A Structured Span Selector

Tianyu Liu   Yuchen Eleanor Jiang   Ryan Cotterell   Mrinmaya Sachan

{`tianyu.liu, yucjiang, ryan.cotterell, mrinmaya.sachan`}@inf.ethz.ch

ETH *zürich*

Abstract

Many natural language processing tasks, e.g., coreference resolution and semantic role labeling, require selecting text spans and making decisions about them. A typical approach to such tasks is to score all possible spans and greedily select spans for task-specific downstream processing. This approach, however, does not incorporate any inductive bias about what sort of spans ought to be selected, e.g., that selected spans tend to be syntactic constituents. In this paper, we propose a novel grammar-based structured span selection model which learns to make use of the partial span-level annotation provided for such problems. Compared to previous approaches, our approach gets rid of the heuristic greedy span selection scheme, allowing us to model the downstream task on an optimal set of spans. We evaluate our model on two popular span prediction tasks: coreference resolution and semantic role labeling; and show improvements on both.

 https://github.com/lyutyuh/
structured-span-selector

1 Introduction

The problem of selecting a continuous segment of the input text, termed a **span**, is a common design pattern[1] in NLP. In this work, we call this design pattern **span selection**. Common tasks that have a span prediction component include coreference resolution (Stede, 2011) where the selected spans are mentions, semantic role labeling (Palmer et al., 2010) where the selected spans are arguments, question answering (Jurafsky and Martin, 2009) where the selected spans are answers, and named entity recognition (Smith, 2011) where the selected spans are entities.

In most of the tasks mentioned above, span selection is the first step.[2] After a set of candidate spans

Figure 1: Examples of two span prediction tasks: *Coreference* and *SRL*. In Coref, $[s]_i$ denotes a span of text s referring to the ith entity. In SRL, $\boxed{i}$ denotes a set of predicates and $[s]_{i-*}$ denotes a set of arguments for the ith predicate.

is determined, a classifier (often a neural network) is typically used to make predictions about the candidate spans. For instance, in coreference resolution, the selected spans (mentions) are clustered according to which entity they refer to; whereas, in SRL the spans (arguments) are classified into a set of roles. Two examples are shown in Fig. 1.

As the number of spans to consider in the input text can be quadratic in the length of the input, candidate spans are greedily selected as potential antecedents, roles, or answers. While greedy span selection has become the de-facto approach in span prediction problems, it has several issues. First, such approaches typically ignore the inherent structure of the problem. For example, spans of interest in problems such as coreference and SRL are typically syntactic constituents, an assumption supported by quantitative results.[3] The lack of syntactic constraints on the spans of interest leads to a waste of computational resources, as all $\mathcal{O}(N^2)$ possible spans are enumerated by the model.

In this paper, we propose a structured span selector for span selection. Our span selector is a syntactically aware, weighted context-free grammar that learns to score partial, possibly nested span annotations. In the case of partial an-

---

[1] A software engineering metaphor, which refers to a reusable solution to a commonly occurring problem.

[2] Note that we also have joint models where we find optimal spans and make downstream decisions simultaneously (Lee et al., 2017; He et al., 2018).

[3] For the OntoNotes dataset, He et al. (2018) reports that 98.7% of the arguments in SRL are constituents. For coreference, we find that 99.1% of the mentions are constituents.

notation, we marginalize out the missing structure and maximize the marginal log-likelihood. Fig. 2 illustrates an example partial parse of our WCFG and the difference between the traditional greedy approach and our approach.

We apply our span selector to both coreference and SRL. In both cases, we optimize the log-likelihood of the joint distribution defined by the span selector and the conditional for the downstream task as defined in Lee et al. (2017) and He et al. (2018). In contrast to previous approaches which heavily rely on heuristics[4] to prune the set of spans to be considered, our span selector directly admits a tractable inference algorithm to determine the highest-scoring set of spans. We observe that the number of spans our model selects is significantly lower than the number of spans considered in previous works, resulting in a reduction in memory required to train these models. Our approach leads to significant gains in both downstream tasks considered in this work: coreference and SRL. We find that our approach improves the performance of the end-to-end coreference model on the OntoNotes English dataset (0.4 F1) and the LitBank dataset (0.7 F1). On SRL, our model also achieves consistent gains over previous work.

2 Background and Related Work

2.1 Span Selection as a Design Pattern

Many NLP tasks involve reasoning over textual spans, e.g., coreference resolution, semantic role labeling, named entity recognition, and question answering. Models for these span prediction tasks often follow a common design pattern: They decompose into two components: (i) a **span selection** component where the model first selects a set of spans of interest, and (ii) a **span prediction** component where a prediction (e.g., entity or role assignment) is made for the chosen set of spans.

As shown in previous works (Zhang et al., 2018; Wu et al., 2020), the quality of the span selector can have a large impact on the overall model performance. The span selector typically selects spans by selecting the start token and the end token in the span. Thus, there are inherently $\binom{N}{2}$ textual spans, which $\mathcal{O}(N^2)$, within a document with N tokens to choose from. Previous span selection models (Lee et al., 2017; He et al., 2018) enumerate all possible
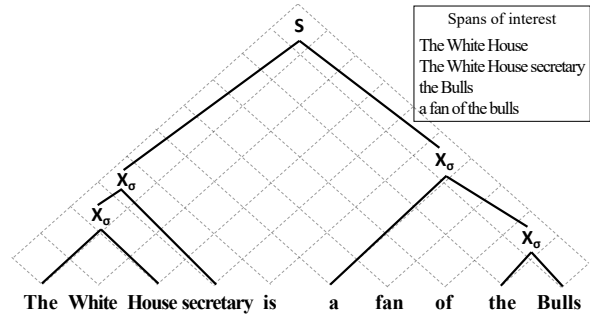

Figure 2: An example partial parse in our WCFG for coreference resolution and the set of spans it corresponds to. Production rules that do not involve the "span of interest" non-terminal (X_σ) are skipped as they do not affect the parsing result (see §3.2). The traditional greedy approach considers $\mathcal{O}(n^2)$ spans denoted by the grid cells unless some pruning heuristics are applied. While the number of nonterminals in a CNF parse is $\mathcal{O}(n)$.

spans in a brute-force manner and feed greedily selected top-k span candidates for the downstream prediction step.

However, several span selection tasks require spans that are syntactic constituents, which is a useful, but often neglected inductive bias in these works. For instance, in coreference resolution, mentions are typically noun phrases, pronouns, and sometimes, verbs. Similarly, in semantic role labeling, semantic arguments of a predicate are also typically syntactic constituents such as noun phrases, prepositional phrases, adverbs, etc.

Our work uses a context-free parsing formulation to enumerate spans. In context-free grammars, the number of valid syntactic constituents in a sequence of length N is bounded by $\mathcal{O}(N)$, as the constituents can be viewed as the internal nodes of a binary parse tree in Chomsky normal form. Compared with brute-force enumeration and greedy pruning, this inductive bias serves us as a natural pruning strategy to reduce the complexity from $\mathcal{O}(N^2)$ to $\mathcal{O}(N)$ and provides us a more natural and linguistically informed way to model span-based tasks in NLP, through which we can employ parsing techniques and retrieve the optimal span selection for downstream tasks.

To further motivate our work, we provide background on two popular span prediction tasks considered in our work: coreference resolution and semantic role labeling (SRL). We also describe some previous works on these tasks and contrast them to our work. Then, we also describe how our model can work with partial span-level annotations

---

[4] Lee et al. (2017) consider spans such that the maximum span width is 10, the number of spans per word is 0.4, etc. Then, the spans are greedily pruned.

provided by datasets for these tasks.

2.2 Coreference Resolution

Most coreference resolution models involve two stages: mention detection and mention clustering. Traditional pipeline systems rely on parse trees and hand-engineered rules to extract mentions (Raghunathan et al., 2010). However, Lee et al. (2017) show that we can directly detect mentions as well as assign antecedents for them in an end-to-end manner.

In addition to this paper, other works (Zhang et al., 2018; Swayamdipta et al., 2018; Miculicich and Henderson, 2020; Yu et al., 2020) have also explored better mention proposers for coreference. Zhang et al. (2018) use multi-task loss to directly optimize the mention detector. Swayamdipta et al. (2018) also leverage syntactic span classification as an auxiliary task to assist coreference. Thirukovalluru et al. (2021), Kirstain et al. (2021), Dobrovolskii (2021) explore token-level representations to both reduce memory consumption and increase performance on longer documents. Miculicich and Henderson (2020) and Yu et al. (2020) both improve the mention detector with better neural network structures. Yet they still need to manually set a threshold to control the number of selected candidate mentions and none of them could produce an optimal span selection for the downstream task. Finkel and Manning (2009) situate NER in a parsing framework by explicitly incorporating named entity types into parse tree labels. In contrast, our work requires neither syntactic annotations nor hyperparameter tuning for mention selection.

Zhang et al. (2018) propose a multi-task style loss to provide direct supervision to mention detection. Yet the seemingly simplistic and heuristic mention detection scheme by greedily taking a fixed portion of all candidate spans still remains unchanged in the aforementioned work. Pretrained text representations (Joshi et al., 2019, 2020; Lee et al., 2018) and higher-order inference techniques (Lee et al., 2018; Kantor and Globerson, 2019; Xu and Choi, 2020) have brought major improvements to the original model by Lee et al.. to achieve very good performance. Our span selector can also be extended to these works.

2.3 Semantic Role labeling

Semantic role labeling (SRL) extracts relations between predicates and their arguments. Two major lines of work in SRL are sequence-tagging models (He et al., 2017; Marcheggiani et al., 2017) and span-based models (He et al., 2018; Ouchi et al., 2018; Li et al., 2019).

Sequence tagging models for SRL convert semantic role annotations to BIO sequences. The tagger generates a label sequence for one single predicate at a time. However, span-based models generate the set of all candidate arguments in one forward pass and classify their semantic roles with regard to each predicate. As discussed in He et al. (2018), span-based models empirically perform better than sequence tagging models as they incorporate span-level features. Span-based models also do better at long-range dependencies as well as agreements with syntactic boundaries. Thus, we focus on span-based models in this work.

2.4 Nested and Partial Span Annotations

Nested mentions and partially annotated mentions are the two major concerns of this work. Most datasets for span prediction problems contain partial annotations of mentions. For example, singletons are not annotated in OntoNotes (Pradhan et al., 2012). In the coreference resolution example given in Fig. 1, the bracketed nested spans are annotated, while the underlined spans are valid mentions that are unannotated, since they do not co-refer with any other mention in the same document. The same is also true in SRL. In the SRL example in Fig. 1, there are two predicates (boxed words in the example) in one sentence. Their arguments are nested (i.e. ARG1 and ARG2 of the second predicate locate within ARG2 of the first predicate).

3 A Structured Span Selection Model

In this section, we develop the primary contribution of our paper: A new model for span selection. Specifically, we assert that almost all spans that a span selector should select are syntactic constituents; see Fig. 1 for two examples. Under this hypothesis, a context-free grammar (CFG) is a natural model for span selection as spans selected by a CFG cannot overlap, i.e., every pair of spans selected by a CFG would either be nested or disjoint.

3.1 Notation

We first start by introducing some basic terminology. We define a **document** D as a sequence of sentences $\mathbf{w}_1, \ldots, \mathbf{w}_{|D|}$. Each **sentence** $\mathbf{w}$ in the document is a sequence of words $[w_1, \ldots, w_{|\mathbf{w}|}]$. A **span** is a contiguous subsequence of words in a

sentence. For instance, we denote the span from position i to position k $w_i \cdots w_k$ as $[i, k]$.

3.2 Weighted Context-Free Grammars

Next, we define weighted context-free grammars (WCFG), the formalism that we will use to build our span selector. A WCFG is a five-tuple $\langle \Sigma, N, \mathrm{S}, R, \rho \rangle$, where Σ is an alphabet[5] of terminal symbols, N is a finite set of non-terminal symbols, $\mathrm{S} \in N$ is the unique start symbol, R is a set of production rules where a rule is of the form $X \to \boldsymbol{\alpha}$ where $X \in N$ and $\boldsymbol{\alpha} \in (N \cup \Sigma)^*$, and $\rho : R \to \mathbb{R}_{\geq 0}$ is a scoring function that maps every production rule to a non-negative real number.[6] We say a WCFG is in Chomksy normal form (CNF) if every production rule has one of three forms: $X \to YZ$, where $X, Y, Z \in N$, $X \to x$, where $X \in N$ and $x \in \Sigma$, or $\mathrm{S} \to \varepsilon$. For an input sentence $\mathbf{w}$, a WCFG defines a weighted set of parse trees, which we will denote $\mathcal{T}(\mathbf{w})$; we drop the argument from $\mathcal{T}$ when the sentence is clear from the context.

We overload the scoring function ρ to assign a weight to each parse tree $t \in \mathcal{T}(\mathbf{w})$. We define ρ applied to a tree as follows

$$\rho(t) = \prod_{r \in t} \rho(r) \geq 0 \qquad (1)$$

Given that ρ returns a non-negative weight, our WCFG can be used to define a distribution over the set of all parses of a sentence

$$p(t \mid \mathbf{w}) = \frac{\rho(t)}{Z} \qquad (2)$$

where $Z = \sum_{t \in \mathcal{T}(\mathbf{w})} \rho(t)$ is the sum of the scores of all parses. Using the familiar inside–outside algorithm (Baker, 1979), we can exactly compute Z in $\mathcal{O}(|\mathbf{w}|^3)$ time.

3.3 A WCFG Span Selector

To convert from parse trees to spans, our paper exploits a simple fact: If our grammar is in CNF, then every parse tree $t \in \mathcal{T}(\mathbf{w})$ corresponds to a *unique* set of labeled spans.[7] Specifically, we write $[i, X, k]$ iff the contiguous subsequence $w_i \cdots w_k$ corresponds to a constituent rooted at X in t. We will denote the tree-to-span bijection as $\mathtt{spans}(\cdot)$

---

[5] An alphabet is a finite, non-empty set.

[6] The scoring function can easily be generalized to map any production to a semiring value (Goodman, 1999).

[7] The necessary and sufficient condition for this to hold is that the grammar has no unary or nullary rules, so every parse tree in the grammar can be written as a bracketed string.

and write $M_t = \mathtt{spans}(t)$ to denote the set of spans t implies. We will also denote the set of all sets of spans viable under a CFG in CNF as $\mathcal{M}(\mathbf{w})$.

To extract spans useful for downstream tasks, we propose a simple WCFG. The grammar has three non-terminals $N \overset{\text{def}}{=} \{\mathrm{S}, \mathrm{X}_\sigma, \mathrm{X}_{\overline{\sigma}}\}$ where S is the distinguished start symbol. A span rooted at non-terminal X_σ, denoted $[i, \mathrm{X}_\sigma, k]$, is termed a **span of interest**; we will abbreviate $[i, \mathrm{X}_\sigma, k]$ as σ_{ik}. Likewise, a span rooted at non-terminal $\mathrm{X}_{\overline{\sigma}}$, denoted $[i, \mathrm{X}_{\overline{\sigma}}, k]$, is termed a **a span of non-interest**; we will abbreviate $[i, \mathrm{X}_{\overline{\sigma}}, k]$ as $\overline{\sigma}_{ik}$. The full grammar is given in App. A.1. We define our weight function $\rho : R \to \mathbb{R}_{\geq 0}$ as follows:

$$\rho\left({}_iX_k \to {}_iY_j \, {}_jZ_k \right) \overset{\text{def}}{=} \begin{cases} \exp s_p\left(\sigma_{ik}\right) & \text{if } X = \mathrm{X}_\sigma \\ 1 & \textbf{otherwise} \end{cases} \qquad (3)$$

where s_p is a learnable span-scorer that assigns a non-negative weight. Note this definition of ρ is an **anchored** scoring mechanism as it also makes use of the span indices i and k.

Under the simplified scoring function in Eq. (3), the score of a tree t can be re-expressed as a product of the scores of the spans of interest in $\mathtt{spans}(t)$. Specifically, for any tree t, we have

$$\rho(t) = \prod_{r \in t} \rho(r) \qquad (4)$$

$$= \exp\left(\sum_{\sigma_{ik} \in \mathtt{spans}(t)} s_p\left(\sigma_{ik}\right) \right) \qquad (5)$$

$$\overset{\text{def}}{=} \exp s_p\left(M_t\right) \qquad (6)$$

where $M_t = \left\{ \sigma_{ik} \mid \sigma_{ik} \in \mathtt{spans}(t) \right\}$. Note that the step from Eq. (4) to Eq. (5) follows from the fact that only those spans rooted at X_σ have a weight other than 1 under our choice of ρ and, furthermore, ρ ignores the body of the context-free rule. The problem of mention detection is hence converted from subset selection to finding an optimal parse tree that maximizes the score:

$$M^* = \underset{M_t \in \mathcal{M}(\mathbf{w})}{\mathrm{argmax}} \; s_p(M_t) \qquad (7)$$

$$= \mathtt{spans}\left(\underset{t \in \mathcal{T}(\mathbf{w})}{\mathrm{argmax}} \, \rho(t) \right) \qquad (8)$$

where the "Viterbi version" of the CKY algorithm (see App. A.3), can be employed to obtain an *exact* solution for the argmax function in Eq. (8)—again, in $\mathcal{O}(|\mathbf{w}|^3)$ time.

Finally, in order to train our WCFG with only partial supervision, i.e., in the case when we do not observe the entire tree, we require the marginal probabilities of the spans of interest. First, let $\mathcal{T}_{ik}$ be the set of parses that contain σ_{ik}. Then, the marginal probability of the span of interest σ_{ik} can be expressed as:

$$p(\sigma_{ik} \mid \mathbf{w}) = \sum_{t \in \mathcal{T}_{ik}} p(t \mid \mathbf{w}) = \sum_{t \in \mathcal{T}_{ik}} \frac{\rho(t)}{Z} \quad (9)$$

As described by Eisner (2016), we can compute $p(\sigma_{ik} \mid \mathbf{w})$ by computing the derivative of the log-normalizer $\log Z$ with respect to $s_p(\sigma_{ik})$, i.e.,

$$p(\sigma_{ik} \mid \mathbf{w}) = \frac{\partial \log Z}{\partial s_p(\sigma_{ik})} \quad (10)$$

Automatic differentiation ensures that this marginal computation will have the *same runtime* as the computation of $\log Z$ itself—to wit in $\mathcal{O}(|\mathbf{w}|^3)$ time (Griewank and Walther, 2008).

4   Adaptations to Downstream Tasks

In this section, we introduce how our structured span selector can be applied in an end-to-end manner to coreference resolution and SRL.

4.1   Coreference Resolution

The goal of coreference resolution is to link a span of interest σ_{ij}, termed a mention in the context of coreference resolution, to its **antecedent**. Note that the antecedent is either another mention in the same document or the dummy antecedent[8], which we denote as ϵ. We write $\sigma_{ij} \rightsquigarrow \sigma_{k\ell}$ to denote that $\sigma_{k\ell}$ is σ_{ij}'s antecedent. When formulating coreference in a probabilistic manner, we have the following natural decomposition:

$$p(\sigma_{ij}, \sigma_{ij} \rightsquigarrow \sigma_{k\ell}) \quad (11)$$
$$= \underbrace{p(\sigma_{ij})}_{\text{pr. } \sigma_{ij} \text{ is a mention}} \times \underbrace{p(\sigma_{ij} \rightsquigarrow \sigma_{k\ell} \mid \sigma_{ij})}_{\text{pr. } \sigma_{ij}\text{'s antecedent is } \sigma_{k\ell}}$$

The support of the above distribution is $\mathcal{X} \times \mathcal{Y}_{ij}$, where $\mathcal{X}$ is the set of all possible textual spans in D, $\mathcal{Y}_{ij} = \{\sigma_{k\ell} \mid j < \ell\} \cup \{\epsilon\}$ is the set of mentions preceding σ_{ij} plus the dummy antecedent ϵ for every $\sigma_{ij} \in \mathcal{X}$. In words, the above decomposition means that the probability of σ_{ij} co-referring with

---

[8]Following (Lee et al., 2017), the dummy antecedent ϵ represents two possible scenarios: (1) the span is not an entity mention or (2) the span is an entity mention but it is not coreferent with any previous span.

the span $\sigma_{k\ell}$ is the probability of first recognizing that σ_{ij} is itself a mention and then determining the link. In practice, this decomposition means that modelers can select $p(\sigma_{ij})$ and $p(\sigma_{ij} \rightsquigarrow \sigma_{k\ell} \mid \sigma_{ij})$ according to their taste and, importantly, independently of each other. In this work, we explore treating $p(\sigma_{ij})$ as the WCFG span selector described in §3.2 and $p(\sigma_{ij} \rightsquigarrow \sigma_{k\ell} \mid \sigma_{ij})$ as Lee et al. (2017)'s popular span ranking model for coreference.

Lee et al. (2017) as a Mention-Linker.   We now describe the mention-linker in Lee et al. (2017). We define the mention-linker distribution as

$$p(\sigma_{ij} \rightsquigarrow m \mid \sigma_{ij}) \quad (12)$$
$$= \frac{\exp s(\sigma_{ij}, m)}{\sum_{m' \in \mathcal{Y}_{ij}} \exp s(\sigma_{ij}, m')}$$

The scoring function $s(\cdot, \cdot)$ is defined in two cases: One case for $m = \epsilon$, the dummy antecedent, and one for $m = \sigma_{k\ell}$, a preceding span:

$$s(\sigma_{ij}, \epsilon) = 0 \quad (13)$$
$$s(\sigma_{ij}, \sigma_{k\ell}) = s_m(\sigma_{ij}) + s_m(\sigma_{k\ell}) + s_a(\sigma_{ij}, \sigma_{k\ell})$$

The first score function, $s_m(\sigma_{ij})$, is a score for span $[i, j]$ being a mention. The second function, $s_a(\sigma_{ij}, \sigma_{k\ell})$, a score that $\sigma_{k\ell}$ is an antecedent of $[i, j]$. In this work, both s_a and s_m are computed by neural networks that take span representations as inputs. However, in principle, they could be computed by any model.

Training.   Lee et al.'s model adopts a naïve greedy algorithm by taking the top $\lambda |D|$ spans with the highest mention scores s_m, where λ is a hyperparameter that has to be manually defined for different datasets. However, finding a proper ratio λ can be very tricky. In contrast, in our setting we can optimize the final objective function which is the log-likelihood of the joint distribution defined at the beginning of §4.1:

$$L_1 = \sum_{\mathbf{w} \in D} \sum_{\sigma_{ij} \in \mathcal{G}_{\mathbf{w}}} \log \sum_{m \in \mathcal{G}_{ij}} p(\sigma_{ij}, \sigma_{ij} \rightsquigarrow m) \quad (14)$$

where $\mathcal{G}$ is the (partially) annotated set of mentions, $\mathcal{G}_{\mathbf{w}}$ is the set of all textual spans of $\mathbf{w}$ in $\mathcal{G}$, and $\mathcal{G}_{ij}$ is the ground truth cluster that σ_{ij} belongs to.

Handling partial annotation with no singletons.   Since in many coreference datasets, e.g., OntoNotes, only the mentions that are referred to more than once are annotated, learning a mention

detector from such data requires the ability to handle the lack of singleton annotations. To handle partial span annotations, we marginalize out the unannotated singletons. This results in the following marginal log-likelihood

$$L_2 = \sum_{\mathbf{w} \in D} \sum_{\sigma_{ij} \in \overline{\mathcal{G}}_{\mathbf{w}}} \log \Big(p(\sigma_{ij} \rightsquigarrow \epsilon \mid \sigma_{ij}) p(\sigma_{ij})$$

$$+ (1 - p(\sigma_{ij})) \Big) \quad (15)$$

We optimize the loss $L = L_1 + L_2$ jointly. Here, $\overline{\mathcal{G}}_{\mathbf{w}}$ denotes the set of all spans of $\mathbf{w}$ *not* in $\mathcal{G}$.

Time Complexity. For each sentence, the inside–outside algorithm Eq. (22) and CKY algorithm Eq. (23) can be reduced to $\mathcal{O}(|\mathbf{w}|)$ semiring matrix multiplications. Sentence-level parallelism can also be applied to all the sentences.

4.2 Semantic Role Labeling

The goal of SRL is to classify the **semantic role** of every argument σ_{ij} with respect to a given predicate. Following the notation style in §4.1, we use $\sigma_{ij} \overset{l}{\rightsquigarrow} v$ to denote that σ_{ij} has the semantic role l in the frame of predicate v. The joint probability $p\left(\sigma_{ij}, \sigma_{ij} \overset{\ell}{\rightsquigarrow} v\right)$ can be written as:

$$\underbrace{p(\sigma_{ij})}_{\text{pr. } \sigma_{ij} \text{ is an argument}} \times \underbrace{p\left(\sigma_{ij} \overset{\ell}{\rightsquigarrow} v \mid \sigma_{ij}\right)}_{\text{pr. } \sigma_{ij}\text{'s role is } \ell \text{ w.r.t. predicate } v}$$

He et al. (2018) as a Role Classifier. As this work focuses on span selection, we choose the role classifier to be the popular and effective one from (He et al., 2018), and use gold predicates v during training and evaluation. The semantic role label l takes value from a discrete label space $\mathcal{L}$ which contains all the semantic roles plus the null relation ϵ. The classifier then models the following probability distribution:

$$p\left(\sigma_{ij} \overset{\ell}{\rightsquigarrow} v \mid \sigma_{ij}\right) = \frac{\exp s(\sigma_{ij}, v, \ell)}{\sum_{\ell \in \mathcal{L}} \exp s(\sigma_{ij}, v, \ell)}$$
$$s(\sigma_{ij}, v, \ell) = s_m(\sigma_{ij}) + s_r(\sigma_{ij}, v, \ell) \quad (16)$$

Similar to the coreference model of Lee et al. (2017), $s_m(\sigma_{ij})$ is the score for span $[i, j]$ to be an argument and $s_r(\sigma_{ij}, v, \ell)$ for $[i, j]$ to play the role l for predicate v. Note that the score for ϵ label (i.e. no relation) $s(\sigma_{ij}, v, \epsilon)$ is set to constant 0, similar to the dummy antecedent case in coreference.

Training Objective. He et al.'s model also suffers from the challenge of tuning the hyperparameter λ. The training objective for SRL with our structured span selection model is then:

$$L_1 = \sum_{\sigma_{ij} \in \mathcal{G}_{\mathbf{w}}} \sum_{v} \log p(\sigma_{ij}, \sigma_{ij} \overset{l}{\rightsquigarrow} v) \quad (17)$$

where l is the correct semantic label of σ_{ij} with regard to predicate v. Similar to coreference resolution, we handle the issue of partial annotation for the span selection model by adding the log-likelihood that σ_{ij} may not be an annotated argument:

$$L_2 = \sum_{\sigma_{ij} \in \overline{\mathcal{G}}_{\mathbf{w}}} \sum_{v} \log \Big(p(\sigma_{ij} \overset{\epsilon}{\rightsquigarrow} v \mid \sigma_{ij}) p(\sigma_{ij})$$

$$+ (1 - p(\sigma_{ij})) \Big) \quad (18)$$

And the final objective function is $L = L_1 + L_2$.

5 Experiments

5.1 The Greedy Baseline

Previous work has mostly considered a greedy procedure for span selection as opposed to Eq. (8). The approach produces a score $s_g(\sigma_{ik})$ *independently* for each span σ_{ik}. As the number of spans σ_{ik} is potentially very large, the set of spans is greedily pruned to a set of size K. For instance, in SRL, spans are selected for each sentence:

$$M^*_{\text{topk}} \quad (19)$$
$$= \text{atop}_K \Big(\Big\{ s_g(\sigma_{ik}) \mid 1 \le i < k \le |\mathbf{w}| \Big\} \Big)$$

However, in coreference resolution, a set of spans are selected for the entire document:

$$M^*_{\text{topk}} \quad (20)$$
$$= \text{atop}_K \Big(\bigcup_{\mathbf{w} \in D} \Big\{ s_g(\sigma_{ik}) \mid 1 \le i < k \le |\mathbf{w}| \Big\} \Big)$$

where atop_K is shorthand for argtop_K. We will see in our experiments that tuning K can be quite challenging (see Fig. 3). Moreover, as the greedy approach scores each span independently, it ignores the structure of the provided span annotation.

5.2 Datasets

Coreference. We experiment on the CoNLL-2012 English shared task dataset (OntoNotes) (Pradhan et al., 2012) and LitBank (Bamman et al.,

2020) in our experiments. As a part of our experiments on OntoNotes, we apply the speaker encoding in (Wu et al., 2020), that is using special tokens (<speaker>, </speaker>) to denote the speaker's name, as opposed to the original binary features used by Lee et al. (2017). This simple change brings a consistent boost to the performance by 0.2 F1. A major difference between these two datasets is that LitBank has singleton mention annotations while OntoNotes does not. For LitBank, we use the standard 10-fold cross-validation setup, as is the standard practice.

SRL. We use the CoNLL-2012 SRL dataset. Gold predicates are provided to the model.

5.3 Coreference Resolution

We report the average precision, recall, and F1 scores of the standard MUC, B^3, and $CEAF_{\phi_4}$ and the average CoNLL F1 score on OntoNotes test set in Tab. 1. The average F1 scores on LitBank are shown in Tab. 2. For OntoNotes, we run the experiments with 5 random initializations and the improvements reported are significant under the two-tailed t-test.

We compare our models with several representative previous works. In order to focus on comparing the impact of mention detection, we apply no higher-order inference techniques in our models and report the non-higher order result from (Xu and Choi, 2020). Joshi et al. (S) is the major baseline that uses SpanBERT (Joshi et al., 2020) and is trained with the speaker encoding discussed in §5.2. This encoding gives 0.2 F1 score improvement over the result reported in (Xu and Choi, 2020).

Our model with the structured mention detector achieves 80.5 F1 score, a 0.4 F1 improvement over the baseline. While on LitBank, our model achieves 76.3 F1, which is a 0.7 F1 improvement over Joshi et al.. It can also be observed that this gain mainly comes from improved recall, which is because we have a superior mention detector that can retrieve mentions with better accuracy. We further analyze this result in the following section.

5.3.1 Analysis of Mention Detector

Next, we examine the performance of our proposed mention detection scheme. As shown in Fig. 3, compared with Joshi et al. (S), our model predicts a far more accurate set of mentions with a higher recall. In contrast to Joshi et al. (S) who select $0.4|D|$ mention spans, our method on aver-
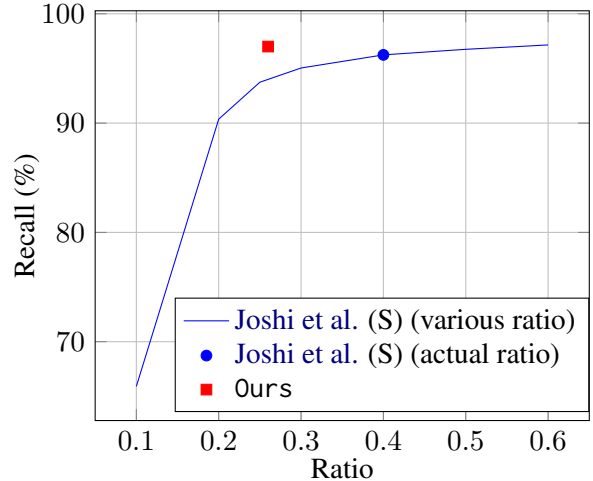

Figure 3: Recall of gold mentions as we vary the ratio of spans kept. Ratio refers to the number of predicted mentions divided by $|D|$. Our mention detector significantly outperforms Joshi et al. (S)'s with a ratio of 0.26 and a recall of 97.0%. Yet Joshi et al. (S) only achieves 96.2% recall with a ratio of 0.40.

age selects $0.26|D|$. The smaller span set makes the coreference model more efficient as well.

5.3.2 Analysis of Structured Modeling

To see how our structured modeling benefits coreference, we further compare our approach with a baseline Sigmoid which replaces the $p(\sigma_{ij})$ in Eq. (10) with a simple non-structured estimator:

$$p_{\text{sigm}}(\sigma_{ij}) = \text{sigmoid}(s_p(\sigma_{ij})) \qquad (21)$$

where $\text{sigmoid}(x) = \frac{1}{1+\exp(-x)}$. The loss function used for Sigmoid is same as the structured model given in §4.1. Through this comparison, we aim to show the effectiveness of structured modeling. We also build a multi-task learning baseline MTL similar to (Swayamdipta et al., 2018). The model performs an auxiliary task that classifies spans into a noun phrase, other syntactic constituents, or non-constituent. The coefficient of the multi-task loss is set to 0.1 as in (Swayamdipta et al., 2018).

This comparison is shown in Tab. 4. We find that replacing $p(\sigma_{ij})$ with unstructured $p_{\text{sigm}}(\sigma_{ij})$ degrades the performance by 0.3 F1 score. Thus, we conclude that the structured probability function is more expressive than $p_{\text{sigm}}(\sigma_{ij})$ as it models the global annotation for each sentence. In contrast, the mention detectors in Joshi et al. (S) and Sigmoid model each span independently.

	MUC			B³			CEAF$_{\phi_4}$			
	P	R	F	P	R	F	P	R	F	Avg. F1
Lee et al. (2017)	78.4	73.4	75.8	68.6	61.8	65.0	62.7	59.0	60.8	67.2
Lee et al. (2018)	81.4	79.5	80.4	72.2	69.5	70.8	68.2	67.1	67.6	73.0
Fei et al.	85.4	77.9	81.4	77.9	66.4	71.7	70.6	66.3	68.4	73.8
Kantor and Globerson	82.6	84.1	83.4	73.3	76.2	74.7	72.4	71.1	71.8	76.6
Joshi et al. (2019)	84.7	82.4	83.5	76.5	74.0	75.3	74.1	69.8	71.9	76.9
Joshi et al. (2020)	85.8	84.8	85.3	78.3	77.9	78.1	76.4	74.2	75.3	79.6
Xu and Choi	85.7	85.3	85.5	78.6	78.6	78.6	76.8	74.8	75.8	79.9
Joshi et al. (S)	86.6	84.5	85.6	80.4	77.3	78.8	77.8	74.0	75.8	80.1
Ours	86.1	85.5	**85.8**	79.8	78.8	**79.3**	77.4	75.4	**76.4**	**80.5**

Table 1: Results on the CoNLL-2012 English shared task test set. Avg. F1 in the last column denotes the average F1 of MUC, B³, and CEAF$_{\phi_4}$. Joshi et al. (S) refers to the original end-to-end model with SpanBERT and trained with speaker encoding. The improvements shown in the table are significant under the two-tailed *t*-test.

	Avg. P	Avg. R	Avg. F1
Bamman et al.	-	-	68.1
LB-MEM	-	-	75.7
U-MEM	-	-	75.9
Joshi et al.	78.7	72.9	75.6
Ours	77.4	75.3	**76.3**

Table 2: Results on the test set of LitBank. The results are averaged over 10 train/dev/test splits. LB-MEM and U-MEM are reported in (Toshniwal et al., 2020).

Nested Depth	1	2	3+
Greedy	96.5	87.1	86.2
Ours	97.8	93.2	93.9

Table 3: Recall rate of mentions of different nested depth on CoNLL-2012 dev set. There are 16873, 2100, and 182 mentions respectively of each nested depth.

	Avg. P	Avg. R	Avg. F1
Joshi et al. (S)	81.6	78.6	80.1
Sigmoid	80.8	79.6	80.2
MTL	80.8	78.9	79.8
Ours	81.1	79.9	**80.5**

Table 4: Comparison with three constructed baselines.

5.3.3 Analysing the source of improvement

Next, we try to explore where the gains of our model come from.

Nested Mentions. We first investigate the capability of our structured model in handling nested mentions. Tab. 3 shows the recall rate of mentions of different nested depths. Here, nested depth refers to the level of nesting in the mentions. E.g. in the first example given in Fig. 1, *The president* is of depth 1, while *he and his wife, now a New York senator* is of depth 2. As shown in Tab. 3, the gains of our method are larger for deeply nested mentions, which highlights the capability of our structured span detector to handle harder nested mentions that cannot be handled by the greedy selector.

Widths of Mentions. We also compare the recall rate for mentions of different widths in Tab. 5. We show that our model can detect longer spans better, which are usually more difficult to detect. For spans with 5-12 words, our structured model still maintains a recall rate of 96.5%, compared to a sharp drop for the greedy unstructured model.

5.4 Semantic Role Labeling

For semantic role labeling, we report the precision, recall, and F1 score on the CoNLL-2012 SRL dataset. The gold predicates are provided during both training and evaluation. Therefore, the model has to focus on extracting the correct arguments and classifying their roles for each predicate. The results are shown in Tab. 6 in comparison with pre-

Span Width	1-4	5-12	12+
Greedy	96.2	92.7	82.5
Ours	97.8	96.5	85.2

Table 5: Recall rate of mentions of different width on CoNLL-2012 dev set. There are 16356, 2180, and 619 mentions respectively of each width interval.

	Avg. P	Avg. R	Avg. F1
He et al. (2018)	-	-	85.5
Ouchi et al. (2018)	87.1	85.3	86.2
Li et al. (2019)	85.7	86.3	86.0
Shi and Lin (2019)	85.9	87.0	86.5
He et al.$_{\text{SpanBERT}}$	88.3	85.9	87.1
Ours	88.1	86.9	**87.5**

Table 6: Results on the test set of CoNLL-2012 semantic role labeling task. The precision, recall, and F1 scores are averaged over all semantic roles.

	P	R	F1
BIO	90.2	91.0	90.6
Greedy	92.8	90.2	91.5
Ours	92.8	91.2	92.0

Table 7: Comparison of unlabeled span accuracy.

vious span-based models. He et al.$_{\text{SpanBERT}}$ refers to the model of He et al. (2018) with SpanBERT$_{\text{large}}$ (Joshi et al., 2020) as sentence encoder.

Next, we report the performance of our span selector on the SRL task. Following the same trend as coreference resolution, we find that our structured model is able to extract much more accurate arguments and thus, significantly reduce the memory consumption for the downstream task. While keeping a comparable recall rate of gold arguments (96.5% for the greedy selector and 96.2% for ours), our span selector reduces the number of enumerated spans by 21.2%. We compare the accuracy of retrieving unlabeled argument spans in Tab. 7. BIO refers to the tagger-style SRL model using the same text encoder. Our model outperforms both baselines.

5.5 Qualitative Examples

In this section, we show a qualitative example to illustrate the grammar learned by our span selec-

Coref	[[The world' s] fifth [Disney] park] will soon [open] to [the public] here .
SRL	[The world' s [fifth] [Disney] [park]] [will] [soon] [[open] [to the public]] [here] .

Table 8: A qualitative example of the grammar learned by the structured span selector.

	Coref	SRL
Greedy	11.5	12.2
Ours	8.5	9.3

Table 9: Comparison of peak GPU memory usage in GBs at inference time on the development set.

tor for coreference resolution and SRL. Two sets of extracted spans for the same input sentence are shown in Tab. 8. For coreference resolution, our model selects maximal NPs (containing all modifiers) and verbs. While in SRL, the parse tree consists of much denser and syntactically heterogeneous spans of NPs, PPs, modal verbs, adverbs, etc. This comparison empirically shows that our model is capable and robust enough to learn a complex underlying grammar from partial annotation.

5.6 Memory Efficiency

We further analyze the memory efficiency of our model. We evaluate the peak GPU memory usage on the development set of OntoNotes. For both tasks, we see a significant reduction in memory usage of 27% for coreference and 24% for SRL.

6 Conclusion

In this paper, we proposed a novel structured model for span selection. In contrast to prior span selection methods, the model is structured, which allows it to model spans better. Instead of a greedy span selection procedure, the span selector uses partial span annotations provided in data to directly obtain the set of optimal spans. We evaluated our span selector on two typical span prediction tasks, namely coreference resolution and semantic role labeling, and achieved consistent gains in terms of accuracy as well as efficiency over greedy span selection.

Ethical Considerations

To the best of our knowledge, the datasets used in our work do not contain sensitive information, and we foresee no further ethical concerns with the work.

Acknowledgements

References

James K. Baker. 1979. Trainable grammars for speech recognition. *The Journal of the Acoustical Society of America*, 65(S1):S132–S132.

David Bamman, Olivia Lewke, and Anya Mansoor. 2020. An annotated dataset of coreference in English literature. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 44–54, Marseille, France. European Language Resources Association.

Vladimir Dobrovolskii. 2021. Word-level coreference resolution. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7670–7675, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jason Eisner. 2016. Inside-outside and forward-backward algorithms are just backprop (tutorial paper). In *Proceedings of the Workshop on Structured Prediction for NLP*, pages 1–17, Austin, TX. Association for Computational Linguistics.

Hongliang Fei, Xu Li, Dingcheng Li, and Ping Li. 2019. End-to-end deep reinforcement learning based coreference resolution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 660–665, Florence, Italy. Association for Computational Linguistics.

Jenny Rose Finkel and Christopher D. Manning. 2009. Joint parsing and named entity recognition. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 326–334, Boulder, Colorado. Association for Computational Linguistics.

Joshua Goodman. 1999. Semiring parsing. *Computational Linguistics*, 25(4):573–606.

Andreas Griewank and Andrea Walther. 2008. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*, second edition. Society for Industrial and Applied Mathematics, USA.

Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. 2018. Jointly predicting predicates and arguments in neural semantic role labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 364–369, Melbourne, Australia. Association for Computational Linguistics.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 473–483, Vancouver, Canada. Association for Computational Linguistics.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Mandar Joshi, Omer Levy, Luke Zettlemoyer, and Daniel Weld. 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)*, pages 5803–5808, Hong Kong, China. Association for Computational Linguistics.

Daniel Jurafsky and James H. Martin. 2009. *Speech and Language Processing*, 2nd edition. Prentice-Hall, USA.

Ben Kantor and Amir Globerson. 2019. Coreference resolution with entity equalization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 673–677, Florence, Italy. Association for Computational Linguistics.

Yuval Kirstain, Ori Ram, and Omer Levy. 2021. Coreference resolution without span representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 14–19, Online. Association for Computational Linguistics.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, Copenhagen, Denmark. Association for Computational Linguistics.

Kenton Lee, Luheng He, and Luke Zettlemoyer. 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)*, pages 687–692, New Orleans, Louisiana. Association for Computational Linguistics.

Zuchao Li, Shexia He, Hai Zhao, Yiqing Zhang, Zhuosheng Zhang, Xi Zhou, and Xiang Zhou. 2019. Dependency or span, end-to-end uniform semantic role labeling. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'19/IAAI'19/EAAI'19. AAAI Press.

Diego Marcheggiani, Anton Frolov, and Ivan Titov. 2017. A simple and accurate syntax-agnostic neural model for dependency-based semantic role labeling. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*,

pages 411–420, Vancouver, Canada. Association for Computational Linguistics.

Lesly Miculicich and James Henderson. 2020. Partially-supervised mention detection. In *Proceedings of the Third Workshop on Computational Models of Reference, Anaphora and Coreference*, pages 91–98, Barcelona, Spain (online). Association for Computational Linguistics.

Hiroki Ouchi, Hiroyuki Shindo, and Yuji Matsumoto. 2018. A span selection model for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1630–1642, Brussels, Belgium. Association for Computational Linguistics.

Martha Palmer, Daniel Gildea, and Nianwen Xue. 2010. Semantic role labeling. *Synthesis Lectures on Human Language Technologies*, 3(1):1–103.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 1–40, Jeju Island, Korea. Association for Computational Linguistics.

Karthik Raghunathan, Heeyoung Lee, Sudarshan Rangarajan, Nathanael Chambers, Mihai Surdeanu, Dan Jurafsky, and Christopher Manning. 2010. A multi-pass sieve for coreference resolution. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 492–501, Cambridge, MA. Association for Computational Linguistics.

Peng Shi and Jimmy Lin. 2019. Simple BERT models for relation extraction and semantic role labeling. *CoRR*, abs/1904.05255.

Noah A. Smith. 2011. Linguistic structure prediction. *Synthesis Lectures on Human Language Technologies*, 4(2):1–274.

Manfred Stede. 2011. Discourse processing. *Synthesis Lectures on Human Language Technologies*, 4(3):1–165.

Swabha Swayamdipta, Sam Thomson, Kenton Lee, Luke Zettlemoyer, Chris Dyer, and Noah A. Smith. 2018. Syntactic scaffolds for semantic structures. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3772–3782, Brussels, Belgium. Association for Computational Linguistics.

Raghuveer Thirukovalluru, Nicholas Monath, Kumar Shridhar, Manzil Zaheer, Mrinmaya Sachan, and Andrew McCallum. 2021. Scaling within document coreference to long texts. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3921–3931, Online. Association for Computational Linguistics.

Shubham Toshniwal, Sam Wiseman, Allyson Ettinger, Karen Livescu, and Kevin Gimpel. 2020. Learning to ignore: Long document coreference with bounded memory neural networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8519–8526, Online. Association for Computational Linguistics.

Wei Wu, Fei Wang, Arianna Yuan, Fei Wu, and Jiwei Li. 2020. CorefQA: Coreference resolution as query-based span prediction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6953–6963, Online. Association for Computational Linguistics.

Liyan Xu and Jinho D. Choi. 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)*, pages 8527–8533, Online. Association for Computational Linguistics.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Neural mention detection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1–10, Marseille, France. European Language Resources Association.

Rui Zhang, Cícero Nogueira dos Santos, Michihiro Yasunaga, Bing Xiang, and Dragomir Radev. 2018. Neural coreference resolution with deep biaffine attention by joint mention detection and mention clustering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 102–107, Melbourne, Australia. Association for Computational Linguistics.

A A Weighted Context-Free Grammar

A.1 The Grammar

In our WCFG $\langle \Sigma, N, S, R, \rho \rangle$, Σ is the set of all tokens in the vocabulary, $N = \{S, X_\sigma, X_{\overline{\sigma}}\}$, where S is the start symbol, X_σ is the span of interest, and $X_{\overline{\sigma}}$ is the spans that are not of interest. The complete set of production rules R is shown in Tab. 10.

S	$\rightarrow$	$X_\sigma\, X_\sigma$	
S	$\rightarrow$	$X_\sigma\, X_{\overline{\sigma}}$	
S	$\rightarrow$	$X_{\overline{\sigma}}\, X_\sigma$	
S	$\rightarrow$	$X_{\overline{\sigma}}\, X_{\overline{\sigma}}$	
X_σ	$\rightarrow$	$X_\sigma\, X_\sigma$	
X_σ	$\rightarrow$	$X_\sigma\, X_{\overline{\sigma}}$	
X_σ	$\rightarrow$	$X_{\overline{\sigma}}\, X_\sigma$	
X_σ	$\rightarrow$	$X_{\overline{\sigma}}\, X_{\overline{\sigma}}$	
$X_{\overline{\sigma}}$	$\rightarrow$	$X_\sigma\, X_\sigma$	
$X_{\overline{\sigma}}$	$\rightarrow$	$X_\sigma\, X_{\overline{\sigma}}$	
$X_{\overline{\sigma}}$	$\rightarrow$	$X_{\overline{\sigma}}\, X_\sigma$	
$X_{\overline{\sigma}}$	$\rightarrow$	$X_{\overline{\sigma}}\, X_{\overline{\sigma}}$	
X_σ	$\rightarrow$	$x,$	$\forall x \in \Sigma$
$X_{\overline{\sigma}}$	$\rightarrow$	$x,$	$\forall x \in \Sigma$

Table 10: Production rules R of our WCFG.

We only assign nontrivial weights to rules $_iX_k \rightarrow {}_iY_j\, {}_jZ_k$ where X is X_σ. That is to say, in Eq. (1), $s_p(_iX_k \rightarrow {}_iY_j\, {}_jZ_k) = 1$ where X is not X_σ.

A.2 The Inside–Outside Algorithm

For a span $[i, X, k]$, its inside value can be expressed as

$$\beta([i, X, k]) = \sum_{X \rightarrow Y\,Z \in R} \left(\exp s_p([i, X, k]) \times \Big(\sum_{j=i+1}^{k-1} \beta([i, Y, j]) \times \beta([j, Z, k]) \Big) \right) \qquad (22)$$

In the case when $k = i + 1$, we have $\beta([i, X, k]) = \exp s_p([i, X, k])$. The inside value of the entire sentence $\beta([0, S, |\mathbf{w}|])$ is exactly Z, the sum of scores of all parses.

A.3 The CKY Algorithm

We use the CKY algorithm to find an optimal parse tree $t^* \in \mathcal{T}(\mathbf{w})$ that maximizes $\rho(t)$. The recursive function when $i < k$ used is:

$$\gamma([i, X, k]) = \max_{X \rightarrow Y\,Z \in R} \left\{ s_p([i, X, k]) + \max_{i < j < k} \Big\{ \gamma([i, Y, j]) + \gamma([j, Z, k]) \Big\} \right\} \qquad (23)$$

In the case when $k = i + 1$, we have $\gamma([i, X, k]) = s_p([i, X, k])$.

B Experimental Settings

The systems are implemented with PyTorch. We use SpanBERT$_{\text{large}}$ as text encoder. We train the model for 20 epochs and select the best-performing model on the development set for testing. The documents are split into 512 word segments to fit in SpanBERT$_{\text{large}}$. Models used for coreference resolution have 402 million learnable parameters, and models for SRL have 382 million learnable parameters. We closely follow the hyperparameter settings of Joshi et al. (2020) and build our models upon the codebase of Xu and Choi (2020)[9] under Apache License 2.0. The learning rate of SpanBERT$_{\text{large}}$ parameters is set to

---

[9]https://github.com/lxucs/coref-hoi

1×10^{-5} with 0.01 decay rate, and the learning rate of task parameters is set to 3×10^{-4}. The dropout rate of feedforward neural network scorers is set to 0.3. When training our model, we randomly sample $0.1|D|$ negative spans that are not mentions and add their negative log-likelihood $-\log p(\sigma_{ij})$ to the training objective. This is to prevent $p(\sigma_{ij})$ from converging to 1. For SRL task, we use a batch size of 32 for 40 epochs and the same learning rate with coreference resolution. The same negative sampling technique is applied. Our models are trained on Nvidia Tesla V100 GPUs with 32GB memory. The average training time is around 8 hours for Joshi et al. (S) baseline and around 9 hours for our model. For SRL models, training takes 25 hours.

C   Results on the Development Set

In this section, we report the results that our models get on the development sets of OntoNotes and LitBank.

	Avg. P	Avg. R	Avg. F1
Joshi et al. (S)	82.0	78.8	80.4
Sigmoid	81.6	79.4	80.5
Ours	81.1	80.2	**80.7**

Table 11: Results on CoNLL-2012 coreference resolution development set.

	Avg. P	Avg. R	Avg. F1
Joshi et al. (S)	78.8	73.7	76.1
Ours	77.4	75.6	**76.6**

Table 12: Results on LitBank development set.

	Avg. P	Avg. R	Avg. F1
He et al.$_{\text{SpanBERT}}$	87.9	85.0	86.4
Ours	87.8	86.2	**87.0**

Table 13: Results on CoNLL-2012 semantic role labeling development set.