering the industry-readiness (Mukherjee et al., 2021b) of our proposed approach since our model is robust towards challenging data instances. We however perform poorly when it comes to identifying triplets with varying sentiment polarities in the same sentence. This is understandable since we do not utilize any specialized sentiment modeling technique. In future, we propose to utilize word-level Valence, Arousal, Dominance scores (Mukherjee et al., 2021a) as additional features to better capture the sentiment of the opinion phrase.

In this work, we propose a new perspective to solve ASTE by investigating the utility of a tagging-free scheme, as against all prior tagging-based methods. Hence, it becomes imperative to analyze how we perform in terms of identifying individual

Dataset	Model	Aspect			Opinion			Sentiment
		P.	R.	F_1	P.	R.	F_1	% Acc.
Laptop	JET^o (M = 4)	0.801	0.495	0.611	0.805	0.528	0.638	0.846
	OTE-MTL	0.812	0.576	0.674	0.826	0.584	0.684	0.858
	GTS-BiLSTM w/o DE	0.725	0.724	0.724	0.692	0.684	0.688	**0.870**
	PASTE-AF	0.792	0.765	0.778	0.757	0.704	0.730	0.840
	PASTE-OF	0.801	0.790	**0.796**	0.763	0.719	**0.740**	0.831
Restaurant	JET^o (M = 4)	0.871	0.638	0.736	0.885	0.666	0.760	**0.947**
	OTE-MTL	0.905	0.706	0.793	0.913	0.718	0.804	0.943
	GTS-BiLSTM w/o DE	0.791	0.835	0.812	0.826	0.837	0.832	0.945
	PASTE-AF	0.837	0.851	**0.844**	0.844	0.852	0.848	0.939
	PASTE-OF	0.836	0.848	0.842	0.848	0.854	**0.851**	0.939

Table 8: Comparative results of aspect, opinion and sentiment prediction on *Laptop* and *Restaurant* datasets

Dataset	Model	P.	R.	F_1	% $F_1 \downarrow$
Laptop	PASTE-AF	0.537	0.486	0.510	-
	- POS & DEP	0.530	0.451	0.488	4.3%
	w/ Random	0.505	0.410	0.453	11.2%
Restaurant	PASTE-OF	0.707	0.706	0.707	-
	- POS & DEP	0.708	0.702	0.705	0.3%
	w/ Random	0.686	0.627	0.655	7.4%

Table 9: Ablation Results

elements of an opinion triplet. Table 8 presents such a comparison. It is encouraging to note that we substantially outperform our baselines on both aspect and opinion span detection sub-tasks. However, as highlighted before, we are outperformed when it comes to sentiment detection.

4.2 Ablation Study:

Since our *Decoder* learns to decode the sequence of triplets from left to right without repetition, while training our models we sort the target triplets in the same order as *generation direction*; i.e. for training PASTE-AF/PASTE-OF, the target triplets are sorted in ascending order of aspect/opinion start positions. As an ablation, we sort the triplets randomly while training the models and report our obtained scores in Table 9. An average drop of 9.3% in F_1 scores for both our model variants establish the importance of sorting the triplets for training our models. When experimenting without the POS and DEP features, we further observe an average drop of 2.3% in F_1 scores, thereby demonstrating their utility for the ASTE task. When experimenting with BERT, although these features helped on the *Laptop* and Rest15 datasets, overall we did not observe any significant improvement.

5 Related Works

ABSA is a collection of several fine-grained sentiment analysis tasks, such as *Aspect Extraction* (Li et al., 2018b, 2020), *Aspect-level Sentiment Classification* (Li et al., 2018a; Xue and Li, 2018), *Aspect-oriented Opinion Extraction* (Fan et al., 2019), *E2E-ABSA* (Li et al., 2019; He et al., 2019), and *Aspect-Opinion Co-Extraction* (Wang et al., 2017; Dai and Song, 2019). However, none of these works offer a complete picture of the aspects being discussed. Towards this end, Peng et al. (2020) recently coined the task of Aspect Sentiment Triplet Extraction (ASTE), and proposed a 2-stage pipeline solution. More recent end-to-end approaches such as OTE-MTL(Zhang et al., 2020), and GTS (Wu et al., 2020) fail to guarantee sentiment consistency over multi-word aspect/opinion spans, since they depend on word-pair dependencies. JET (Xu et al., 2020b) on the other hand requires two different models to be trained to detect aspect-overlapped and opinion-overlapped triplets. Different from all these tagging-based methods, we propose a tagging-free solution for the ASTE task.

6 Conclusion

We investigate the utility of a tagging-free scheme for the task of Aspect Sentiment Triplet Extraction using a Pointer network-based decoding framework. Addressing the limitations of previous tagging-based methods, our proposed architecture, **PASTE**, not only exploits the aspect-opinion interdependence during the span detection process, but also models the span-level interactions for sentiment prediction, thereby truly capturing the interrelatedness between all three elements of an opinion triplet. We demonstrate the better efficacy of PASTE, especially in *recall*, and in predicting multiple and/or overlapping triplets, when experimenting on the *ASTE-Data-V2* dataset.

Acknowledgements

This research is supported by IMPRINT-2, Science and Engineering Research Board (SERB), India.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Yubo Chen, Yunqi Zhang, Changran Hu, and Yongfeng Huang. 2021. Jointly extracting explicit and implicit relational triples with reasoning pattern enhanced binary pointer network. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5694–5703, Online. Association for Computational Linguistics.

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, Online. Association for Computational Linguistics.

Hongliang Dai and Yangqiu Song. 2019. Neural aspect and opinion term extraction with mined rules as weak supervision. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5268–5277, Florence, Italy. Association for Computational Linguistics.

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

Zhifang Fan, Zhen Wu, Xin-Yu 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, Minneapolis, Minnesota. Association for Computational Linguistics.

Hao Fei, Fei Li, Bobo Li, and Donghong Ji. 2021. Encoder-decoder based unified semantic role labeling with label-aware syntax. volume 35, pages 12794–12802.

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, Florence, Italy. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Kun Li, Chengbo Chen, Xiaojun Quan, Qing Ling, and Yan Song. 2020. Conditional augmentation for aspect term extraction via masked sequence-to-sequence generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7056–7066, Online. Association for Computational Linguistics.

Xin Li, Lidong Bing, Wai Lam, and Bei Shi. 2018a. Transformation networks for target-oriented sentiment classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 946–956, Melbourne, Australia. Association for Computational Linguistics.

Xin Li, Lidong Bing, Piji Li, and Wai Lam. 2019. A unified model for opinion target extraction and target sentiment prediction. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6714–6721. AAAI Press.

Xin Li, Lidong Bing, Piji Li, Wai Lam, and Zhimou Yang. 2018b. Aspect term extraction with history attention and selective transformation. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 4194–4200. ijcai.org.

Rajdeep Mukherjee, Atharva Naik, Sriyash Poddar, Soham Dasgupta, and Niloy Ganguly. 2021a. Understanding the role of affect dimensions in detecting emotions from tweets: A multi-task approach. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 2303–2307, New York, NY, USA. Association for Computing Machinery.

Rajdeep Mukherjee, Shreyas Shetty, Subrata Chattopadhyay, Subhadeep Maji, Samik Datta, and

Pawan Goyal. 2021b. Reproducibility, replicability and beyond: Assessing production readiness of aspect based sentiment analysis in the wild. In *Advances in Information Retrieval*, pages 92–106, Cham. Springer International Publishing.

Tapas Nayak and Hwee Tou Ng. 2020. Effective modeling of encoder-decoder architecture for joint entity and relation extraction. In *AAAI*.

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 *AAAI*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014a. 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.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014b. 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.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014c. 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.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Learning to attend via word-aspect associative fusion for aspect-based sentiment analysis. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5956–5963. AAAI Press.

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 Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 3316–3322. AAAI Press.

Zhen Wu, Chengcan Ying, Fei Zhao, Zhifang Fan, Xinyu Dai, and Rui Xia. 2020. Grid tagging scheme for aspect-oriented fine-grained opinion extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2576–2585, Online. Association for Computational Linguistics.

Hu Xu, Bing Liu, Lei Shu, and Philip Yu. 2019. BERT post-training for review reading comprehension and aspect-based sentiment analysis. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2324–2335, Minneapolis, Minnesota. Association for Computational Linguistics.

Hu Xu, Bing Liu, Lei Shu, and Philip Yu. 2020a. DomBERT: Domain-oriented language model for aspect-based sentiment analysis. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1725–1731, Online. Association for Computational Linguistics.

Hu Xu, Bing Liu, Lei Shu, and Philip S. Yu. 2018. Double embeddings and CNN-based sequence labeling for aspect extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 592–598, Melbourne, Australia. Association for Computational Linguistics.

Lu Xu, Hao Li, Wei Lu, and Lidong Bing. 2020b. 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, Online. Association for Computational Linguistics.

Wei Xue and Tao Li. 2018. Aspect based sentiment analysis with gated convolutional networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2514–2523, Melbourne, Australia. Association for Computational Linguistics.

Chen Zhang, Qiuchi Li, Dawei Song, and Benyou Wang. 2020. A multi-task learning framework for opinion triplet extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 819–828, Online. Association for Computational Linguistics.

A Appendix

A.1 Pointer Network-based Decoder

Referring to Figure 2, opinion triplets are decoded using an LSTM-based *Triplet Decoder*, that takes into account the history of previously generated pairs/tuples of aspect and opinion spans, in order to avoid repetition. At each time step t, it generates a hidden representation $\mathbf{h_t^D} \in \mathbb{R}^{d_h}$ that is used by the two Bi-LSTM + FFN-based *Pointer Networks* to respectively predict the aspect and opinion spans, while exploiting their interdependence. The tuple representation tup_t thus obtained is concatenated with $\mathbf{h_t^D}$ and passed through an FFN-based *Sentiment Classifier* to predict the connecting sentiment, thereby decoding an entire opinion triplet at the t^{th} time step. We now elaborate each component of our proposed decoder framework in greater depth.

A.1.1 Span Detection with Pointer Networks

Our pointer network consists of a Bi-LSTM, with hidden dimension d_p, followed by two feed-forward layers (FFN) on top to respectively predict the start and end locations of an entity span. We use two such pointer networks to produce a tuple of hidden vectors corresponding to the aspect and opinion spans of the triplet to be decoded at time step t. We concatenate $\mathbf{h_t^D}$ with each of the encoder hidden state vectors $\mathbf{h_i^E}$ and pass them as input to the first Bi-LSTM. The output hidden state vector corresponding to the i^{th} token of the sentence thus obtained is simultaneously fed to the two FFNs with *sigmoid* to generate a pair of scores $\tilde{s}_i^{p_1}$ and $\tilde{e}_i^{p_1}$ in the range of 0 to 1 as follows:

$$\tilde{s}_i^{p_1} = \mathbf{W_s^{p_1} h_i^{p_1}} + \mathbf{b_s^{p_1}}, \quad \tilde{e}_i^{p_1} = \mathbf{W_e^{p_1} h_i^{p_1}} + \mathbf{b_e^{p_1}}$$

Here, $\mathbf{W_s^{p_1}} \in \mathbb{R}^{d_p \times 1}$, $\mathbf{W_e^{p_1}} \in \mathbb{R}^{d_p \times 1}$, $\mathbf{b_s^{p_1}}$, and $\mathbf{b_e^{p_1}}$ are respectively the weights and bias parameters of the two FFNs for the first pointer network (p_1). After repeating the process for all tokens in the sentence, the normalized probabilities of the i^{th} token to be the start and end positions of an *aspect span* ($s_i^{p_1}$ and $e_i^{p_1}$ respectively) are obtained using *softmax* operations over the two sets of scores thus generated (by the two FFNs) as follows:

$$\mathrm{S}^{\mathrm{p}_1} = \mathrm{softmax}(\tilde{\mathrm{S}}^{\mathrm{p}_1}), \quad \mathrm{E}^{\mathrm{p}_1} = \mathrm{softmax}(\tilde{\mathrm{E}}^{\mathrm{p}_1})$$

Similar equations are used for the second pointer network (p_2) to generate the normalized probabilities, $s_i^{p_2}$ and $e_i^{p_2}$, for the i^{th} token to be the start and end positions of an opinion span respectively; difference being that apart from concatenating $\mathbf{h_t^D}$, we also concatenate the output vectors $\mathbf{h_i^{p_1}}$ from the first Bi-LSTM with encoder hidden states $\mathbf{h_i^E}$ and pass them as input to the second Bi-LSTM. The vector representations for the aspect and opinion spans at time step t are obtained as follows:

$$\mathrm{ap_t} = \sum_{i=1}^{n} s_i^{p_1} \mathbf{h_i^{p_1}} \ \| \ \sum_{i=1}^{n} e_i^{p_1} \mathbf{h_i^{p_1}}; \mathrm{ap_t} \in \mathbb{R}^{2d_p}$$

$$\mathrm{op_t} = \sum_{i=1}^{n} s_i^{p_2} \mathbf{h_i^{p_2}} \ \| \ \sum_{i=1}^{n} e_i^{p_2} \mathbf{h_i^{p_2}}; \mathrm{op_t} \in \mathbb{R}^{2d_p}$$

Here we introduce the term *generation direction* which refers to the order in which we generate the hidden representations for the two entities, i.e. aspect and opinion spans. This allows us to define two **variants** of our model. The variant discussed so far uses p_1 to detect the aspect span before predicting the opinion span using p_2, and is henceforth referred to as **PASTE-AF** (AF stands for *aspect first*). Similarly, we obtain the second variant **PASTE-OF** (*opinion first*) by reversing the *generation direction*. The other two components of our model remain the same for both the variants.

A.2 Attention Modeling

We use *Badhanau Attention* (Bahdanau et al., 2015) to obtain the context representation of the input sentence ($s_t^E \in \mathbb{R}^{d_h}$) at time step t as follows:

$$\tilde{tup}_{prev} = \mathbf{W_{tup}} \, tup_{prev} + \mathbf{b_{tup}}$$

$$\mathbf{u_t^i} = \mathbf{W_u h_i^E}$$

$$\mathbf{\tilde{q}_t^i} = \mathbf{W_{\tilde{q}}} \, \tilde{tup}_{prev} + \mathbf{b_{\tilde{q}}} \ ; \ \mathbf{\tilde{a}_t^i} = \mathbf{v_{\tilde{a}}} \tanh(\mathbf{\tilde{q}_t^i} + \mathbf{u_t^i})$$

$$\mathbf{q_t^i} = \mathbf{W_q h_{t-1}^D} + \mathbf{b_q} \ ; \ \mathbf{a_t^i} = \mathbf{v_a} \tanh(\mathbf{q_t^i} + \mathbf{u_t^i})$$

$$\boldsymbol{\tilde{\alpha}_t} = \mathrm{softmax}(\mathbf{\tilde{a}_t}) \ ; \ \boldsymbol{\alpha_t} = \mathrm{softmax}(\mathbf{a_t})$$

$$\mathbf{s_t^E} = \sum_{i=1}^{n} \frac{\tilde{\alpha}_i + \alpha_i}{2} \mathbf{h_i^E}$$

Here, $\mathbf{W_{\tilde{q}}}, \mathbf{W_q}, \mathbf{W_u} \in \mathrm{R}^{d_h \times d_h}$, $\mathbf{v_{\tilde{a}}}, \mathbf{v_a} \in \mathrm{R}^{d_h}$ are learnable attention parameters, and $\mathbf{b_{\tilde{q}}}, \mathbf{b_q} \in \mathrm{R}^{d_h}$ are bias vectors. First, we obtain $\tilde{tup}_{prev}$ from tup_{prev} using a linear embedding layer, with $\mathbf{W_{tup}} \in \mathrm{R}^{4d_p \times d_h}$ and $\mathbf{b_{tup}}$ as its weights and bias parameters. We then use both $\tilde{tup}_{prev}$ and $\mathbf{h_{t-1}^D}$ separately to obtain two attentive context vectors, $\mathbf{\tilde{q}_t}$ and $\mathbf{q_t}$ respectively. These are then concatenated along with tup_{prev} to define the current context of our LSTM-based decoder. The corresponding normalized attention scores, $\boldsymbol{\tilde{\alpha}_t}$ and $\boldsymbol{\alpha_t}$, are averaged to obtain the attention-weighted sentence representation at decoding time step t.

A.3 Experimental Setup

For our non-BERT experiments, word embeddings are initialized (and kept trainable) using pre-trained 300-dim. Glove vectors (Pennington et al., 2014), and accordingly d_w is set to 300. The dimensions of POS and DEP embeddings, i.e. d_{pos} and d_{dep} are set to 50 each. The decoder (LSTM) hidden dimension d_h is set to 300, and accordingly the hidden state dimensions of both backward and forward LSTMs of the Bi-LSTM-based encoder are set to 150 each. We set the hidden dimension d_p of the Bi-LSTMs in pointer networks to 300. For our BERT experiments, *uncased* version of pre-trained BERT-base (Devlin et al., 2019) is fine-tuned to encode each sentence.

All our model variants are trained end-to-end with *Adam* optimizer (Kingma and Ba, 2015) with 10^{-3} as the learning rate, and 10^{-5} as weight decay. Dropout (0.5) (Srivastava et al., 2014) is applied on embeddings to avoid overfitting. Our non-BERT model variants are trained for 100 epochs with a batch size of 10. Our BERT-based variants are trained for 30 epochs with a batch size of 16. Model selected according to the best F_1 score on the development data is used to evaluate on the test data. We run each model five times and report the median scores. All our experiments are run on Tesla P100-PCIE 16GB GPU.