Improving Constituency Parsing with Span Attention

Yuanhe Tian[♥], **Yan Song**[♠♡†], **Fei Xia**[♥], **Tong Zhang**[♦]
[♥]University of Washington [♠]The Chinese University of Hong Kong (Shenzhen)
[♡]Shenzhen Research Institute of Big Data
[♦]The Hong Kong University of Science and Technology
[♥]{yhtian, fxia}@uw.edu [♠]songyan@cuhk.edu.cn
[♦]tongzhang@ust.hk

Abstract

Constituency parsing is a fundamental and important task for natural language understanding, where a good representation of contextual information can help this task. N-grams, which is a conventional type of feature for contextual information, have been demonstrated to be useful in many tasks, and thus could also be beneficial for constituency parsing if they are appropriately modeled. In this paper, we propose span attention for neural chart-based constituency parsing to leverage n-gram information. Considering that current chart-based parsers with Transformer-based encoder represent spans by subtraction of the hidden states at the span boundaries, which may cause information loss especially for long spans, we incorporate n-grams into span representations by weighting them according to their contributions to the parsing process. Moreover, we propose categorical span attention to further enhance the model by weighting n-grams within different length categories, and thus benefit long-sentence parsing. Experimental results on three widely used benchmark datasets demonstrate the effectiveness of our approach in parsing Arabic, Chinese, and English, where state-of-the-art performance is obtained by our approach on all of them.[1]

1 Introduction

Constituency parsing, which aims to generate a structured syntactic parse tree for a given sentence, is one of the most fundamental tasks in natural language processing (NLP), and plays an important role in many downstream tasks such as relation extraction (Jiang and Diesner, 2019), natural language inference (Chen et al., 2017), and machine translation (Ma et al., 2018). Recently,

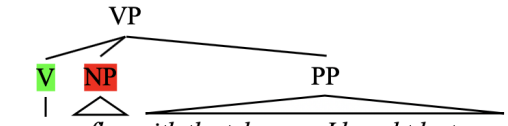

Figure 1: The treelet of an example of the form "*V+NP+PP*", where the "*PP*" should attach to the "*V*" (in green) rather than the "NP" (in red).

neural parsers (Vinyals et al., 2015; Dyer et al., 2016; Stern et al., 2017; Kitaev et al., 2019) without using any grammar rules significantly outperform conventional statistical grammar-based ones (Collins, 1997; Sagae and Lavie, 2005; Glaysher and Moldovan, 2006; Song and Kit, 2009), because neural networks, especially recurrent models (e.g, Bi-LSTM), are adept in capturing long range contextual information, which is essential to modeling the entire sentence. Particularly, a significant boost on the performance of chart-based parsers is observed from some recent studies (Kitaev and Klein, 2018; Kitaev et al., 2019; Zhou and Zhao, 2019) that employ advanced text encoders (i.e., Transformer, BERT, and XLNet), which further demonstrates the usefulness of contexts for parsing.

In general, besides powerful encoders, other extra information (such as pre-trained embeddings and extra syntactic information) can also provide useful contextual information and thus enhance model performance in many NLP tasks (Pennington et al., 2014; Song et al., 2018a; Zhang et al., 2019; Mrini et al., 2019; Tian et al., 2020a,b). As one type of the extra information, n-grams are used as a simple yet effective source of contextual feature in many studies (Song et al., 2009; Song and Xia, 2012; Yoon et al., 2018; Tian et al., 2020c) Therefore, they could be potentially beneficial for parsing as well. However, recent chart-based parers (Stern et al., 2017; Kitaev and Klein, 2018; Gaddy et al., 2018; Kitaev et al., 2019; Zhou and Zhao, 2019) make rare effort to leverage such n-gram information. Another potential issue with current

†Corresponding author.

[1]Our code and the best performing models are released at https://github.com/cuhksz-nlp/SAPar.

Findings of the Association for Computational Linguistics: EMNLP 2020, pages 1691–1703

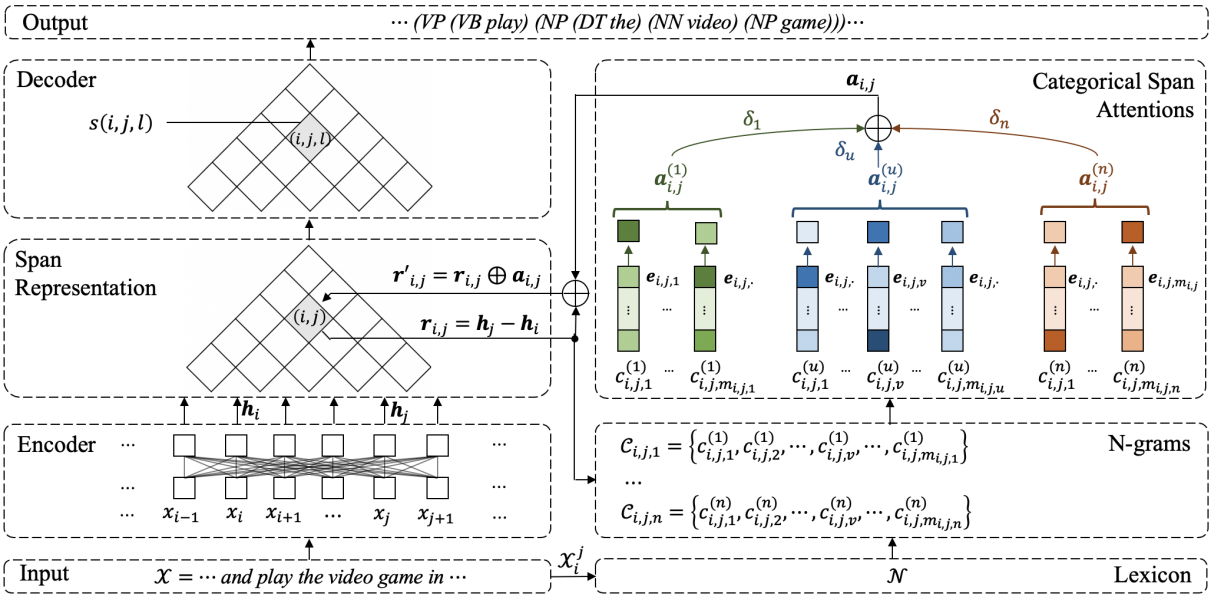

Figure 2: The architecture of the chart-based constituency parser with span attention, with an example partial input sentence and its output. The right part of the figure shows the categorical span attention, where extracted n-grams in span (i, j) are categorized by their length so that n-grams in different categories are weighted separately (different colors refer to different n-gram categories). Note that for normal span attention, all n-grams are weighted together, where attention $\mathbf{a}_{i,j}$ directly corresponds to $\mathbf{e}_{i,j,\cdot}$ in the figure.

chart-based parsers is that they represent spans by subtraction of hidden states at the span boundaries, where the context information in-between may be lost and thus hurt parsing performance especially for long sentences. N-grams can be a simple yet useful source to fill the missing information. For instance, Figure 1 illustrates the treelet of an example in the form of "*V+NP+PP*". As a classic example of PP-attachment ambiguity, a parser may wrongly attach the "*PP*" to the "*NP*" if it only focuses on the words at the boundaries of the text span "*flag ... year*" and in-between information is not represented properly. In this case, n-grams within that span (e.g., the uni-gram "*telescope*") can provide useful cues indicating that the "*PP*" should be attached to the "*V*". Although there are traditional non-neural parsers using n-grams as features to improve parsing (Sagae and Lavie, 2005; Pitler et al., 2010), they are limited in treating them euqally without learning their weights. Therefore, unimportant n-grams may deliver misleading information and lead to wrong predictions.

To address this problem, in this paper, we propose a span attention module to enhance chart-based neural constituency parsing by incorporating appropriate n-grams into span representations. Specifically, for each text span we extract all its substrings that appear in an n-gram lexicon; the span attention uses the normal attention mechanism to weight them with respect to their contributions to predict the constituency label of the span. Because in general short n-grams occur more frequently than long ones, they may dominate in the attention if all n-grams are globally weighted, We further enhance our approach with a categorical mechanism which first groups n-grams into different categories according to their length and then weights them within each category. Thus, n-grams with different lengths are separately treated and the infrequent long ones carrying more contextual information can be better leveraged. The effectiveness of our approach is illustrated by experimental results on three benchmark datasets from different languages (i.e., Arabic, Chinese, and English), on all of which state-of-the-art performance is achieved.

2 The Approach

Our approach follows the chart-based paradigm for constituency parsing, where the parse tree $\mathcal{T}$ of an input sentence $\mathcal{X} = x_1 x_2 \cdots x_i \cdots x_j \cdots x_q$ is represented as a set of labeled spans. A span is denoted by a triplet (i, j, l) with i and j referring to the beginning and ending positions of a span with a label $l \in \mathcal{L}$. Here, $\mathcal{L}$ is the label set containing d_l constituent types. The architecture of our approach is shown in Figure 2. The left side is the backbone chart-based parser. It assigns real value scores $s(i, j, l)$ to the labeled spans, then computes the score of a candidate tree by summing up the

scores of all its spans, and finally chooses a valid tree $\widehat{\mathcal{T}}$ with the highest score s by

$$\widehat{\mathcal{T}} = \arg\max_{\mathcal{T}} \sum_{\substack{(i,j,l)\in\mathcal{T} \\ 0<i<j\leq q}} s(i,j,l) \tag{1}$$

The right side of Figure 2 shows the proposed span attention to enhance the backbone parser, where n-grams in $\mathcal{X}$ are extracted from a pre-constructed lexicon $\mathcal{N}$ and are weighted through the attention module according to their contribution to the parsing process. Therefore, the process of computing $s(i,j,l)$ of the labeled spans through our approach is formalized by

$$s(i,j,l) = p(l|\mathcal{X}, \mathcal{SA}(\mathcal{X}_i^j, \mathcal{N})) \tag{2}$$

where $\mathcal{X}_i^j$ is the text in range $[i,j]$ of $\mathcal{X}$; $\mathcal{SA}$ represents the span attention module and p computes the probability of labeling $l \in \mathcal{L}$ to the span (i,j).

In this section, we start with a brief introduction of neural chart-based parsing, then describe our span attention, and end with an illustration of incorporating span attention into the parsing process.

2.1 Neural Chart-based Parsing

Recent neural chart-based parsers (Stern et al., 2017; Kitaev and Klein, 2018; Kitaev et al., 2019; Zhou and Zhao, 2019) follow the encoder-decoder way, where the encoder receives $\mathcal{X}$ and generates a sequence of context-sensitive hidden vectors (denoted as $\mathbf{h}_i$ and $\mathbf{h}_j$ for x_i and x_j, respectively), which are used to compute the span representation $\mathbf{r}_{i,j} \in \mathbb{R}^{d_r}$ for (i,j) by subtraction: $\mathbf{r}_{i,j} = \mathbf{h}_j - \mathbf{h}_i$. This span representation assumes that, for a recurrent model, e.g., LSTM, its hidden vector at each time step relies on the previous ones so that such subtraction could, to some extent, capture the contextual information of all the words in that span.[2]

For decoders, most recent neural chart-based parsers follow the strategy proposed by Stern et al. (2017), where all span representations $\mathbf{r}_{i,j}$ are fed into a variant of CYK algorithm to generate a globally optimized tree for each sentence. Normally, $\mathbf{r}_{i,j}$ is fed into multi-layer perceptrons (MLP) to compute its scores $s(i,j,\cdot)$ over the label set $\mathcal{L}$. Afterwards, a recursion function is applied to find the highest score $s^*(i,j)$ of span (i,j), which is computed by searching the best constituency label and the corresponding boundary k $(i<k<j)$ by

$$\begin{aligned} s^*(i,j) &= \max_{l\in\mathcal{L}} s(i,j,l) \\ &+ \max_{i<k<j} [s^*(i,k) + s^*(k,j)] \end{aligned} \tag{3}$$

Note that in the special case where $j = i+1$, the best score only relies on the candidate label:

$$s^*(i,j) = \max_{l\in\mathcal{L}} s(i,j,l) \tag{4}$$

Therefore, to parse the entire sentence, one computes $s^*(1,q)$ through the above steps and use a back pointer to recover the full tree structure.

2.2 Span Attention

Although the encoding from subtraction of hidden states is demonstrated to be effective (Stern et al., 2017; Kitaev and Klein, 2018; Kitaev et al., 2019), the subtraction might not represent all the crucial information in the text span. Especially, for Transformer-based encoders, unlike recurrent models, their $\mathbf{h}_i$ and $\mathbf{h}_j$ have no strong dependency on each other so that subtraction may fail to fully capture the contextual information in the span, especially when the span is long. Since n-grams are a good source of the information in the text span, we propose span attention to incorporate weighted n-gram information into span representations to help score the spans (i,j,l).

In detail, for each span (i,j) in $\mathcal{X}$, we extract all n-grams in that span that appear in Lexicon $\mathcal{N}$ to form a set $\mathcal{C}_{i,j} = \{c_{i,j,1}, c_{i,j,2}, \cdots c_{i,j,v}, \cdots c_{i,j,m_{i,j}}\}$ and use the set in span attention. The attention of each n-gram $c_{i,j,v}$ for (i,j) is activated by

$$a_{i,j,v} = \frac{\exp(\mathbf{r}_{i,j}^\top \cdot \mathbf{e}_{i,j,v})}{\sum_{v=1}^{m_{i,j}} \exp(\mathbf{r}_{i,j}^\top \cdot \mathbf{e}_{i,j,v})} \tag{5}$$

where $\mathbf{e}_{i,j,v} \in \mathbb{R}^{d_r}$ is the embedding of $c_{i,j,v}$ whose dimension is identical to that of $\mathbf{r}_{i,j}$. The resulted attention vector $\mathbf{a}_{i,j} \in \mathbb{R}^{d_r}$ is thus computed by the weighted average of n-gram embeddings by

$$\mathbf{a}_{i,j} = \sum_{v=1}^{m_{i,j}} a_{i,j,v} \mathbf{e}_{i,j,v} \tag{6}$$

and it is used to enhance the span representation.

In normal attention, all n-grams are weighted globally and short n-grams may dominate the attention because they occur much more frequently than long ones and are intensively updated. However, there are cases that long n-grams can play

[2]Note that this paper focuses on improving the current best performing span representation (i.e., by hidden vector subtraction) proposed by Stern et al. (2017) so as to make a fair comparison, although there are other possible approaches to representing a span (e.g., max pooling).

an important role in parsing when they carry useful context and boundary information. Therefore, we extend the span attention with a category mechanism (namely, categorical span attention) by grouping n-grams based on their lengths and weighting them within each category.[3] In doing so, all n-grams in $\mathcal{N}$ are categorized into n groups according to their lengths, i.e., $\mathcal{C}_{i,j} = \{\mathcal{C}_{i,j,1}, \mathcal{C}_{i,j,2}, \cdots \mathcal{C}_{i,j,u}, \cdots \mathcal{C}_{i,j,n}\}$, with $u \in [1, n]$ denoting the n-gram length. Then, for each category with n-grams in length u, we follow the same process in Eq. (5) and (6) to compute $a^{(u)}_{i,j,v}$ and $\mathbf{a}^{(u)}_{i,j}$. The final attention is obtained from the concatenation of all categorical attentions by

$$\mathbf{a}_{i,j} = \bigoplus_{1 \le u \le n} \delta_u \mathbf{a}^{(u)}_{i,j} \quad (7)$$

with a trainable parameter $\delta_u \in \mathbb{R}^+$ to balance the contribution of attentions from different categories.

2.3 Parsing with Span Attention

The backbone parser follows Kitaev et al. (2019) to use BERT as the encoder, where $\mathbf{r}_{i,j} = \mathbf{h}_j - \mathbf{h}_i$ is applied to represent the span (i, j). Once $\mathbf{a}_{i,j}$ is obtained from the span attention for (i, j), we incorporate it into the backbone parsing process by directly concatenating it with $\mathbf{r}_{i,j}$: $\mathbf{r}'_{i,j} = \mathbf{r}_{i,j} \oplus \mathbf{a}_{i,j} \in \mathbb{R}^{d_r \cdot (n+1)}$. Then, we apply two fully connected layers with $ReLU$ activation function to $\mathbf{r}'_{i,j}$ and compute the span scores $s(i, j, \cdot)$ over the label set $\mathcal{L}$, which can be formalized by:

$$\mathbf{o}_{i,j} = ReLU(LN(\mathbf{W}_1 \cdot \mathbf{r}'_{i,j} + \mathbf{b}_1)) \quad (8)$$

and

$$s(i, j, \cdot) = \mathbf{W}_2 \cdot \mathbf{o}_{i,j} + \mathbf{b}_2 \quad (9)$$

Here, LN denotes the layer normalization operation; $\mathbf{W}_1$, $\mathbf{W}_2$ and $\mathbf{b}_1$, $\mathbf{b}_2$ are trainable parameters in the fully connected layers. Afterwards, we use Eq. (3) and (4) to recursively find the highest score $s_{best}(1, q)$, and use a back pointer to recover the globally optimized parse tree.

3 Experimental Settings

3.1 Datasets

We test our approach on Arabic, Chinese and English benchmark datasets, namely part 1-3 of the Arabic Penn Treebank 2.0 (ATB) (Maamouri et al., 2004), the Chinese Penn Treebank 5 (CTB5) (Xue et al., 2005), and Penn Treebank 3 (PTB) (Marcus et al., 1993).[4] For ATB, we follow Chiang et al. (2006) and Green and Manning (2010) to use their split[5] to get the training/dev/test sets and convert the texts in the dataset from Buckwalter transliteration[6] to modern standard Arabic. For CTB5 and PTB, we follow Shen et al. (2018) and Kamigaito et al. (2017) to split the datasets. Moreover, we use the Brown Corpus (Marcus et al., 1993) and Genia (Tateisi et al., 2005) for cross-domain experiments.[7] For all datasets, we follow Suzuki et al. (2018) to clean up the raw data[8] and report the statistics of each resulted dataset in Table 1.

Datasets		Sent	Token	ASL
ATB	Train	16K	596K	31.4
	Dev	2K	70K	30.5
	Test	2K	70K	29.9
CTB5	Train	17K	478K	27.4
	Dev	350	7K	19.5
	Test	348	8K	23.0
PTB	Train	40K	950K	23.9
	Dev	2K	40K	23.6
	Test	2K	57K	23.5
Brown (Full)		24K	458K	19.0
Genia (Full)		17K	446K	26.2

Table 1: The statistics of all experimental datasets (with splits) in terms of sentence and token numbers, and average sentence length (ASL).

3.2 N-gram Lexicon Construction

For n-gram extraction, we compute the pointwise mutual information (PMI) of any two adjacent words x', x'' in the dataset by

$$PMI(x', x'') = \log \frac{p(x'x'')}{p(x')p(x'')} \quad (10)$$

where p is the probability of an n-gram (i.e., x', x'' and $x'x''$) in a dataset. A high PMI score suggests

[3] We use length as the categorization criterion because (1) n-gram frequencies vary in different datasets and it is hard to find an appropriate scheme to divide them; (2) n-grams with the same length may have similar ability to deliver contextual information so they are suitable to be grouped by such ability.

[4] All the datasets are obtained from the official release of Linguistic Data Consortium. The catalog numbers for ATB part 1-3 are LDC2003T06, LDC2004T02, LDC2005T20, for CTB5 is LDC2005T01, and for PTB is LDC99T42.

[5] Such split uses the "Johns Hopkins 2005 Workshop" standard, for which we follow the detailed split guideline offered by https://nlp.stanford.edu/software/parser-arabic-data-splits.shtml.

[6] http://languagelog.ldc.upenn.edu/myl/ldc/morph/buckwalter.html

[7] The Brown Corpus is obtained together with PTB (LDC99T42), and the Genia corpus is obtained by its official PTB format from https://nlp.stanford.edu/~mcclosky/biomedical.html.

[8] We use the clean-up code from https://github.com/nikitakit/parser-data-gen.

Data	Models	- POS					+ POS				
		Parm	P	R	F1	M	Parm	P	R	F1	M
ATB	**BERT**	188M	82.99	82.99	82.99	18.87	188M	82.96	83.17	83.07	19.09
	+ SA	191M	**83.36**	83.05	83.21	19.13	191M	83.37	83.12	83.24	19.43
	+ CatSA	192M	83.33	**83.20**	**83.27**	**20.04**	192M	**83.41**	**83.20**	**83.30**	**19.65**
CTB5	**BERT**	113M	93.95	93.35	93.65	47.71	113M	94.30	93.88	94.09	48.86
	+ SA	116M	**94.07**	93.39	93.73	49.43	116M	**94.80**	93.73	94.26	49.14
	+ CatSA	117M	94.02	**93.65**	**93.83**	**50.00**	117M	94.70	**94.00**	**94.35**	**50.00**
	ZEN	235M	93.82	93.65	93.73	50.29	235M	94.37	93.69	94.03	48.87
	+ SA	238M	94.08	93.53	93.80	51.14	238M	94.68	93.81	94.24	51.43
	+ CatSA	239M	**94.23**	**93.66**	**93.94**	**51.41**	239M	**94.69**	**93.91**	**94.30**	**52.00**
PTB	**BERT-LC**	344M	95.71	95.53	95.62	54.06	344M	95.71	95.61	95.66	53.35
	+ SA	349M	95.80	**95.55**	95.68	53.94	349M	95.71	95.70	95.70	54.29
	+ CatSA	350M	**96.02**	95.51	**95.77**	**54.64**	350M	**95.79**	**95.85**	**95.82**	**55.79**
	BERT-LU	345M	95.61	95.59	95.60	54.29	345M	95.59	95.76	95.67	54.24
	+ SA	350M	95.61	95.71	95.66	54.24	350M	95.69	95.75	95.72	54.53
	+ CatSA	351M	**95.76**	**95.74**	**95.75**	**55.29**	351M	**95.77**	**95.84**	**95.80**	**54.71**
	XLNet-LC	371M	95.78	95.79	95.78	54.81	371M	95.97	95.60	95.79	54.70
	+ SA	375M	95.83	**95.95**	95.89	54.94	375M	95.92	95.95	95.93	55.71
	+ CatSA	376M	**96.02**	95.84	**95.93**	**55.88**	376M	**95.97**	**96.02**	**95.99**	**56.06**

Table 2: Experimental results in terms of precision (P), recall (R), F-score (F1) and complete match score (M) of our models on the development set of ATB, CTB5 and PTB with different configurations, i.e., with and without POS, span attention (SA), and categorical span attention (CatSA). The boldface is added to the highest result (P, R, F1, and M) within each group of three models (one from BERT/XLNet baseline, one with SA, and the other with CatSA). For English, we use large cased (LC) version of BERT and XLNet and large uncased (LU) version of BERT. PARM reports the number of trainable parameters in each model.

that the two words co-occur a lot in the dataset and are more likely to form an n-gram. We set the threshold to 0 to determine whether a delimiter should be inserted between the two adjacent words x' and x''. In other words, to build the lexicon $\mathcal{N}$ from a dataset, we use PMI as an unsupervised segmentation method to segment the dataset and collect all n-grams (n $\leq$ 5)[9] appearing at least twice in the training and development sets combined.[10]

3.3 Model Implementation

In our experiments, we use BERT (Devlin et al., 2019) as the basic encoder for all three languages and use ZEN (Diao et al., 2019) and XLNet-large (Yang et al., 2019) for Chinese and English, respectively.[11] For BERT, ZEN, and XLNet, we use the default hyper-parameter settings. (e.g., 24 layers with 1024 dimensional hidden vector for the large models). In addition, following Kitaev et al. (2019), Zhou and Zhao (2019) and Mrini et al. (2019), we add three additional token-level self-attention layers to the top of BERT, ZEN, and XLNet.

For other settings, we randomly initialize all n-gram embeddings used in our attention module[12] with their dimension matching that of the hidden vectors obtained from the encoder (e.g., 1024 for BERT-large). Besides, we run our experiments with and without predicted part-of-speech (POS) tags. Following previous studies, for the experiments without POS tags, we take sentences as the only input; for the experiments with POS tags, we obtain the POS tags from Stanford POS Tagger (Toutanova et al., 2003) and incorporate the POS tags by directly concatenating their embeddings with the output of the BERT/ZEN/XLNet encoder. Following previous studies (Suzuki et al., 2018; Kitaev et al., 2019), we use hinge loss during the training process and evaluate different models by by precision, recall, F1 score, and complete match score via the standard evaluation toolkit Evalb[13].

During the training process, we try three learning rates, i.e., 5e-5, 1e-5, 5e-6, with a fixed random seed, pick the model with the best F1 score on the development set, and evaluate it on the test set.

[9] We empirically set the max n-gram length to 5 as a unified threshold for all three languages.

[10] We show the details of extracting the lexicon with example n-grams in the Appendix.

[11] We download BERT models for Arabic and English from https://github.com/google-research/bert, and for Chinese from https://s3.amazonaws.com/models.huggingface.co/. We download ZEN and XLNet at https://github.com/sinovation/ZEN amd https://github.com/zihangdai/xlnet.

[12] We also try initializing the n-grams with pre-trained embeddings (Pennington et al., 2014; Song et al., 2018b; Yamada et al., 2020), where the results show small differences.

[13] https://nlp.cs.nyu.edu/evalb/

Models	ATB			CTB5			PTB		
	P	R	F1	P	R	F1	P	R	F1
Green and Manning (2010)	78.92	77.72	78.32	-	-	-	-	-	-
Shen et al. (2018)	-	-	-	86.6	86.4	86.5	92.0	91.7	91.8
Teng and Zhang (2018)	-	-	-	88.0	86.6	87.3	92.5	92.2	92.4
Joshi et al. (2018)	-	-	-	-	-	-	94.8	93.8	94.3
Suzuki et al. (2018)	-	-	-	-	-	-	-	-	94.32
Kitaev and Klein (2018)	-	-	-	-	-	-	95.40	94.85	95.13
Kitaev et al. (2019) (BERT)	-	-	-	91.96	91.55	91.75	95.73	95.46	95.59
Fried et al. (2019) (BERT)	-	-	-	-	-	92.14	-	-	95.71
Zhou and Zhao (2019) (BERT)	-	-	-	92.03	92.33	92.18	95.70	95.98	95.84
Zhou and Zhao (2019) (XLNet)	-	-	-	-	-	-	96.21	96.46	96.33
*Mrini et al. (2019) (BERT/XLNet + POS)	-	-	-	91.85	**93.45**	92.64	96.24	**96.53**	96.38
SCT (Manning et al., 2014)	68.33	71.78	70.02	†	†	†	86.21	86.73	86.47
BNP (Kitaev and Klein, 2018)	72.84	76.59	74.67	91.83	91.53	91.68	95.46	94.89	95.17
BERT	83.06	82.87	82.96	92.16	91.98	92.07	95.91	95.17	95.54
+ SA	83.25	82.85	83.05	92.31	92.03	92.17	96.04	95.40	95.72
+ CatSA	83.40	**83.11**	83.26	92.25	92.14	92.20	96.11	95.58	95.85
ZEN/XLNet	-	-	-	92.20	92.05	92.13	96.52	95.70	96.11
+ SA	-	-	-	92.34	92.02	92.18	96.58	96.03	96.31
+ CatSA	-	-	-	92.50	91.98	92.24	**96.64**	96.07	96.36
*BERT + POS	82.98	82.97	82.97	92.52	92.06	92.29	95.92	95.27	95.60
+ SA	83.36	82.80	83.08	92.61	92.20	92.40	95.96	95.51	95.73
+ CatSA	**83.48**	83.07	**83.27**	**92.83**	92.50	**92.66**	96.09	95.62	95.86
*ZEN/XLNet + POS	-	-	-	92.37	92.16	92.26	96.42	95.86	96.14
+ SA	-	-	-	92.40	92.32	92.36	96.56	96.10	96.33
+ CatSA	-	-	-	92.61	92.42	92.52	96.61	96.19	**96.40**

Table 3: Comparing (in terms of Precison, Recall and F1 scores) our best performing models (BERT-LC and ZEN/XLNet-LC) with previous studies and prevailing toolkits (i.e., SCT and BNP) on the test sets of ATB, CTB5 and PTB. The results for SCT are not comparable to other systems including ours (as indicated by †) because SCT is trained on a different dataset. Models marked by * use predicted POS tags as additional input.

4 Results and Analyses

4.1 Overall Performance

In the main experiment, we compare the proposed models with and without the span attention to explore the effect of the span attention on chart-based constituency parsing. For models with the span attention, we also run the settings with and without the categorical mechanism. The results (i.e., precision, recall, F1 score, and complete match scores of all models, as well as their number of trainable parameters) with different configurations (including whether to use the predicted POS tags) on the development sets of ATB, CTB5, and PTB are reported in Table 2.

There are several observations. First, the span attention over n-grams shows its generalization ability, where consistent improvements of F1 over the baseline models are observed on all languages under different settings (i.e., with and without using predicted POS tags; using BERT or XLNet encoders). Second, compared with span attention without the category mechanism, in which n-grams are weighted together, models with categorical span attention perform better on both F1 and complete match scores with a relatively small increase of parameter numbers (around $1M$). Particularly, for the complete match scores, the span attention with normal attentions does not outperform the baseline models in some cases, whereas the categorical span attention mechanism does in all cases. These results could be explained by that frequent short n-grams dominate the general attentions so that the long ones containing more contextual information fail to function well in filling the missing information in the span representation, and thus harm the understanding of long spans, which results in inferior results in complete match score. In contrast, the categorical span attention is able to weight n-grams in different length separately, so that the attentions are not dominated by high-frequency short n-grams and thus reasonable weights can be assigned to long n-grams. Therefore, our model can learn from the important long n-grams and have a good performance on the long spans, which results in consistent improvements over baseline models in complete match scores. Third, on CTB5, models with ZEN encoder consistently outperform the ones with BERT without using POS tags, while they fail to do so with the

MODELS	BROWN	GENIA
BERT (FRIED ET AL., 2019)	93.10	87.54
BERT	93.13	**87.58**
+ SA	93.24	87.50
+ CATSA	**93.29**	87.53

Table 4: Cross-domain experiment results (F1 scores) from previous studies and our models (based on BERT-LC), on the entire Brown and Genia corpora when trained from the training set of PTB.

POS tags as the additional input, which suggests that the predicted POS tags may have more conflict with ZEN compared with BERT.

Moreover, we run our models on the test set of each dataset and compare the results with previous studies, as well as the ones from prevailing parsers, i.e., Stanford CoreNLP Toolkits (SCT)[14] (Manning et al., 2014) and Berkeley Neural Parser (BNP)[15] (Kitaev and Klein, 2018). The results are reported in Table 3, where the models using predicted POS tags are marked with "*".[16] Our models with CATSA outperform previous best performing models from Zhou and Zhao (2019) and Mrini et al. (2019) under different settings (i.e., whether to use the predicted POS tags), and achieve state-of-the-art performance on all datasets. Compared with Zhou and Zhao (2019) and Mrini et al. (2019) which improve constituency parsing by leveraging the dependency information when training their head phrase structure grammar (HPSG) parser, our approach enhances the task from another direction by incorporating n-gram information through the span attentions as a way to address the limitation of using hidden vector subtraction to represent spans.

4.2 Cross-domain Experiments

To further explore whether our approach can be generalized across domains, we follow the setting of Fried et al. (2019) to conduct cross-domain experiments on the Brown and Genia corpus using the models with SA and CATSA, as well as their corresponding baseline. Note that, for fair comparison, we use BERT-large cased as the encoder without using the predicted POS tags. We follow Fried et al. (2019) to train models on the training set of PTB and evaluate them on the entire Brown corpus and the entire Genia corpus. To construct

[14]We use the version of 3.9.2 obtained from https://stanfordnlp.github.io/CoreNLP/.

[15]We obtain their models from https://github.com/nikitakit/self-attentive-parser.

[16]For our models with BERT encoder, we only report the results of the ones using the cased version of BERT-large.

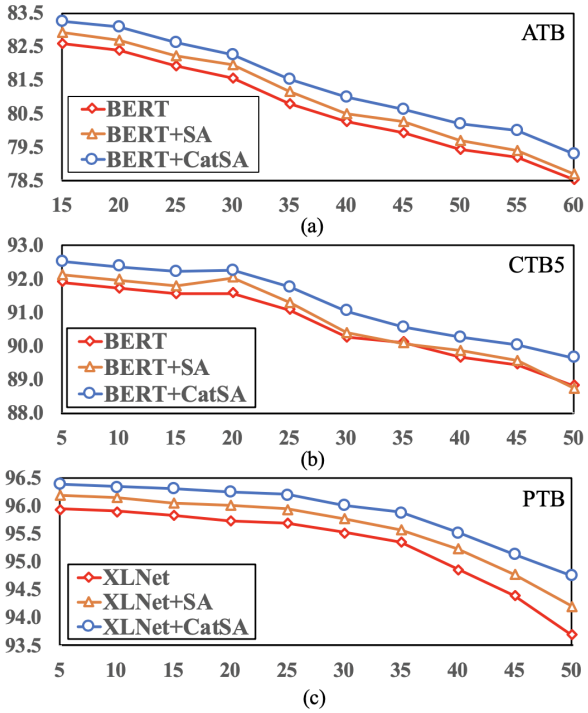

Figure 3: The F1 curves with respect to the minimal test sentence length (the horizontal axis) of different models performed on ATB (a), CTB (b), and PTB (c).

$\mathcal{N}$ in this experiment, we extract n-grams by PMI from the training set of PTB. The results (F1 scores) are reported in Table 4. From the table, we find that our model with categorical span attentions (+ CATSA) outperforms the BERT baseline (Fried et al., 2019) on the Brown corpus while fails to do so on the Genia corpus. The explanation cloud be that the distance between Genia (medical domain) and PTB (news wire domain) is much larger than that between Brown and PTB, so that the n-gram overlap in two domains are limited and thus has little influence to the target domain.

4.3 Effect of CATSA on Long Sentences

To explore the effect of our approach, we investigate our best performing models (where predicted POS tags are used) with the span attention module and the corresponding baselines on different length of sentences in the test sets. The curves of F1 scores with respect to the minimal test sentence length (the horizontal axis) from different models on ATB, CTB5, and PTB are illustrated in Figure 3(a), 3(b), and 3(c), respectively.[17]

In general, long sentences are harder to parse and thus all models' performance degrades when sentence length increases. Yet, our models with CatSA

[17]Given the variance of average sentence length in different datasets (see Table 1), we set the minimal length from 5 to 50 on CTB5 and PTB, and 15 to 60 on ATB, with a step of 5.

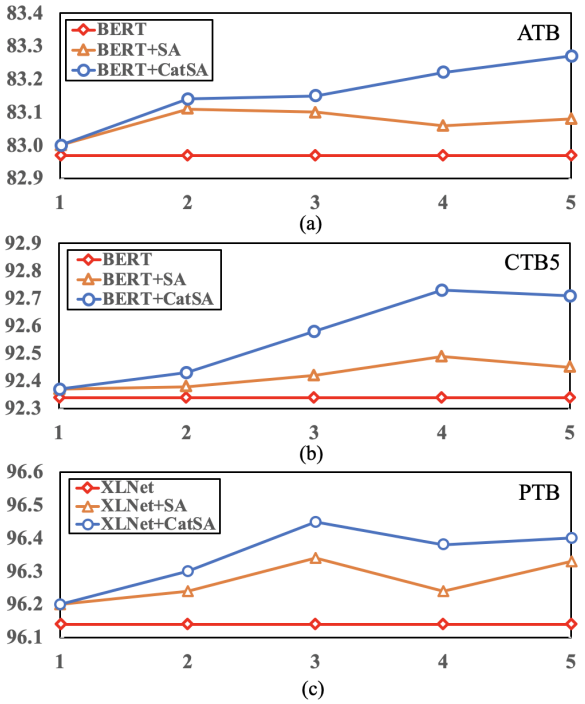

Figure 4: The F1 curves with respect to the max length (the horizontal axis) of n-grams used in different models performed on ATB (a), CTB (b), and PTB (c).

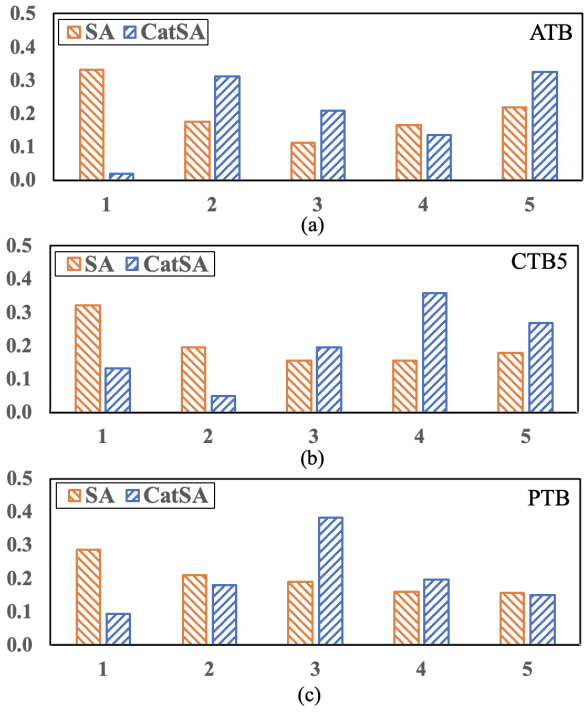

Figure 5: The histograms of average weights assigned to n-gram categories in different lengths, with weights from SA and CATSA show different patterns.

outperform the baseline for all sentence groups and the gap is bigger for long sentences, which indicates our approach can handle long sentences better than the baselines. One possible explanation for this is that long sentences will have larger text spans and may require more long-distance contextual information. Our approach incorporates n-gram information into the span representation and thus can appropriately leverages the infrequent long n-grams by separately weighting them in different categories.

4.4 Analysis on Different N-gram Lengths

To test using n-grams in different length, we conduct an ablation study on the n-grams with respect to their length. In doing so, we conduct experiments on the best performing models (where predicted POS tags are used) with the span attention module, by restricting that n-grams whose length are larger than a threshold is excluded from the lexicon $\mathcal{N}$. We try the threshold from 1 to 5 and demonstrate the curves (F-scores) on the test set of ATB, CTB5, and PTB in Figure 4(a), (b), and (c), respectively. The results of their corresponding baselines are also represented in red curves for reference. It is found from the curves that our models with span attentions consistently outperform the baseline models, which indicates the robustness of our approach with respect to different n-grams used in the model. In addition, for different languages, the n-gram threshold varies when the best performance is obtained. For example, the best performing model on English is with three words as the maximum length of n-grams, while that is five for Arabic and four for Chinese.

Moreover, to investigate how the categorical span attention addresses the problem that high-frequency short n-grams can dominate the general attentions, we run the best performing models with span attentions on the whole ATB, CTB5, and PTB datasets, obtain the total weight assigned to each n-gram, and compute the average weight for the n-grams in each n-gram length category. Figure 5 shows the histograms of the average weights from models with SA and CATSA.

The histograms show that the models with SA (the orange bars) tend to assign short n-grams relatively high weights, especially the uni-grams. This is not surprising because short n-grams occur more frequently and are thus updated more times than long ones. In contrast, the models with CATSA show a different weight distribution (the blue bars) among n-grams with different lengths, which indicates that the CATSA module could balance the weights distribution and thus enable the model to learn from infrequent long n-grams.

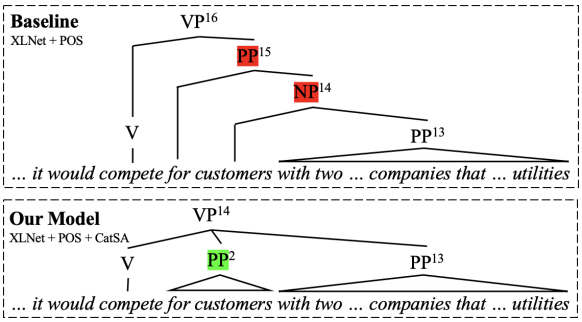

Figure 6: An example sentence with its parsing results from the best performing baseline and our model. The correct and wrong parsing results are highlighted on the span labels by green and red, respectively. The superscripts on the span labels illustrate the heights of them. "V" is a POS tag so there is no height for it.

4.5 Case Study

To illustrate how our model improves baselines with the span attention, especially for long sentences, we show the parse trees produced by the two models for an example sentence in Figure 6, where the superscript for the internal node is the height of the subtree rooted at that node. In this case, our model correctly attaches the "*PP*" ("*with two ... utilities*") containing 24 words to the verb "*compete*", while the baseline attach it to the noun "*customers*". Since the distances between the boundary positions of the wrongly predicted spans (highlighted in red) are relatively long, the baseline system, which simply represents the span as subtraction of the hidden vectors at the boundary positions, may fail to capture the important context information within the text span. In contrast, the span representations used in our model are enhanced by weighted n-gram information and thus contain more context information. Therefore, in deciding which component (i.e., "*compete*" or "*customer*") the *with*-PP should attach to, n-grams (e.g., the uni-gram "*companies*") may provide useful cues, since "customers with companies" is less likely than "compete with companies".

5 Related Work

There are two main types of parsing methodologies. One is the transition-based approaches (Sagae and Lavie, 2005); the other is the chart-based approaches (Collins, 1997; Glaysher and Moldovan, 2006). Recently, neural methods start to play a dominant role in this task, where improvements mainly come from powerful encodings (Dyer et al., 2016; Cross and Huang, 2016; Liu and Zhang, 2017; Stern et al., 2017; Gaddy et al., 2018; Kitaev and Klein, 2018; Kitaev et al., 2019; Fried et al., 2019). Moreover, there are studies that do not follow the aforementioned methodologies, which instead regard the task as a sequence-to-sequence generation task (Vinyals et al., 2015; Suzuki et al., 2018), a language modeling (Choe and Charniak, 2016) task or a sequence labeling task (Gómez-Rodríguez and Vilares, 2018). To further improve the performance, some studies leverage extra resources (such as auto-parsed large corpus (Vinyals et al., 2015), pre-trained word embeddings (Kitaev and Klein, 2018)), HPSG information (Zhou and Zhao, 2019; Mrini et al., 2019), or use model ensembles (Kitaev et al., 2019). Compared to these studies, our approach offers an alternative way to enhance constituency parsing with effective leveraging of n-gram information. Moreover, the proposed span attention addresses the limitation of previous studies (Kitaev and Klein, 2018; Kitaev et al., 2019) that spans are represented by the subtraction of encoded vectors at span boundaries (i.e., the hidden states at initial and ending positions of the span) and thus reduces information loss accordingly. In addition, the categorical span attention provides a simple, yet effective, improvement over the normal attention to process n-grams in a more precise way, which could become a reference for leveraging similar resources in future research.

6 Conclusion

In this paper, we proposed span attention to integrate n-gram into span representations to enhance chart-based neural constituency parsing. Specifically, for each text span in an input sentence, we firstly extracted n-grams in that span from an n-gram lexicon, and then fed them into the span attention to weight them according to their contribution to the parsing process. To better leverage n-grams, especially the long ones, categorical span attention was proposed to improve the normal attention by categorizing n-grams according to their length and weighting them separately within each category. Such span attention not only leverages important contextual information from n-grams but also addresses the limitation of current Transformer-based encoders using subtraction for span representations. To the best our knowledge, this is the first work using n-grams for neural constituency parsing. The effectiveness of our approach was demonstrated by experimental results on three benchmark datasets from Arabic, Chinese, and English, where state-of-the-art performance is obtained on all of them.

References

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced LSTM for Natural Language Inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1657–1668, Vancouver, Canada.

David Chiang, Mona Diab, Nizar Habash, Owen Rambow, and Safiullah Shareef. 2006. Parsing Arabic Dialects. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, Trento, Italy.

Do Kook Choe and Eugene Charniak. 2016. Parsing as Language Modeling. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2331–2336, Austin, Texas.

Michael Collins. 1997. Three Generative, Lexicalised Models for Statistical Parsing. In *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 16–23, Madrid, Spain.

James Cross and Liang Huang. 2016. Span-Based Constituency Parsing with a Structure-Label System and Provably Optimal Dynamic Oracles. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1–11, Austin, Texas.

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.

Shizhe Diao, Jiaxin Bai, Yan Song, Tong Zhang, and Yonggang Wang. 2019. ZEN: Pre-training Chinese Text Encoder Enhanced by N-gram Representations. *ArXiv*, abs/1911.00720.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A. Smith. 2016. Recurrent Neural Network Grammars. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 199–209, San Diego, California.

Daniel Fried, Nikita Kitaev, and Dan Klein. 2019. Cross-Domain Generalization of Neural Constituency Parsers. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 323–330, Florence, Italy.

David Gaddy, Mitchell Stern, and Dan Klein. 2018. What's Going On in Neural Constituency Parsers? An Analysis. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 999–1010, New Orleans, Louisiana.

Elliot Glaysher and Dan Moldovan. 2006. Speeding Up Full Syntactic Parsing by Leveraging Partial Parsing Decisions. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 295–300, Sydney, Australia.

Carlos Gómez-Rodríguez and David Vilares. 2018. Constituent Parsing as Sequence Labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1314–1324, Brussels, Belgium.

Spence Green and Christopher D. Manning. 2010. Better Arabic Parsing: Baselines, Evaluations, and Analysis. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 394–402, Beijing, China.

Ming Jiang and Jana Diesner. 2019. A Constituency Parsing Tree based Method for Relation Extraction from Abstracts of Scholarly Publications. In *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, pages 186–191, Hong Kong.

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.

Hidetaka Kamigaito, Katsuhiko Hayashi, Tsutomu Hirao, Hiroya Takamura, Manabu Okumura, and Masaaki Nagata. 2017. Supervised Attention for Sequence-to-Sequence Constituency Parsing. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 7–12, Taipei, Taiwan.

Nikita Kitaev, Steven Cao, and Dan Klein. 2019. Multilingual Constituency Parsing with Self-Attention and Pre-Training. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3499–3505, Florence, Italy.

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.

Jiangming Liu and Yue Zhang. 2017. In-Order Transition-based Constituent Parsing. *Transactions of the Association for Computational Linguistics*, 5:413–424.

Chunpeng Ma, Akihiro Tamura, Masao Utiyama, Tiejun Zhao, and Eiichiro Sumita. 2018. Forest-Based Neural Machine Translation. In *Proceedings*

of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1253–1263, Melbourne, Australia.

Mohamed Maamouri, Ann Bies, Tim Buckwalter, and Wigdan Mekki. 2004. The penn arabic treebank: Building a large-scale annotated arabic corpus. In *NEMLAR conference on Arabic language resources and tools*, volume 27, pages 466–467.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland.

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.

Khalil Mrini, Franck Dernoncourt, Trung Bui, Walter Chang, and Ndapa Nakashole. 2019. Rethinking Self-Attention: An Interpretable Self-Attentive Encoder-Decoder Parser. *arXiv preprint arXiv:1911.03875*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Emily Pitler, Shane Bergsma, Dekang Lin, and Kenneth Church. 2010. Using Web-scale N-grams to Improve Base NP Parsing Performance. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 886–894, Beijing, China.

Kenji Sagae and Alon Lavie. 2005. A Classifier-Based Parser with Linear Run-Time Complexity. In *Proceedings of the Ninth International Workshop on Parsing Technology*, pages 125–132, Vancouver, British Columbia.

Yikang Shen, Zhouhan Lin, Athul Paul Jacob, Alessandro Sordoni, Aaron Courville, and Yoshua Bengio. 2018. Straight to the Tree: Constituency Parsing with Neural Syntactic Distance. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1171–1180, Melbourne, Australia.

Yan Song and Chunyu Kit. 2009. PCFG Parsing with CRF Tagging for Head Recognition. *Proceedings of CIPS-ParsEval*, pages 133–137.

Yan Song, Chunyu Kit, and Xiao Chen. 2009. Transliteration of Name Entity via Improved Statistical Translation on Character Sequences. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration (NEWS 2009)*, Suntec, Singapore.

Yan Song, Shuming Shi, and Jing Li. 2018a. Joint Learning Embeddings for Chinese Words and their Components via Ladder Structured Networks. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4375–4381.

Yan Song, Shuming Shi, Jing Li, and Haisong Zhang. 2018b. Directional Skip-Gram: Explicitly Distinguishing Left and Right Context for Word Embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2*, pages 175–180, New Orleans, Louisiana.

Yan Song and Fei Xia. 2012. Using a Goodness Measurement for Domain Adaptation: A Case Study on Chinese Word Segmentation. In *LREC*, pages 3853–3860.

Mitchell Stern, Jacob Andreas, and Dan Klein. 2017. A Minimal Span-Based Neural Constituency Parser. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 818–827, Vancouver, Canada.

Jun Suzuki, Sho Takase, Hidetaka Kamigaito, Makoto Morishita, and Masaaki Nagata. 2018. An Empirical Study of Building a Strong Baseline for Constituency Parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 612–618, Melbourne, Australia.

Yuka Tateisi, Akane Yakushiji, Tomoko Ohta, and Jun'ichi Tsujii. 2005. Syntax Annotation for the GENIA Corpus. In *Companion Volume to the Proceedings of Conference including Posters/Demos and tutorial abstracts*.

Zhiyang Teng and Yue Zhang. 2018. Two Local Models for Neural Constituent Parsing. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 119–132, Santa Fe, New Mexico, USA.

Yuanhe Tian, Yan Song, Xiang Ao, Fei Xia, Xiaojun Quan, Tong Zhang, and Yonggang Wang. 2020a. Joint Chinese Word Segmentation and Part-of-speech Tagging via Two-way Attentions of Auto-analyzed Knowledge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8286–8296, Online.

Yuanhe Tian, Yan Song, and Fei Xia. 2020b. Supertagging Combinatory Categorial Grammar with Attentive Graph Convolutional Networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

Yuanhe Tian, Yan Song, Fei Xia, Tong Zhang, and Yonggang Wang. 2020c. Improving Chinese Word Segmentation with Wordhood Memory Networks. In *Proceedings of the 58th Annual Meeting of the*

Association for Computational Linguistics, pages 8274–8285.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 252–259.

Oriol Vinyals, Lukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a Foreign Language. In *Advances in Neural Information Processing Systems 28*, pages 2773–2781.

Naiwen Xue, Fei Xia, Fu-Dong Chiou, and Marta Palmer. 2005. The Penn Chinese TreeBank: Phrase structure annotation of a large corpus. *Natural language engineering*, 11(2):207–238.

Ikuya Yamada, Akari Asai, Jin Sakuma, Hiroyuki Shindo, Hideaki Takeda, Yoshiyasu Takefuji, and Yuji Matsumoto. 2020. Wikipedia2Vec: An Efficient Toolkit for Learning and Visualizing the Embeddings of Words and Entities from Wikipedia. *arXiv preprint 1812.06280v3*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32*, pages 5753–5763.

Seunghyun Yoon, Joongbo Shin, and Kyomin Jung. 2018. Learning to Rank Question-Answer Pairs Using Hierarchical Recurrent Encoder with Latent Topic Clustering. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1575–1584, New Orleans, Louisiana.

Hongming Zhang, Yan Song, and Yangqiu Song. 2019. Incorporating Context and External Knowledge for Pronoun Coreference Resolution. 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 872–881, Minneapolis, Minnesota.

Junru Zhou and Hai Zhao. 2019. Head-Driven Phrase Structure Grammar Parsing on Penn Treebank. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2396–2408, Florence, Italy.

Appendix A: Extracting the Lexicon using PMI

In our experiments, we build the n-gram lexicon $\mathcal{N}$ based on pointwise mutual information (PMI) with n-gram probability estimated from the union of the training and development set of each dataset. Specifically, we compute the PMI of any two adjacent words x', x'' in the dataset by

$$PMI(x', x'') = log\frac{p(x'x'')}{p(x')p(x'')} \tag{11}$$

where p is the probability of an n-gram (i.e., x', x'' and $x'x''$) in the dataset. A high PMI score suggests that the two words co-occur frequently in the dataset and are more likely to form an n-gram. For each pair of adjacent words x_{i-1}, x_i in a sentence $\mathcal{X} = x_1x_2 \cdots x_{i-1}x_i \cdots x_n$, we use a threshold to determine whether a delimiter should be inserted in between them. As a result, the sentence $\mathcal{X}$ is segmented into pieces of n-grams; we extract those n-grams to form the lexicon $\mathcal{N}$. For example, for a given sentence

$$\mathcal{X} = x_1x_2x_3x_4x_5 \tag{12}$$

and the PMI of all adjacent words (i.e., x_1x_2, x_2x_3, x_3x_4, x_4x_5) in it, where

$$PMI(x_2, x_3) > t \tag{13}$$

$$PMI(x_3, x_4) > t \tag{14}$$

and

$$PMI(x_1, x_2) < t \tag{15}$$

$$PMI(x_4, x_5) < t \tag{16}$$

with t denoting the threshold. We add delimiters (denoted by "/") between x_1 and x_2, and x_4 and x_5 since their PMI is lower than t. As a result, we obtain a segmented sentence

$$\mathcal{X}' = x_1/x_2x_3x_4/x_5 \tag{17}$$

and from which we are able to extract three n-grams, i.e., x_1, $x_2x_3x_4$, and x_5 accordingly.

Appendix B: N-gram Examples in the Lexicon $\mathcal{N}$

To explore the effect of each individual n-gram in the lexicon $\mathcal{N}$, we rank the n-grams according to their contributions to the constituency parsing task. In doing so, we firstly run our best performing models (BERT/XLNet encoders with predicted POS tags) with categorical span attentions (+ CATSA)

N-GRAMS	AVG. ATTENTION
said	0.0141
more than	0.0063
as well as	0.0167
from a year earlier	0.0267
in the past few years	0.0184

Table 5: Example n-grams with their average weights obtained from our best performing model (i.e., XLNet + POS + CATSA) on the entire PTB dataset.

for Arabic, Chinese, and English on the entire ATB, CTB5, and PTB datasets, respectively. Then, for each n-gram, we compute its average attention weights according to its appearance in the entire dataset. Afterwards, we group n-grams by their length and rank the n-grams according to their average attention weights within each group. The top 50 n-grams in each group as well as their attention weights for each language are reported in the supplemental material. As a demonstration, Table 5 shows a few n-grams with their average attention weights on the entire PTB dataset.