Constituency Lattice Encoding for Aspect Term Extraction

Yunyi Yang[*], Kun Li[*], Xiaojun Quan[†], Weizhou Shen and Qinliang Su
School of Data and Computer Science
Sun Yat-sen University, Guangzhou, China
{yangyy37, likun36}@mail2.sysu.edu.cn
quanxj3@mail.sysu.edu.cn

Abstract

One of the remaining challenges for aspect term extraction in sentiment analysis resides in the extraction of phrase-level aspect terms, which is non-trivial to determine the boundaries of such terms. In this paper, we aim to address this issue by incorporating the span annotations of constituents of a sentence to leverage the syntactic information in neural network models. To this end, we first construct a constituency lattice structure based on the constituents of a constituency tree. Then, we present two approaches to encoding the constituency lattice using BiLSTM-CRF and BERT as the base models, respectively. We experimented on two benchmark datasets to evaluate the two models, and the results confirm their superiority with respective 3.17 and 1.35 points gained in F1-Measure over the current state of the art. The improvements justify the effectiveness of the constituency lattice for aspect term extraction.[1]

1 Introduction

In aspect-based sentiment analysis, aspect term extraction (ATE) serves as a fundamental task that aims to identify aspect terms from review texts (Hu and Liu, 2004; Popescu and Etzioni, 2005). An aspect term can be a word or a phrase that describes certain attribute of an entity (e.g., *restaurant*). For example, in the sentence "*The food is very average, the Thai fusion stuff is a bit too sweet*", the terms "*food*" and "*Thai fusion stuff*" are the aspect terms to extract in this task. One of the challenges for ATE is to extract uncommon phrase-level aspect terms, which may contain complicated structures of words.

One line of the research work on ATE leverages the syntactic structure of a sentence explicitly. Early research in this direction focuses on rule-based (Hu and Liu, 2004; Qiu et al., 2011) or feature engineering approaches (Jin and Ho, 2009; Li et al., 2010) that require manually designed rules based on part-of-speech tagging or dependency structure. They may also rely on features from predefined lexicons or syntactic analysis from annotated corpus. These approaches have unavoidable limitations in that they are labor-intensive and it takes extensive efforts to construct the rules and features. Besides, there are neural network approaches (Yin et al., 2016; Wang et al., 2016) that encode the dependency parse structure of a sentence in neural models. These work primarily focuses on syntactic dependency information of sentence, given that the dependency structure may indicate the position of a potential aspect term. Nevertheless, their tree-dependent nature of the encoding process makes the optimization inconvenient. Another line of work resorts solely to deep learning techniques such as LSTM (Liu et al., 2015), CNN (Xu et al., 2018), attention mechanism (Wang et al., 2017; Li et al., 2018) and Transformer (Devlin et al., 2019; Xu et al., 2019) to automate the semantic representations and have achieved promising results. Among them, Xu et al. (2019) applies an additional pre-training for BERT (Devlin et al., 2019) with in-domain corpora before fine-tuning, which is the current state of the art for this task.

Although deep learning methods can produce increasingly appealing results, they suffer from phrase-level aspect term extraction without explicit inclusion of sentence syntactic information (Xu et al., 2019).

[*] Contributed equally.

[†] Xiaojun Quan is the corresponding author.

[1] Code is available at https://github.com/leekum2018/CLE4ATE

Proceedings of the 28th International Conference on Computational Linguistics, pages 844–855
Barcelona, Spain (Online), December 8-13, 2020

Particularly, constituency parsing, which generates constituency trees with hierarchical constituents that may indicate the span of a targeted aspect term, is rarely exploited in recent models. Take the sentence "*Other guests enjoyed pizza, santa fe chopped salad and fish and chips*" in Figure 1 for example. The intended aspect terms are "*santa fe chopped salad*" and "*fish and chips*". In this example, constituency parsing provides us with constituents that match the two aspect terms exactly, showing that the constituents can be used to identify the spans of potential aspect terms, especially phrase-level aspect terms.

In this paper, we argue that the span annotations of constituents are the most critical information in incorporating the constituency structure for an ATE model. That means we only need to pay attention to the constituents rather than the whole hierarchical tree structure. To this end, we first propose to construct a constituency lattice structure that contains both the original sentence and the corresponding constituents. The idea is partially motivated by the work on Chinese named-entity recognition (NER) (Zhang and Yang, 2018), in which a lattice structure is also used to help identify the boundaries of target named entities in the sequence labeling process. We then introduce two methods to effectively encode the constituency lattice structure for aspect term extraction, leading to our two models, namely CL-BiLSTM and CL-BERT. For the first method, we propose to encode the constituency lattices into BiLSTM-CRF (Huang et al., 2015) to enhance the representations of sentence words. For the second, we opt for BERT as the base model (Devlin et al., 2019), which is a large pre-trained language model structured by a stack of Transformer blocks. Since Transformers have a different architecture than LSTMs, it is non-trivial to modify the input embedding layer of BERT like LSTMs. To address this issue, we propose to encode the sentence and its constituents separately for several initial layers, and then apply attention to gather constituency spans information from the constituents to enhance the sentence encoding. Experimental results on two benchmarks show that the two proposed methods achieve substantial improvements over the baselines, demonstrating the effectiveness of our constituency lattice encoding methods.

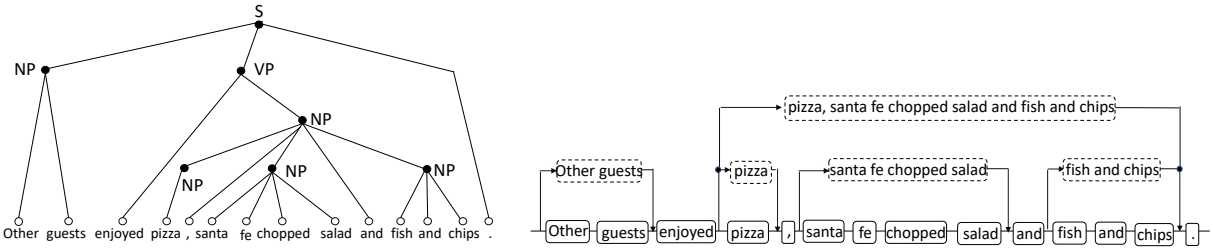
2 Related Work

2.1 Aspect Term Extraction

It tends to be a natural idea to use syntactic information for aspect term extraction (ATE). Early methods in this direction pay most attention to dependency parsing. Qiu et al. (2011) used a dependency parser to augment a seed collection of aspect and opinion terms through double-propagation. Yin et al. (2016) proposed an unsupervised embedding method to encode dependency paths into a recurrent neural network to learn high-level features of words, which are taken as input to conditional random fields (CRFs) for aspect term extraction. Wang et al. (2016) proposed a joint model of recursive neural networks and CRFs. Recently, deep neural networks techniques such as LSTM (Liu et al., 2015), CNN (Xu et al., 2018), attention mechanism (Wang et al., 2017; Li et al., 2018) and Transformer (Devlin et al., 2019; Xu et al., 2019) have been applied to learn better feature representations for supervised aspect extraction. So far, we have not seen any effort in incorporating constituency information for aspect term extraction, which could be essential to solve the extraction of complicated phrase-level aspect terms.

2.2 Lattice

How to encode the lattice structure rather than merely the sequential input has been intensively studied in NLP tasks, such as speech translation (Sperber et al., 2017; Sperber et al., 2019; Zhang et al., 2019), Chinese named-entity recognition (Zhang and Yang, 2018; Peng et al., 2019; Li et al., 2020) and neural machine translation (Xiao et al., 2019). Among them, Sperber et al. (2017) extended TreeLSTM into LatticeLSTM with modified dynamic gated cells to capture multiple speech recognition hypotheses. Similarly, Zhang and Yang (2018) used Lattice-LSTM to capture word lexicons and prevent errors from wrong word segmentation for Chinese named-entity recognition. However, Lattice-LSTM is computationally expensive, due to the complicated architecture of modified RNNs which is rather slow and inconvenient for batch computation. Peng et al. (2019) proposed to incorporate lexicon information into the vector representations of characters, which turns out to be very fast at inference time. While Lattice-LSTM can modify the model structure according to the lattices, it tends to be different for Transformers to do so because of the difference in model architectures. Sperber et al. (2019) and Zhang et al. (2019)



(a) Constituency tree (b) Constituency lattice

Figure 1: An illustrative example of the constituency tree (a) of a sentence and the constituency lattice (b) constructed to encode the syntactic information for aspect term extraction.

introduced attention mechanism with distance masking techniques for lattice to boost translation speech. Xiao et al. (2019) proposed lattice positional encoding and lattice self-attention for Chinese neural machine translation. Compared to the above work, we address the aspect term extraction task by encoding the constituency lattices obtained from constituency parsing into two current models.

3 Approach

In this section, we first formally define the problem of aspect term extraction (ATE) and then provide the details of our constituency lattice encoding methods.

3.1 Problem Definition

Given a word sequence $X = \{x_1, x_2, ..., x_n\}$ of length n, the ATE task can be formulated as a word level classification problem, in which a model takes X as input and produces contextual representations $\{s_1, s_2, ..., s_n\}$, where $s_i \in R^{d_s}$ and d_s is the dimension of the representation. The model outputs a label sequence $Y = \{y_1, y_2, ..., y_n\}$ for the sequence, where $y_i \in \{B, I, O\}$ is used to indicate if the corresponding word is at the *beginning*, *inside* or *outside* of an aspect term.

3.2 Constituency Lattice

As shown in Figure 1(a), constituency parsing produces a constituent-based parse tree for a sentence to represent its syntactic structure. Given the bunch of information in the tree, we assume only a small part of them is related to our task. Therefore, we construct a lattice structure from the constituency tree, namely, constituency lattice. The constituency lattice is made up of a sentence together with a set of constituents, each indicating a span annotation of the sentence, as shown in Figure 1(b). The constituency lattice for a sentence can be regarded as a directed acyclic graph that is constructed as follows. For each constituent, we connect it to the sentence according to the positions of its first and last words in the original sentence. In addition, we filter the constituency lattice to preserve only the constituents with noun phrase (NP) and verb phrase (VP) types to prune unimportant constituents (see our experiments for specification). The resulted constituency lattice eliminates the tree structure and instead provides necessary syntactic information more explicitly for aspect term extraction.

3.3 Constituency Lattice Encoding

In this subsection, we introduce how to encode the constituency lattice structure in BiLSTM-CRF (Huang et al., 2015) and BERT-PT (Xu et al., 2019), which gives rise to our two models, namely Constituency Lattice BiLSTM (CL-BiLSTM) and Constituency Lattice BERT (CL-BERT). BiLSTM-CRF is a BiLSTM network with a subsequent CRF layer and BERT-PT is a variant of BERT (Devlin et al., 2019) with post-training on large-scale domain-related data.

Constituency Lattice LSTM

The word embedding layer is used to map each token x_i of an input sentence to a dense vector:

$$v_i = e^x(x_i), \tag{1}$$

where e^x denotes the word embedding lookup table.

To encode the constituency lattice, we slightly revise this layer by concatenating the embeddings of the words and the constituency lattice. Concretely, for token x_i, we denote the corresponding constituent set as $\mathcal{S}_i = \{C_i^1, C_i^2, ..., C_i^{|\mathcal{S}_i|}\}$, where $C_i^j \in \mathcal{S}_i$ denotes a constituent in the constituency lattice that covers token x_i. Take the constituency lattice in Figure 1(b) as an example, the constituent set of token *"fish"* is {*"fish and chips"*, *"pizza, santa fe chopped salad and fish and chips"* }. Then, the corresponding new vector v_i is obtained as follow:

$$v_i = [e^x(x_i); P_i] \tag{2}$$

$$P_i = \frac{1}{|\mathcal{S}_i|} \sum_{j=1}^{|\mathcal{S}_i|} e^C(C_i^j), \tag{3}$$

where $[;]$ means concatenation of vectors, and e^C denotes the function that maps the word set of a constituent C_i^j to a dense vector, which is defined as:

$$e^C(C_i^j) = \frac{1}{|C_i^j|} \sum_{k=1}^{|C_i^j|} e^x(x_k). \tag{4}$$

Here, $|C_i^j|$ denotes the length of constituent C_i^j in number of words, and x_k is the k-th word in C_i^j.

The new token representation v_i is then fed to BiLSTM to obtain a hidden representation, on top of which the CRF layer is applied to yield a probability distribution for aspect term prediction. This method is partly inspired by the work of Peng et al. (2019), which presents a simplified approach to using lexicon information for Chinese NER to remedy the complicated structure of Lattice-LSTM.

Constituency Lattice BERT

Essentially, pre-trained language models such as BERT derive most of their power from the pre-trained parameters on large-scale general-domain data. Because of the discrepancy between the pre-training and fine-tuning phases, it is non-trivial to revise the embedding layer of BERT like BiLSTM. Therefore, we propose the Constituency Lattice BERT (CL-BERT) which employs both in-sequence attention and lattice attention to overcome the above limitation. Specifically, it first encodes the sentence and each of the corresponding constituents separately for several layers, in which in-sequence attention is applied. Then, lattice attention is applied to gather necessary span information from the constituents to enhance the sentence encoding for aspect term prediction. The overall architecture is shown in Figure 2.

Precisely, the in-sequence attention is the same as the self-attention in vanilla Transformer encoder. After several initial layers, the representations of these sequences become contextualized. Then, we introduce lattice attention for the remaining few layers from each constituent to the sentence in terms of their representations of the special token "[CLS]"s. This allows the in-sequence attention of the subsequent layers of the sentence to further propagate the constituency information to every token in the sentence. Accordingly, the final token representations of the sentence are aware of both the sentence structure and the span annotations of the corresponding constituents.

Formally, given a sentence X and the associated constituency lattice, the sentence X (used interchangeably with C_0) and the constituents $C_1, C_2, ..., C_m$ in the lattice are first encoded by BERT as separate sequences. It is worth noting that the tokens in constituents preserve the same position indices as in the original sentence, which enables the constituents to indicate the original structure of the constituency lattice. For each sequence indexed by τ ($0 \leq \tau \leq m$), the in-sequence attention updates the local representations of the sequence: at layer l, it computes the representation of token i by gathering information from other tokens inside the sequence:

$$h_{\tau,i}^l = \sum_j \text{softmax}_j \left(\frac{q_{\tau,i}^T \cdot k_{\tau,j}}{\sqrt{d_k}} \right) \cdot v_{\tau,j} \tag{5}$$

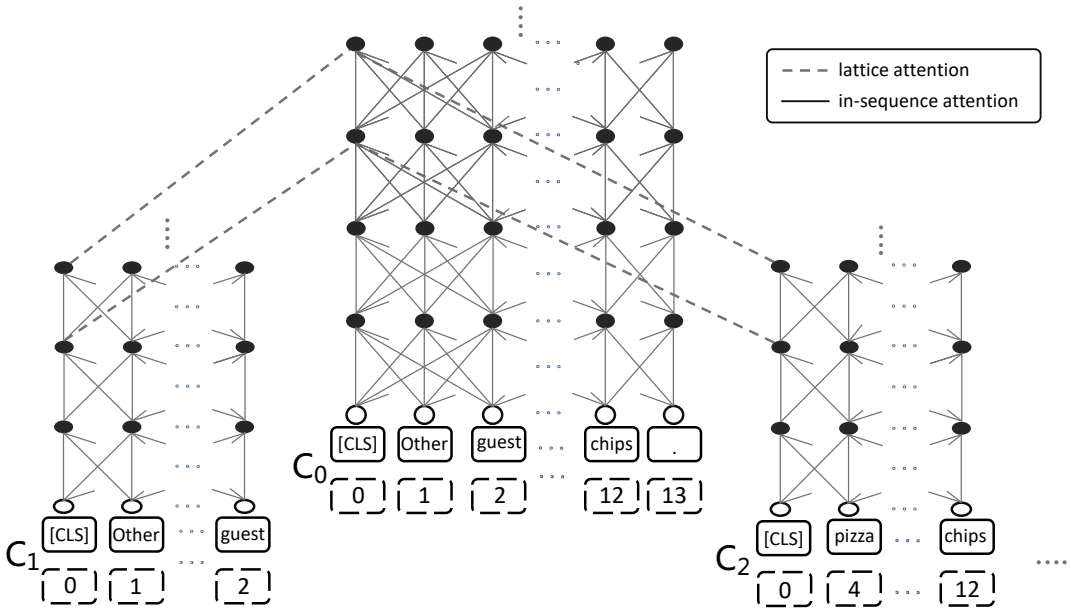


Figure 2: Architecture of CL-BERT, where solid lines denote in-sequence attention and dashed lines denote lattice attention. The tokens in constituents have the same positional indices as in the sentence.

$$[q, k, v] = \left[W^q, W^k, W^v\right] \cdot h^{l-1} \tag{6}$$

where d_k denotes the dimension of hidden state h.

After some initial number L of layers, the encoder for C_0 starts to gather information from the other constituent encoders by means of lattice attention to update the representation of the "[CLS]" token of C_0. The lattice attention is cast by attending to each constituent C_η with the sentence sequence C_0 as query. As the token "[CLS]" is expected to capture the representation of the whole corresponding sequence, the lattice attention is based on the "[CLS]"s of all the sequences:

$$\hat{h}^j_{0,0} = \sum_\eta \text{softmax}_\eta \left(\frac{\hat{q}^T_{0,0} \cdot \hat{k}_{\eta,0}}{\sqrt{d_k}} \right) \cdot \hat{v}_{\eta,0} \tag{7}$$

$$\left[\hat{q}, \hat{k}, \hat{v}\right] = \left[W^{\hat{q}}, W^{\hat{k}}, W^{\hat{v}}\right] \cdot h^j \tag{8}$$

where $j > L$ and the attention weights are calculated for C_0 and each constituent C_η using query $\hat{q}_{0,0}$ and key $\hat{k}_{\eta,0}$ ("[CLS]" is the first token of a sequence in BERT). Then, it uses the weights to aggregate the values of $\hat{v}_{\eta,0}$ with η from 1 to m, and generates an augmented representation $\hat{h}^j_{0,0}$, which is then combined with the original representation to form a new representation for the "[CLS]" token of C_0 in layer j:

$$h^j_{0,0} = \text{Linear}\left(\left[h^j_{0,0}; \hat{h}^j_{0,0}\right]\right) \tag{9}$$

The information stored in the updated $h^j_{0,0}$ is accessible by other tokens in C_0 through the in-sequence attention in the subsequent layers. In doing so, the information of the constituency lattice can be encoded into the representations of all tokens in the sequence C_0.

In this paper, we ask CL-BERT to perform the lattice attention in the last 3 layers and get the final representation for each token in X from the last layer, which is then mapped into the classification space by a linear transformation. The setting with respect to the lattice attention is studied in the experiments.

4 Experiments

4.1 Datasets

We conduct experiments on two widely-used benchmark datasets, including the Laptop dataset from SemEval 2014 Task 4 (Pontiki et al., 2014) and the Restaurants dataset from SemEval 2016 Task 5 (Pontiki et al., 2016). Statistics of the two datasets are presented in Table 1. As in Xu et al. (2018), we randomly hold out 150 examples from each training set for validation. The F1-Measure metric is used to evaluate the performance of the baselines and our models.

4.2 Implementation Details

Dataset	Train		Test	
	#Sent	#Aspect	#Sent	#Aspect
Restaurant	2000	1743	676	622
Laptop	3045	2358	800	654

Table 1: Statistics of our datasets. #Sent and #Aspect denote the number of sentence and aspect, respectively.

For CL-LSTM, the word embeddings are initialized with GloVe-840B-300d (Pennington et al., 2014) and fixed during training, and the hidden size of BiLSTM layers is set to 300. We add a CRF (Lafferty et al., 2001) layer to avoid illegal transitions between labels. Adam (Kingma and Ba, 2014) is used to optimize this model with a learning rate of 1e-4 and two momentum coefficients set to 0.9 and 0.999, respectively. We apply dropout with a rate of 0.5 for the input word embeddings and a rate of 0.2 for the other layers.

The in-sequence attention and other Transformer components in CL-BERT are initialized by the pre-trained BERT-PT (Xu et al., 2019), which is post-trained on a large domain-specific corpus with parameters initialized from BERT-base.[2] The additional parameters of lattice attention are initialized randomly and trained from scratch. We use AdamW (Loshchilov and Hutter, 2017) to optimize this model with an initial learning rate of 1e-5, which decreases linearly during training.

When constructing the constituency lattices, we filter the constituents based on their types and preserve only those with type NP for the Restaurant dataset, and type NP or VP for the Laptop dataset.

4.3 Baselines

The baselines used for comparison can be organized into two categories.

The first category are models using multi-task learning to extract aspect terms and opinion terms jointly. **RNCRF** (Wang et al., 2016) combines dependency tree with RNN and CRF for aspect and opinion terms co-extraction. **CMLA** (Wang et al., 2017) uses a multi-layer coupled-attention network for the co-extraction. **MIN** (Li and Lam, 2017) employs three LSTMs and dependency rules to identify aspect and opinion terms. **HAST** (Li et al., 2018) uses opinion information to assist aspect term extraction.

The second category are single-task approaches. **BiLSTM-CRF** (Huang et al., 2015) adopts BiLSTM with a subsequent CRF layer. **WDEmb** (Yin et al., 2016) enhances CRF with word embeddings, linear context embeddings and dependency path embeddings. **DE-CNN** (Xu et al., 2018) employs two types of pre-trained embeddings: general-purpose embeddings and domain-specific embeddings, which are stacked and passed into a convolution neural network. **BERT-base** (Devlin et al., 2019) is a large pre-trained language model fine-tuned for ATE. **BERT-PT** (Xu et al., 2019) adopts the same architecture and initial parameters of BERT-base and then post-trains on large-scale in-domain data and a machine reading comprehension dataset (Rajpurkar et al., 2016). It is the current state-of-the-art method for ATE.

Our two models are represented as **CL-LSTM** and **CL-BERT**, which perform constituency lattice encoding in BiLSTM-CRF and BERT-PT, respectively.

5 Results and Analysis

5.1 Overall Performance

The overall results are shown in Table 2, from which several observations can be noted. First, our CL-LSTM model outperforms most of the baseline models. Second, the performance of BiLSTM-CRF

[2]The weights of BERT-PT are available at `https://github.com/howardhsu/BERT-for-RRC-ABSA`

is significantly improved when incorporated with our constituency lattice structure. CL-LSTM also outperforms the baselines such as RNCRF, WDEmb, and MIN, which explicitly model the dependency structure in different ways. This may prove that constituency structure is more prominent for ATE than dependency structure. Third, the DE-CNN and BERT-PT models can already outperform all the existing ATE models by a significant margin, demonstrating the power of domain-specific post-training for this task. Nevertheless, after encoded with our constituency lattice, CL-LSTM has comparable performance with DE-CNN, while CL-BERT also sees considerable improvement and achieves a new state of the art. These results demonstrate the success of our constituency lattice encoding in capturing important constituency information for aspect term extraction.

Moreover, we can also observe that the performance gained by our constituency lattice encoding over the baselines is more obvious on the Restaurant dataset than on Laptop. We speculate the reason is that the Restaurant dataset has more phrase-level aspect terms than Laptop, which is to be further discussed in Section 5.2.

Model	Restaurant	Laptop
RNCRF	69.72	78.42
HAST	73.61	79.52
MIN	73.44	77.58
CMLA	72.77	77.80
BiLSTM-CRF	71.3	76.91
WDEmb	-	75.16
DE-CNN	74.37	81.59
BERT	74.1	79.28
BERT-PT	77.97	84.26
CL-LSTM	76.56	79.29
CL-BERT	**81.14**	**85.61**

Table 2: Overall results (F1:%) of different methods on the Restaurant and Laptop datasets.

5.2 Effectiveness of Constituency Lattice

The constituency lattices provide information about the phrase structure of a sentence such that models with constituency lattice encoding can benefit from the span annotations. To demonstrate this point, we first examine the relation between aspect terms and constituents by counting how many aspect terms in a dataset that match exactly one of the corresponding constituents. The results under the *Coverage* column in Table 3 show that most of the aspect terms in the Restaurant dataset coincide with NP constituents, while most aspect terms in Laptop coincide with NP or VP constituents.

We further conduct an investigation over the extracted aspect terms on the test sets with an evaluation criterion *Ill Extraction Rate* (*IE-Rate*), which is computed by: IE-Rate $= \frac{\text{\#ill extraction}}{\text{\#aspect term}} \times 100$, where "#aspect term" denotes the number of extracted aspect terms and "#ill extraction" is the number of extracted aspect terms with incorrect boundaries, excluding the cases that have no span intersection with gold aspect terms. As shown in Table 3, even though BERT-PT has better F1-score than BiLSTM-CRF, they both struggle in terms of *IE-Rate*, which indicates their limited capabilities of extracting accurate boundaries. On the other hand, both CL-LSTM and CL-BERT reduce the *IE-Rate* scores over their baselines, which demonstrates the effectiveness of our constituency lattice encoding in helping models capture the correct span segmentation of aspect terms. Moreover, similar to the overall performance, the improvement in IE-Rate on the Restaurant dataset is more significant than on Laptop.

Dataset	Coverage		IE-Rate			
	Train	Test	BiLSTM-CRF	CL-LSTM	BERT-PT	CL-BERT
Restaurant	90.65	86.76	9.75	6.91	9.20	5.76
Laptop	86.08	82.87	12.19	11.75	12.81	9.77

Table 3: Percentages of aspect terms that coincide with a specific constituent (NP for Restaurant and NP/VP for Laptop) and the Ill Extraction Rate (IE-Rate: %) scores of different methods.

To better understand how the two methods for constituency lattice encoding work, as well as the difference between the two ATE datasets, we present a case study with several testing examples. BERT-PT and CL-BERT are used as the extraction models. As shown in Table 4, CL-BERT is able to extract long and complicated aspect terms, while BERT-PT tends to extract shorter aspect terms, leading to

incomplete extractions. Additionally, CL-BERT fails to extract the full term in the last case which is from the Laptop dataset. This error indicates that some aspect terms in the Laptop dataset contain complicated structures which are difficult for the constituency lattices to distinguish.

No.	Input and Result
1	**Input**: I ended the meal with the unusual dessert of a port and chocolate tasting ....yummy! **BERT-PT**: dessert, port and chocolate tasting; **CL-BERT**: dessert of a port and chocolate tasting
2	**Input**: Bring your cell phone cause you may have to wait to get into the best sushi restaurant in the world: BLUE RIBBON SUSHI . **BERT-PT**: sushi, RIBBON; **CL-BERT**: sushi, BLUE RIBBON SUSHI
3	**Input**: I am exceedingly pleased to report that my dinner at Ray's Boathouse last Friday completely exceeded my expectations. **BERT-PT**: Ray; **CL-BERT**: Ray's Boathouse
4	**Input**: I have had it over a year now with out a Glitch of any kind..I love the lit up keys and screen display ...this thing is Fast and clear as can be. **BERT-PT**: keys, screen display; **CL-BERT**: lit up keys, screen display
5	**Input**: ... $450 savings to buy 16GB of RAM , Two Seagate Momentus XT hybrid drives and an OWC upgrade kit to install the second hard drive . **BERT-PT**: 16GB of RAM, Seagate, XT hybrid drives, OWC upgrade kit, hard drive; **CL-BERT**: 16GB of RAM, Seagate Momentus XT hybrid drives, OWC upgrade kit, hard drive
6	**Input**: I opted for the SquareTrade 3-Year Computer Accidental Protection Warranty ($1500 -2000) which also support ... that are NOT covered by AppleCare . **BERT-PT**: Protection Warranty, AppleCare; **CL-BERT**: Computer Accidental Protection Warranty, AppleCare

Table 4: Case analysis on BERT-PT and CL-BERT. The ground truth aspect terms in the sentences are shaded. The first three cases are from the Restaurant dataset and the last three are from Laptop.

5.3 Effect of Lattice Attention

Recall that CL-BERT performs lattice attention in the last few layers of BERT to propagate information from the constituents to the sentence. In this experiment we investigate the effect of this attention in different layers. That means we change the number of layers for the attention to take place from 1 to 7. For example, when the number is 5, it means the attention is performed for the last five layers. The results of this experiment on both datasets are plotted in Figure 3. We can note that CL-BERT achieves the best performance when the number of layers to perform lattice attention is 3, and the performance goes down when the number is greater than 3. We suspect that the representations of sequences have not been adequately learned by in-sequence attention in the earlier layers (too many layers for lattice attention). To verify this conjecture, we further conduct an experiment by sliding the 3-layer lattice attention along the different levels of CL-BERT. The results in Table 5 show that the performance consistently increases when lattice attention is performed in deeper layers, meaning that late lattice attention is better than early one.

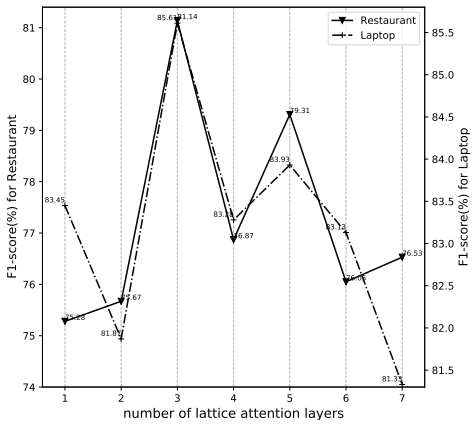


Figure 3: Performance of CL-BERT with different number of lattice attention.

Dataset	Index of layer with lattice attention			
	[1, 2, 3]	[4, 5, 6]	[7, 8, 9]	[10, 11, 12]
Restaurant	75.78	77.03	76.53	81.14
Laptop	82.74	83.28	85.12	85.61

Table 5: Performance (F1, %) of CL-BERT with three-layer lattice attention at different layers.

5.4 Effect of Pruning

The statistics of the datasets show that 98.0% of the aspect terms in the Restaurant dataset are noun phrases, and 97.7% of the aspect terms in the Laptop dataset are noun phrases or verb phrases. So we preserve the constituents of type NP for the Restaurant dataset and types NP and VP for Laptop. In this subsection, we conduct an ablation study to evaluate the influence of this pruning strategy. We present the two proposed models' results with full and pruned constituency lattice respectively for comparison. From the results in Table 6, we can observe that employing pruned constituency lattice consistently outperforms that without it. This proves that the irrelevant types of constituents can bring in noises when incorporating the syntactic information, and thus it is crucial to only preserve the constituents that are most related to aspect terms.

5.5 Effect of Different Parsers

Undoubtedly, constituency parsing plays a pivotal in our models. To evaluate the impact of different constituency parsers, we conduct a study to implement our approaches based on two well-known constituency parsers: self-attentive parser (Kitaev and Klein, 2018)[3] and a constituency parser (Joshi et al., 2018) implemented by AllenNLP (Gardner et al., 2017).[4] Table 7 shows the results of our models with different parsers for aspect term extraction. The performance of the two parsers in F1-score on the Penn Treebank WSJ test set (Marcus et al., 1993) is also provided for reference. From the table, we can observe that the better AllenNLP parser consistently leads to better ATE performance. This implies that while the proposed constituency lattice encoding methods can capture useful constituency information based on existing constituency parsers, they can be further improved with the advances of constituency parsing.

Dataset	CL-LSTM		CL-BERT	
	Prune	Full	Prune	Full
Restaurant	**76.56**	72.83	**81.14**	80.34
Laptop	**79.29**	77.36	**85.61**	85.32

Table 6: Performance of CL-LSTM and CL-BERT on pruned or full constituency lattice.

Parser	Parsing Quality	CL-LSTM		CL-BERT	
	WSJ test set	Restaurant	Laptop	Restaurant	Laptop
Self-attentive Parser	93.55	73.51	78.59	78.93	84.13
AllenNLP Parser	94.30	76.56	79.29	81.14	85.61

Table 7: Results of our models based on two different parsers together with the performance of the parsers. Higher F1-score on the Penn Treebank WSJ test set always denotes better parsing performance.

6 Conclusion

This paper presents our efforts to address the problem of aspect term extraction, especially phrase-level aspect term extraction, in aspect-based sentiment analysis. Motivated by the recent process in sequence labeling problems such as Chinese named-entity recognition, we propose to incorporate constituency

[3] https://github.com/nikitakit/self-attentive-parser

[4] The AllenNLP Parser is used for the previous experiments.

lattices in neural models to leverage the syntactic information explicitly for aspect term extraction. We demonstrate how to acquire the constituency lattices from constituency parse trees and encode them in two existing neural models, namely BiLSTM-CRF and BERT. Extensive experiments are conducted on two benchmark datasets to evaluate the two models, and the experimental results confirm their effectiveness with respective 3.17 points and 1.35 points gained in F1-Measure over the current state of the art. Since constituency parsing plays a critical role in our constituency lattice encoding, we show that the two models can be further improved once better parsers are provided. Finally, although our models are shown to deal with most of the phrase-level aspect term extractions, the error analysis shows that there are still long aspect terms that cannot be extracted completely by either our models or the baselines. But we believe this issue can be relieved as soon as there are more such samples provided in the training set.

Acknowledgments

This work was supported by the Fundamental Research Funds for the Central Universities (No.19lgpy220), the Program for Guangdong Introducing Innovative and Entrepreneurial Teams (No.2017ZT07X355), and the National Natural Science Foundation of China (No. 61806223).

References

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, June. Association for Computational Linguistics.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. Allennlp: A deep semantic natural language processing platform.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. ACM.

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

Wei Jin and Hung Hay Ho. 2009. A novel lexicalized hmm-based learning framework for web opinion mining. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 465–472.

Vidur Joshi, Matthew Peters, and Mark Hopkins. 2018. Extending a parser to distant domains using a few dozen partially annotated examples. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1190–1199, Melbourne, Australia, July. Association for Computational Linguistics.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686, Melbourne, Australia, July. Association for Computational Linguistics.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Xin Li and Wai Lam. 2017. Deep multi-task learning for aspect term extraction with memory interaction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2886–2892, Copenhagen, Denmark, September. Association for Computational Linguistics.

Fangtao Li, Chao Han, Minlie Huang, Xiaoyan Zhu, Ying-Ju Xia, Shu Zhang, and Hao Yu. 2010. Structure-aware review mining and summarization. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 653–661.

Xin Li, Lidong Bing, Piji Li, Wai Lam, and Zhimou Yang. 2018. Aspect term extraction with history attention and selective transformation. In *IJCAI*, pages 4194–4200.

Xiaonan Li, Hang Yan, Xipeng Qiu, and Xuanjing Huang. 2020. Flat: Chinese ner using flat-lattice transformer. *arXiv preprint arXiv:2004.11795*.

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.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in adam. *ArXiv*, abs/1711.05101.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Minlong Peng, Ruotian Ma, Qi Zhang, and Xuanjing Huang. 2019. Simplify the usage of lexicon in chinese ner. *arXiv preprint arXiv:1908.05969*.

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.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland, August. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, AL-Smadi Mohammad, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, et al. 2016. Semeval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th international workshop on semantic evaluation (SemEval-2016)*, pages 19–30.

Anamaria Popescu and Oren Etzioni. 2005. Extracting product features and opinions from reviews. pages 339–346.

Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2011. Opinion word expansion and target extraction through double propagation. *Computational Linguistics*, 37(1):9–27.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas, November. Association for Computational Linguistics.

Matthias Sperber, Graham Neubig, Jan Niehues, and Alex Waibel. 2017. Neural lattice-to-sequence models for uncertain inputs. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1380–1389.

Matthias Sperber, Graham Neubig, Ngoc-Quan Pham, and Alex Waibel. 2019. Self-attentional models for lattice inputs. In *ACL 2019 : The 57th Annual Meeting of the Association for Computational Linguistics*, pages 1185–1197.

Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2016. Recursive neural conditional random fields for aspect-based sentiment analysis. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 616–626.

Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2017. Coupled multi-layer attentions for co-extraction of aspect and opinion terms. In *AAAI*, pages 3316–3322.

Fengshun Xiao, Jiangtong Li, Hai Zhao, Rui Wang, and Kehai Chen. 2019. Lattice-based transformer encoder for neural machine translation. In *ACL 2019 : The 57th Annual Meeting of the Association for Computational Linguistics*, pages 3090–3097.

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*. Association for Computational Linguistics.

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

Yichun Yin, Furu Wei, Li Dong, Kaimeng Xu, Ming Zhang, and Ming Zhou. 2016. Unsupervised word and dependency path embeddings for aspect term extraction. In *IJCAI'16 Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 2979–2985.

Yue Zhang and Jie Yang. 2018. Chinese ner using lattice lstm. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1554–1564.

Pei Zhang, Niyu Ge, Boxing Chen, and Kai Fan. 2019. Lattice transformer for speech translation. In *ACL 2019 : The 57th Annual Meeting of the Association for Computational Linguistics*, pages 6475–6484.