Decomposed Local Models for Coordinate Structure Parsing

Hiroki Teranishi[1] **Hiroyuki Shindo**[1] **Yuji Matsumoto**[1,2]
[1]Nara Institute of Science and Technology
[2]RIKEN Center for Advanced Intelligence Project (AIP)
{teranishi.hiroki.sw5, shindo, matsu}@is.naist.jp

Abstract

We propose a simple and accurate model for coordination boundary identification. Our model decomposes the task into three subtasks during training; finding a coordinator, identifying inside boundaries of a pair of conjuncts, and selecting outside boundaries of it. For inference, we make use of probabilities of coordinators and conjuncts in the CKY parsing to find the optimal combination of coordinate structures. Experimental results demonstrate that our model achieves state-of-the-art results, ensuring that the global structure of coordinations is consistent.

1 Introduction

Coordination is a frequently occurring structure that consists of conjuncts joined by a coordinator word. Since conjunct spans are one of the major ambiguities, identifying them is difficult, even for humans. For instance, in the sentence "*Toshiba's line of portables, for example, features the T-1000, which is in the same weight class* *but* *is much slower* *and* *has less memory,* *and* *the T-1600, which also uses a 286 microprocessor,* *but* *which weighs almost twice as much* *and* *is three times the size*," we cannot find correct conjuncts for each coordinator at a glance. The presence of coordination makes a sentence more ambiguous and longer, resulting in errors in syntactic parsing.

To identify the conjuncts of a given coordinator, previous studies have explored two properties of coordinate structures: (1) similarity – conjuncts tend to be similar; (2) replaceability – conjuncts can be replaced. Ficler and Goldberg (2016b) combine the syntactic parser and neural networks to compute the similarity and replaceability features of conjuncts. Teranishi et al. (2017) also exploit the two properties without deploying any syntactic parser, and achieve state-of-the-art results. Although both approaches outperform the similarity-based approaches (Shimbo and Hara, 2007; Hara et al., 2009), they cannot handle more than two conjuncts in a coordination, and multiple coordinations in a sentence at one time. Hence, their systems may produce coordinations that conflict with each other. In contrast, Hara et al. (2009) define production rules for coordination in order to output consistent coordinate structures.

Here, we propose a new framework for coordination boundary identification. We generalize a scoring function that takes a pair of spans with a coordinator and returns a higher score when the two spans appear to be coordinated. Using this function in the CKY parsing with production rules for coordination, our system produces globally consistent coordinations in a given sentence. To obtain such a function, we decompose the task into three independent subtasks – finding a coordinator, identifying the inner boundaries of a pair of conjuncts and delineating its outer boundaries. We use three different neural networks for the tasks, and the networks are trained on the basis of their local decisions. Our method is inspired by recent successes with locally-trained models for structured inference problems such as constituency parsing (Teng and Zhang, 2018) and dependency parsing (Dozat and Manning, 2017) without globally-optimized training. Experimental results reveal that our model outperforms existing systems and our strong baseline, an extension of Teranishi et al. (2017), and ensures that the global structure of the coordinations is consistent.

In summary, our contributions include the following:

- We propose a simple framework that trains a generalized scoring function of a pair of conjuncts and uses it for inference.
- We decompose the task and use three local models that interoperate for the CKY parsing.

Proceedings of NAACL-HLT 2019, pages 3394–3403
Minneapolis, Minnesota, June 2 - June 7, 2019. ©2019 Association for Computational Linguistics

- We establish a system that can accommodate more than two conjuncts in a sentence.
- Our system outperforms existing ones, particularly because it produces globally consistent coordinate structures.

2 Coordination Boundary Identification

2.1 Coordinate Structure

A *coordinate structure* or *coordination* is a syntactic structure in which two or more elements, known as *conjuncts*, are linked by *coordinator*(s). In addition to coordinating words, such as "and," "or," or "but," some punctuation marks function secondarily to connect two conjuncts. We refer to those punctuation marks as *sub-coordinator*s. Sub-coordinators cannot independently conjoin phrases to form a coordinate structure. The presence of a coordination is usually signaled by the appearance of a coordinator; however, coordinating words do not always lead to coordinations. For instance, "but" is not a coordinator when it functions as a preposition. In this paper, we refer to a word that can be a coordinator or sub-coordinator as a *coordinator key*.

2.2 Task Definition and Difficulties

The task of coordination boundary identification is to find conjunct spans of a given coordinating word. If a coordinating word does not act as a coordinator, a system must return NONE; denoting the absence of a coordinate structure. The difficulties in this task arise when there are multiple coordinate structures in a sentence or more than two conjuncts in a single coordinate structure. If there is more than one coordinate structure in a sentence, each coordinate structure must be isolated from the others or integrated into the other(s). In other words, coordinate structures cannot be partially overlapped. When there are more than two conjuncts in a coordinate structure, it has to be ascertained whether the punctuation marks are sub-coordinators that bring one more conjunct, and if so, which coordinate structure they belong to. Thus, we must identify how many conjuncts a coordinate structure contains and the location of those conjuncts in the coordinate structure — whether it is nested in or isolated from other coordinate structures.

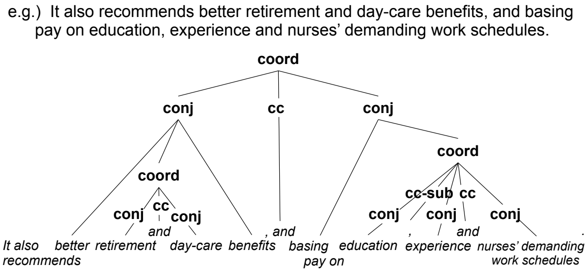
e.g.) It also recommends better retirement and day-care benefits, and basing pay on education, experience and nurses' demanding work schedules.
coord
conj
cc
conj
coord
coord
cc-sub
cc
conj
cc
conj
conj
conj
conj
and
, and
,
and
.
It also recommends
better
retirement
day-care
benefits
basing pay on
education
experience
nurses' demanding work schedules

Figure 1: An example of a coordinate tree.

2.3 Coordinate Structures as a Tree

Invoking Shimbo and Hara (2007), we use a tree to represent the coordinate structures in a sentence. We call this tree a **coordinate tree**. Figure 1 shows an example of a coordinate tree. Tree structures are particularly suitable because the ranges of coordinate structures are always consistent, and conjuncts are shown as nodes without being limited by the frequency of their occurrence. Our system produces a coordinate tree using the CKY algorithm and then retrieves well-formed coordinate structures from the tree. In this work, we focus on how to learn the scoring function that assigns higher scores to probable pairs of conjuncts for the CKY parsing.

3 Proposed Method

Our proposed model consists of three parts: a coordinator classifier and the inner and outer-boundary scoring models. Figure 2 is the overview of our framework. The coordinator classifier is a binary classifier that ascertains whether a word functions as a coordinator or not. The inner-boundary scoring model computes the score for a pair of conjuncts on the basis of their boundaries that are in proximity to a coordinator. This means that the model produces a score based on the end of the left conjunct and the beginning of the right conjunct. Similarly, the outer-boundary scoring model assigns a score to a pair of the beginning of the left conjunct and the end of the right conjunct. Using the inner and outer-boundary scoring models, our model calculates all possible combinations of the four boundaries, and then produces their probabilities. Given the local probabilities, we run the CKY algorithm to find the globally optimal coordinate structures in the sentence. In this section, we formulate our model based on the details of the neural networks' architecture; afterward, we describe the parsing method.

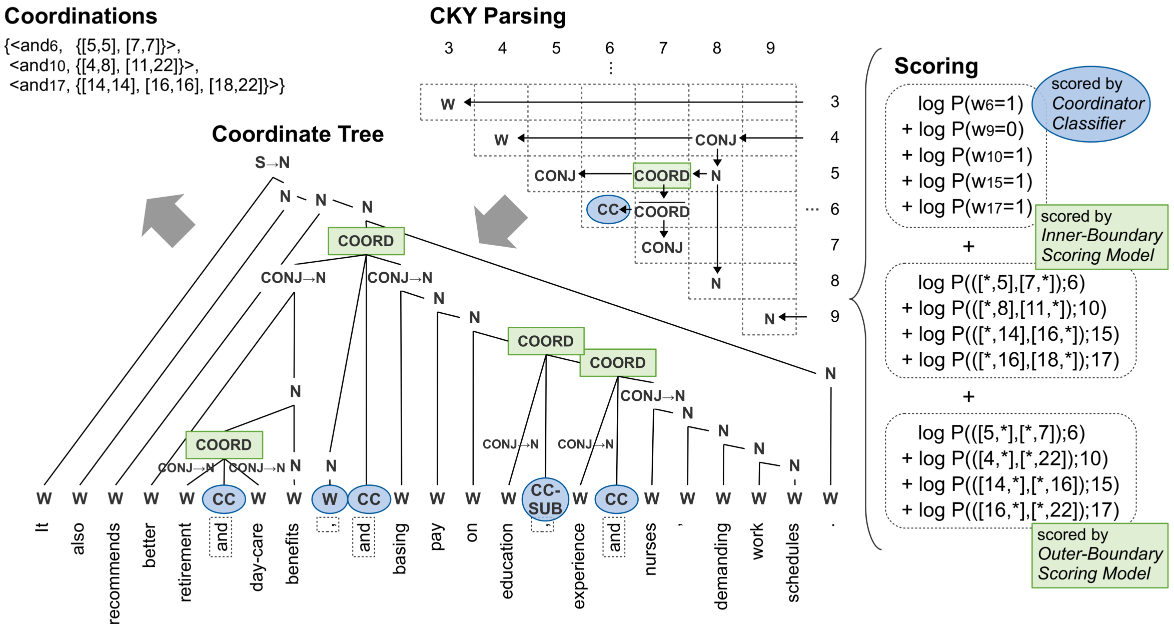

Figure 2: Overview of the proposed framework for coordination boundary identification. The scores of circular nodes are assigned by the coordinator classifier, and the scores of rectangular nodes are assigned by the inner and outer-boundary scoring models.

3.1 Model

Given a sentence that consists of N words $w_{1:N} = w_1, \ldots, w_N$ with the corresponding part-of-speech (POS) tags $p_{1:N} = p_1, \ldots, p_N$, our model outputs a set of coordinate structures $\{\langle c, \{[b_1, e_1], \ldots, [b_n, e_n]\}\rangle\} (n \geq 2)$ where c is a coordinator and $[b_k, e_k]$ is the k-th conjunct spanning from the b_k-th word to the e_k-th word. Although we cannot know the number of coordinate structures and conjuncts in each coordinate structure, we can use coordinator keys as clues to find pairs of conjuncts. Our model tries to find pairs of conjuncts, rather than coordinate structures, in a sentence.

$$\begin{aligned} X &= \{w_{1:N}, p_{1:N}, C\} \\ C &= \{t | w_t \in \mathbb{S}_{cc} \cup \mathbb{S}_{sub\text{-}cc}\} \\ Y &= \{\langle y_t^{ckey}, y_t^{pair}\rangle | t \in C\} \end{aligned} \tag{1}$$

where y_t^{ckey} is a label that indicates whether w_t is the actual coordinator ($y_t^{ckey} = 1$) or not ($y_t^{ckey} = 0$), and y_t^{pair} is a pair of conjunct spans. $y_t^{pair} = \varnothing$ when $y_t^{ckey} = 0$. When $t = 1$ or $t = N$, $y_t^{ckey} = 0$ because it does not form a coordinate structure within the sentence. In this paper, we define $\mathbb{S}_{cc}$ and $\mathbb{S}_{sub\text{-}cc}$ as {"and", "or", "but", "nor", "and/or"} and {",", ";", ":"}, respectively. We use two different models to identify inner and outer boundaries of y_t^{pair}, because enumerating all possible inner and outer boundaries of y_t^{pair} requires time complexity $\mathcal{O}(N^2) + \mathcal{O}(N^2) = \mathcal{O}(N^2)$, whereas enumerating all possible y_t^{pair} requires time complexity $\mathcal{O}(N^4)$[1].

Coordinator Classifier

The coordinator classifier is a binary classifier that predicts the label of a coordinator key.

$$P(y_t^{ckey} | w_t, \theta) = \mathrm{softmax}(f_{ckey}(w_t)) \tag{2}$$

The training loss of the binary classification is computed by the following equation:

$$\ell_\theta^{ckey}(X, Y) = - \sum_{\langle y_t^{ckey}, y_t^{pair}\rangle \in Y} \log P(y_t^{ckey} | w_t, \theta) \tag{3}$$

Inner-Boundary Scoring Model

The inner-boundary scoring model assigns a score to a pair of conjunct spans on the basis of inner boundaries. We use b^l, e^l, b^r, e^r to denote the beginning of a left conjunct, the end of the left conjunct, the beginning of a right conjunct, and the end of the right conjunct, respectively. The score

[1] For division of four boundaries, "two beginnings and two ends" or "left span and right span" can be chosen instead. In preliminary experiments, "left span and right span" models perform poorly, and "two beginnings and two ends" models perform well, but worse than "inner and outer-boundary" models.

of the inner-boundary pair (e^l, b^r) for a coordinator key w_t is calculated as follows:

$$\textsc{Score}_\theta^{inner}(e^l, b^r, w_t) = f_{inner}(e^l, b^r, w_t) \quad (4)$$

The probabilities of the inner boundaries are normalized distributions over all possible inner boundary pairs:

$$\begin{aligned} I_{w_t} = \{&(1, t+1), (1, t+2), \dots, (1, N), \\ &(2, t+1), \dots, (t-1, N)\} \end{aligned} \quad (5)$$

$$\begin{aligned} &P(y_t^{pair} = ([*, e^l], [b^r, *]) | w_t, \theta) = \\ &\frac{\exp\left(\textsc{Score}_\theta^{inner}(e^l, b^r, w_t)\right)}{\sum\limits_{(e'^l, b'^r) \in I_{w_t}} \exp\left(\textsc{Score}_\theta^{inner}(e'^l, b'^r, w_t)\right)} \end{aligned} \quad (6)$$

$$\begin{aligned} &\ell_\theta^{inner}(X, Y) = \\ &\quad - \sum_{\langle y_t^{ckey}, y_t^{pair} \rangle \in Y} y_t^{ckey} \log P(y_t^{pair} | w_t, \theta) \end{aligned} \quad (7)$$

The term $y_t^{ckey} \log P(y_t^{pair} | w_t, \theta)$ means the cross-entropy loss is activated only for positive coordinator keys ($y_t^{ckey} = 1$) and is disabled otherwise ($y_t^{ckey} = 0$).

Outer-Boundary Scoring Model

Similarly to the inner-boundary scoring model, we define the probability $P(y_t^{pair} = ([b^l, *], [*, e^r]) | w_t, \theta)$ based on the set of all the outer-boundary pairs O_{w_t}; the loss is defined as ℓ_θ^{outer} using the scoring function $\textsc{Score}_\theta^{outer}(b^l, e^r, w_t) = f_{outer}(b^l, e^r, w_t)$. Note that I_{w_t} and O_{w_t} are identical because their possible pairs are the same. Based on the inner pair probability $P(y_t^{pair} = ([*, e^l], [b^r, *]) | w_t, \theta)$ and the outer pair probability $P(y_t^{pair} = ([b^l, *], [*, e^r]) | w_t, \theta)$, the most probable pair is produced by:

$$\begin{aligned} y_t^{pair} &= \operatorname*{arg\,max}_{(\hat{e}^l, \hat{b}^r)} P(([*, \hat{e}^l], [\hat{b}^r, *]) | w_t, \theta) \\ &\cup \operatorname*{arg\,max}_{(\hat{b}^l, \hat{e}^r)} P(([\hat{b}^l, *], [*, \hat{e}^r]) | w_t, \theta) \end{aligned} \quad (8)$$

3.2 CKY Parsing

Our three models predict coordinators including sub-coordinators, and the inner and outer boundaries of their coordinating conjuncts. Such local predictions may cause conflicts between different coordinate structures. Furthermore, two conjuncts linked by a sub-coordinator must be embedded in another coordinate structure formed by a coordinator. To overcome these limitations, we use the CKY algorithm to find the optimal coordinations in a sentence. In particular, we define the CFG rules to produce a coordinate tree, as used in Hara et al. (2009). Our CFG rules, distinct from those of Hara et al. (2009) [2], are shown in Table 1. Based on these rules, we can map a coordinate tree to the one-to-one corresponding syntactic tree, covering 99.5% coordinations in the Penn Treebank [3].

Non-terminals			
COORD	Coordination		
CONJ	Conjunct		
CC	Coordinating conjunction		
CC-SUB	Sub-coordinator		
W	Word		
N	Non-coordination		
S	Sentence		
Rules for coordinations			
(1)	COORD	→	CONJ N? CC N? CONJ
(2)	COORD	→	CONJ CC-SUB COORD
(3)	CONJ	→	COORD
(4)	CONJ	→	N
Rules for non-coordinations			
(5)	S	→	COORD
(6)	S	→	N
(7)	N	→	COORD N
(8)	N	→	W COORD
(9)	N	→	W N
(10)	N	→	W
Rules for pre-terminals			
(11)	CC	→	(and|or|but|nor|and/or)
(12)	CC-SUB	→	(,|;|:)
(13)	W	→	*

Table 1: Production rules for coordinate trees. (. . . | . . .) matches one of the elements and "*" matches any word. "?" indicates zero or one occurrence of the preceding element.

3.2.1 Scoring

We give scores only to coordination nodes denoted as COORD, and pre-terminals. When scoring pre-terminals, we assign $\log P(w_k = 1)$ to CC and CC-SUB, and $\log(P(w_k = 0))$ to W if $w_k \in \mathbb{S}_{cc} \cup \mathbb{S}_{\textit{sub-cc}}$, otherwise 0. When scoring the CO-

[2] Our rules can produce coordinate structures that contain arbitrary length phrase(s) around coordinators, while conjuncts always appear next to coordinators in their rules.

[3] Most of the non-derivable coordinations are in the form like "A and B and C" where a coordinating word is regarded as a sub-coordinator. Even so, this expression can be parsed as a nested coordinate structure by the rules.

ORD, we take the left conjunct and the right conjunct which are linked by the CC. Thus, in the rule (2), the conjunct pair linked by a CC-SUB is the incoming CONJ and the leftmost CONJ in the child COORD. Using a coordinator and its pair of conjuncts, we assign $\log P(([i,j],[l,m])) = \log P(([*,j],[l,*])) + \log P(([i,*],[*,m]))$ to the COORD. The best scoring coordinate tree can be found efficiently using dynamic programming with time complexity $\mathcal{O}(N^3)$.

3.3 Neural Network Models

We use neural networks as instantiations of f_{ckey}, f_{inner}, and f_{outer} that we have introduced in this section.

Encoder

To get sentence-level representations for a sequence of words and POS tags, we use bidirectional long short-term memories (BiLSTMs) (Hochreiter and Schmidhuber, 1997).

$$\mathbf{h}_{1:N} = \text{BiLSTMs}(f_{input}(w_{1:N}, p_{1:N})) \qquad (9)$$

The dimensionality of each resulting vector $\mathbf{h}_t$ is $2d^{hidden}$. For the BiLSTMs inputs, we use f_{input} to map words and POS tags onto their representations. We can use different word representations including a pretrained word model, ELMo (Peters et al., 2018), BERT (Devlin et al., 2018) or character-level LSTMs/convolutional neural networks (CharCNNs). We demonstrate the differences between the different choices in Section 4. The entire network consisting of f_{input} and BiLSTMs is referred to as the encoder; it is shared by the three neural networks in the higher layer.

Coordinator Classifier

We use a linear transformation of the sentence-level representation of a coordinator key for f_{ckey}.

$$f_{ckey}(w_t) = \mathbf{W}^{ckey}\mathbf{h}_t + \mathbf{b}^{ckey} \qquad (10)$$

where $\mathbf{W}^{ckey} \in \mathbb{R}^{2 \times 2d^{hidden}}$ and $\mathbf{b}^{ckey} \in \mathbb{R}^2$ are the model parameters of the classifier.

Inner-Boundary Scoring Model

From the sentence-level representations produced by the encoder, the inner-boundary scoring model concatenates two representations of inner boundaries, and then feeds the produced vector into a multilayered perceptron (MLP).

$$\begin{aligned} &f_{inner}(e^l, b^r, w_t) = \\ &\quad \mathbf{w}_2^{in}\,\text{ReLU}(\mathbf{W}_1^{in}[\mathbf{h}_{e^l}; \mathbf{h}_{b^r}] + \mathbf{b}_1^{in}) + \mathrm{b}_2^{in} \end{aligned} \qquad (11)$$

where $\mathbf{W}_1^{in} \in \mathbb{R}^{d^{in} \times 4d^{hidden}}$, $\mathbf{b}_1^{in} \in \mathbb{R}^{d^{in}}$, $\mathbf{w}_2^{in} \in \mathbb{R}^{d^{in}}$ and $\mathrm{b}_2^{in} \in \mathbb{R}^1$ are the parameters of the inner-boundary scoring model.

Outer-Boundary Scoring Model

Using sentence-level representations, the outer-boundary scoring model takes two vectors that are calculated by subtracting the adjacent vectors to the coordinator from the boundary vectors. These subtraction operations are intended to capture the semantic distance and relatedness between two spans (Teranishi et al., 2017). The model then passes the vector to a MLP.

$$\begin{aligned} &f_{feature}(b^l, e^r, w_t, \mathbf{h}_{1:N}) = \\ &\quad [\mathbf{h}_{b^l} - \mathbf{h}_{t+1}; \mathbf{h}_{e^r} - \mathbf{h}_{t-1}] \end{aligned} \qquad (12)$$

$$\begin{aligned} &f_{outer}(b^l, e^r, w_t) = \\ &\quad \mathbf{w}_2^{out}\,\text{ReLU}(\mathbf{W}_1^{out}\,\mathbf{r}) + \mathbf{b}_1^{out}) + \mathrm{b}_2^{out} \\ &\quad \mathbf{r} = f_{feature}(b^l, e^r, w_t, \mathbf{h}_{1:N}) \end{aligned} \qquad (13)$$

where $\mathbf{W}_1^{out} \in \mathbb{R}^{d^{out} \times 4d^{hidden}}$, $\mathbf{b}_1^{out} \in \mathbb{R}^{d^{out}}$, $\mathbf{w}_2^{out} \in \mathbb{R}^{d^{out}}$ and $\mathrm{b}_2^{out} \in \mathbb{R}^1$ are the parameters of the outer-boundary scoring model.

3.4 Learning

To train the set of parameter θ of our neural networks, we minimize the following loss function:

$$\begin{aligned} L(\theta) = \sum_{(X,\hat{Y}) \in D} \Big(&\ell_\theta^{ckey}(X,\hat{Y}) \\ &+\ell_\theta^{inner}(X,\hat{Y}) \\ &+\ell_\theta^{outer}(X,\hat{Y})\Big) \end{aligned} \qquad (14)$$

where D is a set of pairs of a sentence and its correct coordinate structures in a training dataset. Thus, our submodels are trained jointly.

Why local training?

Instead of learning the scoring functions on the basis of local decisions, we can directly train our models combined with the CKY parsing using a structured max-margin objective between the scores of the best predicted and gold trees. In preliminary experiments, however, such a global training requires careful hyperparameter tuning and is hard to optimize stably, resulting in slightly better performance than the method of Teranishi et al. (2017).

4 Experiments

4.1 Settings

4.1.1 Datasets

We use the coordination-annotated Penn Treebank (Ficler and Goldberg, 2016a) (PTB) and Genia Treebank beta (Kim et al., 2003) (GENIA). Unlike the evaluation by Teranishi et al. (2017) and Ficler and Goldberg (2016b), we strip the PTB of all quotation marks (“) and (”) to normalize irregular coordinations such as ⟨... *“Daybreak,” “Daywatch,” “Newsday,” and “Newsnight,”* ...⟩. We follow the standard train/development/test split on the PTB. For the GENIA, we do not apply the preprocessing described above. We evaluate the model through a five-fold cross-validation, as in Hara et al. (2009).

4.1.2 Model

We use pretrained word vectors, POS tags, and character vectors produced by the CharCNN (Ma and Hovy, 2016), regarded as the *default*. We also investigate the performance of the model, using three different word representations for the encoder: (1) pretrained word embeddings; GloVe (Pennington et al., 2014) for the PTB, BioASQ (Tsatsaronis et al., 2012) for the GENIA, (2) contextualized sentence embeddings; ELMo, (3) randomly initialized word vectors. For the PTB, POS tags are obtained using the Stanford POS Tagger (Toutanova et al., 2003) with 10-way jackknifing. For the GENIA, we use the gold POS tags, as in Hara et al. (2009). To optimize the model parameters, we use Adam (Kingma and Ba, 2015). Other hyperparameters are described in Appendix A.

4.1.3 Baseline Model

We adopt our implementation of Teranishi et al. (2017) as the baseline. The original model of Teranishi et al. (2017) predicts the beginning and the end of a coordinate structure, and then splits it into conjuncts by commas. Their model decides the boundary of a coordinate structure individually, which may cause conflicts with that of other coordinate structure(s). Thus, we extend their model to find the best combination of coordinate structures, greedily choosing most probable boundaries without conflicts[4]. For the baseline model, we use the same encoder as that of our default model. Hereinafter, we refer to this baseline model as *Teranishi+17:+ext*.

4.1.4 Evaluation

We evaluate the systems on the basis of the ability to predict conjunct spans with the precision, recall, and F1 measures on the PTB. To compare the performance of our model with Teranishi et al. (2017), we adjudge the predicted conjuncts correct based on the following metrics.

- **whole**: matches at the beginning of the first conjunct and the end of the last conjunct.
- **outer**: matches in the first conjunct and the last conjunct.
- **inner**: matches in the two conjuncts adjacent to the coordinator.
- **exact**: matches in all the conjuncts.

In addition, we pay particular attention to the evaluation of NP coordination.

For the GENIA, we measure the recall values of coordinate structures by the aforementioned metrics; previous studies, on the other hand, evaluated their systems based only on the whole metric. Also, we evaluate the performance of our model based on syntactic categories.

4.2 Results

Tables 2 and 3 show the experimental results on the PTB and GENIA datasets. On the PTB, our model outperforms the baseline and existing methods for all metrics. We cannot compare its performance with that of existing methods because of its use of the preprocessing for quotation marks; nevertheless, our model achieves significant improvements. Our model is more accurate than the baseline because ours learns both the inner and outer boundaries of conjunct pairs including those of sub-coordinators, while the baseline learns only the coordination boundaries. On the GENIA, our model also outperforms the baseline on the exact metric. While our model has some limitations when it comes to predicting the beginning and the end of coordinations, it performs better on the inner metric. In contrast, Teranishi+17:+ext achieves the best results on the whole metric, whereas it performs poorly on the other metrics. This performance reflects the differences between the algorithms of the two systems. Our model builds a coordinate tree in a bottom-up manner and predicts inner conjuncts accurately. On the other hand, the baseline model predicts the entire span of a coordinate structure and splits them into conjuncts in a top-down fashion. That is why the base-

[4] We did not add the constraint to situate a nested coordination in the parent conjunct.

		Development						Test					
		All			NP			All			NP		
		P	R	F	P	R	F	P	R	F	P	R	F
Ours	whole	78.60	78.41	78.51	79.26	78.71	78.98	76.88	77.16	77.02	78.75	78.50	78.62
	outer	77.18	77.00	77.09	78.57	78.03	78.30	75.33	75.61	75.47	77.95	77.70	77.83
	inner	79.19	79.00	79.10	80.64	80.09	80.36	77.60	77.88	77.74	80.19	79.93	80.06
	exact	**76.95**	**76.76**	**76.85**	**78.11**	**77.57**	**77.84**	**75.33**	**75.61**	**75.47**	**77.95**	**77.70**	**77.83**
Teranishi+17 :+ext	whole	78.78	77.94	78.36	78.52	77.80	78.16	77.36	76.52	76.94	78.72	78.34	78.53
	outer	74.49	73.70	74.09	76.67	75.97	76.32	72.03	71.24	71.63	75.36	75.00	75.17
	inner	76.04	75.23	75.63	77.82	77.11	77.47	74.14	73.33	73.74	77.44	77.07	77.25
	exact	74.13	73.34	73.74	76.21	75.51	75.86	71.48	70.70	71.08	75.20	74.84	75.01
Teranishi+17*	whole	75.92	72.87	74.36	77.90	75.05	76.45	-	-	-	-	-	-
	outer	72.48	69.57	70.99	76.24	73.45	74.82	-	-	-	-	-	-
	inner	74.07	71.10	72.56	77.43	74.59	75.99	73.46	72.16	72.81	75.87	74.76	75.31
	exact	72.11	69.22	70.63	75.77	72.99	74.35	-	-	-	-	-	-
Ficler+16*	inner	72.34	72.25	72.29	75.17	74.82	74.99	72.81	72.61	72.7	76.91	75.31	76.1

Table 2: Evaluation per coordination by the different metrics. Preprocessing for quotation marks are not reported in "Teranishi+17" and "Ficler+16".

		NP	VP	ADJP	S	PP	UCP	SBAR	ADVP	Others	All
	#	2317	465	321	188	167	60	56	21	3	3598
Ours	whole	59.30	65.16	78.19	53.19	55.68	48.33	66.07	90.47	0.00	61.31
	outer	59.21	64.94	78.19	53.19	55.68	48.33	66.07	90.47	0.00	61.22
	inner	70.60	67.74	81.61	55.31	55.68	53.33	69.64	90.47	33.33	69.51
	exact	**59.21**	**64.94**	**78.19**	**53.19**	**55.68**	**48.33**	**66.07**	**90.47**	**0.00**	**61.22**
Teranishi+17 :+ext	whole	67.19	63.65	76.63	53.19	61.67	35.00	78.57	85.71	33.33	66.31
	outer	57.14	54.83	72.27	8.51	55.68	28.33	57.14	85.71	0.00	55.22
	inner	57.61	54.83	72.27	8.51	55.68	28.33	57.14	85.71	0.00	55.53
	exact	57.14	54.83	72.27	8.51	**55.68**	28.33	57.14	85.71	**0.00**	55.22
Teranishi+17	whole	66.59	63.87	78.50	52.65	53.89	50.00	78.57	85.71	33.33	65.98
Ficler+16	whole	65.08	71.82	74.76	17.02	56.28	51.66	91.07	80.95	33.33	64.14
Hara+09	whole	64.2	54.2	80.4	22.9	59.9	36.7	51.8	85.7	66.7	61.5

Table 3: Recall with Genia Treebank beta. The numbers in the rows "Teranishi+17," "Ficler+16" and "Hara+09" are taken from their papers.

line model cannot predict coordinated clauses labeled as "S," that are likely to be longer and to contain non-coordinating commas. The shortcoming of our model is that our bottom-up parsing may cause errors due to wrong decisions in the early stage of the parsing; this is observed as poor performance in the whole metric.

4.3 Analysis

Complete match in a sentence

We investigate the ability of our system to predict all the coordinate structures in a sentence precisely. We categorize sentences into the following four groups[5].

All: All sentences that have any coordinate structure.

- **Simple**: Sentences that have only one coordinate structure consisting of two conjuncts.
- **Complex**: Sentences that are categorized as Consecutive and/or Multiple.
  - **Consecutive**: Sentences that have a coordinate structure consisting of more than two conjuncts.
  - **Multiple**: Sentences that have multiple coordinate structures.

Sentences categorized as "All" are the union of the mutually exclusive sets of Simple and Complex.

Table 4 shows complete match rates on the PTB. Both on the development and test sets, our system records significant gain, in comparison to Teranishi+17:+ext, on Simple coordination sentences. It might be because the inner and outer-boundary scoring models learn to predict four boundaries of two spans, whereas the baseline model predicts only two outer boundaries on Simple coordination sentences. Since an appositive or adverbial phrase can appear between a coordinator and its conjunct, the assumption that two conjuncts must be next to

[5] *Consecutive* and *Multiple* both contain sentences that are Consecutive and Multiple.

Model	Sentence	Development	Test
Ours	All	489 / 673 = **72.65**	619 / 873 = **70.90**
	- Simple	378 / 481 = **78.58**	476 / 609 = **78.16**
	- Complex	111 / 192 = **57.81**	143 / 264 = **54.16**
	- Consecutive	41 / 66 = **62.12**	56 / 96 = **58.33**
	- Multiple	79 / 146 = **54.10**	96 / 197 = **48.73**
Teranishi+17 :+ext	All	468 / 673 = 69.53	577 / 873 = 66.09
	- Simple	358 / 481 = 74.42	444 / 609 = 72.90
	- Complex	110 / 192 = 57.29	133 / 264 = 50.37
	- Consecutive	40 / 66 = 60.60	48 / 96 = 50.00
	- Multiple	78 / 146 = 53.42	92 / 197 = 46.70

Table 4: Complete match rates of coordinations per sentence.

	All (exact)			All (inner)
	P	R	F	F
default	76.95	76.76	76.85	79.10
-POS tags	71.59	71.34	71.47	74.42
-CharCNNs	76.41	76.41	76.41	78.53
-GloVe	75.05	75.23	75.14	77.03
+ELMo	76.35	76.17	76.26	78.15
concat feature	74.85	74.41	74.63	76.64

Table 5: Performance comparison between different settings of the proposed models.

a coordinator fails and causes errors. Our system also outperforms Teranishi+17:+ext on Consecutive and Multiple coordination sentences. Teranishi+17:+ext predicts a coordination span, and then splits it into conjunct spans. Therefore, it can mistakenly segment coordinations when false sub-coordinators appear in a sentence. In contrast, our approach ascertains whether sub-coordinating words are true sub-coordinators; thus, it can lead to more robust production of Consecutive sentences.

What helps for Coordination Parsing?

We conduct an ablation study for our model. Table 5 shows the results. Without the POS tags, the model performs poorly. It is worthy of note that the pretrained word embedding is beneficial information for the task. On the other hand, the use of contextual embedding, ELMo, does not improve performance. We deduce that POS tags and morphological information, and not contextual word senses, are clues for shorter and similar coordinations such as NP coordinations. For the feature extraction function of the outer-boundary scoring model, the *concat* function that performs the same function as the inner-boundary scoring model does not achieve competitive advantage. The feature function described as Eq. 12 is designed to capture the similarity and replaceability of two spans; while the *concat* function has only the contextual information of the outer boundaries of a pair.

5 Related Work

5.1 Similarity-based Approaches

For the coordination identification task in Japanese, Kurohashi and Nagao (1994) used a chart to find the highest similarity pair of conjuncts using dynamic programming. Hogan (2007) developed a generative parsing model for coordinated noun phrases, incorporating symmetry in conjunct structures and head words. Shimbo and Hara (2007) proposed a discriminative model that computes scores based on the syntactic and morphological features assigned to edges and nodes in a sequence alignment. While their method focused on non-nested coordinations, Hara et al. (2009) extended their work to accommodate nested coordinations using CFG rules. A consistent global structure of coordinations is produced using discriminative functions based on the similarity of conjuncts with dynamic programming. Our concept of the CKY parsing is borrowed from their work; however, a key difference of our approach lies in how it computes the score of conjuncts and trains the score function. Hanamoto et al. (2012) used dual decomposition to combine HPSG parsing with the discriminative model developed by Hara et al. (2009).

5.2 Non Similarity-based Approaches

Kawahara and Kurohashi (2008) focused on resolving the ambiguities of coordinate structures without the use of any similarities. Their method relied on the dependency relations surrounding the conjuncts and the generative probabilities of phrases. Yoshimoto et al. (2015) extended the Eisner algorithm by adding new rules to accommodate coordinations during dependency parsing.

5.3 Coordination Boundary Identification using Neural Networks

Ficler and Goldberg (2016b) used neural networks for the coordination boundary identification task. They incorporated the replaceability property between conjuncts, in addition to the similarity property, in the computation of a score for a pair of conjuncts. They first used a binary classifier for coordinating words; then, they extracted probable candidate pairs of conjuncts using the Berkeley Parser (Petrov et al., 2006); afterward, they assigned scores to the pairs using neural networks. However, the shortcoming of their work is that it is highly dependent on the external parser. The work of Teranishi et al. (2017) developed an end-to-end model, as opposed to the pipeline approach of Ficler and Goldberg (2016b). They also used similarity and replaceability feature representations without information from a syntactic parser. While Ficler and Goldberg (2016b) cut off improbable pairs of conjuncts ahead of training, Teranishi et al. (2017) calculated scores for all possible pairs of the beginning and the end of coordinate structures instead of conjuncts. We apply the same strategy to the inner-boundary pairs and the outer-boundary pairs because assigning low probabilities to improbable inner and outer pairs makes the model robust for the CKY parsing.

6 Conclusion

We proposed a simple and accurate model for coordination boundary identification. Our system decomposes this task into three subtasks, and uses three different neural networks to tackle them. For inference, the CKY algorithm is applied using the CFG rules in order to produce globally consistent coordinate structures in a sentence. Experimental results demonstrated that our locally-trained models interoperate to obtain the optimal combination of coordinate structures and outperform existing systems and the strong baseline. Through empirical analysis, we found that our system performs better than the baseline in complete matches of sentences that contain more than two conjuncts and/or multiple coordinations.

Acknowledgments

This work was partly supported by JST CREST Grant Number JPMJCR1513 and JSPS KAKENHI Grant Number 18K18109. We are grateful to the anonymous reviewers for their helpful insights and comments.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *Proceedings of the 5th International Conference on Learning Representations*.

Jessica Ficler and Yoav Goldberg. 2016a. Coordination annotation extension in the penn tree bank. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 834–842, Berlin, Germany. Association for Computational Linguistics.

Jessica Ficler and Yoav Goldberg. 2016b. A neural network for coordination boundary prediction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 23–32, Austin, Texas. Association for Computational Linguistics.

Atsushi Hanamoto, Takuya Matsuzaki, and Jun'ichi Tsujii. 2012. Coordination structure analysis using dual decomposition. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 430–438, Avignon, France. Association for Computational Linguistics.

Kazuo Hara, Masashi Shimbo, Hideharu Okuma, and Yuji Matsumoto. 2009. Coordinate structure analysis with global structural constraints and alignment-based local features. 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*, pages 967–975, Suntec, Singapore. Association for Computational Linguistics.

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

Deirdre Hogan. 2007. Coordinate noun phrase disambiguation in a generative parsing model. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 680–687, Prague, Czech Republic. Association for Computational Linguistics.

Daisuke Kawahara and Sadao Kurohashi. 2008. Coordination disambiguation without any similarities. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 425–432, Manchester, UK. Coling 2008 Organizing Committee.

Jin Dong Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. 2003. Genia corpusa semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(suppl 1):i180–i182.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*.

Sadao Kurohashi and Makoto Nagao. 1994. A syntactic analysis method of long japanese sentences based on the detection of conjunctive structures. *Computational Linguistics*, 20(4):507–534.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

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

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 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)*, pages 2227–2237. Association for Computational Linguistics.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440, Sydney, Australia. Association for Computational Linguistics.

Masashi Shimbo and Kazuo Hara. 2007. A discriminative learning model for coordinate conjunctions. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 610–619, Prague, Czech Republic. Association for Computational Linguistics.

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. Association for Computational Linguistics.

Hiroki Teranishi, Hiroyuki Shindo, and Yuji Matsumoto. 2017. Coordination boundary identification with similarity and replaceability. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 264–272, Taipei, Taiwan. Asian Federation of Natural Language Processing.

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 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 173–180, Stroudsburg, PA, USA. Association for Computational Linguistics.

George Tsatsaronis, Michael Schroeder, Georgios Paliouras, Yannis Almirantis, Ion Androutsopoulos, Eric Gaussier, Patrick Gallinari, Thierry Artieres, Michael R Alvers, Matthias Zschunke, and Axel-Cyrille Ngonga Ngomo. 2012. Bioasq: A challenge on large-scale biomedical semantic indexing and question answering. In *AAAI Fall Symposium Series: Information Retrieval and Knowledge Discovery in Biomedical Text*.

Akifumi Yoshimoto, Kazuo Hara, Masashi Shimbo, and Yuji Matsumoto. 2015. Coordination-aware dependency parsing (preliminary report). In *Proceedings of the 14th International Conference on Parsing Technologies*, pages 66–70, Bilbao, Spain. Association for Computational Linguistics.

A Hyperparameters

Name	Value
Dimention of the word embeddings (GloVe)	100
Dimention of the word embeddings (BioASQ)	200
Dimention of the POS tag embeddings	50
Dimention of the character embeddings in the CharCNNs	10
Window size of the the CharCNNs	5
Dimention of the produced representation from the CharCNNs	50
Dimension of the LSTM hidden vector d^{hidden}	512
Number of BiLSTMs layers	2
MLP units in the hidden layer d^{in}	1024
MLP units in the hidden layer d^{out}	1024
Dropout ratio (all)	0.50
Initial learning rate	0.001
Regularization term λ (PTB)	0.0
Regularization term λ (GENIA)	0.0001
Gradient clipping threshold	5.0

Table 6: The final hyperparameters used in the experiments.