A Unified Generative Framework for Various NER Subtasks

Hang Yan[1], Tao Gui[2], Junqi Dai[1], Qipeng Guo[1], Zheng Zhang[3], Xipeng Qiu[1,4]*
[1]Shanghai Key Laboratory of Intelligent Information Processing, Fudan University
[1]School of Computer Science, Fudan University
[2]Institute of Modern Languages and Linguistics, Fudan University
[3]New York University
[4]Pazhou Lab, Guangzhou, China
{hyan19,tgui16,jqdai19,qpguo16,xpqiu}@fudan.edu.cn
zz@nyu.edu

Abstract

Named Entity Recognition (NER) is the task of identifying spans that represent entities in sentences. Whether the entity spans are nested or discontinuous, the NER task can be categorized into the flat NER, nested NER, and discontinuous NER subtasks. These subtasks have been mainly solved by the token-level sequence labelling or span-level classification. However, these solutions can hardly tackle the three kinds of NER subtasks concurrently. To that end, we propose to formulate the NER subtasks as an entity span sequence generation task, which can be solved by a unified sequence-to-sequence (Seq2Seq) framework. Based on our unified framework, we can leverage the pre-trained Seq2Seq model to solve all three kinds of NER subtasks without the special design of the tagging schema or ways to enumerate spans. We exploit three types of entity representations to linearize entities into a sequence. Our proposed framework is easy-to-implement and achieves state-of-the-art (SoTA) or near SoTA performance on eight English NER datasets, including two flat NER datasets, three nested NER datasets, and three discontinuous NER datasets [1].

1 Introduction

Named entity recognition (NER) has been a fundamental task of Natural Language Processing (NLP), and three kinds of NER subtasks have been recognized in previous work (Sang and Meulder, 2003; Pradhan et al., 2013a; Doddington et al., 2004; Kim et al., 2003; Karimi et al., 2015), including flat NER, nested NER, and discontinuous NER. As shown in Figure 1, the nested NER contains overlapping entities, and the entity in the discontinuous NER may contain several nonadjacent spans.

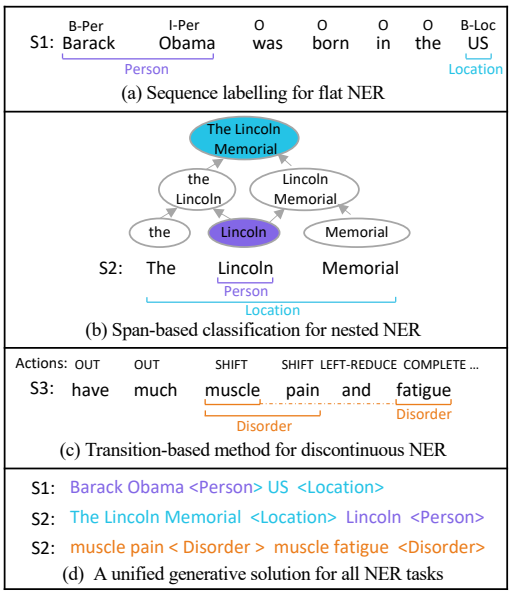

Figure 1: Examples of three kinds of NER subtasks. (a) - (c) illustrate flat NER, nested NER, discontinuous NER, and their corresponding mainstream solutions respectively. (d) Our proposed generative solution to solve all NER subtasks in a unified way.

The sequence labelling formulation, which will assign a tag to each token in the sentence, has been widely used in the flat NER field (McCallum and Li, 2003; Collobert et al., 2011; Huang et al., 2015; Chiu and Nichols, 2016; Lample et al., 2016; Straková et al., 2019; Yan et al., 2019; Li et al., 2020a). Inspired by sequence labelling's success in the flat NER subtask, Metke-Jimenez and Karimi (2016); Muis and Lu (2017) tried to formulate the nested and discontinuous NER into the sequence labelling problem. For the nested and discontinuous NER subtasks, instead of assigning labels to each token directly, Xu et al. (2017); Wang and Lu (2019); Yu et al. (2020); Li et al. (2020b) tried to enumerate all possible spans and conduct the span-level classification. Another way to efficiently represent spans is to use the hypergraph (Lu

*Corresponding author.
[1]Code is available at https://github.com/yhcc/BARTNER.

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pages 5808–5822
August 1–6, 2021.

and Roth, 2015; Katiyar and Cardie, 2018; Wang and Lu, 2018; Muis and Lu, 2016).

Although the sequence labelling formulation has dramatically advanced the NER task, it has to design different tagging schemas to fit various NER subtasks. One tagging schema can hardly fit for all three NER subtasks[2] (Ratinov and Roth, 2009; Metke-Jimenez and Karimi, 2016; Straková et al., 2019; Dai et al., 2020). While the span-based models need to enumerate all possible spans, which is quadratic to the length of the sentence and is almost impossible to enumerate in the discontinuous NER scenario (Yu et al., 2020). Therefore, span-based methods usually will set a maximum span length (Xu et al., 2017; Luan et al., 2019; Wang and Lu, 2018). Although hypergraphs can efficiently represent all spans (Lu and Roth, 2015; Katiyar and Cardie, 2018; Muis and Lu, 2016), it suffers from the spurious structure problem, and structural ambiguity issue during inference and the decoding is quite complicated (Muis and Lu, 2017). Because the problems lie in different formulations, no publication has tested their model or framework in three NER subtasks simultaneously to the best of our knowledge.

In this paper, we propose using a novel and simple sequence-to-sequence (Seq2Seq) framework with the pointer mechanism (Vinyals et al., 2015) to generate the entity sequence directly. On the source side, the model inputs the sentence, and on the target side, the model generates the entity pointer index sequence. Since flat, continuous and discontinuous entities can all be represented as entity pointer index sequences, this formulation can tackle all the three kinds of NER subtasks in a unified way. Besides, this formulation can even solve the crossing structure entity[3] and multi-type entity[4]. By converting the NER task into a Seq2Seq generation task, we can smoothly use the Seq2Seq pre-training model BART (Lewis et al., 2020) to enhance our model. To better utilize the pre-trained BART, we propose three kinds of entity representations to linearize entities into entity pointer index sequences.

Our contribution can be summarized as follows:

- We propose a novel and simple generative solution to solve the flat NER, nested NER, and discontinuous NER subtasks in a unified framework, in which NER subtasks are formulated as an entity span sequence generation problem.
- We incorporate the pre-trained Seq2Seq model BART into our framework and exploit three kinds of entity representations to linearize entities into sequences. The results can shed some light on further exploration of BART into the entity sequence generation.
- The proposed framework not only avoids the sophisticated design of tagging schema or span enumeration but also achieves SoTA or near SoTA performance on eight popular datasets, including two flat NER datasets, three nested NER datasets, and three discontinuous NER datasets.

2 Background

2.1 NER Subtasks

The term "Named Entity" was coined in the Sixth Message Understanding Conference (MUC-6) (Grishman and Sundheim, 1996). After that, the release of CoNLL-2003 NER dataset has greatly advanced the flat NER subtask (Sang and Meulder, 2003). Kim et al. (2003) found that in the field of molecular biology domain, some entities could be nested. Karimi et al. (2015) provided a corpus that contained medical forum posts on patient-reported Adverse Drug Events (ADEs), some entities recognized in this corpus may be discontinuous. Despite the difference between the three kinds of NER subtasks, the methods adopted by previous publications can be roughly divided into three types.

Token-level classification The first line of work views the NER task as a token-level classification task, which assigns to each token a tag that usually comes from the Cartesian product between entity labels and the tag scheme, such as BIO and BILOU (Ratinov and Roth, 2009; Collobert et al., 2011; Huang et al., 2015; Chiu and Nichols, 2016; Lample et al., 2016; Alex et al., 2007; Straková et al., 2019; Metke-Jimenez and Karimi, 2016; Muis and Lu, 2017; Dai et al., 2020), then Conditional Random Fields (CRF) (Lafferty et al., 2001) or tag sequence generation methods can be used for decoding. Though the work of (Straková et al., 2019; Wang et al., 2019; Zhang et al., 2018; Chen and Moschitti, 2018) are much like our method, they all

[2] Attempts made for discontinuous constituent parsing may tackle three NER subtasks in one tagging schema (Vilares and Gómez-Rodríguez, 2020).

[3] Namely, for span ABCD, both ABC and BCD are entities. Although this is rare, it exists (Dai et al., 2020).

[4] An entity can have multiple entity types, as proteins can be annotated as drug/compound in the EPPI corpus (Alex et al., 2007).

tried to predict a tagging sequence. Therefore, they still need to design tagging schemas for different NER subtasks.

Span-level classification When applying the sequence labelling method to the nested NER and discontinous NER subtasks, the tagging will be complex (Straková et al., 2019; Metke-Jimenez and Karimi, 2016) or multi-level (Ju et al., 2018; Fisher and Vlachos, 2019; Shibuya and Hovy, 2020). Therefore, the second line of work directly conducted the span-level classification. The main difference between publications in this line of work is how to get the spans. Finkel and Manning (2009) regarded the parsing nodes as a span. Xu et al. (2017); Luan et al. (2019); Yamada et al. (2020); Li et al. (2020b); Yu et al. (2020); Wang et al. (2020a) tried to enumerate all spans. Following Lu and Roth (2015), hypergraph methods which can effectively represent exponentially many possible nested mentions in a sentence have been extensively studied in the NER tasks (Katiyar and Cardie, 2018; Wang and Lu, 2018; Muis and Lu, 2016).

Combined token-level and span-level classification To avoid enumerating all possible spans and incorporate the entity boundary information into the model, Wang and Lu (2019); Zheng et al. (2019); Lin et al. (2019); Wang et al. (2020b); Luo and Zhao (2020) proposed combining the token-level classification and span-level classification.

2.2 Sequence-to-Sequence Models

The Seq2Seq framework has been long studied and adopted in NLP (Sutskever et al., 2014; Cho et al., 2014; Luong et al., 2015; Vaswani et al., 2017; Vinyals et al., 2015). Gillick et al. (2016) proposed a Seq2Seq model to predict the entity's start, span length and label for the NER task. Recently, the amazing performance gain achieved by PTMs (pre-trained models) (Qiu et al., 2020; Peters et al., 2018; Devlin et al., 2019; Dai et al., 2021; Yan et al., 2020) has attracted several attempts to pre-train a Seq2Seq model (Song et al., 2019; Lewis et al., 2020; Raffel et al., 2020). We mainly focus on the newly proposed BART (Lewis et al., 2020) model because it can achieve better performance than MASS (Song et al., 2019). And the sentence-piece tokenization used in T5 (Raffel et al., 2020) will cause different tokenizations for the same token, making it hard to generate pointer indexes to conduct the entity extraction.

BART is formed by several transformer encoder and decoder layers, like the transformer model used in the machine translation (Vaswani et al., 2017). BART's pre-training task is to recover corrupted text into the original text. BART uses the encoder to input the corrupted sentence and the decoder to recover the original sentence. BART has base and large versions. The base version has 6 encoder layers and 6 decoder layers, while the large version has 12. Therefore, the number of parameters is similar to its equivalently sized BERT [5].

3 Proposed Method

In this part, we first introduce the task formulation, then we describe how we use the Seq2Seq model with the pointer mechanism to generate the entity index sequences. After that, we present the detailed formulation of our model with BART.

3.1 NER Task

The three kinds of NER tasks can all be formulated as follows, given an input sentence of n tokens $X = [x_1, x_2, ..., x_n]$, the target sequence is $Y = [s_{11}, e_{11}, ..., s_{1j}, e_{1j}, t_1, ..., s_{i1}, e_{i1}, ..., s_{ik}, e_{ik}, t_i]$, where s, e are the start and end index of a span, since an entity may contain one (for flat and nested NER) or more than one (for discontinuous NER) spans, each entity is represented as $[s_{i1}, e_{i1}, ..., s_{ij}, e_{ij}, t_i]$, where t_i is the entity tag index. We use $G = [g_1, ..., g_l]$ to denote the entity tag tokens (such as "Person", "Location", etc.), where l is the number of entity tags. We make $t_i \in (n, n + l]$, the n shift is to make sure t_i is not confusing with pointer indexes (pointer indexes will be in range $[1, n]$).

3.2 Seq2Seq for Unified Decoding

Since we formulate the NER task in a generative way, we can view the NER task as the following equation:

$$P(Y|X) = \prod_{t=1}^{m} P(y_t|X, Y_{<t}) \quad (1)$$

where y_0 is the special "start of sentence" control token.

We use the Seq2Seq framework with the pointer mechanism to tackle this task. Therefore, our model consists of two components:

[5] Because of the cross-attention between encoder and decoder, the number of parameters of BART is about 10% larger than its equivalently sized of BERT (Lewis et al., 2020).

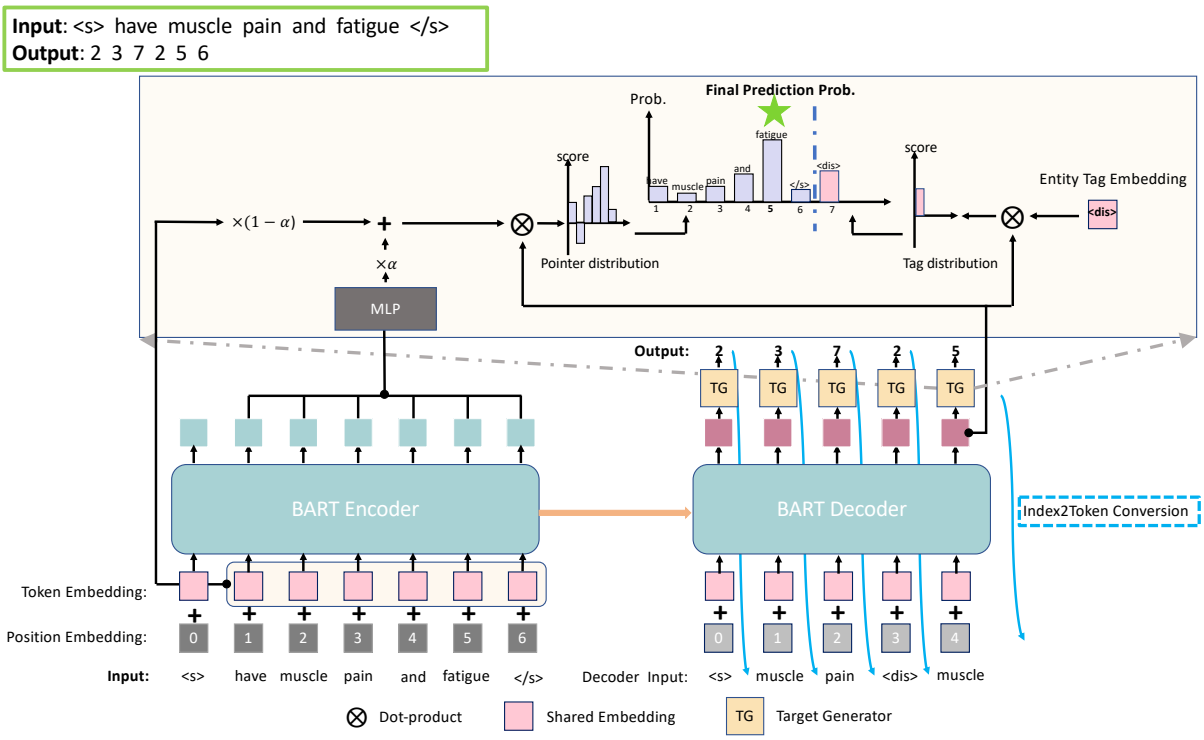

Figure 2: Model structure used in our method. The encoder encodes input sentences, and the decoder uses the pointer mechanism to generate indexes autoregressively. "<s>" and "</s>" are the predefined start-of-sentence and end-of-sentence tokens in BART. In the output sequence, "7" means the entity tag "<dis>", and other numbers indicate the pointer index (in range [1, 6]).

(1) **Encoder** encodes the input sentence X into vectors $\mathbf{H}^e$, which formulates as follows:

$$\mathbf{H}^e = \text{Encoder}(X) \tag{2}$$

where $\mathbf{H}^e \in \mathbb{R}^{n \times d}$, and d is the hidden dimension.

(2) **Decoder** is to get the index probability distribution for each step $P_t = P(y_t|X, Y_{<t})$. However, since $Y_{<t}$ contains the pointer and tag index, it cannot be directly inputted to the Decoder. We use the Index2Token conversion to convert indexes into tokens

$$\hat{y}_t = \begin{cases} X_{y_t}, & \text{if } y_t \leq n, \\ G_{y_t - n}, & \text{if } y_t > n \end{cases} \tag{3}$$

After converting each y_t this way, we can get the last hidden state $\mathbf{h}_t^d \in \mathbb{R}^d$ with $\hat{Y}_{<t} = [\hat{y}_1, ..., \hat{y}_{t-1}]$ as follows

$$\mathbf{h}_t^d = \text{Decoder}(\mathbf{H}^e; \hat{Y}_{<t}) \tag{4}$$

Then, we can use the following equations to achieve the index probability distribution P_t

$$\mathbf{E}^e = \text{TokenEmbed}(X) \tag{5}$$

$$\hat{\mathbf{H}}^e = \text{MLP}(\mathbf{H}^e) \tag{6}$$

$$\bar{\mathbf{H}}^e = \alpha * \hat{\mathbf{H}}^e + (1 - \alpha) * \mathbf{E}^e \tag{7}$$

$$\mathbf{G}^d = \text{TokenEmbed}(G) \tag{8}$$

$$P_t = \text{Softmax}([\bar{\mathbf{H}}^{\mathbf{e}} \otimes \mathbf{h}_t^d; \mathbf{G}^d \otimes \mathbf{h}_t^d]) \tag{9}$$

where TokenEmbed is the embeddings shared between the Encoder and Decoder; $\mathbf{E}^e, \hat{\mathbf{H}}^e, \bar{\mathbf{H}}^e \in \mathbb{R}^{n \times d}$; $\alpha \in \mathbb{R}$ is a hyper-parameter; $\mathbf{G}^d \in \mathbb{R}^{l \times d}$; $[\cdot\,;\cdot]$ means concatenation in the first dimension; $\otimes$ means the dot product.

During the training phase, we use the negative log-likelihood loss and the teacher forcing method. During the inference, we use an autoregressive manner to generate the target sequence. We use the decoding algorithm presented in Algorithm 1 to convert the index sequence into entity spans.

3.3 Detailed Entity Representation with BART

Since our model is a Seq2Seq model, it is natural to utilize the pre-training Seq2Seq model BART to enhance our model. We present a visualization of

Algorithm 1 Decoding Algorithm to Convert the Entity Representation Sequence into Entity Spans

Input: Target sequence $Y = [y_1, ..., y_m]$ and $y_i \in [1, n + |G|]$
Output: Entity spans $E = \{(e_1, t_1), ..., (e_i, t_i)\}$

$E = \{\}, e = [], i = 1$
while $i <= m$ **do**
  $y_i = Y[i]$
  if $y_i > n$ **then**
    if $len(e) > 0$ **then**
      $E.add((e, G_{y_i - n}))$
    end if
    $e = []$
  else
    $e.append(y_i)$
  end if
  $i = i + 1$
end while
return E

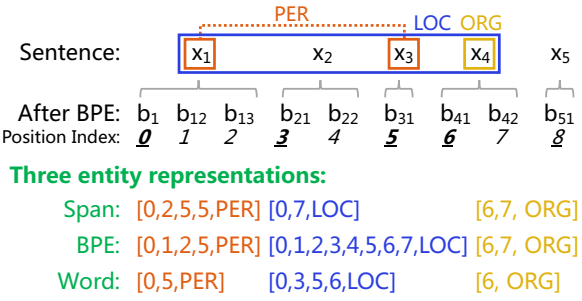

Figure 3: The bottom three lines are examples of the three kinds of entity representations to determine the entity in the sentence unambiguously. Words in the boxes are entity words, words within the same color box belong to the same entity, and their corresponding entity representation is also with the same color. There are three entities, (x_1, x_3, PER), $(x_1, x_2, x_3, x_4, LOC)$, (x_4, FAC), where LOC, PER, FAC are their corresponding entity tags. The underlined position index means this is the starting BPE of a word.

our model based on BART in Figure 2. However, BART's adoption is non-trivial because the Byte-Pair-Encoding (BPE) tokenization used in BART might tokenize one token into several BPEs. To exploit how to use BART efficiently, we propose three kinds of pointer-based entity representations to locate entities in the original sentence unambiguously. The three entity representations are as follows:

Span The position index of the first BPE of the starting entity word and the last BPE of the ending entity word. If this entity includes multiple discontinuous spans of words, each span is represented in the same way.

BPE The position indexes of all BPEs of the entity words.

Word Only the position index of the first BPE of each entity word is used.

For all cases, we will append the entity tag to the entity representation. An example of the entity representations is presented in Figure 3. If a word does not belong to any entity, it will not appear in the target sequence. If a whole sentence has no entity, the prediction should be an empty sequence (only contains the "start of sentence" (<s>) token and the "end of sentence" (</s>) token).

4 Experiment

4.1 Datasets

To show that our proposed method can be used in various NER subtasks, we conducted experiments on eight datasets.

Flat NER Datasets We adopt the CoNLL-2003 (Sang and Meulder, 2003) and the OntoNotes dataset [6] (Pradhan et al., 2013b). For CoNLL-2003, we follow Lample et al. (2016); Yu et al. (2020) to train our model on the concatenation of the train and development sets. For the OntoNotes dataset, we use the same train, development, test splits as Pradhan et al. (2012); Yu et al. (2020), and the New Testaments portion were excluded since there is no entity in this portion (Chiu and Nichols, 2016).

Nested NER Datasets We conduct experiments on ACE 2004[7] (Doddington et al., 2004), ACE 2005[8] (Walker and Consortium, 2005), Genia corpus (Kim et al., 2003). For ACE2004 and ACE2005, we use the same data split as Lu and Roth (2015); Muis and Lu (2017); Yu et al. (2020), the ratio between train, development and test set is 8:1:1. For Genia, we follow Wang et al. (2020b); Shibuya and Hovy (2020) to use five types of entities and split the train/dev/test as 8.1:0.9:1.0.

[6] https://catalog.ldc.upenn.edu/LDC2013T19
[7] https://catalog.ldc.upenn.edu/LDC2005T09
[8] https://catalog.ldc.upenn.edu/LDC2006T06
[9] In the reported experiments, they included the document context. We rerun their code with only the sentence context. The lack of document context might cause performance degradation is also confirmed by the author himself in https://github.com/juntaoy/biaffine-ner/issues/8#issuecomment-650813813.

Models	CoNLL2003			OntoNotes		
	P	R	F	P	R	F
Clark et al. (2018)[GloVe300d]	-	-	92.6	-	-	-
Peters et al. (2018)[ELMo]	-	-	92.22	-	-	-
Akbik et al. (2019)[Flair]	-	-	93.18	-	-	-
Straková et al. (2019)[BERT-Large]	-	-	93.07	-	-	-
Yamada et al. (2020)[RoBERTa-Large]	-	-	92.40	-	-	-
Li et al. (2020b)[BERT-Large]†	92.47	93.27	92.87	**91.34**	88.39	89.84
Yu et al. (2020)[BERT-Large]‡	**92.85**	92.15	92.5	89.92	89.74	89.83
Ours(Span)[BART-Large]	92.31	93.45	92.88	88.94	90.33	89.63
Ours(BPE)[BART-Large]	92.60	93.22	92.96	90.00	89.52	89.76
Ours(Word)[BART-Large]	92.61	**93.87**	**93.24**	89.99	**90.77**	**90.38**

Table 1: Results for the flat NER datasets. "†" indicates we rerun their code. "‡" means our reproduction with only the sentence-level context [9].

Models	ACE2004			ACE2005			Genia		
	P	R	F	P	R	F	P	R	F
Luan et al. (2019)[ELMO]	-	-	84.7	-	-	82.9	-	-	76.2
Straková et al. (2019)[BERT-Large]	-	-	84.33	-	-	83.42	-	-	76.44
Shibuya and Hovy (2020)[BERT-Large]⋆	85.23	84.72	84.97	83.30	84.69	83.99	77.46	76.65	77.05
Li et al. (2020b)[BERT-Large]†	85.83	85.77	85.80	**85.01**	84.13	84.57	**81.25**	76.36	78.72
Yu et al. (2020)[BERT-Large] ‡	85.42	85.92	85.67	84.50	84.72	84.61	79.43	78.32	78.87
Wang et al. (2020a)[BERT-Large]⋆	86.08	**86.48**	86.28	83.95	85.39	84.66	79.45	78.94	79.19
Ours(Span)[BART-Large]	84.81	83.64	84.22	81.41	83.24	82.31	78.87	**79.6**	**79.23**
Ours(BPE)[BART-Large]	86.69	83.83	85.24	82.08	83.44	82.75	78.15	79.06	78.60
Ours(Word)[BART-Large]	**87.27**	86.41	**86.84**	83.16	**86.38**	**84.74**	78.57	79.3	78.93

Table 2: Results for nested NER datasets,"†" means our rerun of their code. "‡" means our reproduction with only sentence-level context[9]. "⋆" for a fair comparison, we only present results with the BERT-Large model.

Discontinuous NER Datasets We follow Dai et al. (2020) to use CADEC (Karimi et al., 2015), ShARe13 (Pradhan et al., 2013a) and ShARe14 (Mowery et al., 2014) corpus. Since only the Adverse Drug Events (ADEs) entities include discontinuous annotation, only these entities were considered (Dai et al., 2020; Metke-Jimenez and Karimi, 2016; Tang et al., 2018).

4.2 Experiment Setup

We use the BART-Large model, whose encoder and decoder each has 12 layers for all experiments, making it the same number of transformer layers as the BERT-Large and RoBERTa-Large model. We did not use any other embeddings, and the BART model is fine-tuned during the optimization. We put more detailed experimental settings in the Supplementary Material. We report the span-level F1.

5 Results

5.1 Results on Flat NER

Results are shown in Table 1. We do not compare with Yamada et al. (2020) since they added entity information during the pre-training process. Clark et al. (2018); Peters et al. (2018); Akbik et al. (2019); Straková et al. (2019) assigned a label to each token, and Li et al. (2020b); Yu et al. (2020) are based on span-level classifications, while our method is based on the entity sequence generation. And for both datasets, our method achieves better performance. We will discuss the performance difference between our three entity representations in Section 5.4.

5.2 Results on Nested NER

Table 2 presents the results for the three nested NER datasets, and our proposed BART-based gen-

Model	CADEC			ShARe13			ShARe14		
	P	R	F	P	R	F	P	R	F
Metke-Jimenez and Karimi (2016)	64.4	56.5	60.2	-	-	-	-	-	-
Tang et al. (2018)	67.8	64.9	66.3	-	-	-	-	-	-
Dai et al. (2020)[ELMo]	68.9	69.0	69.0	80.5	75.0	77.7	**78.1**	81.2	79.6
Ours(Span)[BART-Large]	**71.55**	68.59	70.04	80.42	**78.15**	79.27	76.85	83.59	80.08
Ours(BPE)[BART-Large]	69.45	70.51	69.97	82.07	76.45	79.16	75.88	**84.37**	79.90
Ours(Word)[BART-Large]	70.08	**71.21**	**70.64**	**82.09**	77.42	**79.69**	77.2	83.75	**80.34**

Table 3: Results for discontinuous NER datasets.

Entity Representation	Flat NER		Nested NER			Discontinuous NER		
	CoNLL2003	OntoNotes	ACE2004	ACE2005	Genia	CADEC	ShARe13	ShARe14
Span	3.0/3.0	3.0/3.0	**3.0/3.0**	**3.0/3.0**	**3.0/3.0**	3.17/3.0	3.15/3.0	**3.2/3.0**
BPE	3.55/3.0	3.39/3.0	4.15/3.0	3.84/3.0	5.21/5.0	4.08/4.0	3.92/3.0	4.34/4.0
Word	**2.44/2.0**	**2.86/2.0**	3.53/2.0	3.26/2.0	3.09/3.0	**2.72/3.0**	**2.63/3.0**	3.74/3.0

Table 4: The average (before /) and median entity length (including the entity label) for each entity representations in the respective testing set.

erative models are comparable to the token-level classication (Straková et al., 2019; Shibuya and Hovy, 2020) and span-level classification (Luan et al., 2019; Li et al., 2020b; Wang et al., 2020a) models.

5.3 Results on Discontinuous NER

Results in Table 3 show the comparison between our model and other models in three discontinuous NER datasets. Although Dai et al. (2020) tried to utilize BERT to enhance the model performance, they found that ELMo worked better. In all three datasets, our model achieves better performance.

5.4 Comparison Between Different Entity Representations

In this part, we discuss the performance difference between the three entity representations. The "Word" entity representation achieves better performance almost in all datasets. And the comparison between the "Span" and "BPE" representations is more involved. To investigate the reason behind these results, we calculate the average and median length of entities when using different entity representations, and the results are presented in Table 4. It is clear that for a generative framework, the shorter the entity representation the better performance it should achieve. Therefore, as shown in Table 4, the "Word" representation with smaller average entity length in CoNLL2003, OntoNotes, CADEC, ShARe13 achieves better performance in these datasets. However, although the average entity length of the "BPE" representation is longer than the "Span" representation, it achieves better performance in CoNLL2003, OntoNotes, ACE2004, ACE2005, this is because the "BPE" representation is more similar to the pre-training task, namely, predicting continuous BPEs. And we believe this task similarity is also the reason why the "Word" representation (Most of the words will be tokenized into a single BPE, making the "Word" representation still continuous.) achieves better performance than the "Span" representation in ACE2004, ACE2005, and ShARe14, although the former has longer entity length.

A clear outlier is the Genia dataset, where the "Span" representation achieves better performance than the other two. We presume this is because in this dataset, a word will be tokenized into a longer BPE sequence (this can be inferred from the large entity length gap between the "Word" and "BPE" representation.) so that the "Word" representation will also be dissimilar to the pre-training tasks. For example, the protein "lipoxygenase isoforms" will be tokenized into the sequence "['Ġlip', 'oxy', 'gen', 'ase', 'Ġiso', 'forms']", which makes the target sequence of the "Word" representation be "['Ġlip', 'Ġiso']", resulting a discontiguous BPE

	Flat NER		Nested NER			Discontinuous NER		
Errors	CoNLL2003	OntoNotes	ACE2004	ACE2005	Genia	CADEC	ShARe13	ShARe14
E_1	0.05%	0.02%	0.23%	0.06%	0.0%	0.31%	0.0%	0.01%
E_2	0.04%	0.03%	0.13%	0.22%	0.11%	1.02%	0.18%	0.16%
E_3	0.05%	0.02%	0.30%	0.26%	0.06%	0.0%	0.08%	0.02%

Table 5: Different invalid prediction probability for the "Word" entity representation. E_1 means the predicted indexes contain index which is not the start index of a word, E_2 means the predicted indexes within an entity are not increasing, E_3 means duplicated entity prediction.

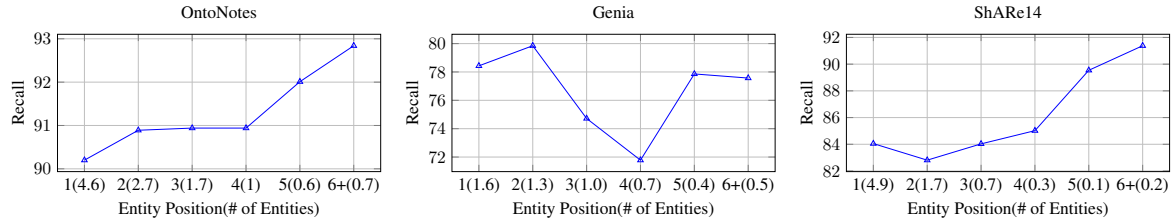

Figure 4: The recall of entities in different entity sequence positions, the number of entities in that position is the number in the bracket (the unit is 1000).

sequence. Therefore, the shorter "Span" representation achieves better performance in this dataset.

6 Analysis

6.1 Recall of Discontinuous Entities

Since only about 10% of entities in the discontinuous NER datasets are discontinuous, only evaluating the whole dataset may not show our model can recognize the discontinuous entities. Therefore, like in Dai et al. (2020); Muis and Lu (2016) we report our model's performance on the discontinuous entities in Table 6. As shown in Table 6, our model can predict the discontinuous named entities and achieve better performance.

	ShARe13			ShARe14		
Model	P	R	F	P	R	F
Dai et al. (2020)	**78.5**	39.4	52.5	**56.1**	43.8	49.2
Ours(Word)	57.5	**52.8**	**55.0**	49.6	**56.2**	**52.7**

Table 6: Performance on the discontinuous entities of the tesing dataset of ShARe13 and ShARe14.

6.2 Invalid Prediction

In this part, we mainly focus on the analysis of the "Word" representation since it generally achieves better performance. We do not restrict the output distribution; therefore, the entity prediction may contain invalid predictions as show in Table 5, this table shows that the BART model can learn the prediction representations quite well since, in most cases, the invalid prediction is less than 1%. We exclude all these invalid predictions during evaluation.

6.3 Entity Order Vs. Entity Recall

Its appearance order in the sentence determines the entity order, and we want to study whether the entity that appears later in the target sequence will have worse recall than entities that appear early. The results are provided in Figure 4. The latter the entity appears, the larger probability that it can be recalled for the flat NER and discontinuous NER. While for the nested NER, the recall curve is quite involved. We assume this phenomenon is because, for the flat NER and discontinuous NER (more than 91.1% of entities are continuous) datasets, different entities have less dependence on each other. While in the nested NER dataset, entities in the latter position may be the outermost entity that contains the former entities. The wrong prediction of former entities may negatively influence the later entities.

7 Conclusion

In this paper, we formulate NER subtasks as an entity span sequence generation problem, so that we can use a unified Seq2Seq model with the pointer mechanism to tackle flat, nested, and discontinuous NER subtasks. The Seq2Seq formulation en-

ables us to smoothly incorporate the pre-training Seq2Seq model BART to enhance the performance. To better utilize BART, we test three types of entity representation methods to linearize the entity span into sequences. Results show that the entity representation with a shorter length and more similar to continuous BPE sequences achieves better performance. Our proposed method achieves SoTA or near SoTA performance for eight different NER datasets, proving its generality to various NER subtasks.

Acknowledgements

We would like to thank the anonymous reviewers for their insightful comments. The discussion with colleagues in AWS Shanghai AI Lab was quite fruitful. We also thank the developers of fastNLP[10] and fitlog[11]. We thank Juntao Yu for helpful discussion about dataset processing. This work was supported by the National Key Research and Development Program of China (No. 2020AAA0106700) and National Natural Science Foundation of China (No. 62022027).

Ethical Considerations

For the consideration of ethical concerns, we would make detailed description as following:

(1) All of the experiments are conducted on existing datasets, which are derived from public scientific papers.

(2) We describe the characteristics of the datasets in a specific section. Our analysis is consistent with the results.

(3) Our work does not contain identity characteristics. It does not harm anyone.

(4) Our experiments do not need a lots of computer resources compared to pre-trained models.

References

Alan Akbik, Tanja Bergmann, and Roland Vollgraf. 2019. Pooled contextualized embeddings for named entity recognition. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 724–728. Association for Computational Linguistics.

Beatrice Alex, Barry Haddow, and Claire Grover. 2007. Recognising nested named entities in biomedical text. In *Biological, translational, and clinical language processing, BioNLP@ACL 2007, Prague, Czech Republic, June 29, 2007*, pages 65–72. Association for Computational Linguistics.

Lingzhen Chen and Alessandro Moschitti. 2018. Learning to progressively recognize new named entities with sequence to sequence models. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 2181–2191. Association for Computational Linguistics.

Jason P. C. Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional lstm-cnns. *Trans. Assoc. Comput. Linguistics*, 4:357–370.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1724–1734. ACL.

Kevin Clark, Minh-Thang Luong, Christopher D. Manning, and Quoc V. Le. 2018. Semi-supervised sequence modeling with cross-view training. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 1914–1925. Association for Computational Linguistics.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537.

Junqi Dai, Hang Yan, Tianxiang Sun, Pengfei Liu, and Xipeng Qiu. 2021. Does syntax matter? A strong baseline for aspect-based sentiment analysis with roberta. *CoRR*, abs/2104.04986.

Xiang Dai, Sarvnaz Karimi, Ben Hachey, and Cécile Paris. 2020. An effective transition-based model for discontinuous NER. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 5860–5870. 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, NAACL-HLT 2019, Minneapolis, MN,*

[10] https://github.com/fastnlp/fastNLP. FastNLP is a natural language processing python package.

[11] https://github.com/fastnlp/fitlog. Fitlog is an experiment tracking package.

USA, June 2-7, 2019, Volume 1 (Long and Short Papers), pages 4171–4186. Association for Computational Linguistics.

George R. Doddington, Alexis Mitchell, Mark A. Przybocki, Lance A. Ramshaw, Stephanie M. Strassel, and Ralph M. 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 2004, May 26-28, 2004, Lisbon, Portugal*. European Language Resources Association.

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, EMNLP 2009, 6-7 August 2009, Singapore, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 141–150. ACL.

Joseph Fisher and Andreas Vlachos. 2019. Merge and label: A novel neural network architecture for nested NER. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5840–5850. Association for Computational Linguistics.

Dan Gillick, Cliff Brunk, Oriol Vinyals, and Amarnag Subramanya. 2016. Multilingual language processing from bytes. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 1296–1306. The Association for Computational Linguistics.

Ralph Grishman and Beth Sundheim. 1996. Message understanding conference- 6: A brief history. In *16th International Conference on Computational Linguistics, Proceedings of the Conference, COLING 1996, Center for Sprogteknologi, Copenhagen, Denmark, August 5-9, 1996*, pages 466–471.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991.

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, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1446–1459. Association for Computational Linguistics.

Sarvnaz Karimi, Alejandro Metke-Jimenez, Madonna Kemp, and Chen Wang. 2015. Cadec: A corpus of adverse drug event annotations. *J. Biomed. Informatics*, 55:73–81.

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, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 861–871. Association for Computational Linguistics.

Jin-Dong Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. 2003. GENIA corpus - a semantically annotated corpus for bio-textmining. In *Proceedings of the Eleventh International Conference on Intelligent Systems for Molecular Biology, June 29 - July 3, 2003, Brisbane, Australia*, pages 180–182.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 260–270. The Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Xiaonan Li, Hang Yan, Xipeng Qiu, and Xuanjing Huang. 2020a. FLAT: chinese NER using flat-lattice transformer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6836–6842. Association for Computational Linguistics.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020b. A unified MRC framework for named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 5849–5859. Association for Computational Linguistics.

Hongyu Lin, Yaojie Lu, Xianpei Han, and Le Sun. 2019. Sequence-to-nuggets: Nested entity mention detection via anchor-region networks. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5182–5192. 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, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 857–867. The Association for Computational Linguistics.

Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019. A general framework for information extraction using dynamic span graphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 3036–3046. Association for Computational Linguistics.

Ying Luo and Hai Zhao. 2020. Bipartite flat-graph network for nested named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6408–6418. Association for Computational Linguistics.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1412–1421. The Association for Computational Linguistics.

Andrew McCallum and Wei Li. 2003. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the Seventh Conference on Natural Language Learning, CoNLL 2003, Held in cooperation with HLT-NAACL 2003, Edmonton, Canada, May 31 - June 1, 2003*, pages 188–191. ACL.

Alejandro Metke-Jimenez and Sarvnaz Karimi. 2016. Concept identification and normalisation for adverse drug event discovery in medical forums. In *Proceedings of the First International Workshop on Biomedical Data Integration and Discovery (BMDID 2016) co-located with The 15th International Semantic Web Conference (ISWC 2016), Kobe, Japan, October 17, 2016*, volume 1709 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Danielle L. Mowery, Sumithra Velupillai, Brett R. South, Lee M. Christensen, David Martínez, Liadh Kelly, Lorraine Goeuriot, Noémie Elhadad, Sameer Pradhan, Guergana K. Savova, and Wendy W. Chapman. 2014. Task 2: Share/clef ehealth evaluation lab 2014. In *Working Notes for CLEF 2014 Conference, Sheffield, UK, September 15-18, 2014*, volume 1180 of *CEUR Workshop Proceedings*, pages 31–42. CEUR-WS.org.

Aldrian Obaja Muis and Wei Lu. 2016. Learning to recognize discontiguous entities. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 75–84. The Association for Computational Linguistics.

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, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2608–2618. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics.

Sameer Pradhan, Noémie Elhadad, Brett R. South, David Martínez, Lee M. Christensen, Amy Vogel, Hanna Suominen, Wendy W. Chapman, and Guergana K. Savova. 2013a. Task 1: Share/clef ehealth evaluation lab 2013. In *Working Notes for CLEF 2013 Conference , Valencia, Spain, September 23-26, 2013*, volume 1179 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013b. Towards robust linguistic analysis using ontonotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning, CoNLL 2013, Sofia, Bulgaria, August 8-9, 2013*, pages 143–152. ACL.

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 Empirical Methods in Natural Language Processing and Computational Natural Language Learning - Proceedings of the Shared Task: Modeling Multilingual Unrestricted Coreference in OntoNotes, EMNLP-CoNLL 2012, July 13, 2012, Jeju Island, Korea*, pages 1–40. ACL.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *CoRR*, abs/2003.08271.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Lev-Arie 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, Boulder, Colorado, USA, June 4-5, 2009*, pages 147–155. ACL.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning, CoNLL 2003, Held in cooperation with HLT-NAACL 2003, Edmonton, Canada, May 31 - June 1, 2003*, pages 142–147. ACL.

Takashi Shibuya and Eduard H. Hovy. 2020. Nested named entity recognition via second-best sequence learning and decoding. *Trans. Assoc. Comput. Linguistics*, 8:605–620.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. MASS: masked sequence to sequence pre-training for language generation. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 5926–5936. PMLR.

Jana Straková, Milan Straka, and Jan Hajic. 2019. Neural architectures for nested NER through linearization. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5326–5331. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112.

Buzhou Tang, Jianglu Hu, Xiaolong Wang, and Qingcai Chen. 2018. Recognizing continuous and discontinuous adverse drug reaction mentions from social media using LSTM-CRF. *Wirel. Commun. Mob. Comput.*, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

David Vilares and Carlos Gómez-Rodríguez. 2020. Discontinuous constituent parsing as sequence labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 2771–2785. Association for Computational Linguistics.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2692–2700.

C. Walker and Linguistic Data Consortium. 2005. *ACE 2005 Multilingual Training Corpus*. LDC corpora. Linguistic Data Consortium.

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, Brussels, Belgium, October 31 - November 4, 2018*, pages 204–214. Association for Computational Linguistics.

Bailin Wang and Wei Lu. 2019. Combining spans into entities: A neural two-stage approach for recognizing discontiguous entities. 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 2019, Hong Kong, China, November 3-7, 2019*, pages 6215–6223. Association for Computational Linguistics.

Jue Wang, Lidan Shou, Ke Chen, and Gang Chen. 2020a. Pyramid: A layered model for nested named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 5918–5928. Association for Computational Linguistics.

Yu Wang, Yun Li, Hanghang Tong, and Ziye Zhu. 2020b. HIT: nested named entity recognition via head-tail pair and token interaction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6027–6036. Association for Computational Linguistics.

Yu Wang, Yun Li, Ziye Zhu, Bin Xia, and Zheng Liu. 2019. SC-NER: A sequence-to-sequence model with sentence classification for named entity recognition. In *Advances in Knowledge Discovery and Data Mining - 23rd Pacific-Asia Conference, PAKDD 2019, Macau, China, April 14-17, 2019, Proceedings, Part I*, volume 11439 of *Lecture Notes in Computer Science*, pages 198–209. Springer.

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, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long*

Papers, pages 1237–1247. Association for Computational Linguistics.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: deep contextualized entity representations with entity-aware self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6442–6454. Association for Computational Linguistics.

Hang Yan, Bocao Deng, Xiaonan Li, and Xipeng Qiu. 2019. TENER: adapting transformer encoder for named entity recognition. *CoRR*, abs/1911.04474.

Hang Yan, Xipeng Qiu, and Xuanjing Huang. 2020. A graph-based model for joint chinese word segmentation and dependency parsing. *Trans. Assoc. Comput. Linguistics*, 8:78–92.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Named entity recognition as dependency parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6470–6476. Association for Computational Linguistics.

Yuan Zhang, Hongshen Chen, Yihong Zhao, Qun Liu, and Dawei Yin. 2018. Learning tag dependencies for sequence tagging. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 4581–4587. ijcai.org.

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 2019, Hong Kong, China, November 3-7, 2019*, pages 357–366. Association for Computational Linguistics.

A Supplemental Material

A.1 Hyper-parameters

The detailed hyper-parameter used in different datasets are listed in Table 7. We use the slanted triangular learning rate warmup. All experiments are conducted in the Nvidia Ge-Force RTX-3090 Graphical Card with 24G graphical memory.

Hyper	Value
Epoch	30
Warmup step	0.01
Learning rate	[1e-5,2e-5,4e-5]
Batch size	16
BART	Large
α	0.5
Beam size	[1, 4]

Table 7: Hyper-parameters used for CoNLL2003, OntoNotes, ACE2004, ACE2005, Genia, CADEC, ShARe13, ShARe14.

A.2 Beam Search

Since our framework is based on generation, we want to study whether using beam search will increase the performance, results are depicted in Figure 5, it shows the beam search almost has no effect on the model performance. The litte effect on the F1 value might be caused the the small searching space when generating.

A.3 Efficiency Metrics

In this section, we compare the memory footprint, training and inference time of our proposed model and BERT-based models. The experiments are conducted on the flat NER datasets, CoNLL-2003 (Sang and Meulder, 2003) and OntoNotes (Pradhan et al., 2012). We use the BERT-MLP and BERT-CRF models as our baseline models. BERT-MLP and BERT-CRF are sequence labelling based models. For an input sentence $X = [x_1, ..., x_n]$, both models use BERT (Devlin et al., 2019) to encode X as follows

$$\mathbf{H} = \text{BERT}(X) \tag{10}$$

where $\mathbf{H} \in \mathbb{R}^{n \times d}$, d is the hidden state dimension.

Then for the BERT-MLP model, it decodes the tags as follows

$$\mathbf{F} = \text{Softmax}(\max(\mathbf{H}W_b + b_b, 0)W_a + b_a) \tag{11}$$

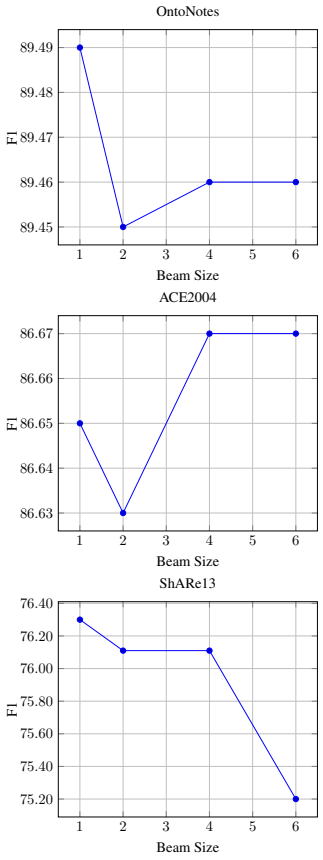

Figure 5: The F1 change curve with the increment of beam size. The beam size has limited effect on the F1 score.

where $W_a \in \mathbb{R}^{d \times |T|}$ and $|T|$ is the number of tags, $b_a \in \mathbb{R}^{|T|}$, $W_b \in \mathbb{R}^{d \times d}$, $b_b \in \mathbb{R}^d$, $\mathbf{F} \in \mathbb{R}^{n \times |T|}$ is the tag probability distribution. Then we use the negative log likelihood loss. And during the inference, for each token, the tag index with the largest probability is deemed as the prediction.

For the BERT-CRF model, we use the conditional random fields (CRF) (Lafferty et al., 2001) to decode tags. We assue the golden label sequence is $Y = [y_1, ..., y_n]$, then we use the following equations to get the probability of Y

$$\mathbf{M} = \max(\mathbf{H}W_b + b_b, 0)W_a + b_a \tag{12}$$

$$\mathbf{M} = \log\_\text{softmax}(\mathbf{M}) \tag{13}$$

$$P(Y|X) = \frac{\sum_{i=1}^{n} e^{\mathbf{M}[i,y_i]+\mathbf{T}[y_{i-1},y_i]}}{\sum_{y'}^{\mathbf{Y}(\mathbf{s})} \sum_{i=1}^{n} e^{\mathbf{M}[i,y'_i]+\mathbf{T}[y'_{i-1},y'_i]}}, \tag{14}$$

where $\mathbf{M} \in \mathbb{R}^{n \times |T|}$, $\mathbf{Y}(\mathbf{s})$ is all valid label sequences, $\mathbf{T} \in \mathbb{R}^{|T| \times |T|}$ is the transitation matrix, an entry (i, j) in $\mathbf{T}$ means the transition score from tag i to tag j. After getting the $P(Y|X)$, we use negative log likelihood loss to optimize the model. Dur-

Dataset	Model	Memory	Training Time	Evaluation Time
CoNLL-2003	BERT-MLP	7G	98s	3s
	BERT-CRF	7G	122s	5s
	Ours(Word)[BART]	8G	115s	12s
OntoNotes	BERT-MLP	7G	421s	9s
	BERT-CRF	7G	523s	13s
	Ours(Word)[BART]	7G	493s	38s

Table 8: The training memory usage, training time and evaluation time comparison between three models.

ing the inference, the Viterbi Algorithm is used to find the label sequence achieves the highest score.

We use the BERT-base version and BART-base version to calculate the memory footprint during training, seconds needed to iterate one epoch (one epoch means iterating over all training samples), and seconds needed to evaluate the development set. The batch size is 16 and 48 for training and evaluation, respectively. The comparison is presented in Table 8.

During the training phase, we can use the casual mask to make the training of our model in parallel. Therefore, our proposed model can train faster than the BERT-CRF model, which needs sequential computation. While during the evaluating phase, we have to autoregressively generate tokens, which will make the inference slow. Therefore, further work like the usage of a non-autoregressive method can be studied to speed up the decoding (Gu et al., 2018).