Incorporating Constituent Syntax for Coreference Resolution

Fan Jiang and Trevor Cohn

School of Computing and Information Systems
The University of Melbourne, Victoria, Australia
fan.jiang1@student.unimelb.edu.au, t.cohn@unimelb.edu.au

Abstract

Syntax has been shown to benefit Coreference Resolution from incorporating long-range dependencies and structured information captured by syntax trees, either in traditional statistical machine learning based systems or recently proposed neural models. However, most leading systems use only dependency trees. We argue that constituent trees also encode important information, such as explicit span-boundary signals captured by nested multi-word phrases, extra linguistic labels and hierarchical structures useful for detecting anaphora. In this work, we propose a simple yet effective graph-based method to incorporate constituent syntactic structures. Moreover, we also explore to utilise higher-order neighbourhood information to encode rich structures in constituent trees. A novel message propagation mechanism is therefore proposed to enable information flow among elements in syntax trees. Experiments on the English and Chinese portions of OntoNotes 5.0 benchmark show that our proposed model either beats a strong baseline or achieves new state-of-the-art performance.[1]

1 Introduction

As one of the most fundamental and important tasks in Natural Language Processing (NLP), Coreference Resolution aims to group all *mentions* that refer to the same real-world entity. This is framed as a span-level classification problem over sequences of words, as illustrated in Figure 1. Recently, neural models have achieved strong performance in coreference resolution (Lee, He, and Zettlemoyer 2018; Joshi et al. 2019, 2020; Wu et al. 2020) with the help of pretrained language models (Peters et al. 2018; Devlin et al. 2019).

Syntax was widely used in early learning-based methods, through the use of features derived from syntax trees, which were shown to significantly improve statistical coreference systems (Ge, Hale, and Charniak 1998; Bergsma and Lin 2006; Kong et al. 2010; Kong and Zhou 2011). However, it is not clear whether these syntax-related features can improve modern neural coreference models. Meanwhile, many research in other tasks have got positive effects by introducing syntactic information into neural models. They normally apply graph neural networks (Schlichtkrull et al. 2017; Veličković et al. 2018) to automatically incorporate dependency trees to better capture the non-local relationships between words (Marcheggiani and Titov 2017; Bastings et al. 2017; Schlichtkrull et al. 2017). A very recent work (Jiang and Cohn 2021) shows that incorporating dependency syntax together with semantic role features using heterogeneous graph networks can significantly improve a strong neural coreference model. However, few attempts have been made for the application of constituent syntax, especially on neural coreference models.

DA: It's because of what *both of you* are doing to have things change. I think that's what's… Go ahead Linda.
LW: Thanks goes to you and to the media to help *us*.
DA: Absolutely.
LW: Obviously *we* couldn't seem loud enough to bring the attention, so *our* hat is off to all of you as well.

Figure 1: An example of coreference resolution (Pradhan et al. 2012). Coreferent mentions are with the same colour.

In this work, we aim to investigate the use of constituent syntax in neural coreference models, and we argue that compared to dependency syntax, incorporating constituent syntax is more natural for coreference resolution. In constituent trees, the information encoding boundaries of non-terminal phrases is explicitly presented. By contrast, such information is either implicitly encoded or not revealed in dependency trees. Besides, gold mentions usually map to a limited range of constituent types (See §4.5). We thus believe constituent tree structures capture signals for mention detection more effectively and explicitly, which we suspect is more helpful to the overall coreference resolution task. Besides, extra linguistic labels are also shown to reveal linguistic constraints for coreference resolution (Ng 2010) (e.g., indefinite mentions are less likely to refer to any preceding mentions).

In order to effectively incorporate constituent syntax structures, we propose a graph-based neural coreference model. Previous works (Trieu et al. 2019; Kong and Jian 2019) introduced constituent trees as hard constraints to filter invalid mentions, which helps obtain better mention detectors and overall coreference resolvers. However, these methods do not preserve the full structure of original trees, as a result of their

[1]Code is available at https://github.com/Fantabulous-J/Coref-Constituent-Graph

simplification of the tree as node traversal sequences with a collection of path features, or by ignoring the hierarchical structure entirely. In contrast, our method builds a graph consisting of terminal and non-terminal nodes and applies graph neural networks to encode such structures more flexibly.

Our model consists of three parts: document encoder, graph encoder and coreference layer. We use pretrained language model such as SpanBERT (Joshi et al. 2020) as document encoder. We propose a novel graph encoder by leveraging graph attention networks (Veličković et al. 2018), which encodes bidirectional graphs separately. Moreover, in order to capture higher-order neighbourhood information and reduce the over-smoothing problem (Chen et al. 2020) of graph neural networks by stacking too many layers, we additionally add two-hop edges based on original constituent tree structures. New message propagation mechanisms over the underlying extended graph are therefore designed, where constituent nodes are updated iteratively using bidirectional graph attention networks, and explicit hierarchical syntax and span boundary information are propagated to enhance the contextualized token embeddings. We conduct experiments on the English and Chinese portions of OntoNotes 5.0 benchmark, and show that our proposed model significantly outperforms a strong baseline and achieves new state-of-the-art performance on the Chinese dataset.

2 Baseline Model

Our model is based on the C2F-COREF+*SpanBERT* model (Joshi et al. 2019), which improves over Lee, He, and Zettlemoyer (2018) with the document encoder replaced with SpanBERT model. It exhaustively enumerates all text spans up to a certain length limit as candidate mentions and aggressively prunes spans with low confidence. For each mention i, the model will learn a distribution over its possible antecedents $\mathcal{Y}(i)$:

$$P(y) = \frac{e^{s(i,y)}}{\sum_{y' \in \mathcal{Y}(i)} e^{s(i,y')}} \quad (1)$$

where the scoring function $s(i, j)$ measures how likely span i and j comprise valid mentions and corefer to one another:

$$s(i,j) = s_m(i) + s_m(j) + s_c(i,j) \quad (2)$$

$$s_m(i) = \mathbf{FFNN}_m(\mathbf{g_i}) \quad (3)$$

$$s_c(i,j) = \mathbf{FFNN}_c(\mathbf{g_i}, \mathbf{g_j}, \mathbf{g_i} \odot \mathbf{g_j}, \phi(i,j)) \quad (4)$$

where $\mathbf{g_i}$ and $\mathbf{g_j}$ are span representations formed by the concatenation of contextualized embeddings of span endpoints, head vector using attention mechanism and span width embeddings. **FFNN** represents a two-layers feedforward neural network with **ReLU** activation function inside. $\mathbf{g_i} \odot \mathbf{g_j}$ is the element-wise similarity between span i and j. $\phi(i, j)$ are meta features including bucket span distances, genre information and binary speaker features. s_m means the predicted mention score which will be used to prune unlikely candidate mentions, while s_c is the final pairwise coreference score. As suggested by Xu and Choi (2020), we further discard the higher-order based span refinement module, which uses antecedent distribution $P(y)$ as the attention mechanism to obtain the weighted average sum of antecedent embeddings, when replicating the baseline model.

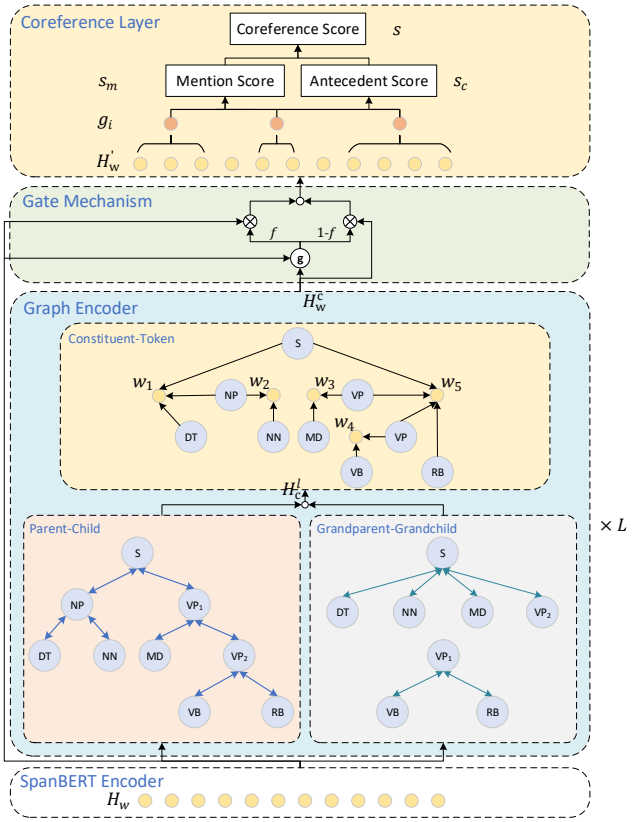

Figure 2: The architecture of our proposed model.

3 Proposed Model

3.1 Document Encoder

In order to fit long documents, Joshi et al. (2019) chooses to split documents into independent segments. But the drawback is that it has limited modelling capacity as tokens can only attend to other tokens within the same segment, especially for tokens at the boundary of each segment (Joshi et al. 2019). Instead, we choose to create overlapped segments and treat speakers (speaker's name of each utterance) as part of the input (Wu et al. 2020). Specifically, we use a sliding-window approach to create T-sized segments with a stride of $\frac{T}{2}$ tokens and insert speakers into the beginning of their belonged utterances. Overlapped segments with attached speaker information are then encoded by SpanBERT (Joshi et al. 2020) to obtain contextualized representations, which can be denoted as $\mathbf{H}_w = (\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n)$, where $\mathbf{h}_i \in \mathcal{R}^d$ and n is the document length.

3.2 Graph Construction

For each sentence in the document, we have an associated constituent tree which consists of words (terminals) and constituents (non-terminals). Therefore, we have two types of nodes in our graph: token nodes ($W = \{w_1, w_2, \ldots, w_n\}$) and constituent nodes ($C = \{c_1, c_2, \ldots, c_m\}$), where n and m are the number of token and constituent nodes, respectively. For a given constituent node c_i, we use START(c_i)

and END(c_i) to denotes its start and end token indices.

Node Initialization Token node representations are initialized using the contextualized token representations $\mathbf{H}_w$ from the document encoder. Span boundary information has been shown to be extremely important to span-based tasks. Therefore, a novel span yield enhanced method is proposed to initialize each constituent node $c_i \in C$:

$$\mathbf{h}_{c_i} = [\mathbf{h}_{\text{START}(c_i)}; \mathbf{h}_{\text{END}(c_i)}; \mathbf{e}_{\text{type}}(c_i)] \quad (5)$$

where $\mathbf{h}_{\text{START}(c_i)}$ and $\mathbf{h}_{\text{END}(c_i)}$ are the embeddings of start and end tokens of constituent c_i. $\mathbf{e}_{\text{type}}(c_i)$ is the constituent type embeddings obtained from a randomly initialized look-up table, which will be optimized during the training process. Thus, we can obtain a set of initialized constituent node representations: $\mathbf{H}_c = \{\mathbf{h}_{c_1}, \mathbf{h}_{c_2}, \ldots, \mathbf{h}_{c_m}\}$.

Edge Construction

Constituent-Constituent We design two categories of edges in our graph, namely *parent-child* and *grandparent-grandchild*, to capture longer-range dependencies. For each edge category, we further add reciprocal edges for each edge in the graph and label them with *forward* and *backward* types, respectively. Additionally, self-loop edges are added to each node in the graph. Thus, the edges are constructed based on following rules:

(1). A pair of **parent-child** and **child-parent** edges between node c_i and c_j are constructed if they are directly connected in the constituent tree.

(2). A pair of **grandparent-grandchild** and **grandchild-grandparent** edges between node c_i and c_j are constructed if node c_i can reach node c_j using two hops, and vice versa.

Constituent-Token A token node w_i is linked to c_j if it is the left or rightmost token in the yield of c_j. Such edges are made unidirectional to make sure that information can only be propagated from constituent nodes to token nodes, which aims to enrich basic token representations with span boundary information and the hierarchical syntax structures.

3.3 Graph Encoder

We use a Graph Attention Network (Veličković et al. 2018) to update the representation of constituent nodes and propagate syntactic information to basic token nodes. For a node i, the attention mechanism allows it to selectively incorporate information from its neighbour nodes:

$$\alpha_{ij} = \text{softmax}(\sigma(\mathbf{a}^T[\mathbf{W}\mathbf{h}_i; \mathbf{W}\mathbf{h}_j])) \quad (6)$$

$$\mathbf{h}'_i = \|_{k=1}^{K} \text{ReLU}(\sum_j \alpha_{ij}^k \mathbf{W}^k \mathbf{h}_j) \quad (7)$$

where K is the number of heads, $\mathbf{h}_i$ and $\mathbf{h}_j$ are embeddings of node i and j, $\mathbf{a}^T$, $\mathbf{W}$ and $\mathbf{W}^k$ are trainable parameters. σ is the LeakyReLU activation function (Xu et al. 2015). $\|$ and $[;]$ means concatenation. Eqs. 6 and 7 are designated as an operation $\mathbf{h}'_i = \text{GAT}(\mathbf{h}_i, \mathbf{h}_j | j \in \mathcal{N}_i)$, where $\mathcal{N}_i$ is the set of target node i's neighbours, $\mathbf{h}_i$ and $\mathbf{h}_j$ are the embeddings of target and neighbour node. $\mathbf{h}'_i$ is the updated embedding of target node.

Bidirectional GAT Layer We design a bidirectional GAT layer to model the constituent-constituent edges. Specifically, for a given constituent node c_i, we obtain its neighbour nodes with edge type t in *forward* (outgoing) and *backward* (incoming) directions: $\mathcal{N}_{c_i}^{tf}$ and $\mathcal{N}_{c_i}^{tb}$, respectively. Then we use two separate GAT encoders to derive the updated representation of node c_i in different directions:

$$\mathbf{h}_{c_i}^{tf} = \text{GAT}(\mathbf{h}_{c_i}, \mathbf{h}_{c_j} | c_j \in \mathcal{N}_{c_i}^{tf}) \quad (8)$$

$$\mathbf{h}_{c_i}^{tb} = \text{GAT}(\mathbf{h}_{c_i}, \mathbf{h}_{c_j} | c_j \in \mathcal{N}_{c_i}^{tb}) \quad (9)$$

Then the updated representation of constituent node c_i is obtained by the summation of the representations of two directions: $\mathbf{h}_{c_i}^{t} = \mathbf{h}_{c_i}^{tf} + \mathbf{h}_{c_i}^{tb}$.

Multi-type Integration Layer In order to aggregate updated node representations using different types of edges, we use the self-attentive mechanism (Lee et al. 2017):

$$\alpha_{c_i,t} = \text{softmax}(\textbf{FFNN}(\mathbf{h}_{c_i}^{t})) \quad (10)$$

$$\mathbf{h}'_{c_i} = \sum_{t=1}^{T} \alpha_{c_i,t} \mathbf{h}_{c_i}^{t} \quad (11)$$

where T is the number of edge types and **FFNN** is a two-layer feedforward neural network with **ReLU** function inside. An operation is designated to summarise Eqs. 8 to 11: $\mathbf{h}'_{c_i} = \text{Multi-BiGAT}(\mathbf{h}_{c_i}, \mathbf{h}_{c_j} | c_j \in \mathcal{N}_{c_i})$.

3.4 Message Propagation

The message propagation mechanism is defined to enable information flow from constituent nodes to basic token nodes. First, we update the constituent node representation using our defined bidirectional GAT layer with multi-type edges:

$$\mathbf{h}_{c_i}^{l+1} = \text{Multi-BiGAT}(\mathbf{h}_{c_i}^{l}, \mathbf{h}_{c_j}^{l} | c_j \in \mathcal{N}_{c_i}) \quad (12)$$

where $\mathbf{h}_{c_i}^{l}$ is the constituent node representation from previous layer l and $\mathbf{h}_{c_i}^{0}$ is the initial constituent node embedding.

Then the updated constituent node representations propagate information to update the token node embeddings through constituent-token edges:

$$\mathbf{h}_i^{l+1} = \text{GAT}(\mathbf{h}_i^{l}, \mathbf{h}_{c_j}^{l+1} | c_j \in \mathcal{N}_i) \quad (13)$$

where $\mathbf{h}_i^{l}$ is the token representation from layer l and $\mathbf{h}_i^{0}$ is the encoding from document encoder.

The updated token representation is then used to reconstruct the updated constituent node embeddings using Eq. 5, which will be employed in the next graph encoder layer. After L iterations, we could obtain the final constituent syntax enhanced token representations, which are denoted as $\mathbf{H}_w^c$.

Finally, we use a gating mechanism to infuse the syntax-enhanced token representation dynamically:

$$\mathbf{f} = \sigma(\mathbf{W}_g \cdot [\mathbf{H}_w; \mathbf{H}_w^c] + \mathbf{b}_g) \quad (14)$$

$$\mathbf{H}'_w = \mathbf{f} \odot \mathbf{H}_w + (\mathbf{1} - \mathbf{f}) \odot \mathbf{H}_w^c \quad (15)$$

where $\mathbf{W}_g$ and $\mathbf{b}_g$ are trainable parameters, $\odot$ and σ are element-wise multiplication and sigmoid function respectively.

The enhanced token representations $\mathbf{H}'_w$ will be used to form span embeddings and compute coreference scores (Lee, He, and Zettlemoyer 2018) (See §2).

		MUC			B^3			$CEAF_{\phi_4}$			
		P	R	F1	P	R	F1	P	R	F1	Avg. F1
English	E2E-COREF (Lee et al. 2017)	78.4	73.4	75.8	68.6	61.8	65.0	62.7	59.0	60.8	67.2
	C2F-COREF (Lee, He, and Zettlemoyer 2018)	81.4	79.5	80.4	72.2	69.5	70.8	68.2	67.1	67.6	73.0
	SpanBERT-base (Joshi et al. 2020)	84.3	83.1	83.7	76.2	75.3	75.8	74.6	71.2	72.9	77.4
	Our baseline + SpanBERT-base*†	83.9	84.2	84.0	76.2	76.9	76.6	74.3	73.1	73.7	78.1 ($\pm$0.1)
	Jiang and Cohn (2021) + SpanBERT-base†‡§	85.3	85.0	85.2	77.9	77.7	77.8	75.6	74.1	74.8	79.3 ($\pm$0.2)
	Our model + SpanBERT-base†§	**85.6**	85.8	85.7	78.2	**79.0**	**78.6**	**76.3**	74.8	75.5	**80.0** ($\pm$0.2)
	CorefQA (Wu et al. 2020) + SpanBERT-base¶	85.2	**87.4**	**86.3**	**78.7**	76.5	77.6	76.0	**75.6**	**75.8**	79.9
	SpanBERT-large (Joshi et al. 2020)	85.8	84.8	85.3	78.3	77.9	78.1	76.4	74.2	75.3	79.6
	Our baseline + SpanBERT-large*†	86.0	86.0	86.0	79.6	79.6	79.6	77.2	75.8	76.5	80.7 ($\pm$0.1)
	Jiang and Cohn (2021) + SpanBERT-large†‡§	87.2	86.7	87.0	81.1	80.5	80.8	78.6	77.0	77.8	81.8 ($\pm$0.2)
	Our model + SpanBERT-large†§	87.3	87.1	87.2	81.1	80.9	81.0	78.8	77.2	78.0	82.1 ($\pm$0.2)
	CorefQA (Wu et al. 2020) + SpanBERT-large¶	**88.6**	**87.4**	**88.0**	**82.4**	**82.0**	**82.2**	**79.9**	**78.3**	**79.1**	**83.1**
Chinese	Clark and Manning (2016)	73.9	65.4	69.4	67.5	56.4	61.5	62.8	57.6	60.1	63.7
	Kong and Jian (2019)§	77.0	64.6	70.2	70.6	54.7	61.6	64.9	55.4	59.8	63.9
	Our baseline + BERT-wwm-base*†	76.7	70.9	73.7	68.3	62.4	65.2	67.4	60.8	63.9	67.6 ($\pm$0.3)
	Our model + BERT-wwm-base†§	**84.1**	**78.6**	**81.3**	**77.4**	**71.5**	**74.4**	**76.5**	**70.0**	**73.1**	**76.3** ($\pm$0.2)
	Our baseline + RoBERTa-wwm-ext-large*†	79.9	72.2	75.8	71.6	64.3	67.7	70.8	62.8	66.5	70.0 ($\pm$0.3)
	Our model + RoBERTa-wwm-ext-large†§	**85.8**	**80.9**	**83.3**	**79.8**	**74.5**	**77.0**	**78.7**	**72.9**	**75.7**	**78.7** ($\pm$0.2)

Table 1: The results on the test set of the OntoNotes English and Chinese shared task compared with previous systems when using gold constituent trees. $*$ indicates our replicated baseline. § indicates methods using gold features. ¶ indicates methods using substantial training resources and extra datasets for pretraining. † means averaged performance over 5 runs. ‡ means results obtained by running the publicly released code of Jiang and Cohn (2021) with the document encoding method in Section 3.1.

4 Experiments

4.1 Dataset

Our model is evaluated on the English and Chinese portions of OntoNotes 5.0 dataset (Pradhan et al. 2012). The English corpus consists of 2802, 343 and 348 documents in the training, development and test splits, respectively, while the Chinese corpus contains 1810, 252 and 218 documents for train/dev/test splits. The model is evaluated using three coreference metrics: MUC, B^3 and CEAFϕ_4 and the average F1 score (Avg. F1) of the three are reported. We use the latest version of the official evaluation scripts (version 8.01),[2] which implements the original definitions of the metrics.

4.2 Experimental Settings

We reimplement the C2F-COREF+*SpanBERT*[3] baseline using PyTorch. For English model, we use SpanBERT-base and large model to encode documents;[4] while for Chinese, we use BERT-wwm-base and RoBERTa-wwm-ext-large[5] as the document encoders. Graph attention networks and the message propagation module are implemented based on Deep Graph Library (Wang et al. 2019). Gold constituent trees annotated on the datasets are used in this experiment for consistent comparison with previous work.

Most hyperparameters are adopted from Joshi et al. (2019) and newly introduced hyperparameters are determined through grid search. The learning rates of finetuning base and large model are 2×10^{-5} and 1×10^{-5}. The learning rates of task-specific parameters are 3×10^{-4} and 5×10^{-4} for English, and 5×10^{-4} for Chinese when using base and large model, respectively. Both BERT and task parameters are trained using Adam optimizer (Kingma and Ba 2015), with a warmup learning scheduler for the first 10% of training steps and linear decay scheduler decreasing to 0, respectively. The number of heads is set to 4 and 8 for base and large models. The size of constituent type embeddings is 300. We set the number of graph attention layers as 2. For all experiments, we choose the best model according to Avg. F1 on dev set, which is then evaluated on the test set.

4.3 Baselines and State-of-the-Art

We compare our proposed model with a variety of previous competitive models: Clark and Manning (2016) is a neural network based model which incorporates entity-level information. E2E-COREF (Lee et al. 2017) is the first end-to-end neural model for coreference resolution which jointly detects and groups entity mention spans. Kong and Jian (2019) improves the E2E-COREF model by treating constituent trees as constraints to filter invalid candidate mentions and encoding the node traversal sequences of parse trees to enhance document representations. C2F-COREF (Lee, He, and Zettlemoyer 2018) extends the E2E-COREF model by introducing a *coarse-to-fine* candidate mention pruning strategy and a higher-order span refinement mechanism. Joshi et al. (2020) improves over C2F-COREF with the document encoder replaced by SpanBERT. CorefQA (Wu et al. 2020) employs the machine

[2] http://conll.cemantix.org/2012/software.html

[3] https://github.com/mandarjoshi90/coref

[4] https://github.com/facebookresearch/SpanBERT

[5] https://github.com/ymcui/Chinese-BERT-wwm

reading comprehension framework to recast the coreference resolution problem as a query-based span-prediction task, which achieves current state-of-the-art performance. Jiang and Cohn (2021) enhances neural coreference resolution by incorporating dependency syntax and semantic role labels using heterogeneous graph attention networks.

4.4 Overall Results

Table 1 shows the results of our model compared with a range of high-performing neural coreference resolution models on English and Chinese. For English, we observe that our replicated baseline surpasses the SpanBERT baseline (Joshi et al. 2020) by 0.7% and 1.1%, demonstrating the effectiveness of the sliding-window based document encoding approach and modified representations of speaker identities. Our model further improves the replicated baseline significantly with improvements of 1.9% and 1.4%, respectively, a result which is also comparable to the state-of-the-art performance of CorefQA (Wu et al. 2020).[6] Improvements can also be observed over Jiang and Cohn (2021) (0.7% with $p < 0.002$ and 0.3% with $p < 0.06$).[7] For Chinese, our replicated baseline has already achieved state-of-the-art performance. With the help of constituent syntax, our model again beats the baseline model with significant improvements of 8.7%. This indicates that constituent syntax is far more useful to Chinese than English, and we suspect that word-level segmentation encoded in constituent trees brings extra benefits in Chinese.

4.5 Analysis

Effects of Constituency Quality To evaluate how the quality of constituent trees affects the performance, we test two off-the-shelf parsers (Zhang, Zhou, and Li 2020) (achieving 95.26% and 91.40% F1 score on PTB and CTB7) to obtain predicted trees. When using predicted trees with our base model, we get Avg. F1 of 78.7% (+0.6% with $p < 0.05$) and 73.0% (+5.4%) on both languages, consistently outperforming the baseline. Similarly, the effects in large models are also noticeable, resulting in Avg. F1 of 81.0% (+0.3% with $p < 0.05$) and 75.0% (+5.0%) respectively. However, the performance is still worse than using gold trees, indicating the necessity of high-quality constituency parsers.

Ablation Study We modify several components of our model to validate their effects. Results are reported in Table 2 for the following ablations: 1) Using vanilla constituent trees by only keeping parent-child edges; 2) Removing the gating mechanism and directly use representations from the graph encoder; 3) Changing the way of representing constituent node to initialize only with type embeddings; 4) Using the *independent* setting (Joshi et al. 2019) for document encoding; 5) Using the similar way as in this paper to incorporate dependency trees; 6) Using the same method to encode constituent and dependency syntax as in this paper alongside the method in Jiang and Cohn (2021) for attention fusion of the dependency-syntax and constituent-syntax representations.

From Table 2 we observe that: 1) The bidirectional graph and higher-order edges show positive impacts in capturing long-range dependencies; 2) Removing the gate mechanism leads to significant performance degradation, especially in English. We believe that the gate mechanism plays an important role in dynamically choosing useful information from original sequential representations and graph-enhanced representations, and keeping information such as position embeddings from being lost after the graph attention network; 3) Although only using type embeddings to initialize constituent node representations also yields competitive performance, our span yield enhanced initialization method can capture span-boundary information more effectively; 4) Splitting documents into independent segments is less beneficial, especially for tokens at the boundary of their segment; 5) Incorporating dependency syntax achieves inferior performance, showing that explicit span-boundary information encoded in constituent trees is more beneficial; 6) Combining both types of syntax is better than using dependency syntax solely but is inferior to only using constituent syntax.

Mentions With Different Lengths Table 3 shows the performance comparison in terms of different mention lengths on both datasets. As shown in the table, we can observe that our proposed model consistently outperforms the baseline model for both two languages. This indicates that the improved overall performance in the coreference resolution task has benefited largely from better mention detectors, which is consistent with our hypothesis. The performance gain is more significant for mentions with longer length on both languages, demonstrating that leveraging constituent syntax is highly effective for modelling long-range dependencies and becomes more crucial when entity length becomes longer.

Variants	English		Chinese	
	Avg. F1	$\Delta F1$	Avg. F1	$\Delta F1$
-	80.0	-	76.3	-
Vanilla Tree	79.7	-0.3	75.9	-0.4
No Gate	77.3	-2.7	75.1	-1.2
Only Type Embedding	79.5	-0.5	75.5	-0.8
No Sliding Window	79.6	-0.4	76.0	-0.3
Dependency Syntax	79.2	-0.8	70.8	-5.5
Constituent & Dependency	79.7	-0.3	75.3	-1.0

Table 2: Results when ablating different modules compared to our base model on English and Chinese datasets.

Dataset	Model	Mention Length					Overall
		1-2	3-4	5-7	8-10	11+	
English	baseline	90.3	82.8	78.3	75.1	65.9	87.0
	our method	**90.5**	**84.3**	**80.1**	**79.2**	**74.0**	**88.1**
Chinese	baseline	85.1	79.0	71.0	69.1	66.4	80.1
	our method	**88.5**	**85.0**	**78.3**	**80.0**	**73.6**	**85.1**

Table 3: The F1 score based on mention length on English and Chinese development sets when using base model.

[6] We do not use their model as a baseline mainly due to hardware limitations, as it requires 128G GPU memory for training. It can also be easily incorporated with our method by adding the proposed graph encoder on top of their document encoder with minor modification, which we expect would lead to further improvements.

[7] Pitman's permutation statistical test (Dror et al. 2018).

Doc length	#Docs	Baseline	Ours	$+\Delta F1$
0 – 128	57	82.8	86.1	+3.3
129 – 256	73	81.3	83.2	+1.9
257 – 512	78	82.0	83.7	+1.7
513 – 768	71	78.3	79.0	+0.7
769 – 1152	52	77.5	78.9	+1.4
1153+	12	68.0	70.7	+2.7
All	343	78.2	79.7	+1.5

Table 4: The Avg. F1 on the English dev set when using base model, broken down by document length.

Dataset	Methods	Avg. F1	$\Delta F1$
English	Baseline	78.1	-
	Our Method	80.0	+1.9
	Baseline + Mention Filter	77.3	-0.8
Chinese	Baseline	67.6	-
	Our Method	76.3	+8.7
	Baseline + Mention Filter	71.8	+4.2

Table 5: Results when utilising syntactic parse trees as mention filter compared to the baseline and our base model.

Document Length In Table 4, we show that the performance of our model against the baseline on the English development set as a function of document lengths. As expected, our model consistently outperforms the baseline model on all document sizes, especially for documents with length larger than 1153 tokens. This demonstrates that the incorporated constituent syntax and our modelling choices are beneficial for capturing longer-range dependencies. Besides, the improvements on short documents (<128 tokens) are also significant. We find that most anaphoric mentions have very short distances between their nearest antecedents. The Binding Theory (Chomsky 1988) argues that constituent syntax is more effective in keeping anaphoric mentions locally bounded by short-distance antecedents. Thus, it is possible that our model implicitly learns this principle, which results in better performance. Nevertheless, our model shows similar pattern as the baseline model, performing distinctly worse as document length increases. This indicates that the sentence-level syntax used in this work are not sufficient enough to tackle the deficiency of modelling long-range dependency. One possible solution is to incorporate document-level features such as hierarchical discourse structures.

Constituent Tree as Mention Filter An alternative use of syntax is through constraining mention types. We use the constituent parse tree as hard constraints on top of the baseline to filter out invalid candidate mentions, assuming that only candidate mentions that have matched phrases in the parse tree are valid. We observe that about 99% of gold mentions correspond to a small set of syntactic phrases and POS types.[8] We thus use these two phrase sets as filters to prune unlikely candidate mentions. Table 5 shows the corresponding results. We can find that the syntactic constraint harms the performance slightly on the English baseline (-0.8%) but improves the Chinese baseline by 4.2%. However, in both cases this constrained baseline is substantially worse than using the syntax tree as part of our neural model, as proposed in this paper (with scores of 2.7% and 4.5% lower for English and Chinese, respectively).

[8]en: the set of phrases tagged with NP, NML, PRP, PRP$, WP, WDT, WRB, NNP, VB, VBD, VBN, VBG, VBZ, VBP (Wu and Gardner 2020) includes 99.63% gold mentions. zh: the set of VV, NT, PN, DFL, NR, NP, QP, NN covers 99.79% gold mentions.

4.6 Resolution Classes

To further understand the behaviour of our proposed model, we follow Stoyanov et al. (2009) and Lu and Ng (2020) to classify gold mentions into different resolution classes and compare it with the baseline on each of them.

Proper Names Gold mentions associated with named entity types belong to this class, and four sub-classes are defined accordingly. 1) *exact string match* (*e*): at least one preceding mention in a proper name's gold cluster exactly has the same string; 2) *partial string match* (*p*): at least one preceding mention in a proper name's gold cluster shares some words; 3) *no string match* (*n*): no preceding mention in a proper name's gold cluster shares some words; 4) *non-anaphoric* (*na*): a proper name does not refer to any preceding mention.

Common NPs Gold mentions without named entity types belong to this class, with four sub-classes as in *proper names*.

Pronouns Five pronoun sub-classes are defined. 1) *1/2*: 1st and 2nd person pronouns (e.g., you); 2) *G3*: gendered 3rd person pronouns (e.g., she); 3) *U3*: ungendered 3rd person pronouns (e.g., they); 4) *oa*: any anaphoric pronouns not in 1), 2), and 3) (e.g., demonstrative pronouns); 5) *na*: non-anaphoric pronouns (e.g., pleonastic pronouns).

Results For performance measurements, we follow Lu and Ng (2020) to use mention detection recall (MD) and resolution accuracy (RA). For MD, we count the percentage of gold mentions that are correctly detected in each resolution class; while for RA, we compute the percentage of correctly detected mentions that are correctly resolved.

Table 6 shows the performance of the baseline and our proposed model on each resolution class. Firstly, we can see that both models perform the best on proper names, followed by common nouns and pronouns. Secondly, by analysing the fine-grained classes, the *exact match* class in proper names and common nouns are easier than the *partial match* one, which is easier than the *no string match* class. For pronouns, the 3rd person gendered pronoun is the easiest one, followed by the 1st/2nd person noun, while both models find it difficult to resolve other pronouns such as reflective pronouns. Thirdly, we find that our model gains significant improvements on non-anaphoric mentions, showing its superiority in dealing with the difficulty of anaphoricity determination, with improvements up to 2.7% and 8.0% RA in English and Chinese, respectively. Moreover, considerable improvements on 3rd ungendered pronouns (2.0% and 1.4%) are also observed. Constituent syntax is also especially helpful in detecting partial and no string match classes for proper names (8.7% and 8.3%) and common NPs (6.2% and 3.4%) in Chinese. These demonstrate that the harder a resolution class is, the more

	OntoNotes 5.0 English					OntoNotes 5.0 Chinese				
		Baseline		Ours			Baseline		Ours	
Class	Size %	RA	MD	RA	MD	Size %	RA	MD	RA	MD
PN-e	15.52	96.4	94.0	**96.5**	**94.3**	15.05	**97.5**	93.5	97.3	95.9
PN-p	6.06	90.0	86.3	**92.6**	**89.1**	2.81	76.0	84.7	**84.7**	**88.2**
PN-n	6.63	85.7	89.1	**86.4**	**89.5**	0.47	35.9	59.1	**44.2**	**65.2**
PN-na	6.74	95.5	83.0	**95.9**	**85.8**	6.05	85.1	78.4	**85.7**	**85.4**
CN-e	6.17	96.7	90.8	**96.9**	**91.9**	16.01	91.5	80.1	**93.5**	**85.8**
CN-p	8.60	83.8	80.5	**85.9**	**82.3**	11.99	70.0	65.1	**76.2**	**71.7**
CN-n	3.39	72.4	68.6	**73.6**	**70.7**	4.55	62.8	68.7	**66.2**	**73.2**
CN-na	15.47	91.8	69.7	**92.6**	**72.2**	26.88	**67.1**	65.2	65.9	**74.3**
PR-1/2	11.64	93.8	**95.1**	**93.9**	94.9	8.12	85.7	**97.5**	**88.0**	95.7
PR-G3	5.99	96.5	99.6	**97.0**	**99.7**	4.91	87.9	**99.4**	**89.0**	99.3
PR-UG3	10.14	86.8	**95.2**	**88.8**	94.8	1.02	72.5	91.0	**73.9**	**95.8**
PR-oa	1.45	63.0	**68.2**	**67.8**	63.9	0.83	**70.7**	**63.6**	70.2	48.3
PR-na	2.20	54.7	**85.3**	**57.4**	85.1	1.33	62.4	**87.8**	**70.4**	84.6

Table 6: The results of resolution classes in the development set of OntoNotes English and Chinese dataset.

significant our model's improvement is. Besides, this also shows that the incorporated constituent syntax help resolve traditionally difficult anaphors. Overall, by maintaining comparable performance in other easier classes simultaneously, our model has achieved significantly better final results on these two languages compared with the baseline.

5 Related Work

Syntax for Coreference Resolution Syntactic features derived from syntactic parse trees were dominant in early research for coreference resolution. Ge, Hale, and Charniak (1998) proposes Hobbs distances to encode the rank of candidate antecedents of a given pronoun based on Hobbs's syntax parse tree based pronoun resolution algorithm (Hobbs 1978). Bergsma and Lin (2006) implements path-related features based on syntactic parse trees, where the sequence of words and dependency labels in the path between a given pronoun and its candidate antecedent is utilised. Statistical information collected from such paths is used to measuring the likelihood of being coreferent for the pronoun and antecedent. Syntactic information has also been applied in the anaphoricity determination task by using tree-kernel-based methods: Kong et al. (2010) and Kong and Zhou (2011) design various kinds of path-related features such as root path between the root node and current mention. By contrast, few attempts have been made to evaluate the utility of syntax for neural coreference models. Trieu et al. (2019) and Kong and Jian (2019) treated constituent trees as signals to filter invalid candidate mentions for coreference resolution. Nevertheless, their methods either ignore the hierarchical structures of constituent trees or fail to well preserve tree structures by encoding constituent trees using node traversal sequences. To fill this gap, we propose a novel graph-based method to fully model constituent tree structures more flexibly.

Enhancing Neural Models with Syntax External syntax has long been used for enhancing neural models. Mainstream methods typically use graph neural networks to capture the structural information encoded in dependency trees. Marcheggiani and Titov (2017) and Bastings et al. (2017) applied Graph Convolutional Networks (GCNs) (Schlichtkrull et al. 2017) to incorporate dependency trees to capture the non-local relationships between words. Wang et al. (2020) employed reshaped dependency trees using relational graph attention networks to effectively capture long-range dependencies while ignoring noisy relations.

Compared to dependency syntax, the utility of constituent syntax in neural models is less well studied. Some early work utilised recursive neural networks to incorporate constituent trees by recursively updating the representation of constituent phrases (Socher et al. 2013; Tai, Socher, and Manning 2015). Nevertheless, such a method is less efficient than applying graph neural networks since the recursive way of encoding means that later steps should depend on earlier ones. The most similar method to our own is Marcheggiani and Titov (2020), who developed a neural model of semantic role labelling based on GCN encoding of a constituency tree with a message propagation mechanism. Nevertheless, our method differs in extending plain parse trees with higher-order edges and bidirectional graphs to capture longer-range neighbourhood information. We also proposes a novel span yield enhanced method to represent constituent nodes instead of initializing them with zero vectors, which is better suited to conference resolution and similar to the way of representing mention spans. Our work also differs in terms of the task: we consider coreference resolution rather than SRL, a document-level task requiring modelling inter-sentence phenomena.

6 Conclusion

In this paper, we successfully incorporated constituent trees with added higher-order edges and bidirectional graphs, which are encoded via our designed bidirectional graph attention networks and message propagation mechanism. Emperical results on a English and Chinese benchmark confirm the superiority of our proposed method, significantly beating a strong baseline and achieving state-of-the-art performance.

Acknowledgments

We thank the anonymous reviewers for their helpful feedback. This research was undertaken using the LIEF HPC-GPGPU Facility hosted at the University of Melbourne. This Facility was established with the assistance of LIEF Grant LE170100200.

References

Bastings, J.; Titov, I.; Aziz, W.; Marcheggiani, D.; and Sima'an, K. 2017. Graph Convolutional Encoders for Syntax-aware Neural Machine Translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 1957–1967. Copenhagen, Denmark: Association for Computational Linguistics.

Bergsma, S.; and Lin, D. 2006. Bootstrapping Path-Based Pronoun Resolution. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, 33–40. Sydney, Australia: Association for Computational Linguistics.

Chen, D.; Lin, Y.; Li, W.; Li, P.; Zhou, J.; and Sun, X. 2020. Measuring and Relieving the Over-Smoothing Problem for Graph Neural Networks from the Topological View. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04): 3438–3445.

Chomsky, N. 1988. *Language and Problems of Knowledge: The Managua Lectures*. Cambridge, MA: MIT Press.

Clark, K.; and Manning, C. D. 2016. Improving Coreference Resolution by Learning Entity-Level Distributed Representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 643–653. Berlin, Germany: Association for Computational Linguistics.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 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)*, 4171–4186.

Dror, R.; Baumer, G.; Shlomov, S.; and Reichart, R. 2018. The Hitchhiker's Guide to Testing Statistical Significance in Natural Language Processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1383–1392. Melbourne, Australia: Association for Computational Linguistics.

Ge, N.; Hale, J.; and Charniak, E. 1998. A Statistical Approach to Anaphora Resolution. In *In Proceedings of the Sixth Workshop on Very Large Corpora*, 161–170.

Hobbs, J. 1978. Resolving pronoun references. *Lingua 44*, 311–338.

Jiang, F.; and Cohn, T. 2021. Incorporating Syntax and Semantics in Coreference Resolution with Heterogeneous Graph Attention Network. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1584–1591. Online: Association for Computational Linguistics.

Joshi, M.; Chen, D.; Liu, Y.; Weld, D. S.; Zettlemoyer, L.; and Levy, O. 2020. SpanBERT: Improving Pre-training by Representing and Predicting Spans. *Transactions of the Association for Computational Linguistics*, 8: 64–77.

Joshi, M.; Levy, O.; Zettlemoyer, L.; and Weld, D. 2019. BERT for Coreference Resolution: Baselines and Analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 5803–5808.

Kingma, D. P.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In Bengio, Y.; and LeCun, Y., eds., *3rd International Conference on Learning Representations, ICLR 2015*.

Kong, F.; and Jian, F. 2019. Incorporating Structural Information for Better Coreference Resolution. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, 5039–5045. International Joint Conferences on Artificial Intelligence Organization.

Kong, F.; and Zhou, G. 2011. Combining Dependency and Constituent-based Syntactic Information for Anaphoricity Determination in Coreference Resolution. In *Proceedings of the 25th Pacific Asia Conference on Language, Information and Computation*, 410–419. Singapore: Institute of Digital Enhancement of Cognitive Processing, Waseda University.

Kong, F.; Zhou, G.; Qian, L.; and Zhu, Q. 2010. Dependency-driven Anaphoricity Determination for Coreference Resolution. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, 599–607. Beijing, China: Coling 2010 Organizing Committee.

Lee, K.; He, L.; Lewis, M.; and Zettlemoyer, L. 2017. End-to-end Neural Coreference Resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 188–197. Association for Computational Linguistics.

Lee, K.; He, L.; and Zettlemoyer, L. 2018. Higher-Order Coreference Resolution with Coarse-to-Fine Inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, 687–692.

Lu, J.; and Ng, V. 2020. Conundrums in Entity Coreference Resolution: Making Sense of the State of the Art. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 6620–6631. Online: Association for Computational Linguistics.

Marcheggiani, D.; and Titov, I. 2017. Encoding Sentences with Graph Convolutional Networks for Semantic Role Labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 1506–1515. Copenhagen, Denmark: Association for Computational Linguistics.

Marcheggiani, D.; and Titov, I. 2020. Graph Convolutions over Constituent Trees for Syntax-Aware Semantic Role Labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 3915–3928. Online: Association for Computational Linguistics.

Ng, V. 2010. Supervised Noun Phrase Coreference Research: The First Fifteen Years. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, 1396–1411. Uppsala, Sweden: Association for Computational Linguistics.

Peters, M.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; and Zettlemoyer, L. 2018. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2227–2237.

Pradhan, S.; Moschitti, A.; Xue, N.; Uryupina, O.; and Zhang, Y. 2012. CoNLL-2012 Shared Task: Modeling Multilingual Unrestricted Coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, 1–40.

Schlichtkrull, M.; Kipf, T. N.; Bloem, P.; van den Berg, R.; Titov, I.; and Welling, M. 2017. Modeling Relational Data with Graph Convolutional Networks. arXiv:1703.06103.

Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C. D.; Ng, A.; and Potts, C. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 1631–1642. Seattle, Washington, USA: Association for Computational Linguistics.

Stoyanov, V.; Gilbert, N.; Cardie, C.; and Riloff, E. 2009. Conundrums in Noun Phrase Coreference Resolution: Making Sense of the State-of-the-Art. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, 656–664. Suntec, Singapore: Association for Computational Linguistics.

Tai, K. S.; Socher, R.; and Manning, C. D. 2015. Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 1556–1566. Beijing, China: Association for Computational Linguistics.

Trieu, H.-L.; Duong Nguyen, A.-K.; Nguyen, N.; Miwa, M.; Takamura, H.; and Ananiadou, S. 2019. Coreference Resolution in Full Text Articles with BERT and Syntax-based Mention Filtering. In *Proceedings of The 5th Workshop on BioNLP Open Shared Tasks*, 196–205. Hong Kong, China: Association for Computational Linguistics.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. In *International Conference on Learning Representations*.

Wang, K.; Shen, W.; Yang, Y.; Quan, X.; and Wang, R. 2020. Relational Graph Attention Network for Aspect-based Sentiment Analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 3229–3238.

Wang, M.; Yu, L.; Zheng, D.; Gan, Q.; Gai, Y.; Ye, Z.; Li, M.; Zhou, J.; Huang, Q.; Ma, C.; Huang, Z.; Guo, Q.; Zhang, H.; Lin, H.; Zhao, J.; Li, J.; Smola, A. J.; and Zhang, Z. 2019. Deep Graph Library: Towards Efficient and Scalable Deep Learning on Graphs. *CoRR*, abs/1909.01315.

Wu, W.; Wang, F.; Yuan, A.; Wu, F.; and Li, J. 2020. CorefQA: Coreference Resolution as Query-based Span Prediction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 6953–6963.

Wu, Z.; and Gardner, M. 2020. Understanding Mention Detector-Linker Interaction for Neural Coreference Resolution. arXiv:2009.09363.

Xu, B.; Wang, N.; Chen, T.; and Li, M. 2015. Empirical Evaluation of Rectified Activations in Convolutional Network. arXiv:1505.00853.

Xu, L.; and Choi, J. D. 2020. Revealing the Myth of Higher-Order Inference in Coreference Resolution. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 8527–8533.

Zhang, Y.; Zhou, H.; and Li, Z. 2020. Fast and Accurate Neural CRF Constituency Parsing. In *Proceedings of IJCAI*, 4046–4053.