A dynamic programming algorithm for span-based nested named-entity recognition in $\mathcal{O}(n^2)$

Caio Corro
Universite Paris-Saclay, CNRS, LISN, 91400, Orsay, France
caio.corro@lisn.upsaclay.fr

Abstract

Span-based nested named-entity recognition (NER) has a cubic-time complexity using a variant of the CYK algorithm. We show that by adding a supplementary structural constraint on the search space, nested NER has a quadratic-time complexity, that is the same asymptotic complexity than the non-nested case. The proposed algorithm covers a large part of three standard English benchmarks and delivers comparable experimental results.

1 Introduction

Named entity recognition (NER) is a fundamental problem in information retrieval that aims to identify mentions of entities and their associated types in natural language documents. As such, the problem can be reduced to the identification and classification of segments of texts. In particular, we focus on mentions that have the following properties:

1. continuous, i.e. a mention corresponds to a contiguous sequence of words;
2. potentially nested, i.e. one mention can be inside another, but they can never partially overlap.

Four examples are shown in Figure 1.

In a span-based setting, recognition for nested NER has a cubic-time complexity (Finkel and Manning, 2009; Fu et al., 2021) using variants of the Cocke-Younger-Kasami (CYK) algorithm (Kasami, 1965; Younger, 1967; Cocke, 1970). If we restrict the search space to non-nested mentions, then recognition can be realized in quadratic time using a semi-Markov model (Sarawagi and Cohen, 2004). An open question is whether it is possible to design algorithms with better time-complexity/search space trade-offs.

In this paper, we propose a novel span-based nested NER algorithm with a quadratic-time complexity, that is with the same time complexity as the semi-Markov algorithm for the non-nested case. Our approach is based on the observation that many mentions only contain at most one nested mention of length strictly greater than one. As such, we follow a trend in the syntactic parsing literature that studies search-spaces that allow the development of more efficient parsing algorithms, both for dependency and constituency structures (Pitler et al., 2012, 2013; Satta and Kuhlmann, 2013; Gómez-Rodríguez et al., 2010; Corro, 2020), *inter alia*.

Our main contributions can be summarized as follows:

- We present the semi-Markov and CYK-like models for non-nested and nested NER, respectively — although we do not claim that these approaches for NER are new, our presentation of the CYK-like algorithm differs from previous work as it is tailored to the NER problem and guarantees uniqueness of derivations;
- We introduce a novel search space for nested NER that has no significant loss in coverage compared to the standard one, see Table 5;
- We propose a novel quadratic-time recognition algorithm for the aforementioned search space;
- We experiment our quadratic-time algorithm on three English datasets (ACE-2004, ACE-2005 and GENIA) and show that it obtains comparable results to the cubic-time algorithm.

2 Related work

Span-based methods: Semi-Markov models have been first proposed in the generative modeling framework for time-serie analysis and word segmentation (Janssen and Limnios, 1999; Ge, 2002).

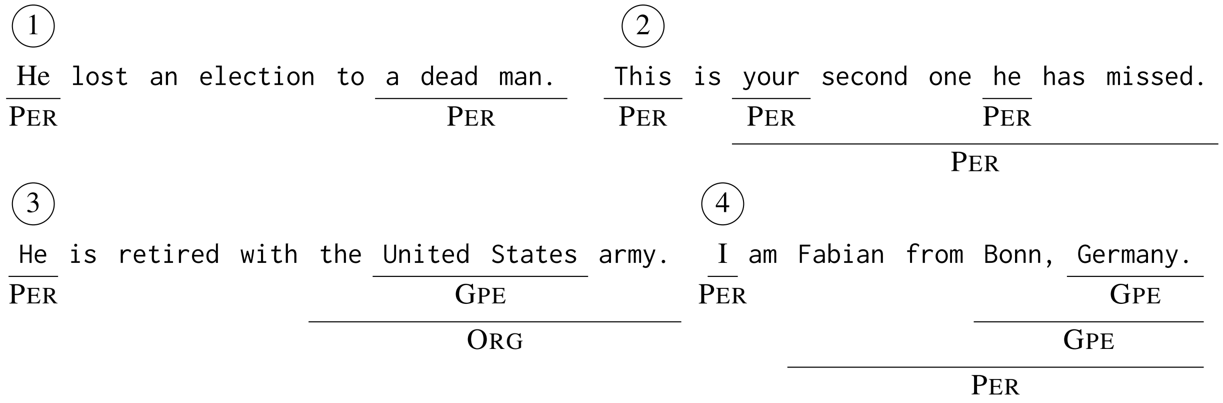
1
He lost an election to a dead man.
PER
PER
2
This is your second one he has missed.
PER
PER
PER
PER
3
He is retired with the United States army.
PER
GPE
ORG
4
I am Fabian from Bonn, Germany.
PER
GPE
GPE
PER

Figure 1: Sentence examples and their associated analyses from the ACE-2005 dataset.

Sarawagi and Cohen (2004) introduced a discriminative variant for NER. Arora et al. (2019) extended this approach with a task-tailored structured SVM loss (Tsochantaridis et al., 2004). Inference algorithms for semi-Markov models have a $\mathcal{O}(n^2)$ time complexity, where n is the length of the input sentence. Unfortunately, semi-Markov models can only recognize non-nested mentions. Finkel and Manning (2009) proposed a representation of nested mentions (together with part-of-speech tags) as a phrase structure, enabling the use of the CYK algorithm for MAP inference. Influenced by recent work in the syntactic parsing literature on span-based model, i.e. models without an explicit grammar (Hall et al., 2014; Stern et al., 2017), Fu et al. (2021) proposed to rely on these span-based phrase structure parsers for nested NER. As structures considered in NER are not *stricto sensu* complete phrase-structures, they use a latent span model. Inference in this model has a $\mathcal{O}(n^3)$ time complexity. Lou et al. (2022) extended this approach to lexicalized structures (i.e. where each mention has an explicitly identified head), leading to a $\mathcal{O}(n^4)$ time complexity for inference due to the richer structure.

Tagging-based methods: NER can be reduced to a sentence tagging problem using BIO and BILOU schemes (Ratinov and Roth, 2009) to bypass the quadratic-time complexity of semi-Markov models. MAP Inference (resp. marginal inference) is then a linear-time problem using the Viterbi algorithm (resp. forward-backward algorithm).[1] However, this approach cannot incorporate span features neither be used for nested entities. Alex et al. (2007) and Ju et al. (2018) proposed to rely on several tagging layers to predict nested entities. Shibuya and Hovy (2020) introduced an extension of the Viterbi algorithm that allows to rely on BIO tagging for nested NER by considering second-best paths. To leverage the influence of outer entities, Wang et al. (2021) rely on different potential functions for inner entities. Note that algorithms for the second-best paths method have a $\mathcal{O}(n^2)$ time complexity, that is similar to the span-based alogithm we propose.

Hypergraph-based methods: Lu and Roth (2015) proposed an hypergraph-based method for nested NER. Although this approach is appealing for its $\mathcal{O}(n)$ (approximate) inference algorithms, it suffers from two major issues: (1) the marginal inference algorithm overestimate the partition function; (2) the representation is ambiguous, that is a single path in the hypergraph may represent different analysis of the same sentence. Muis and Lu (2017) proposed a different hypergraph with $\mathcal{O}(n^2)$ inference algorithms that solves issue (1) but still exhibits issue (2). Katiyar and Cardie (2018) extended hypergraph methods to rely on neural network scoring. Wang and Lu (2018) proposed a novel hypergraph method that fixes issues (1) and (2) but their approach does not forbid partially overlapping mentions.

Unstructured methods: Several authors proposed to predict the presence of a mention on each span independently, sometimes with specialized neural architectures (Xu et al., 2017; Sohrab and Miwa, 2018; Zheng et al., 2019; Xia et al., 2019; Wang et al., 2020; Tan et al., 2020; Zaratiana et al., 2022), *inter alia*. Note that these approaches classify $\mathcal{O}(n^2)$ spans of text independently, hence the time-complexity is similar to the approach proposed in this paper but they cannot guarantee well-formedness of the prediction.

[1] It is quadratic in the number of tags, but we assume the input of the algorithm is the sentence only.

3 Nested named-entity recognition

In this section, we introduce the nested NER problem and the vocabulary we use through the paper.

3.1 Notations and vocabulary

Let $\boldsymbol{s} = s_1 ... s_n$ be a sentence of n words. Without loss of generality, we assume that all sentences are of the same size. We use interstice (or fencepost) notation to refer to spans of $\boldsymbol{s}$, i.e. $\boldsymbol{s}_{i:j} = s_{i+1} ... s_j$ if $0 \leq i < j \leq n$, the empty sequence if $0 \leq i = j \leq n$ and undefined otherwise. We denote M the set of possible mentions in a sentence and T the set of mention types, e.g. $T = \{\textsc{Per}, \textsc{Org}, \textsc{Gpe}, ...\}$. Without loss of generality, we assume that $T \cap \{\rightarrow, \leftrightarrow, \mapsto, \mapsfrom\} = \emptyset$. A mention is denoted $\langle t, i, j\rangle \in M$ s.t. $t \in T, 0 \leq i < j \leq n$, where i (resp. j) is called the **left border** (resp. **right border**). An analysis of sentence $\boldsymbol{s}$ is denoted $\boldsymbol{y} \in \{0, 1\}^M$ where $y_m = 1$ (resp. $y_m = 0$) indicates that mention $m \in M$ is included in the analysis (resp. is not included). For example, the analysis of sentence 1 in Figure 1 is represented by a vector $\boldsymbol{y}$ where $y_{\langle \textsc{Per},0,1\rangle} = 1$, $y_{\langle \textsc{Per},5,8\rangle} = 1$ and all other elements are equal to zero. A mention $\langle t, i, j\rangle$ is said to be inside another mention $\langle t', i', j'\rangle$ iff $i' < i < j \leq j'$ or $i' \leq i < j < j'$.

Let $\boldsymbol{y}$ be the analysis of a sentence. We call **first level mentions** all mentions in $\boldsymbol{y}$ that are not inside another mention of the analysis. We call **nested mentions** all mentions that are not first level mentions. For example, the first level mentions of the analysis of sentence 2 in Figure 1 are $\langle \textsc{Per}, 0, 1\rangle$ “`this`” and $\langle \textsc{Per}, 2, 8\rangle$ “`your second one he has missed`”. We call **children** of mention $m \in M$ the set $C \subseteq M$ of mentions that are inside m but not inside another mention that is inside m. Conversely, m is said to be the parent of each mention in C. For example, in sentence 2 in Figure 1, the mention $\langle \textsc{Per}, 2, 8\rangle$ “`your second one he has missed`” has two children, $\langle \textsc{Per}, 2, 3\rangle$ “`your`” and $\langle \textsc{Per}, 5, 6\rangle$ “`he`”. In sentence 4 in Figure 1, $\langle \textsc{Gep}, 5, 6\rangle$ “`Germany`” is a child of $\langle \textsc{Gep}, 4, 6\rangle$ “`Bonn, Germany`” but it is not a child of $\langle \textsc{Per}, 2, 6\rangle$“`Fabian from Bonn, Germany`”. The **left neighborhood** (resp. **right neighborhood**) of a nested mention is the span between the left border of its parent and its left border (resp. between its right border and the right border of its parent). For example, in sentence 2 in Figure 1, mention $\langle \textsc{Per}, 5, 6\rangle$ “`he`” has left neighborhood $\boldsymbol{s}_{2:5}$ “`your second one`” and right neighborhood $\boldsymbol{s}_{6,8}$ “`has missed`”.

The set of possible analyses is denoted Y. We will consider three different definitions of Y:

1. the set of analyses where no disjoint mention spans overlap, corresponding to non-nested NER;
2. the set of analyses where one mention span can be inside another one but cannot partially overlap, corresponding to nested NER;
3. the set 2 with additional constraint that a mention must contain at most one child with a span length strictly greater to one.

3.2 Inference problems

The weight of an analysis $\boldsymbol{y} \in Y$ is defined as the sum of included mention weights. Let $\boldsymbol{w} \in \mathbb{R}^M$ be a vector of mention weights. The probability of an analysis is defined via the Boltzmann or “softmax” distributions:

$$p(\boldsymbol{y}|\boldsymbol{w}) = \frac{\exp(\boldsymbol{w}^\top \boldsymbol{y})}{Z(\boldsymbol{w})},$$

where $Z(\boldsymbol{w}) = \sum_{\boldsymbol{y}' \in Y} \exp(\boldsymbol{w}^\top \boldsymbol{y}')$ is the partition function. Note that, in general, the set Y is of exponential size but $Z(\boldsymbol{w})$ can nonetheless be efficiently computed via dynamic programming.

The training problem aims to minimize a loss function over the training data. We focus on the negative log-likelihood loss function defined as:

$$\ell(\boldsymbol{w}, \boldsymbol{y}) = -\boldsymbol{w}^\top \boldsymbol{y} + \log Z(\boldsymbol{w}).$$

Note that this loss function is convex in $\boldsymbol{w}$. This differentiates us from previous work that had to rely on non-convex losses (Fu et al., 2021; Lou et al., 2022). Moreover, note that the loss function used by Fu et al. (2021) and Lou et al. (2022) requires to compute the log-partition twice, one time with “normal” weights and one time with masked weights. The difference lays in the fact that we will use algorithms that are tailored for the considered search space Y whereas Fu et al. (2021) and Lou et al. (2022) introduced latent variables in order to be able to rely on algorithms designed for a different problem, namely syntactic constituency parsing. Note that the partial derivatives of $\log Z(\boldsymbol{w})$ are the marginal distributions of mentions (Wainwright et al., 2008). Hence, we will refer to computing $\log Z(\boldsymbol{w})$ and its derivatives as marginal inference, a required step for gradient based optimization at training time.

At test time, we aim to compute the highest scoring structure given weights $\boldsymbol{w}$:

$$\boldsymbol{y}^* = \arg\max_{\boldsymbol{y} \in Y} p(\boldsymbol{y}|\boldsymbol{w}) = \arg\max_{\boldsymbol{y} \in Y} \boldsymbol{w}^\top \boldsymbol{y}$$

We call this problem MAP inference.

For many problems in natural language processing, marginal inference and MAP inference can be computed via dynamic programming over different semirings (Goodman, 1999) or dynamic programming with smoothed max operators (Mensch and Blondel, 2018). However, we need to ensure the uniqueness of derivations property so that a single analysis $\boldsymbol{y} \in Y$ has exactly one possible derivation under the algorithm. Otherwise, the same analysis would be counted several times when computing the partition function, leading to an overestimation of its value.

4 Related algorithms

In this section, we present semi-Markov and CYK-like algorithms for non-nested and nested NER, respectively. Our presentation is based on the weighted logic programming formalism, also known as parsing-as-deduction (Pereira and Warren, 1983). We refer the reader to Kallmeyer (2010, Chapter 3) for an introduction to this formalism. The space and time complexities can be directly inferred by counting the maximum number of free variables in items and deduction rules, respectively. To the best of our knowledge, the presentation of the CYK-like algorithm is novel as previous work relied on the "actual" CYK algorithm (Finkel and Manning, 2009) or its variant for span-based syntactic parsing (Lou et al., 2022; Fu et al., 2021).

4.1 Non-nested named-entity recognition

The semi-Markov algorithm recognizes a sentence from left to right. Items are of the following forms:

- $[t, i, j]$ s.t. $t \in T$ and $0 \leq i < j \leq n$: represent the mention $\langle t, i, j \rangle$;
- $[\rightarrow, i]$ s.t. $0 \leq i \leq n$: represent a partial analysis of the sentence covering words $\boldsymbol{s}_{0:i}$.

Axioms are items of the form $[\rightarrow, 0]$ and $[t, i, j]$. The first axiom form represents an empty partial analysis and the second set of axioms represent all possible mentions in the sentence. We assign weight $w_{\langle t,i,j \rangle}$ to axiom $[t, i, j]$, for all $t \in T, i, j \in \mathbb{N}$ s.t. $0 \leq i < j \leq n$. The goal of the algorithm is the item $[\rightarrow, n]$.

Deduction rules are defined as follows:

$$\textbf{(a)} \frac{[\rightarrow, i] \quad [t, i, j]}{[\rightarrow, j]} \qquad \textbf{(b)} \frac{[\rightarrow, i-1]}{[\rightarrow, i]}$$

Rule **(a)** appends a mention spanning words $\boldsymbol{s}_{i:j}$ to a partial analysis, whereas rule **(b)** advances one position by assuming word $\boldsymbol{s}_{i:i+1}$ is not covered by a mention.

A trace example of the algorithm is given in Table 1. Soundness, completeness and uniqueness of derivations can be directly induced from the deduction system. The time and space complexities are both $\mathcal{O}(n^2|T|)$.

4.2 Nested named-entity recognition

We present a CYK-like algorithm for nested named entity recognition. Contrary to algorithms proposed by Finkel and Manning (2009) and Fu et al. (2021), *inter alia*, our algorithm directly recognizes the nested mentions and does not require any "trick" to take into account non-binary structures, words that are not covered by any mention or the fact that a word in a mention may not be covered by any of its children. As such, we present an algorithm that is tailored for NER instead of the usual "hijacking" of constituency parsing algorithms. This particular presentation of the algorithm will allow us to simplify the presentation of our novel contribution in Section 5.

Items are of the following forms:

- $[t, i, j]$ as defined previously;
- $[\rightarrow, i]$ as defined previously;
- $[\mapsto, i, j]$ with $0 \leq i < j \leq n$: represent the partial analysis of a mention and its nested structure starting at position i.
- $[\leftrightarrow, i, j]$ with $0 \leq i < j \leq n$: represent the full analysis of a mention spanning $\boldsymbol{s}_{i:j}$, including its internal structure (i.e. full analysis of its children).

Axioms and goals are the same as the ones of the semi-Markov algorithm presented in Section 4.1, with supplementary set of items of form $[\mapsto, i, i]$ that are used to start recognizing the internal structure of a mention starting at position i.

The algorithm consists of two steps. First, the internal structure of mentions are constructed in a bottom-up fashion. Second, first level mentions (and their internal structures) are recognized in a

	Items	Rules	Comments
1.	$[\rightarrow, 0]$	Axiom	Recognize mention “He”
2.	$[\textsc{Per}, 0, 1]$	Axiom	
3.	$[\rightarrow, 1]$	**(a)** with 1 and 2	
4.	$[\rightarrow, 2]$	**(b)** with 3	Recognize empty space between the two mentions
5.	$[\rightarrow, 3]$	**(b)** with 4	
6.	$[\rightarrow, 4]$	**(b)** with 5	
7.	$[\rightarrow, 5]$	**(b)** with 6	
8.	$[\textsc{Per}, 5, 8]$	Axiom	Recognize mention “a dead man”
9.	$[\rightarrow, 8]$	**(a)** with 7 and 8	

Table 1: Example of recognition trace with the semi-Markov algorithm on sentence 1.

	Items	Rules $\mathcal{O}(n^3)$	$\mathcal{O}(n^2)$	Comments
1.	$[\mapsto, 0, 0]$	Axiom	//	Recognize mention “This”
2.	$[\mapsto, 0, 1]$	**(d)** with 1	//	
3.	$[\textsc{Per}, 0, 1]$	Axiom	//	
4.	$[\leftrightarrow, 0, 1]$	**(g)** with 2 & 3	//	
5.	$[\mapsto, 2, 2]$	Axiom	//	Recognize mention “your”
6.	$[\mapsto, 2, 3]$	**(d)** with 5	//	
7.	$[\textsc{Per}, 2, 3]$	Axiom	//	
8.	$[\leftrightarrow, 2, 3]$	**(g)** with 7	//	
9.	$[\mapsto, 0, 5]$	Axiom	//	Recognize mention “he”
10.	$[\mapsto, 0, 6]$	**(d)** with 9	//	
11.	$[\textsc{Per}, 5, 6]$	Axiom	//	
12.	$[\leftrightarrow, 5, 6]$	**(g)** with 11	//	
13.	$[\mapsto, 2, 4]$	**(f)** with 8	//	Recognize mention “your second one he has missed”
14.	$[\mapsto, 2, 5]$	**(d)** with 13	//	
15.	$[\mapsto, 2, 6]$	**(c)** with 14 & 12	**(j)**	
16.	$[\mapsto, 2, 7]$	**(d)** with 15	//	
17.	$[\mapsto, 2, 8]$	**(d)** with 16	//	
18.	$[\textsc{Per}, 2, 8]$	Axiom	//	
19.	$[\leftrightarrow, 2, 8]$	**(g)** with 17 & 18	//	
20.	$[\rightarrow, 0]$	Axiom	//	Combine all first-level mentions
21.	$[\rightarrow, 1]$	**(h)** with 20 & 4	//	
22.	$[\rightarrow, 2]$	**(i)** with 21	//	
23.	$[\rightarrow, 8]$	**(h)** with 22 & 19	//	

Table 2: Example of recognition trace with the CYK-like and the proposed $\mathcal{O}(n^2)$ algorithm on sentence 2. There is only one rule that differs, but they both share the same antecedents.

	Items	Rules	Comments
1.	$[\mapsto, 0, 0]$	Axiom	Recognize mention “He”
2.	$[\mapsto, 0, 1]$	**(d)** with 1	
3.	$[\textsc{Per}, 0, 1]$	Axiom	
4.	$[\leftrightarrow, 0, 1]$	**(g)** with 2 and 3	
5.	$[\mapsto, 5, 5]$	Axiom	Recognize mention “United states”
6.	$[\mapsto, 5, 6]$	**(d)** with 5	
7.	$[\mapsto, 5, 7]$	**(d)** with 6	
8.	$[\textsc{Gpe}, 5, 7]$	Axiom	
9.	$[\leftrightarrow, 5, 7]$	**(a)** with 1 and 2	
10.	$[\mapsfrom, 4, 7]$	**(m)** with 9	Recognize mention “the United States army”
11.	$[\mapsto, 4, 7]$	**(p)** with 10	
11.	$[\mapsto, 4, 8]$	**(d)** with 11	
12.	$[\textsc{Org}, 4, 8]$	Axiom	
13.	$[\leftrightarrow, 4, 8]$	**(g)** with 11 and 12	
14.	$[\rightarrow, 0]$	Axiom	Combine all first-level mentions
15.	$[\rightarrow, 1]$	**(h)** with 14 and 4	
16.	$[\rightarrow, 2]$	**(i)** with 15	
17.	$[\rightarrow, 3]$	**(i)** with 16	
18.	$[\rightarrow, 4]$	**(i)** with 17	
19.	$[\rightarrow, 8]$	**(h)** with 18 & 13	

Table 3: Example of recognition trace of the proposed algorithm on sentence 3.

	Items	Rules	Comments
1.	$[\mapsto, 0, 0]$	Axiom	Recognize mention “I”
2.	$[\mapsto, 0, 1]$	**(d)** with 1	
3.	$[\textsc{Per}, 0, 1]$	Axiom	
4.	$[\leftrightarrow, 0, 1]$	**(g)** with 2 & 3	
5.	$[\mapsto, 5, 5]$	Axiom	Recognize mention “Germany”
6.	$[\mapsto, 5, 6]$	**(d)** with 5	
7.	$[\textsc{Gpe}, 5, 6]$	Axiom	
8.	$[\leftrightarrow, 5, 6]$	**(g)** with 6 & 7	
9.	$[\mapsto, 4, 4]$	Axiom	Recognize mention “Bonn, Germany”
10.	$[\mapsto, 4, 5]$	**(d)** with 9	
11.	$[\mapsto, 4, 6]$	**(j)** with 10 & 8	
12.	$[\textsc{Gpe}, 4, 6]$	Axiom	
13.	$[\leftrightarrow, 4, 6]$	**(g)** with 11 & 12	
14.	$[\mapsfrom, 3, 6]$	**(m)** with 13	Recognize mention ‘Fabian from Bonn, Germany”
15.	$[\mapsfrom, 2, 6]$	**(n)** with 14	
16.	$[\mapsto, 2, 6]$	**(p)** with 15	
17.	$[\textsc{Per}, 2, 6]$	Axiom	
18.	$[\leftrightarrow, 2, 6]$	**(g)** with 16 & 17	
19.	$[\rightarrow, 0]$	Axiom	Combine all first-level mentions
20.	$[\rightarrow, 1]$	**(h)** with 19 & 4	
21.	$[\rightarrow, 2]$	**(i)** with 10	
22.	$[\rightarrow, 6]$	**(h)** with 21 & 18	

Table 4: Example of recognition trace of the proposed algorithm on sentence 4.

similar fashion to the semi-Markov model. The deduction rules for bottom-up construction are defined as follows:

$$\textbf{(c)}\ \frac{[\mapsto, i, k]\quad [\leftrightarrow, k, j]}{[\mapsto, i, j]}\ \text{i < k} \qquad \textbf{(d)}\ \frac{[\mapsto, i, j-1]}{[\mapsto, i, j]}$$

$$\textbf{(e)}\ \frac{[\leftrightarrow, i, k]\quad [\leftrightarrow, k, j]}{[\mapsto, i, j]} \qquad \textbf{(f)}\ \frac{[\leftrightarrow, i, j-1]}{[\mapsto, i, j]}$$

$$\textbf{(g)}\ \frac{[\mapsto, i, j]\quad [t, i, j]}{[\leftrightarrow, i, j]}\ \text{i < j}$$

Rule **(c)** concatenates an analyzed mention to a partial analysis of another mention — note that the constraint forbids that right antecedent shares its left border with its parent. Rule **(d)** advances of one position in the partial structure, assuming the analyzed mention starting at i does not have a child mention covering $s_{j-1:j}$. Rules **(e)** and **(f)** are used to recognize the internal structure of a mention that has a child sharing the same left border. Although the latter two deduction rules may seem far-fetched, they cannot be simplified without breaking the uniqueness of derivations property or breaking the prohibition of self loop construction of $\leftrightarrow$ items. Finally, rule **(g)** finishes the analysis of a mention and its internal structure.

Note that this construction is highly similar to the dotted rule construction in the Earley algorithm (Earley, 1970). Moreover, contrary to Stern et al. (2017), we do not introduce null labels for implicit binarization. The benefit of our approach is that there is no spurious ambiguity in the algorithm, i.e. we guaranty uniqueness of derivations. Therefore, we can use the same deduction rules to compute the log-partition function of the negative log-likelihood loss. This is not the case of the approach of Stern et al. (2017), which forces them to rely on a structured hinge loss.

Deduction rules for the second step are defined as follows:

$$\textbf{(h)}\ \frac{[\rightarrow, i]\quad [\leftrightarrow, i, j]}{[\rightarrow, j]} \qquad \textbf{(i)}\ \frac{[\rightarrow, i-1]}{[\rightarrow, i]}$$

They have similar interpretation to the rules of the semi-Markov model where we replaced mentions by possibly nested structures.

A trace example of the algorithm is given in Table 2. Although the algorithm is more involved than usual presentations, our approach directly maps a derivation to nested mentions and guarantee uniqueness of derivations. The space and time complexities are $\mathcal{O}(n^2|T|)$ and $\mathcal{O}(n^3|T|)$, respectively.

5 $\mathcal{O}(n^2)$ nested named-entity recognition

In this section, we describe our novel algorithm for quadratic-time nested named entity recognition. Our algorithm limits its search space to mentions that contain at most one child of length strictly greater to one.

Items are of the following forms:

- $[t, i, j]$ as defined previously;
- $[\rightarrow, i]$ as defined previously;
- $[\mapsto, i, j]$ as defined previously;
- $[\leftrightarrow, i, j]$ as defined previously;
- $[\mapsfrom, i, j]$ with $0 \leq i < j \leq 0$: represents a partial analysis of a mention and its internal structure, where its content will be recognized by appending content on the left instead of the right.

Axioms and goals are the same than the one of the CYK-like algorithm presented in Section 4.2 — importantly, there is no extra axiom for items of the form $[\mapsfrom, i, j]$.

For the moment, assume we restrict nested mentions that have a length strictly greater to the ones that share their left boundaries with their parent. We can re-use rules **(d)**, **(f)**, **(g)**, **(h)** and **(i)** together with the following two deduction rules:

$$\textbf{(j)}\ \frac{[\mapsto, i, j-1]\quad [\leftrightarrow, j-1, j]}{[\mapsto, i, j]}\ i < j-1$$

$$\textbf{(k)}\ \frac{[\leftrightarrow, i, j-1]\quad [\leftrightarrow, j-1, j]}{[\mapsto, i, j]}$$

More precisely, we removed the two rules inducing a cubic-time complexity in the CYK-like algorithm and replaced them with quadratic-time rules. This transformation is possible because our search space forces the rightmost antecedents of these two rules to cover a single word, hence we do not need to introduce an extra free variable. However, in this form, the algorithm only allows the child mention of length strictly greater to one to share its left boundary with its parent.

We now extend the algorithm to the full targeted search space. The intuition is as follows: for a given mention, if it has a child mention of length strictly greater than one that does not share its left border with its parent, we first start recognizing this child mention and its left neighborhood and then move to right neighborhood using previously defined rules. We start the recognition of the left neighborhood using the two following rules:

$$\textbf{(l)}\ \frac{[\leftrightarrow, i, i+1] \quad [\leftrightarrow, i+1, j]}{[\mapsfrom, i, j]}\ i+2<j$$

$$\textbf{(m)}\ \frac{[\leftrightarrow, i+1, j]}{[\mapsfrom, i, j]}\ i+2<j$$

where the constraints ensure antecedents $[\leftrightarrow, i+1, j]$ are non-unary (otherwise we will break the uniqueness of derivations constraint). Rule **(l)** (resp. **(m)**) recognizes the case where span $s_{i:i+1}$ contains (resp. does not contain) a mention. The following rules are analogous to rules **(d)** and **(j)** but for visiting the left neighborhood instead of the right one:

$$\textbf{(n)}\ \frac{[\mapsfrom, i+1, j]}{[\mapsfrom, i, j]} \quad \textbf{(o)}\ \frac{[\leftrightarrow, i, i+1] \quad [\mapsfrom, i+1, j]}{[\mapsfrom, i, j]}$$

Finally, once the left neighborhood has been recognized, we move to the right one using the following rule:

$$\textbf{(p)}\ \frac{[\mapsfrom, i, j]}{[\mapsto, i, j]}$$

Using the aforementioned rules, our algorithm has time and space complexities of $\mathcal{O}(n^2|T|)$. We illustrate the difference with the CYK-like algorithm with a trace example in Table 2: in this specific example, the two analyses differ only by the application of a single rule. Table 3 contains a trace example where all nested mentions have a size one, so the parent mention is visited from left to right. Table 4 contains a trace example where we need to construct one internal structure by visiting the left neighborhood of the non-unary child mention from right to left.

Soundness and completeness can be proved by observing that, for a given mention, any children composition can be parsed with deduction rules as long as there is at most one child with a span strictly greater to one. Moreover, these are the only children composition that can be recognized. Finally, uniqueness of derivations can be proved as there is a single construction order of the internal structure of a mention.

Infinite recursion. An important property of our algorithm is that it does not bound the number of allowed recursively nested mentions. For example, consider the phrase "*[Chair of [the Committee of [Ministers of [the Council of [Europe]]]]]*". Not only can this nested mention structure be recognized by our algorithm, but any supplementary "of" precision would also be recognized.

	ACE-2004	ACE-2005	GENIA
Non-nested $\mathcal{O}(n^2)$	78.19	80.89	91.21
Nested $\mathcal{O}(n^3)$	99.97	99.96	99.95
Nested $\mathcal{O}(n^2)$	98.92	99.31	99.83

Table 5: Maximum recall that can be achieved on the full datasets (train, dev and test) for the three algorithms.

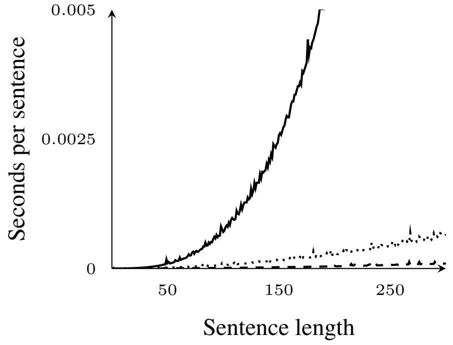

Figure 2: MAP decoding time (dynamic programming algorithm only) in seconds on an Intel Core i5 (2.4 GHz) processor for sentences of lengths 1 to 300. **(dashed)** quadratic-time semi-markov algorithm. **(solid)** CYK-like cubic-time algorithm. **(dotted)** quadratic-time algorithm proposed in this paper.

Possible extension. Note that we could extend the algorithm so that we allow each mention to have at most one child of length strictly greater to a predefined constant c, and other children should have a length less or equal to c. However, as fixing $c = 1$ results in a good cover of datasets we use, we do not consider this extension in this work.

6 Experimental results

Data. We evaluate our algorithms on the ACE-2004 (Doddington et al., 2004), ACE-2005 (Walker et al., 2006) and GENIA (Kim et al., 2003) datasets. We split and pre-process the data using the tools distributed by Shibuya and Hovy (2020).

Data coverage. As our parsing algorithm considers a restricted search space, an important question is whether it has a good coverage of NER datasets. Table 5 shows the maximum recall we can achieve with the algorithms presented in this paper. Note that no algorithm achieve a coverage of 100% as there is a small set of mentions with exactly the same span[2] and mentions that overlap partially. We observe that the loss of coverage for our quadratic-

[2] This can be easily fixed by collpasing theses mentions, a standard trick used in the constituency parsing literature, see (Stern et al., 2017)

	Dataset								
	ACE-2004			ACE-2005			GENIA		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
Comparable models based on BERT									
Shibuya and Hovy (2020)	85.23	84.72	84.97	83.30	84.69	83.99	77.46	76.65	77.05
Wang et al. (2020)	86.08	86.48	86.28	83.95	85.39	84.66	79.45	78.94	79.19
Wang et al. (2021, max)	86.27	85.09	85.68	85.28	84.15	84.71	79.20	78.16	78.67
Fu et al. (2021)†	87.62	87.57	87.60	83.34	85.67	84.49	79.10	76.53	77.80
Tan et al. (2021)†	87.05	86.26	86.65	83.92	84.75	84.33	78.33	76.66	77.48
Shen et al. (2021)†	87.27	86.61	86.94	86.02	85.62	85.82	76.80	79.02	77.89
Yan et al. (2021)	87.27	86.41	86.84	83.16	86.38	84.74	78.57	79.30	78.93
Model based on BERT with lexicalization									
Lou et al. (2022)	87.39	88.40	87.90	85.97	87.87	86.91	78.39	78.50	78.44
This work									
Semi-Markov algorithm, $\mathcal{O}(n^2)$	89.06	68.63	77.52	84.39	68.51	75.63	80.87	71.37	75.82
CYK-like algorithm, $\mathcal{O}(n^3)$	87.18	86.06	86.62	84.20	85.84	85.01	79.20	77.31	78.24
Proposed algorithm, $\mathcal{O}(n^2)$	87.37	85.04	86.19	84.42	85.28	84.85	79.28	77.25	78.25

Table 6: Precision, recall and F1-measure results. We compare ourselves to other BERT-based models — some of the cited papers includes richer models that we omit for brievity as our goal is only to asses the performance of our algorithm compared to the CYK-like one. Results marked with † are the reproduction of Lou et al. (2022) as the original papers experimented on different data splits.

time algorithm is negligible compared to the cubic-time algorithm for all datasets.

Timing. We implemented the three algorithms in C++ and compare their running time for MAP inference in Figure 2. The proposed algorithm is way faster than the CYK-like. If we would parse only sentences of 300 words and we only consider the time spend in the decoding algorithm (i.e. ignoring the forward pass in the neural network), the CYK-like algorithm couldn't even decode 50 sentences in a second whereas our algorithm could decode more than 1500 sentences on an Intel Core i5 (2.4 GHz) processor. As such, we hope that our algorithm will allow future work to consider NER on longer spans of text.

Neural architecture and hyperparameters. Our neural network is composed of a finetuned BERT model[3] (Devlin et al., 2019) followed by 3 bidirectional LSTM layers (Hochreiter and Schmidhuber, 1997) with a hidden size of 400. When the BERT tokenizer splits a word, we use the output embedding of the the first token. Mention weights (i.e. values in vector $\boldsymbol{w}$) are computed using two biaffine layers (Dozat and Manning, 2017), one labeled and one unlabeled, with independent left and right projections of dimension 500 and RELU activation functions.

We use a negative log-liklihood loss (i.e. CRF loss) with 0.1-label smoothing (Szegedy et al., 2016). The learning rate is 1×10^{-5} for BERT parameters and 1×10^{-3} for other parameters. We use an exponential decay scheduler for learning rates (decay rate of 0.75 every 5000 steps). We apply dropout with probability of 0.1 at the output of BERT, LSTM layers and projection layers. We keep the parameters that obtains the best F1-measure on development data after 20 epochs.

Results. We report experimental results in Table 6. Note that our goal is not to establish a novel SOTA for the task but to assess whether our quadratic-time algorithm is well-suited for the nested NER problem, therefore we only compare our models with recent work using the same datasplit and comparable neural architectures (i.e. BERT-based and without lexicalization). **Any method that modifies the cubic-time parser to improve results can be similarly introduced in our parser.** Our implementation of the CYK-like cubic-time parser obtains results close to comparable work in the literature. Importantly, we observe that, with the proposed quadratic-time algorithm, F1-measure results are (almost) the same on GENIA and the the degradation is negligible on ACE-2004 and ACE-2005 (the F1-measure decreases by less than 0.5).

[3] *bert-base-uncased* as distributed at `https://huggingface.co/bert-base-uncased`

7 Conclusion

In this work, we proposed a novel quadratic-time parsing algorithm for nested NER, an asymptotic improvement of one order of magnitude over previously proposed span-based algorithms. We showed that the novel search-space has a good coverage of English datasets for nested NER. Despite having the same time-complexity than semi-Markov models, our approach achieves comparable experimental results to the cubic-time CYK-like algorithm.

As such, we hope that our algorithm will be used as a drop-in fast replacement for future work in nested NER, where the cubic-time algorithm has often been qualified of slow. Future work could consider the extension to lexicalized mentions.

Limitations

An obvious limitation of our work is the considered search space. Although we showed that it is well suited for the data used in practice by the NLP community, this may not hold in more general settings.

Moreover, we only experiment in English. We suspect that similar results would hold for morphologically-rich languages as we expect, in the latter case, that constituents are shorter (i.e. morphologically-rich languages heavily rely on morphological inflection, so we expect more mentions spanning a single word), see (Haspelmath and Sims, 2013, Section 1.2 and Table 1.1). However, this is not guaranteed and future work needs to explore the multilingual setting.

Finally, in this work we do not consider discontinuous mentions, which is an important setting in real world scenario.

Acknowledgments

We thank François Yvon, Songlin Yang and the anonymous reviewers for their comments and suggestions. This work benefited from computations done on the Saclay-IA platform and on the HPC resources of IDRIS under the allocation 2022-AD011013727 made by GENCI.

I apologize to the reviewers for not adding the supplementary experimental results: being the sole author of this article, I unfortunately did not find the time to do so.

References

Beatrice Alex, Barry Haddow, and Claire Grover. 2007. Recognising nested named entities in biomedical text. In *Biological, translational, and clinical language processing*, pages 65–72, Prague, Czech Republic. Association for Computational Linguistics.

Ravneet Arora, Chen-Tse Tsai, Ketevan Tsereteli, Prabhanjan Kambadur, and Yi Yang. 2019. A semi-Markov structured support vector machine model for high-precision named entity recognition. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5862–5866, Florence, Italy. Association for Computational Linguistics.

John Cocke. 1970. Programming languages and their compilers: Preliminary notes.

Caio Corro. 2020. Span-based discontinuous constituency parsing: a family of exact chart-based algorithms with time complexities from O(n^6) down to O(n^3). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2753–2764, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

George Doddington, Alexis Mitchell, Mark Przybocki, Lance Ramshaw, Stephanie Strassel, and Ralph Weischedel. 2004. The automatic content extraction (ACE) program – tasks, data, and evaluation. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal. European Language Resources Association (ELRA).

Timothy Dozat and Christopher D Manning. 2017. Deep biaffine attention for neural dependency parsing. In *Proceedings of the International Conference of Representation Learning (ICLR)*.

Jay Earley. 1970. An efficient context-free parsing algorithm. *Communications of the ACM*, 13(2):94–102.

Jenny Rose Finkel and Christopher D. Manning. 2009. Nested named entity recognition. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 141–150, Singapore. Association for Computational Linguistics.

Yao Fu, Chuanqi Tan, Mosha Chen, Songfang Huang, and Fei Huang. 2021. Nested named entity recognition with partially-observed treecrfs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):12839–12847.

Xianping Ge. 2002. Segmental semi-markov models and applications to sequence analysis.

Carlos Gómez-Rodríguez, Marco Kuhlmann, and Giorgio Satta. 2010. Efficient parsing of well-nested linear context-free rewriting systems. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 276–284, Los Angeles, California. Association for Computational Linguistics.

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

David Hall, Greg Durrett, and Dan Klein. 2014. Less grammar, more features. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 228–237, Baltimore, Maryland. Association for Computational Linguistics.

Martin Haspelmath and Andrea Sims. 2013. *Understanding morphology*. Routledge.

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

Jacques Janssen and Nikolaos Limnios. 1999. *Semi-Markov models and applications*. Kluwer Academic.

Meizhi Ju, Makoto Miwa, and Sophia Ananiadou. 2018. A neural layered model for nested named entity recognition. 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 1446–1459, New Orleans, Louisiana. Association for Computational Linguistics.

Laura Kallmeyer. 2010. *Parsing Beyond Context-Free Grammars*. Springer Science & Business Media.

Tadao Kasami. 1965. An efficient recognition and syntax-analysis algorithm for context-free languages.

Arzoo Katiyar and Claire Cardie. 2018. Nested named entity recognition revisited. 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 861–871, New Orleans, Louisiana. Association for Computational Linguistics.

J.-D. Kim, T. Ohta, Y. Tateisi, and J. Tsujii. 2003. GENIA corpus—a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(suppl):i180–i182.

Chao Lou, Songlin Yang, and Kewei Tu. 2022. Nested named entity recognition as latent lexicalized constituency parsing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6183–6198, Dublin, Ireland. Association for Computational Linguistics.

Wei Lu and Dan Roth. 2015. Joint mention extraction and classification with mention hypergraphs. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 857–867, Lisbon, Portugal. Association for Computational Linguistics.

Arthur Mensch and Mathieu Blondel. 2018. Differentiable dynamic programming for structured prediction and attention. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 3462–3471. PMLR.

Aldrian Obaja Muis and Wei Lu. 2017. Labeling gaps between words: Recognizing overlapping mentions with mention separators. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2608–2618, Copenhagen, Denmark. Association for Computational Linguistics.

Fernando C. N. Pereira and David H. D. Warren. 1983. Parsing as deduction. In *21st Annual Meeting of the Association for Computational Linguistics*, pages 137–144, Cambridge, Massachusetts, USA. Association for Computational Linguistics.

Emily Pitler, Sampath Kannan, and Mitchell Marcus. 2012. Dynamic programming for higher order parsing of gap-minding trees. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 478–488, Jeju Island, Korea. Association for Computational Linguistics.

Emily Pitler, Sampath Kannan, and Mitchell Marcus. 2013. Finding optimal 1-endpoint-crossing trees. *Transactions of the Association for Computational Linguistics*, 1:13–24.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, Boulder, Colorado. Association for Computational Linguistics.

Sunita Sarawagi and William W Cohen. 2004. Semi-markov conditional random fields for information extraction. In *Advances in Neural Information Processing Systems*, volume 17. MIT Press.

Giorgio Satta and Marco Kuhlmann. 2013. Efficient parsing for head-split dependency trees. *Transactions of the Association for Computational Linguistics*, 1:267–278.

Yongliang Shen, Xinyin Ma, Zeqi Tan, Shuai Zhang, Wen Wang, and Weiming Lu. 2021. Locate and label: A two-stage identifier for nested named entity recognition. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2782–2794, Online. Association for Computational Linguistics.

Takashi Shibuya and Eduard Hovy. 2020. Nested named entity recognition via second-best sequence learning and decoding. *Transactions of the Association for Computational Linguistics*, 8:605–620.

Mohammad Golam Sohrab and Makoto Miwa. 2018. Deep exhaustive model for nested named entity recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2843–2849, Brussels, Belgium. Association for Computational Linguistics.

Mitchell Stern, Jacob Andreas, and Dan Klein. 2017. A minimal span-based neural constituency parser. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 818–827, Vancouver, Canada. Association for Computational Linguistics.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826.

Chuanqi Tan, Wei Qiu, Mosha Chen, Rui Wang, and Fei Huang. 2020. Boundary enhanced neural span classification for nested named entity recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):9016–9023.

Zeqi Tan, Yongliang Shen, Shuai Zhang, Weiming Lu, and Yueting Zhuang. 2021. A sequence-to-set network for nested named entity recognition. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 3936–3942. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Ioannis Tsochantaridis, Thomas Hofmann, Thorsten Joachims, and Yasemin Altun. 2004. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the twenty-first international conference on Machine learning*, page 104.

Martin J Wainwright, Michael I Jordan, et al. 2008. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1–2):1–305.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. Ace 2005 mul tilingual training corpus.

Bailin Wang and Wei Lu. 2018. Neural segmental hypergraphs for overlapping mention recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 204–214, Brussels, Belgium. Association for Computational Linguistics.

Jue Wang, Lidan Shou, Ke Chen, and Gang Chen. 2020. Pyramid: A layered model for nested named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5918–5928, Online. Association for Computational Linguistics.

Yiran Wang, Hiroyuki Shindo, Yuji Matsumoto, and Taro Watanabe. 2021. Nested named entity recognition via explicitly excluding the influence of the best path. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3547–3557, Online. Association for Computational Linguistics.

Congying Xia, Chenwei Zhang, Tao Yang, Yaliang Li, Nan Du, Xian Wu, Wei Fan, Fenglong Ma, and Philip Yu. 2019. Multi-grained named entity recognition. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1430–1440, Florence, Italy. Association for Computational Linguistics.

Mingbin Xu, Hui Jiang, and Sedtawut Watcharawittayakul. 2017. A local detection approach for named entity recognition and mention detection. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1237–1247, Vancouver, Canada. Association for Computational Linguistics.

Hang Yan, Tao Gui, Junqi Dai, Qipeng Guo, Zheng Zhang, and Xipeng Qiu. 2021. A unified generative framework for various NER subtasks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5808–5822, Online. Association for Computational Linguistics.

Daniel H. Younger. 1967. Recognition and parsing of context-free languages in time n3. *Information and Control*, 10(2):189–208.

Urchade Zaratiana, Nadi Tomeh, Pierre Holat, and Thierry Charnois. 2022. GNNer: Reducing overlapping in span-based NER using graph neural networks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 97–103, Dublin, Ireland. Association for Computational Linguistics.

Changmeng Zheng, Yi Cai, Jingyun Xu, Ho-fung Leung, and Guandong Xu. 2019. A boundary-aware neural model for nested named entity recognition. 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 357–366, Hong Kong, China. Association for Computational Linguistics.