Sentence Identification with BOS and EOS Label Combinations

Takuma Udagawa, Hiroshi Kanayama, Issei Yoshida
IBM Research - Tokyo, Japan
Takuma.Udagawa@ibm.com, {hkana,issei}@jp.ibm.com

Abstract

The sentence is a fundamental unit in many NLP applications. Sentence segmentation is widely used as the first preprocessing task, where an input text is split into consecutive sentences considering the end of the sentence (EOS) as their boundaries. This task formulation relies on a strong assumption that the input text consists only of sentences, or what we call the sentential units (SUs). However, real-world texts often contain non-sentential units (NSUs) such as metadata, sentence fragments, nonlinguistic markers, etc. which are unreasonable or undesirable to be treated as a part of an SU. To tackle this issue, we formulate a novel task of sentence identification, where the goal is to identify SUs while excluding NSUs in a given text. To conduct sentence identification, we propose a simple yet effective method which combines the beginning of the sentence (BOS) and EOS labels to determine the most probable SUs and NSUs based on dynamic programming. To evaluate this task, we design an automatic, language-independent procedure to convert the Universal Dependencies corpora into sentence identification benchmarks. Finally, our experiments on the sentence identification task demonstrate that our proposed method generally outperforms sentence segmentation baselines which only utilize EOS labels.

1 Introduction

The sentence, which we refer to as the sentential unit (SU), is a fundamental unit of processing in many NLP applications including syntactic parsing (Dozat and Manning, 2017), semantic parsing (Dozat and Manning, 2018), and machine translation (Liu et al., 2020). Existing works mostly rely on *sentence segmentation* (a.k.a. *sentence boundary detection*) as the first preprocessing task, where we predict the end of the sentence (EOS) to split a text into consecutive SUs (Kiss and Strunk, 2006; Gillick, 2009). This approach relies on a strong assumption that the text only consists of SUs; however, real-world texts like web contents often contain non-sentential units (NSUs) such as the metadata of attachments embedded in the email body, repetition of symbols for separating texts, irregular series of nouns, etc. (just to name a few). Such NSUs may cause detrimental or unexpected results in the downstream tasks if considered as parts of the SUs and are more desirable to be distinguished from SUs in the first preprocessing step.

To tackle this problem, we formulate a novel task of *sentence identification*, where the goal is to identify SUs while excluding NSUs in a given text (§3). This can be regarded as an SU span extraction task, where each SU span is represented by the beginning of the sentence (BOS) and the EOS labels.[1] We illustrate the difference between sentence segmentation and sentence identification in Table 1. In sentence segmentation, the text fragment of an embedded file ("`- TEXT.htm << File: TEXT.htm >>`") needs to be considered as a part of an SU. In contrast, sentence identification can regard it as an NSU and exclude it for downstream applications such as dependency parsing.

To conduct sentence identification, we propose a simple method which effectively combines the BOS and EOS probabilities to determine both SUs and NSUs (§4). To be specific, we first train the BOS and EOS labeling models based on either the sentence identification dataset (with SUs and NSUs) or sentence segmentation dataset (only SUs). Then, we search for the most probable spans of SUs and NSUs using a simple dynamic programming framework. Theoretically, our method can be considered as a natural generalization of existing sentence segmentation algorithms.

To evaluate this task, we design an automatic pro-

[1] For simplicity, we assume that the input text can be segmented into consecutive, non-overlapping units of SUs and NSUs. This way, we can also represent and evaluate SU extraction as an equivalent BIO labeling task (§5-§7).

Findings of the Association for Computational Linguistics: EACL 2023, pages 343–358
May 2-6, 2023

Input Text (from EWT)	Thank you. - TEXT.htm << File: TEXT.htm >> I was thinking of converting it to a hover vehicle. I might just sell the car and get you to drive me around all winter.
Sentence Segmentation	Thank you. **E** - TEXT.htm << File: TEXT.htm >> I was thinking of converting it to a hover vehicle. **E** I might just sell the car and get you to drive me around all winter. **E**
Sentence Identification	**B** Thank you. **E** - TEXT.htm << File: TEXT.htm >> **B** I was thinking of converting it to a hover vehicle. **E** **B** I might just sell the car and get you to drive me around all winter. **E**

Table 1: Illustration of sentence segmentation and sentence identification. In sentence segmentation, EOS labels (**E**) are used to segment the input text into consecutive SUs (in blue). In sentence identification, only the spans bracketed by the BOS (**B**) and EOS labels are extracted as SUs, while the rest can be excluded as NSUs.

cedure to convert the Universal Dependencies (UD) corpora (de Marneffe et al., 2021) into sentence identification benchmarks (§5). To be specific, (i) we use the original sentence boundaries in UD as the unit (SU and NSU) boundaries and (ii) classify each unit as an SU iff it contains at least one clausal predicate with a core/non-core argument. Importantly, our classification rule follows the definition of *lexical sentence* in linguistics (Nunberg, 1990), is easily customizable with language-independent rules, and makes reasonable classification within the scope of our experiments.

To conduct our experiments, we focus on the English Web Treebank (Silveira et al., 2014) as the primary benchmark for sentence identification and train the BOS/EOS labeling models by finetuning RoBERTa (Liu et al., 2019) (§6). We also propose techniques to develop these models using a standard sentence segmentation dataset, i.e. the Wall Street Journal corpus (Marcus et al., 1993), which only contains clean, edited SUs without any NSUs.

Based on our experimental results, we demonstrate that our proposed method generally outperforms sentence segmentation baselines which only utilize EOS labels (§7). These results highlight the importance of combining the BOS labels in addition to the EOS labels for accurate sentence identification under various conditions.

2 Background

Sentence segmentation, a.k.a. sentence boundary detection, is the task of segmenting an input text into the unit of sentences. Despite the long history of study (Riley, 1989) and its importance in the entire NLP pipeline (Walker et al., 2001), this area has received relatively little attention. For one reason, the task has been recognized as "long solved" (Read et al., 2012) with the most recent approach reporting 99.8% F1 score on the standard English Wall Street Journal (WSJ) dataset (Wicks and Post, 2021). Their state-of-the-art method ERSATZ combines (i) a regular-expression based detector of candidate sentence boundaries, followed by (ii) a Transformer-based (Vaswani et al., 2017) binary classifier which predicts whether the candidate boundary is EOS based on the local context, i.e. surrounding few words. This modern context-based approach has been shown to outperform competitive, widely used baselines such as SPLITTA (Gillick, 2009), PUNKT (Kiss and Strunk, 2006), and MOSES (Koehn et al., 2007).

However, two important aspects are not fully addressed in the current literature. First is the coverage of *diverse domains, genres, and writing styles*. Existing works (including Wicks and Post, 2021) focus on formal/edited text and assume the existence of sentence ending punctuations (e.g. full stops) at the sentence boundaries. However, social media texts often lack such punctuations and contain various types of non-linguistic noise, which can lead to a substantial degradation in the segmentation performance (Read et al., 2012; Rudrapal et al., 2015). Speech transcription texts also usually contain disfluent, ungrammatical, or fragmented structures and lack both punctuations and casing (Wang et al., 2019; Rehbein et al., 2020). Considering the amount of such informal or non-standard texts in the real world, it is compelling to expand the capability of sentence segmentation beyond formal, standardized text.

The second aspect is the coverage of *multiple languages*. Different languages involve different complexities in sentence segmentation, e.g. Chinese requires the disambiguation of commas as the

sentence ending punctuation (Xue and Yang, 2011) and Thai does not mark EOS with any type of punctations (Aroonmanakun et al., 2007; Zhou et al., 2016). To advance NLP from a multilingual perspective, it is crucial to develop and evaluate models in multiple languages: Wicks and Post (2021) make an important step in this direction, proposing a language-agnostic, unified sentence segmentation model covering a total of 87 languages.

Based on these observations, we first propose to extend the task of sentence segmentation to *sentence identification*, which expands the capability of sentence segmentation beyond formal, standardized text (§3, §4). Secondly, we propose a cross-lingual method of benchmarking sentence identification based on the UD corpora, considering every word or character as the candidate boundary to cover diverse domains, genres, and languages that lack sentence ending punctuations (§5). Finally, we follow Wicks and Post (2021) to develop modern neural-based models that require no language-specific engineering and can be developed for different languages in a unified manner (§6).

3 Task Formulation

3.1 Sentence Segmentation Task

First, we introduce a precise (re-)formulation of the sentence segmentation task. Let $\boldsymbol{W} = (w_0, w_1, ..., w_{N-1})$ represent the input text, where each w_i denotes a word (but can also be a subword or character). We also define the text span $\boldsymbol{W}[i:j] = (w_i, ..., w_{j-1})$, their concatenation $\boldsymbol{W}[i:j] \oplus \boldsymbol{W}[j:k] = \boldsymbol{W}[i:k]$, and SU boundary indices $\boldsymbol{B} = (b_0, b_1, ..., b_M)$ where $b_0 = 0$, $b_M = N$, and $\bigoplus_{i=1}^{M} \boldsymbol{W}[b_{i-1}:b_i] = \boldsymbol{W}$ (i.e. the concatenation of all SUs recovers the input text).

Next, we introduce the SU probability $p_{\text{SU}}(\boldsymbol{W}[i:j])$ which corresponds to the probability of the text span $\boldsymbol{W}[i:j]$ being an SU. Based on this probability, the task of sentence segmentation can be formalized as searching for the boundaries $\boldsymbol{B}$ which maximize the following probability:[2]

$$\arg\max_{\boldsymbol{B}} \prod_{i=1}^{M} p_{\text{SU}}(\boldsymbol{W}[b_{i-1}:b_i]) \tag{1}$$

The most standard approach is to define $p_{\text{SU}}(\boldsymbol{W}[i:j])$ based on a pretrained EOS labeling model, as we describe in §4.1. However, our (re-)formulation as Eq. (1) is more general and permits other definitions of SU probability as well.

[2] M is a variable and need not be fixed during the search.

3.2 Sentence Identification Task

In sentence identification, we consider the input text $\boldsymbol{W}$ can be segmented into consecutive, non-overlapping units of SUs and NSUs. Hence, we regard $\boldsymbol{B} = (b_0, b_1, ..., b_M)$ as the unit (SU and NSU) boundaries and define the unit indicators $\boldsymbol{A} = (a_1, a_2, ..., a_M)$ for each unit as follows:

$$a_i = \begin{cases} 1 & \text{if } \boldsymbol{W}[b_{i-1}:b_i] \text{ is an SU} \\ 0 & \text{if } \boldsymbol{W}[b_{i-1}:b_i] \text{ is an NSU} \end{cases}$$

Next, we introduce the NSU probability $p_{\text{NSU}}(\boldsymbol{W}[i:j])$ which corresponds to the probability of the text span $\boldsymbol{W}[i:j]$ being an NSU. Based on p_{SU} and p_{NSU}, we can formalize the task of sentence identification as searching for the unit boundaries $\boldsymbol{B}$ and unit indicators $\boldsymbol{A}$ which maximize the following probability:

$$\arg\max_{\boldsymbol{B},\boldsymbol{A}} \prod_{i=1}^{M} p_{\text{SU}}(\boldsymbol{W}[b_{i-1}:b_i])^{a_i}\, p_{\text{NSU}}(\boldsymbol{W}[b_{i-1}:b_i])^{1-a_i} \tag{2}$$

Note that this strictly generalizes the sentence segmentation task in Eq. (1), which is a special case where $a_i = 1, \forall a_i \in \boldsymbol{A}$. Based on this task formulation, we discuss how we can define $p_{\text{SU}}(\boldsymbol{W}[i:j])$ and $p_{\text{NSU}}(\boldsymbol{W}[i:j])$ to derive our sentence identification algorithm in §4.2.

4 Methods

4.1 Sentence Segmentation Method

In the most standard approach, sentence segmentation employs an EOS labeling model p_{EOS} to define the SU probability p_{SU} in Eq. (1). To be specific, let $p_{\text{EOS}}(w_i|\boldsymbol{W};\theta)$ denote the EOS labeling model, which computes the probability of w_i being EOS in $\boldsymbol{W}$ (θ denotes the model parameters). Typically, it is straightforward to train this model in a *supervised learning* setup using a dataset annotated with gold EOS boundaries (Wicks and Post, 2021). For brevity, we use the notation $p_{\text{EOS}}(w_i)$ as a shorthand for $p_{\text{EOS}}(w_i|\boldsymbol{W};\theta)$, i.e. we omit $\boldsymbol{W}$ and θ (unless required) in the rest of this paper.

Based on the pretrained model p_{EOS}, we can define the SU probability as $p_{\text{SU}}(\boldsymbol{W}[i:j]) = p_{\text{EOS}}(w_{j-1}) \prod_{i \le k < j-1} (1 - p_{\text{EOS}}(w_k))$, which requires the last word w_{j-1} to be EOS and all other

words to be non-EOS. By substituting this definition, we can decompose Eq. (1) as follows:

$$
\begin{aligned}
(1) &= \arg\max_{\boldsymbol{B}} \sum_{i=1}^{M} \log\, p_{\mathrm{SU}}(\boldsymbol{W}[b_{i-1}\!:\!b_i]) \\
&= \arg\max_{\boldsymbol{B}} \sum_{i=1}^{M} \Big\{ \log\, p_{\mathrm{EOS}}(w_{b_i-1}) + \sum_{b_{i-1}\le j<b_i-1} \log\,(1-p_{\mathrm{EOS}}(w_j)) \Big\} \\
&= \arg\max_{\boldsymbol{B}} \sum_{i\in \boldsymbol{B}_{\mathrm{EOS}}} \log\, p_{\mathrm{EOS}}(w_i) + \sum_{i\notin \boldsymbol{B}_{\mathrm{EOS}}} \log\,(1-p_{\mathrm{EOS}}(w_i))
\end{aligned}
\tag{3}
$$

where $\boldsymbol{B}_{\mathrm{EOS}} = \{b_i - 1 \,|\, i \in (1, 2, ..., M)\}$ represents all the EOS indices defined by $\boldsymbol{B}$.

This is a trivial optimization problem where we can simply choose $\boldsymbol{B}_{\mathrm{EOS}} = \{i \in (0, 1, ..., N-1) \,|\, p_{\mathrm{EOS}}(w_i) \geq 0.5\}$ to maximize Eq. (3). This also shows that sentence segmentation can be conducted by predicting the EOS independently for each w_i based on $p_{\mathrm{EOS}}(w_i)$. In contrast, sentence identification involves a more complex optimization problem which we solve using dynamic programming (§4.2).

4.2 Sentence Identification Method

We extend the method of sentence segmentation (§4.1) to conduct sentence identification. To be specific, we employ pretrained BOS and EOS labeling models $p_{\mathrm{BOS}}, p_{\mathrm{EOS}}$ to define the SU and NSU probabilities $p_{\mathrm{SU}}, p_{\mathrm{NSU}}$ in Eq. (2). As a first step, we need to train the BOS and EOS labeling models: this can be conducted in a supervised manner using a dataset containing gold BOS and EOS labels, as we explain in §6.1.

Based on the pretrained BOS and EOS labeling models, we can define the SU and NSU probabilities as follows:

$$
\begin{aligned}
p_{\mathrm{SU}}(\boldsymbol{W}[i\!:\!j]) = \; & p_{\mathrm{BOS}}(w_i) \prod_{i<k\le j-1} (1-p_{\mathrm{BOS}}(w_k)) \\
& \times\; p_{\mathrm{EOS}}(w_{j-1}) \prod_{i\le k<j-1} (1-p_{\mathrm{EOS}}(w_k)) \\
p_{\mathrm{NSU}}(\boldsymbol{W}[i\!:\!j]) = & \prod_{i\le k\le j-1} (1-p_{\mathrm{BOS}}(w_k)) \times \prod_{i\le k\le j-1} (1-p_{\mathrm{EOS}}(w_k))
\end{aligned}
$$

In the SU probability p_{SU}, the first word w_i is required to be BOS, the last word w_{j-1} to be EOS, and all other words to be neither BOS nor EOS. Note that this definition of p_{SU} is a natural generalization from §4.1 which only relies on the EOS probability p_{EOS}.

In contrast, the NSU probability p_{NSU} requires all words to be neither BOS nor EOS. Notably, this definition does not distinguish contiguous NSUs in the sense that $p_{\mathrm{NSU}}(\boldsymbol{W}[i\!:\!k]) = p_{\mathrm{NSU}}(\boldsymbol{W}[i\!:\!j]) \times p_{\mathrm{NSU}}(\boldsymbol{W}[j\!:\!k])$ if $\boldsymbol{W}[i\!:\!j] \oplus \boldsymbol{W}[j\!:\!k] = \boldsymbol{W}[i\!:\!k]$. This is convenient as we are only interested in the extraction of SUs and do not need to seek the exact boundaries between consecutive NSUs.

By substituting these definitions of p_{SU} and p_{NSU}, we can decompose Eq. (2) as follows:

$$
\begin{aligned}
(2) &= \arg\max_{\boldsymbol{B},\boldsymbol{A}} \sum_{i=1}^{M} \Big\{ a_i \log\, p_{\mathrm{SU}}(\boldsymbol{W}[b_{i-1}\!:\!b_i]) \\
&\qquad\qquad + \; (1-a_i) \log\, p_{\mathrm{NSU}}(\boldsymbol{W}[b_{i-1}\!:\!b_i]) \Big\} \\
&= \arg\max_{\boldsymbol{B},\boldsymbol{A}} \sum_{i\in \boldsymbol{B}^{\boldsymbol{A}}_{\mathrm{BOS}}} \log\, p_{\mathrm{BOS}}(w_i) + \sum_{i\notin \boldsymbol{B}^{\boldsymbol{A}}_{\mathrm{BOS}}} \log\,(1-p_{\mathrm{BOS}}(w_i)) \\
&\qquad + \sum_{i\in \boldsymbol{B}^{\boldsymbol{A}}_{\mathrm{EOS}}} \log\, p_{\mathrm{EOS}}(w_i) + \sum_{i\notin \boldsymbol{B}^{\boldsymbol{A}}_{\mathrm{EOS}}} \log\,(1-p_{\mathrm{EOS}}(w_i))
\end{aligned}
\tag{4}
$$

where $\boldsymbol{B}^{\boldsymbol{A}}_{\mathrm{BOS}} = \{b_{i-1} \,|\, i \in (1, 2, ..., M), a_i = 1\}$ denotes the BOS indices and $\boldsymbol{B}^{\boldsymbol{A}}_{\mathrm{EOS}} = \{b_i - 1 \,|\, i \in (1, 2, ..., M), a_i = 1\}$ denotes the EOS indices, both defined by $\boldsymbol{B}$ and $\boldsymbol{A}$.

Therefore, our goal is to choose $\boldsymbol{B}^{\boldsymbol{A}}_{\mathrm{BOS}}$ and $\boldsymbol{B}^{\boldsymbol{A}}_{\mathrm{EOS}}$ which maximize Eq. (4). To this end, we need to consider the restrictions that (i) the first label should be BOS, (ii) the last label should be EOS, and (iii) BOS and EOS labels need to appear alternately. These restrictions can be incorporated in our dynamic programming framework to find the argmax of Eq. (4). For the precise algorithm, we refer the readers to Appendix A.

5 Evaluation

Due to the novelty of the task, currently there exists no benchmark for evaluating sentence identification. To address this issue, we propose a fully automatic procedure to convert the Universal Dependencies (UD) corpora (de Marneffe et al., 2021) into sentence identification benchmarks.

Concretely speaking, we conduct the following two steps based on the gold UD annotation: (i) the detection of unit (SU and NSU) boundaries and (ii) the classification of each unit into SU or NSU. As for (i), we simply use the original *sentence boundaries* in the UD annotation, where UD uses the term *sentence* in a broader sense including both SUs and NSUs (e.g. sentence fragments). Note that the exact boundaries between consecutive NSUs (which we call NSU–NSU boundaries) do not need to be accurate or consistent, since we are only interested in extracting the spans of SUs. However, we do expect that the original boundaries are generally reliable in all other cases (SU–SU and SU–NSU boundaries), which seems to be the case.

The main problem is (ii), i.e. how to classify

∼∼*∼*∼*∼*∼*∼*∼*∼*
**********NOTE**********
By video conference from _____
Excerpt:
02/13/2001 08:02 PM
5:00 PT ** 6:00 MT ** 7:00 CT ** 8:00 ET
Time: 11:30am / 1:30pm Central / 2:30pm Eastern
Sunshine Coast, British Columbia, Canada
- UnleadedStocks.pdf
t r u t h o u t — Perspective
(Answered, 2 Comments)
The federal sites of Washington, DC.
From Madrid to Seville to Barcelona an Valencia.

Table 2: Examples of gold NSUs in the English Web Treebank (EWT) identified based on our procedure. Each line corresponds to one example of NSU.

		Train	Dev	Test
Total SUs		10,356	1,523	1,490
Total NSUs		2,187	478	587
Word-Level	B-Label	10,356	1,523	1,490
	I-Label	160,127	18,791	18,222
	O-Label	6,939	1,302	1,822
Character-Level	B-Label	10,356	1,523	1,490
	I-Label	773,223	92,309	88,441
	O-Label	47,107	9,925	13,232

Table 3: EWT dataset statistics.

each unit as an SU or NSU. To this end, we follow the notion of *lexical sentence* in linguistics (Nunberg, 1990) which defines an SU based on the dependencies among the lexical items, e.g. a group of words that contain a subject and predicate. In this work, we build upon the UD dependency relations and define an SU as a unit that contains at least one *clausal predicate* with a *core* or *non-core argument*.[3] Here, a *clause* expresses an event or proposition which we regard as an essential aspect of SUs. A clausal predicate and a core argument form the backbone of a clause, while a non-core argument modifies it (de Marneffe et al., 2021).

Note that our current definition excludes *noun phrases* appearing by themselves, since they only consist of the *nominal dependent* relations. However, we can flexibly customize the definition of SUs to include or exclude such phrases.

Due to the reliance on UD, our conversion procedure can be applied to a wide variety of languages supported in UD (currently over 100 languages). However, as a first set of experiments, we focus on the English Web Treebank (EWT) (Silveira et al., 2014) as the primary benchmark of sentence identification. This dataset comprises five genres of web media texts: namely weblogs, newsgroup threads, emails, product reviews, and Q&A websites. Consequently, the dataset contains formal SUs, informal SUs (e.g. without capitalization or punctuations) as well as a wide variety of NSUs.

We show some examples of NSUs in Table 2 (and more in Appendix B) identified based on our procedure. As shown by the results, our procedure can identify various NSUs including nonlinguistic markers, timestamps, lists, contact information, etc. We can also see that noun/prepositional phrases are classified as NSUs based on our criteria. By excluding such NSUs and identifying SUs, we can clearly separate the portions of the text that are worth sophisticated linguistic analyses, e.g. based on dependency parsing or manual inspection.

Finally, we summarize the dataset statistics of EWT in Table 3. Overall, 17∼28% of the units were classified as NSUs, with the test set containing the highest proportion of NSUs. We also regard SU extraction as a word-level or character-level BIO labeling task and report the number of gold BIO labels in Table 3.[4] At the word-level, we can see that the proportion of O-labels (indicating NSUs) is only 4∼8% and much smaller than the proportion of NSUs in terms of units: this is because NSUs are usually short and contain only a few words. At the character-level, the proportion of O-labels is slightly larger (6∼13%): this is because NSUs often contain extraordinarily long words like URLs and long sequences of nonlinguistic symbols.

Overall, we could verify that there exists a non-negligible amount of NSUs in the EWT dataset, which we aim to exclude with sentence identification in our experiments.

6 Experimental Setup

6.1 Model Setup

As we discussed in §4.2, our sentence identification method requires pretrained BOS and EOS labeling models to identify SUs and NSUs. To develop these models, we simply finetune RoBERTa$_{\text{BASE}}$

[3] To check this condition, we simply need to verify whether there is at least one core argument (e.g. *nsubj*, *obj*, *ccomp*) or non-core dependent (e.g. *obl*, *advcl*, *aux*). For a full list of the UD relations, see https://universaldependencies.org/u/dep/.

[4] B = Beginning of SU, I = Inside of SU, and O = Outside of SU. Details of how we assign the gold BIO labels (at the word-level and character-level) are provided in Appendix C.

by adding a binary BOS/EOS classifier on top of the encoder.

To enable our models to handle various lengths of the input texts, we concatenate the consecutive L units of gold SUs and NSUs as the input during training, where L is sampled from a geometric distribution with parameter p_{CC}.[5] However, the RoBERTa encoder has the restriction that the input text size cannot exceed 512 subwords. Therefore, if the input text size is too large, we replace L with the maximum $L' < L$ which satisfies this restriction. Note that this is a common procedure to sample variable (instead of fixed) lengths of concatenated units (Joshi et al., 2020).

Assuming the existence of the in-domain sentence identification dataset (EWT Train/Dev), it is straightforward to train the BOS/EOS labeling models based on our unit concatenation procedure. However, we may not always have the gold annotation of SUs and NSUs for the target domain. To take such cases into account, we also consider a setup where we only have the standard sentence segmentation dataset (WSJ Train/Dev) to train the BOS/EOS labeling models.

When using the sentence segmentation dataset (WSJ), we need to apply the unit concatenation procedure using only clean, edited SUs. Unfortunately, this can yield the following data priors which do not actually hold in a sentence identification dataset (EWT): (i) an SU (almost) always starts with a capitalization and ends with punctuation, (ii) the first word of the input is always BOS and the last word is always EOS, and (iii) BOS always directly follows EOS.

To address (i) and (ii), we propose a simple data augmentation technique to alleviate the discrepancy in the data priors. To address (iii), we propose an ensembling technique with the unidirectional (instead of bidirectional) models which are agnostic to this data prior.

6.1.1 Data Augmentation (+AUG)

To address (i), we conduct a unit-level data augmentation, i.e. we modify each unit based on the following rules with a small probability p_{DA}:

- Convert all words in the unit to lower-case, upper-case, or title-case (e.g. "*hello world*", "*HELLO WORLD*", or "*Hello World*").
- Remove sentence ending punctuations based on a regular-expression matcher (following ERSATZ, Wicks and Post, 2021).

[5] With parameter $p_{CC} \in (0, 1]$, the probability mass function of the geometric distribution is $p(L = l) = (1 - p_{CC})^{l-1} p_{CC}$ where $l \in \{1, 2, 3, ...\}$. As p_{CC} decreases, the distribution gets more skewed towards larger L. With $p_{CC} = 0$, we consider $p(L = \infty) = 1$.

Orig.		B Joe went to school. E	B After that he ...
(i)	Unit Aug.	B Joe went to school E	B AFTER THAT HE ...
(ii)	Unit Trunc.	~~B~~ ~~Joe went~~ to school ~~E~~	B AFTER THAT HE ...

Table 4: Illustration of our data augmentation technique. In (i) *unit-level augmentation*, we randomly change the casing or remove the last punctuations of each unit. In (ii) *unit truncation*, we randomly truncate the first and last units of the input (and regard them as NSUs).

After the unit-level augmentation, we can apply the unit concatenation in the exact same manner.

Finally, to address (ii), we randomly apply a unit truncation to the first and last units of the concatenated input. To be specific, we choose a random word in the first (last) unit and remove all words prior (posterior) to it with a small probability p_{TR}. If the truncation is conducted, we regard the unit as an NSU and fix the gold BOS/EOS labels accordingly. See Table 4 for an illustration.

Based on this procedure, we can expect to alleviate the data priors (i) and (ii). For more details, we refer the readers to Appendix D.

6.1.2 Unidirectional Model (+UNI)

Simply concatenating SUs (without NSUs) yields the data prior (iii), i.e. BOS always directly follows EOS. This prior can be easily captured by the bidirectional models $p_{\text{BOS}}(w_i|\boldsymbol{W})$, $p_{\text{EOS}}(w_i|\boldsymbol{W})$ conditioned on the whole input $\boldsymbol{W}$, including our RoBERTa-based models. For instance, as shown in Figure 1, the model may predict EOS at the end of the first unit ($w_2 = \#$) just because the next word ($w_3 = \textit{This}$) is likely predicted as BOS.

To alleviate this issue, we propose to combine the predictions of the unidirectional models for BOS and EOS labeling. To be precise, let $\boldsymbol{W}^{\leq i} = (w_0, ..., w_i)$ and $\boldsymbol{W}^{\geq i} = (w_i, ..., w_{N-1})$. Then, we can represent the unidirectional BOS model as $p^{\text{Uni}}_{\text{BOS}}(w_i|\boldsymbol{W}^{\geq i})$ (looking the context right-to-left) and EOS model as $p^{\text{Uni}}_{\text{EOS}}(w_i|\boldsymbol{W}^{\leq i})$ (looking left-to-right). As illustrated in Figure 1, these models are agnostic to the data prior (iii). In practice, we can simply use different attention masks and share the

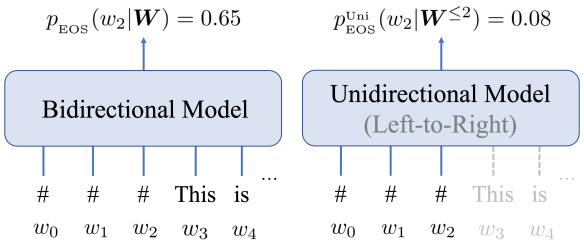


Figure 1: Illustration of the bidirectional EOS model (left) and the unidirectional EOS model (right).

encoder parameters (except the last classifier) for the unidirectional and bidirectional models.

We can utilize these unidirectional models by taking a linear intepolation with the bidirectional models as follows:

$$p^{+\text{Uni}}_{\text{BOS}}(w_i|\boldsymbol{W}) = \lambda \cdot p^{\text{Uni}}_{\text{BOS}}(w_i|\boldsymbol{W}^{\geq i}) + (1-\lambda) \cdot p_{\text{BOS}}(w_i|\boldsymbol{W})$$
$$p^{+\text{Uni}}_{\text{EOS}}(w_i|\boldsymbol{W}) = \lambda \cdot p^{\text{Uni}}_{\text{EOS}}(w_i|\boldsymbol{W}^{\leq i}) + (1-\lambda) \cdot p_{\text{EOS}}(w_i|\boldsymbol{W})$$

Then, we can use $p^{+\text{Uni}}_{\text{BOS}}$ and $p^{+\text{Uni}}_{\text{EOS}}$ in place of p_{BOS} and p_{EOS} (respectively) to conduct sentence identification, as described in §4.2.

Finally, we compare our proposed methods against sentence segmentation baselines which only utilize EOS labels.[6] As for the baselines, we use the EOS labeling model developed in the same manner to segment the input text based on EOS. Note that we can optionally force the last word in the input to be EOS: in this case, the result will only contain SUs since all segments will end with EOS. By default, we do not force the last EOS: in this case, the segment after the last EOS (if exists) is considered as an NSU.

As a default configuration, we use $p_{CC} = 0.5$, $p_{DA} = 0.3$, $p_{TR} = 0.1$, and $\lambda = 0.5$ in our experiments. To ensure reproducibility, we report more details on the hyperparameters and model setup in Appendix D. For the precise procedure on how we convert between the word-, character-, and subword-level labels (for RoBERTa), we refer the readers to Appendix C.

6.2 Evaluation Setup

In the evaluation phase, we consider three ways of assembling the input texts on which we conduct sentence identification. Firstly, we can apply the same unit concatenation procedure as described in §6.1. To be specific, we use $p_{CC} = 0.5$ (same as the training phase) and $p_{CC} = 0$ (which concatenates the units up to the maximal length) to simulate both shorter and longer lengths of the input texts.

However, this approach is relatively *synthetic* in the sense that we take the gold unit boundaries for granted. They are usually unavailable at the inference time, so we should consider a more realistic setting for evaluating the methods without relying on the gold unit boundaries.

To this end, we propose to evaluate sentence identification as a *postprocessing* of sentence segmentation. To be specific, we first apply the state-of-the-art method ERSATZ (Wicks and Post, 2021) on the raw text of EWT and then apply sentence identification to each segmented text. Note that ERSATZ has high precision but still predicts false EOS which can fragment a gold SU: in such cases, we consider the fragmented SUs as NSUs and fix the labels accordingly (just as we did in unit truncation, cf. §6.1 and Table 4).

As for the evaluation metrics, we convert the predictions of our methods into word/character-level BIO labels (cf. Appendix C) and compute the F1 score for each label prediction. Then, we summarize the results as the macro average F1 and weighted average F1. We also compute the F1 score of the exact SU span extraction at the word/character-level. Finally, we run each experiment (from model training to testing) five times with different random seeds and report the average and standard deviation as the final results.

7 Results

Table 5 summarizes the word-level evaluation results. The results for the character-level evaluation show similar tendencies, so we put them in Appendix E. The F1 score for each BIO label prediction is also available in Appendix E.

Firstly, we take a look at the results when we have the in-domain sentence identification dataset (EWT Train/Dev) for model development. In this setup, we can verify that our proposed method (BOS&EOS) significantly outperforms the baselines (EOS-Only) in all metrics. For instance, our method achieves consistently high performance of 84∼89% F1 for the exact SU span extraction, both at the word- and character-level. This is a very promising result that demonstrates the effectiveness of our method when we can leverage the gold SUs and NSUs from the target domain.

Secondly, we focus on the results where we

[6] This EOS-only method is the most reasonable baseline to quantify the precise advantage from combining BOS labels in addition to EOS, which is proposed in our methods.

Train/Dev Datasets	Model	EWT Test ($p_{CC} = 0.5$)			EWT Test ($p_{CC} = 0$)			EWT Test (Postprocess)		
		BIO Macro	BIO Weighted	Span	BIO Macro	BIO Weighted	Span	BIO Macro	BIO Weighted	Span
EWT Train/Dev	EOS-Only	83.2±1.5	93.9±0.6	72.8±1.8	59.7±0.2	86.4±0.1	58.2±1.1	86.3±2.7	94.6±1.1	81.6±2.4
	EOS-Only (force last)	58.6±0.1	86.6±0.0	60.4±0.8	57.6±0.2	85.9±0.1	57.7±1.0	59.1±0.1	85.7±0.0	62.3±0.3
	BOS&EOS	**93.0**±1.4	**97.3**±0.6	**87.3**±1.6	**91.0**±1.8	**96.4**±0.7	**84.1**±2.6	**92.3**±1.0	**96.7**±0.4	**88.8**±0.9
WSJ Train/Dev	EOS-Only	71.7±0.7	88.9±0.4	59.2±2.4	56.9±0.6	85.2±0.3	48.2±2.5	71.5±0.3	87.8±0.3	67.8±0.3
	EOS-Only (force last)	57.5±0.3	86.2±0.2	53.6±2.1	55.4±0.7	85.0±0.3	48.2±2.5	58.9±0.1	85.7±0.0	61.1±0.2
	EOS-Only (+AUG)	66.4±1.5	88.3±0.4	59.5±1.4	58.3±0.5	86.1±0.3	54.4±2.5	71.1±1.3	88.5±0.6	66.2±1.9
	BOS&EOS	71.5±0.2	89.1±0.2	59.1±1.5	57.7±0.9	85.4±0.2	48.8±1.6	71.0±0.3	87.9±0.2	68.4±0.3
	BOS&EOS (+UNI)	70.4±0.7	88.2±0.3	60.0±1.1	63.3±0.8	86.0±0.4	53.0±1.3	70.8±0.4	87.6±0.2	68.4±0.1
	BOS&EOS (+UNI +AUG)	**72.5**±0.4	**89.5**±0.1	**66.6**±0.2	**72.4**±1.3	**89.1**±0.5	**63.7**±1.0	**74.3**±1.1	**89.6**±0.4	**71.9**±1.4

Table 5: **Overall Results** (Word-Level). We report the macro/weighted average F1 of the BIO labeling task and the F1 score of the exact SU span extraction task. Details of our experimental setup are discussed in §6.

only utilize the standard sentence segmentation dataset (WSJ Train/Dev) for model development. In this setup, we also report the results of applying our data augmentation (+AUG) and unidirectional model (+UNI) techniques from §6.1.[7]

Due to the data discrepancy between WSJ and EWT, we find a natural drop in performance compared to the previous setup using in-domain EWT Train/Dev. However, we can verify that our techniques (+AUG, +UNI) generally help to alleviate this issue, and our proposed method performs on par or slightly better than the EOS-only baselines when applying these techniques. It is especially worth noting the improvement in the exact SU span extraction task (reaching 64∼72% F1), where the advantage of our method is the most conspicuous and consistent in both word- and character-level evaluation. This improvement can also be explained by the higher performance in the B-label prediction with our method (Appendix E), which is a prerequisite for accurate SU span extraction.

Finally, we note that the EOS-only baseline without forcing the last EOS can be quite competitive with shorter inputs ($p_{CC} = 0.5$ and postprocessing) but performs considerably worse when the input texts are longer ($p_{CC} = 0$). This is because the baseline can only predict the last segment of the input as an NSU, which is less problematic when the input texts are shorter but becomes increasingly problematic with longer inputs (since most NSUs will not be able to be removed). In contrast, our proposed method performs much more robustly under various input lengths.

Through further experiments and analyses, we found that (i) the results are stable across different hyperparameter choices, (ii) predictions are reasonable especially when using the in-domain dataset (EWT Train/Dev) for model development, and (iii) our methods do not sacrifice performance on the formal/edited texts of the sentence segmentation dataset (WSJ Test). These detailed evidences can be found in Appendix F.

[7]We did not observe any improvement from applying these techniques to the in-domain dataset (EWT Train/Dev), which is consistent with our motivation and expectation.

8 Conclusion

In this paper, we introduced a novel task of sentence identification, where we aim to identify SUs while excluding NSUs in a given text (§3). Through sentence identification, we can clearly distinguish the portions of the text that are appropriate (or not) for the prediction and evaluation of sophisticated linguistic analyses, such as dependency parsing, semantic role labeling, etc.

To conduct sentence identification, we proposed a simple yet effective method of combining the BOS and EOS labeling models to determine the SUs and NSUs (§4). To evaluate sentence identification, we designed an automatic, language-independent procedure to convert the UD corpora into sentence identification benchmarks (§5).

In our experiments, we developed the BOS/EOS labeling models by finetuning pretrained RoBERTa (§6). Based on the experimental results, we showed that our proposed method combining the BOS and EOS labels outperforms sentence segmentation baselines which only utilize EOS labels in all of the considered settings (§7). Overall, we expect sentence identification to be a fundamental framework for the preprocessing of noisy, informal, or non-standard texts in the real world.

Limitations

Firstly, our current experiments are limited to English and cover only five domains of web media texts in EWT. However, our task formulation (§3), method (§4), and evaluation framework (§5) are fully agnostic to the language and domain. Hence it is straightforward to conduct experiments in different languages or domains (as long as they are supported in the UD). While we expect similar results with different languages/domains, we leave further investigation as a future work.

Secondly, while our method performs reliably when the in-domain dataset is available, there is still a huge room left for improvement without relying on such resources (e.g. only using the standard sentence segmentation dataset). To make our method fully practical, we still need to improve on the accuracy and robustness in such cross-domain scenarios. One potential approach is to refine the definitions of SU and NSU probabilities from §4.2 to make sentence identification more robust. For instance, we can incorporate span-level scores instead of only using word-level BOS/EOS probabilities to define the SU/NSU probabilities. We leave further improvement and extension of our approach as an important future work.

Finally, our methods are currently evaluated on the (exact) SU span extraction task. Ideally, we should also evaluate the methods on downstream applications such as POS tagging, syntactic parsing, semantic role labeling, etc. However, we still expect that the (exact) SU span extraction will play a primary role in the evaluation, since accurate (say human-level) identification of SUs/NSUs will likely provide unprecedented benefits on a wide variety of NLP applications dealing with real-world texts. While we leave the precise analyses on downstream applications as future work, our contributions make the first foundational step towards expanding the capability of the long-established sentence segmentation task.

References

Wirote Aroonmanakun et al. 2007. Thoughts on word and sentence segmentation in thai. In *Proceedings of the Seventh Symposium on Natural language Processing*, pages 85–90.

Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal Dependencies. *Computational Linguistics*, 47(2):255–308.

Timothy Dozat and Christopher D Manning. 2017. Deep biaffine attention for neural dependency parsing. In *Proc. of ICLR*.

Timothy Dozat and Christopher D. Manning. 2018. Simpler but more accurate semantic dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 484–490, Melbourne, Australia. Association for Computational Linguistics.

Dan Gillick. 2009. Sentence boundary detection and the problem with the U.S. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 241–244, Boulder, Colorado. Association for Computational Linguistics.

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

Tibor Kiss and Jan Strunk. 2006. Unsupervised multilingual sentence boundary detection. *Computational Linguistics*, 32(4):485–525.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Geoffrey Nunberg. 1990. *The linguistics of punctuation*. 18. Center for the Study of Language (CSLI).

Jonathon Read, Rebecca Dridan, Stephan Oepen, and Lars Jørgen Solberg. 2012. Sentence boundary detection: A long solved problem? In *Proceedings of COLING 2012: Posters*, pages 985–994, Mumbai, India. The COLING 2012 Organizing Committee.

Ines Rehbein, Josef Ruppenhofer, and Thomas Schmidt. 2020. Improving sentence boundary detection for spoken language transcripts. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 7102–7111, Marseille, France. European Language Resources Association.

Michael D. Riley. 1989. Some applications of tree-based modelling to speech and language. In *Speech and Natural Language: Proceedings of a Workshop Held at Cape Cod, Massachusetts, October 15-18, 1989*.

Dwijen Rudrapal, Anupam Jamatia, Kunal Chakma, Amitava Das, and Björn Gambäck. 2015. Sentence boundary detection for social media text. In *Proceedings of the 12th International Conference on Natural Language Processing*, pages 254–260, Trivandrum, India. NLP Association of India.

Natalia Silveira, Timothy Dozat, Marie-Catherine de Marneffe, Samuel Bowman, Miriam Connor, John Bauer, and Chris Manning. 2014. A gold standard dependency corpus for English. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 2897–2904, Reykjavik, Iceland. European Language Resources Association (ELRA).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proc. of NeurIPS*.

Daniel J. Walker, David E. Clements, Maki Darwin, and Jan W. Amtrup. 2001. Sentence boundary detection: a comparison of paradigms for improving MT quality. In *Proceedings of Machine Translation Summit VIII*, Santiago de Compostela, Spain.

Xiaolin Wang, Masao Utiyama, and Eiichiro Sumita. 2019. Online sentence segmentation for simultaneous interpretation using multi-shifted recurrent neural network. In *Proceedings of Machine Translation Summit XVII: Research Track*, pages 1–11, Dublin, Ireland. European Association for Machine Translation.

Rachel Wicks and Matt Post. 2021. A unified approach to sentence segmentation of punctuated text in many languages. 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 3995–4007, Online. Association for Computational Linguistics.

Nianwen Xue and Yaqin Yang. 2011. Chinese sentence segmentation as comma classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 631–635, Portland, Oregon, USA. Association for Computational Linguistics.

Nina Zhou, AiTi Aw, Nattadaporn Lertcheva, and Xuancong Wang. 2016. A word labeling approach to Thai sentence boundary detection and POS tagging. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 319–327, Osaka, Japan. The COLING 2016 Organizing Committee.

A Dynamic Programming Algorithm

To find the maximum value (and the argmax) of Eq. 4 from §4.2, we rely on a simple dynamic programming framework. To be specific, we consider the partial labeling of BOS and EOS up to $\boldsymbol{W}^{\leq k} = (w_0, ..., w_k)$, where $k \leq N-1$. Then, we aim to compute the maximum log probability of Eq. 4 based on the partial labeling, i.e. using $\boldsymbol{W}^{\leq k}$ in place of $\boldsymbol{W}$.

Since the labeling is partial, $\boldsymbol{W}^{\leq k}$ may end inside the SU (i.e. the last label is BOS) or outside the SU (i.e. the last label is EOS). Let $\log\, p_{\text{IS}}(k{+}1)$ denote the maximum log probability when $\boldsymbol{W}^{\leq k}$ ends inside the SU and $\log\, p_{\text{OS}}(k+1)$ the maximum log probability when $\boldsymbol{W}^{\leq k}$ ends outside the SU. Then, we can initialize $\log\, p_{\text{IS}}(0) = \log\, 0 = -\infty$, $\log\, p_{\text{OS}}(0) = \log\, 1 = 0$ (since we always start outside the SU) and iteratively update the two values as follows:

$$\begin{aligned}
\log\, p'_{\text{IS}}(i) &= \max\{\log\, p_{\text{IS}}(i) + \log\,(1{-}p_{\text{BOS}}(w_i)),\\
&\qquad \log\, p_{\text{OS}}(i) + \log\, p_{\text{BOS}}(w_i)\}\\
\log\, p'_{\text{OS}}(i) &= \log\, p_{\text{OS}}(i) + \log\,(1{-}p_{\text{BOS}}(w_i))\\
\log\, p_{\text{IS}}(i{+}1) &= \log\, p'_{\text{IS}}(i) + \log\,(1{-}p_{\text{EOS}}(w_i))\\
\log\, p_{\text{OS}}(i{+}1) &= \max\{\log\, p'_{\text{IS}}(i) + \log\, p_{\text{EOS}}(w_i),\\
&\qquad \log\, p'_{\text{OS}}(i) + \log\,(1{-}p_{\text{EOS}}(w_i))\}
\end{aligned} \tag{5}$$

Note that we first update $p_{\text{IS}}(i) \rightarrow p'_{\text{IS}}(i)$ and $p_{\text{OS}}(i) \rightarrow p'_{\text{OS}}(i)$ based on the BOS probability $p_{\text{BOS}}(w_i)$. Then, we update $p'_{\text{IS}}(i) \rightarrow p_{\text{IS}}(i{+}1)$ and $p'_{\text{OS}}(i) \rightarrow p_{\text{OS}}(i{+}1)$ based on the EOS probability $p_{\text{EOS}}(w_i)$.[8] The iterative procedure is illustrated in Figure 2.

Finally, we can compute the log probability $\log\, p_{\text{OS}}(N)$ (since we always end outside the SU), which corresponds to the maximum value of Eq. 4. To obtain the argmax, we can simply incorporate backtracking during the iterative updates of Eq. 5. Through this dynamic programming framework, we can ensure that the restrictions from §4.2 are satisfied: namely, (i) the first label should be BOS, (ii) the last label should be EOS, and (iii) BOS and EOS labels need to appear alternately.

In practice, we can limit the candidates of BOS indices to the subset where $p_{\text{BOS}}(w_i)$ is higher than a certain threshold c. This can be efficiently implemented by simply skipping the updates of $p'_{\text{IS}}(i)$ and $p'_{\text{OS}}(i)$, i.e. using $p'_{\text{IS}}(i) = p_{\text{IS}}(i)$ and $p'_{\text{OS}}(i) = p_{\text{OS}}(i)$, if $p_{\text{BOS}}(w_i) < c$.[9] Likewise, we can limit the candidates of EOS indices by skipping the updates of $p_{\text{IS}}(i+1)$ and $p_{\text{OS}}(i+1)$ if $p_{\text{EOS}}(w_i) < c$. Generally speaking, this leads to a more efficient algorithm: therefore, we use the candidate threshold of $c = 0.1$ for restricting both BOS and EOS indices throughout our experiments.

[8] Note that if a single word w_i is labeled as both BOS and EOS at the same time, we can extract it as a single SU.

[9] This is equivalent to forcing w_i to be non-BOS, i.e. setting $p_{\text{BOS}}(w_i) = 0$ in Eq. 5.

B SU and NSU Examples

In Table 6, we provide more examples of SUs and NSUs identified based on our procedure described in §5. As for the SUs, we can verify that EWT contains clean, formal SUs with appropriate capitalization and punctuation. We can also verify that EWT contains various types of *informal* SUs, e.g. that lack capitalization/punctuation, use non-standard casing, end with emoticons, include spelling errors, concatenate consecutive SUs without a space, etc.

C Label Assignment and Conversion

In this section, we explain the precise procedure on how we (i) assign the gold character-level labels, (ii) convert the character-level labels to word/subword-level labels, and (iii) convert the subword-level labels to character/word-level labels. We limit our explanation to BIO labels, since it is straightforward to convert them to the combination of BOS and EOS labels (and vice versa).

Firstly, we can assign the gold character-level labels from the UD annotation by taking the character-level alignment, which determines the exact spans of SUs and NSUs. From the character-level labels, we can assign the word- or subword-level labels based on the following rule:

- If the word (or subword) contains a character with the B-label, assign it the B-label.
- Else if it contains a character with the I-label, assign the I-label.
- Otherwise assign the O-label.

For instance, this procedure is used to create the subword-level labels for training our BOS/EOS labeling models.

To evaluate our methods, we need to convert the subword-level labels produced by our methods into the character-level labels, which can then be converted into the word-level labels (based on the previous procedure). To convert a subword-level label into a sequence of character-level labels, we

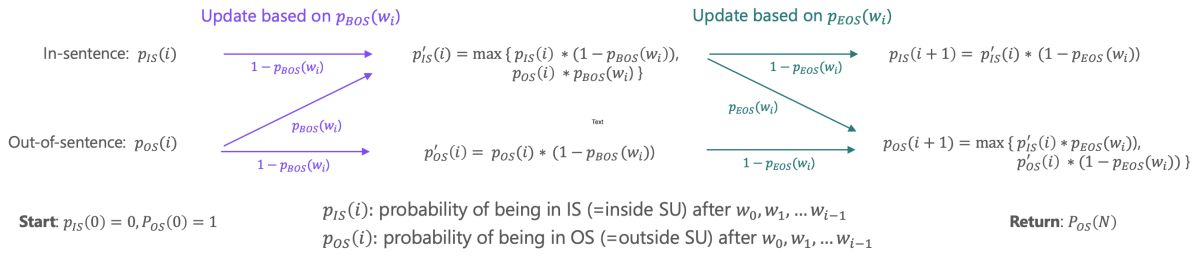


Figure 2: Illustration of the dynamic programming procedure.

SUs	President Bush on Tuesday nominated two individuals to replace retiring jurists on federal courts in the Washington area. Unfortunately, Mr. Lay will be in San Jose, CA participating in a conference, where he is a speaker, on June 14. “In 1972, there was an enormous glut of pilots,” Campenni says. PS – There is a happy hour tonight at Scudeiros on Dallas Street (just west of the Met Garage) beginning around 5:00. 2) Your vet would not prescribe them if they didn’t think it would be helpful. BUT EVERYONE HAS THERE OWN WAY!!!!!! The motel is very well maintained, and the managers are so accomodating, it’s kind of like visiting family each year! ;-) where can I find the best tours to the Mekong Delta at reasonable prices? it seems like its healthier too, but its prolly not. I have wifi at my house, but thats just at my house...is there anyway i can buy some card to make the ipod itself have wifi?
NSUs	—->===}*{===<—- - Lisa_coverletter.doc << File: Lisa_coverletter.doc >> Thur. Sept. 28 - Paris (Versailles or Fontainbleu - half day side trip) 9.3m - Number of US unemployed in April 2004. Game 1: Monday, May 28 @ 2:00PM vs. Los Angeles SPARKS Mixed Tempura.....8.25 Shrimp or vegetable tempura & salad. Infinity stereo, bucket seats, nerf bars, tool box, bed liner, camper tow package, 5 speed manual. printing, printing, copies, printing, copies, printing, A++++ !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Dear Sir / Madam,

Table 6: Examples of gold SUs and NSUs in the English Web Treebank (EWT) identified based on our procedure (§5). Each line corresponds to one example of SU or NSU.

apply the following rule (where n denotes the number of characters in the subword):

- If the subword has the B-label, the character-level labels are 1 B-label followed by $n - 1$ I-labels.
- If the subword has the I-label, the character-level labels are n I-labels.
- If the subword has the O-label, the character-level labels are n O-labels.

D Details on the Model Setup

As discussed in §6.1, we finetune the pretrained RoBERTa$_{\text{BASE}}$ publicly available on the HuggingFace model hub[10]. We add a binary BOS/EOS classifier on top of the encoder, which is a single-layer MLP with a hidden size of 768. We share the encoder parameters and use different classifiers for the BOS/EOS predictions. The BOS/EOS models are trained jointly by summing their losses.

When we combine the unidirectional models (+UNI), we take the same approach and use different classifiers for the unidirectional/bidirectional models. Again, the encoder parameters are shared and all models are trained jointly.

As for the training data preparation, we apply the unit concatenation and data augmentation (+AUG) *on the fly*, i.e. we see different concatenation and augmentation of the units in each iteration. The same procedure is applied on the validation set.

During data augmentation, we remove the last sentence ending punctuation based on the following regular-expression, similar to the candidate boundary detector in ERSATZ (Wicks and Post, 2021):

- $(.^*P_eP^*)$ where P denotes the set of punctuations and $P_e \subset P$ denotes the sentence ending punctuations.

Since our experiments are conducted on English, we use $P = \{.?!")'\}$ and $P_e = \{.?!\}$.

Finally, all models are implemented in Pytorch and trained on a single Tesla V100-SXM2-32GB

[10] https://huggingface.co/models

GPU. We use a batch size of 8, accumulate the gradients for 32 batches, and apply the gradient clipping at 1.0 before updating the model weights. As for the optimizer, we use Adam with the initial learning rate of 0.0001 and exponentially decay the learning rate by $\gamma = 0.95$ after each epoch. We check the validation loss every 200 batches and stop the training early if there is no improvement for 5 consecutive evaluations.

E The Full Experimental Results

In this section, we report the full results of our experiments which did not fit in §7. Table 7 shows the word-level F1 scores for each B-, I-, and O-label prediction. Table 8 shows the overall results for the character-level evaluation.

Generally speaking, we can confirm the same results as observed in §7. Firstly, our proposed method significantly outperforms the baselines when we use the EWT Train/Dev dataset for model development. Secondly, our method performs slightly better than (or at least on par with) the baselines when developed on the WSJ Train/Dev dataset. Finally, the baseline without forcing the last EOS is competitive with shorter inputs ($p_{CC} = 0.5$ and postprocessing) but performs considerably worse when the input texts are longer ($p_{CC} = 0$).

F Further Experiments and Analyses

In this section, we provide further experiments and analyses to complement our study. To be specific, we provide discussions on the effect of the choice of hyperparameters (F.1), qualitative analyses based on example model outputs (F.2), and evaluation of sentence identification based on the sentence segmentation dataset (F.3).

F.1 Effect of Hyperparameters

As a default configuration, we used $p_{DA} = 0.3, p_{TR} = 0.1$ for the data augmentation (+AUG) and $\lambda = 0.5$ for the unidirectional model ensembling (+UNI). To examine the effect of the choice of these hyperparameters, we conducted further experiments by changing these default hyperparameters. Note that all evaluation results in this subsection are based on BOS&EOS (+UNI +AUG) developed on WSJ Train/Dev.

Firstly, we focus on the data augmentation and report the results of our method trained with different sets of p_{DA} and p_{TR} (with λ fixed at 0.5). Since increasing p_{DA} leads to higher recall (and lower precision) of SU extraction and increasing p_{TR} leads to higher precision (and lower recall), we used a fixed ratio of $p_{DA} : p_{TR} = 3 : 1$ which seemed to make a good trade-off. As shown in Table 9, the results are generally stable with the different choices of the hyperparameters. However, more data augmentation (with larger values of p_{DA} and p_{TR}) tends to slightly improve the performance, especially for the exact SU span extraction.

Secondly, we focus on the unidirectional model ensembling and report the results of changing the linear interpolation rate $\lambda \in [0, 1]$, where $\lambda = 0$ is equivalent to using only the bidirectional models and $\lambda = 1$ only the unidirectional models. We fix $p_{DA} = 0.3$ and $p_{TR} = 0.1$ and only change λ at the inference time without retraining the unidirectional or bidirectional models. As shown in Figure 3, we found that unidirectional and bidirectional models generally have complementary benefits, and choosing the intermediate value of λ leads to the best performance. The results also indicate that we may be able to obtain further improvement by tuning λ on the validation set, although we simply fixed $\lambda = 0.5$ throughout our experiments.

F.2 Qualitative Analyses

In Table 10 and 11, we show the actual predictions made by our proposed method developed on EWT Train/Dev and WSJ Train/Dev. For the latter, we applied +UNI and +AUG with the default hyperparameters.

In the first example (Table 10), we can verify that both models identify the correct SU span while removing the non-sentential header as the NSU. This is a relatively easy example, since the start of the SU is capitalized and less ambiguous.

In the second example (Table 11), we can observe that our method using in-domain data (EWT Train/Dev) extracts the correct SU span, while our method developed on out-of-domain data (WSJ Train/Dev) incorrectly excludes a part of an SU. This seems to be a relatively difficult example, since the start of the SU is not capitalized and more ambiguous. It is worth noting that such SUs can be reliably extracted when we can leverage the in-domain annotation of gold SUs and NSUs.

F.3 Evaluation on the Sentence Segmentation Dataset

Finally, we report the results of sentence identification on the standard sentence segmentation dataset (WSJ Test).

Train/Dev Datasets	Model	EWT Test ($p_{CC} = 0.5$)			EWT Test ($p_{CC} = 0.0$)			EWT Test (Postprocess)		
		B-Label	I-Label	O-Label	B-Label	I-Label	O-Label	B-Label	I-Label	O-Label
EWT Train/Dev	EOS-Only	85.6±0.8	97.3±0.3	66.6±3.5	78.0±0.6	95.1±0.1	6.0±0.4	90.2±1.2	97.5±0.5	71.3±6.6
	EOS-Only (force last)	79.8±0.2	95.9±0.0	0.0±0.0	77.8±0.6	95.1±0.1	0.0±0.0	81.7±0.2	95.7±0.0	0.0±0.0
	BOS&EOS	**94.3**±0.6	**98.7**±0.3	**86.1**±3.5	**93.0**±1.0	**98.2**±0.4	**81.7**±4.0	**94.7**±0.3	**98.4**±0.2	**83.9**±2.4
WSJ Train/Dev	EOS-Only	78.7±1.3	94.4±0.4	**42.1**±1.3	71.8±1.9	94.3±0.2	4.5±0.4	83.3±0.2	93.8±0.3	37.3±0.9
	EOS-Only (force last)	76.7±0.9	**95.6**±0.1	0.0±0.0	71.7±1.9	**94.6**±0.2	0.0±0.0	81.0±0.4	**95.6**±0.0	0.0±0.0
	EOS-Only (+AUG)	79.4±1.0	95.4±0.2	24.5±4.1	78.1±1.8	93.3±0.2	1.4±1.1	82.7±1.1	94.9±0.5	35.6±3.2
	BOS&EOS	79.4±0.9	94.8±0.2	40.5±0.6	72.9±1.2	94.3±0.2	5.8±2.0	83.9±0.2	94.1±0.2	35.2±0.9
	BOS&EOS (+UNI)	79.8±0.6	93.9±0.2	37.5±1.5	76.2±1.3	93.3±0.3	20.2±1.2	83.8±0.1	93.8±0.1	34.7±0.9
	BOS&EOS (+UNI +AUG)	**83.7**±0.1	95.0±0.2	38.7±1.2	**83.0**±0.6	**94.6**±0.3	**39.7**±3.5	**85.9**±0.6	95.2±0.2	**41.9**±2.8

Table 7: **BIO Labeling Results** (Word-Level). We report the F1 scores for each B-, I- and O-label prediction.

Train/Dev Datasets	Model	EWT Test ($p_{CC} = 0.5$)			EWT Test ($p_{CC} = 0.0$)			EWT Test (Postprocess)		
		BIO Macro	BIO Weighted	Span	BIO Macro	BIO Weighted	Span	BIO Macro	BIO Weighted	Span
EWT Train/Dev	EOS-Only	83.8±1.1	92.7±0.5	72.8±1.8	58.5±0.2	81.5±0.0	58.2±1.1	87.7±2.3	93.9±1.2	81.6±2.4
	EOS-Only (force last)	57.7±0.1	81.0±0.0	60.4±0.8	56.9±0.2	80.9±0.0	57.7±1.0	58.1±0.1	79.9±0.0	62.3±0.3
	BOS&EOS	**94.0**±1.0	**97.2**±0.6	**87.3**±1.6	**92.2**±1.5	**96.3**±0.7	**84.1**±2.6	**93.5**±0.6	**96.6**±0.4	**88.9**±0.8
WSJ Train/Dev	EOS-Only	**72.8**±0.6	86.9±0.4	59.1±2.3	56.0±0.6	80.9±0.1	48.2±2.5	**73.3**±0.4	85.6±0.2	67.7±0.4
	EOS-Only (force last)	56.6±0.3	80.9±0.0	53.5±2.0	54.9±0.6	80.7±0.1	48.2±2.5	57.8±0.2	79.9±0.0	61.0±0.3
	EOS-Only (+AUG)	64.3±1.5	83.5±0.6	59.5±1.4	57.4±0.5	81.0±0.2	54.4±2.5	69.2±1.7	84.0±0.8	66.2±1.9
	BOS&EOS	72.7±0.7	**87.1**±0.2	59.1±1.5	57.8±1.9	81.6±0.7	48.8±1.6	72.4±1.0	85.2±0.5	68.3±0.3
	BOS&EOS (+UNI)	72.4±0.6	86.3±0.3	59.6±1.0	65.3±1.0	83.6±0.5	52.9±1.3	72.8±0.4	85.3±0.2	68.0±0.2
	BOS&EOS (+UNI +AUG)	72.2±1.3	86.1±0.6	**66.5**±0.3	**72.8**±1.8	**86.5**±0.9	**63.6**±1.0	73.2±1.9	**85.7**±0.9	**71.8**±1.5

Table 8: **Overall Results** (Character-Level). We report the macro/weighted average F1 of the BIO labeling task and the F1 score of the exact SU span extraction task.

In Table 12, we summarize the WSJ dataset statistics. Note that WSJ only contains SUs and do not contain any NSUs (O-labels). However, we can still evaluate the performance using the same metrics, i.e. the macro/weighted average F1 of the BIO labeling task and the F1 of the exact SU span extraction task.[11]

Table 13 summarizes the word-level evaluation results. Since we are evaluating on WSJ Test, the performance is naturally better when the models are trained on WSJ Train/Dev rather than EWT Train/Dev (which is now out-of-domain).

When the models are trained on EWT, we found that the baseline (EOS-Only) forcing the last EOS performs the best. This is natural, since this baseline better reflects the nature of the sentence segmentation dataset where all units are SUs. However, our method (BOS&EOS) is still comparable to this baseline and do not (or minimally) sacrifice performance on such datasets.

When the models are trained on WSJ, we found that our method without +UNI or +AUG performs the best. This is most likely because we can leverage the knowledge of BOS to predict EOS. When we apply the data augmentation (+AUG) and unidirectional model ensembling (+UNI), we observe a slight decrease in performance compared to our vanilla method. However, the results are still comparable and even outperforms the baselines in some metrics (e.g. the exact SU span extraction task).

Overall, we can conclude that our methods do not sacrifice the performance on the the clean, edited texts of the sentence segmentation dataset.

[11] Since the O-label does not exist, we report the macro average F1 as the average F1 scores of the B-label and I-label predictions.

Evaluation	Augmentation Rates	EWT Test ($p_{CC} = 0.5$)			EWT Test ($p_{CC} = 0$)			EWT Test (Postprocess)		
		BIO Macro	BIO Weighted	Span	BIO Macro	BIO Weighted	Span	BIO Macro	BIO Weighted	Span
Word-Level	$p_{DA}=0.15, p_{TR}=0.05$	71.3±1.1	89.0±0.5	65.7±1.3	71.5±0.9	88.6±0.5	62.3±1.7	73.5±1.4	89.2±0.6	71.2±1.8
	$p_{DA}=0.3, \ p_{TR}=0.1$	72.5±0.4	89.5±0.1	66.6±0.2	72.4±1.3	89.1±0.5	63.7±1.0	74.3±1.1	89.6±0.4	71.9±1.4
	$p_{DA}=0.45, p_{TR}=0.15$	**73.2**±1.0	**90.0**±0.1	**67.3**±0.8	**73.0**±0.9	**89.5**±0.6	**64.0**±1.8	**75.1**±1.3	**90.0**±0.4	**72.1**±0.7
Character-Level	$p_{DA}=0.15, p_{TR}=0.05$	**72.3**±1.9	**86.3**±1.0	65.4±1.4	**73.6**±0.7	**86.8**±0.3	62.2±1.7	**73.8**±1.5	**86.1**±0.8	71.0±1.8
	$p_{DA}=0.3, \ p_{TR}=0.1$	72.2±1.3	86.1±0.6	66.5±0.3	72.8±1.8	86.5±0.9	63.6±1.0	73.2±1.9	85.7±0.9	71.8±1.5
	$p_{DA}=0.45, p_{TR}=0.15$	71.9±1.1	86.1±0.3	**67.2**±0.8	72.3±0.9	86.3±0.6	**64.0**±1.8	73.6±1.5	86.0±0.6	**72.1**±0.7

Table 9: **Effect of Data Augmentation Rates** (Word/Character-Level). We use different data augmentation rates (p_{DA} and p_{TR}) and evaluate BOS&EOS (+UNI +AUG) developed on WSJ Train/Dev. We report the macro/weighted average F1 of the BIO labeling task and the F1 score of the exact SU span extraction task.

Developed on EWT	... 06/04/2001 05:54 PM **B** Can you pass this along to Elizabeth to ensure Sanders **E** is on board as well?
Developed on WSJ	... 06/04/2001 05:54 PM **B** Can you pass this along to Elizabeth to ensure Sanders **E** is on board as well?

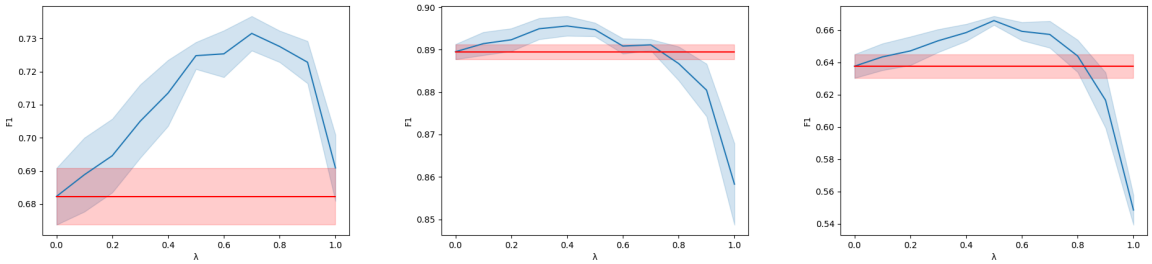
Table 10: **Example Outputs (Both Correct)**. We show the predictions made by our proposed method (BOS&EOS) developed on EWT Train/Dev (top) or WSJ Train/Dev (bottom). We can verify that both methods identify the correct SU span while removing the non-sentential header as the NSU.

Developed on EWT	**B** with my breakfast I like bacon and sausage when I having a big breakfast like **E** a grand slam with pancakes and the works.
Developed on WSJ	with my breakfast **B** I like bacon and sausage when I having a big breakfast like **E** a grand slam with pancakes and the works.

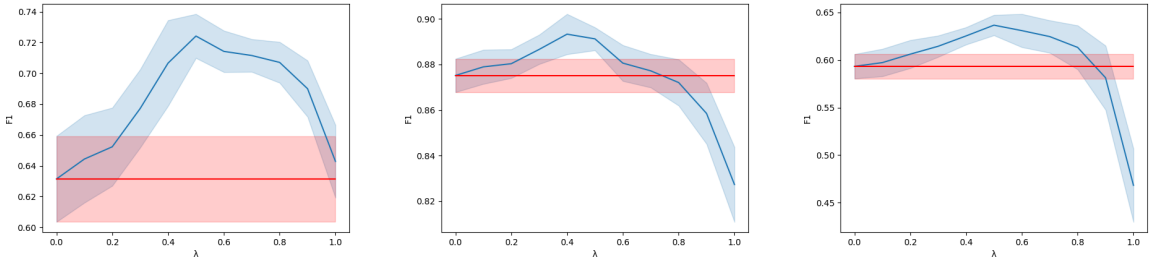
Table 11: **Example Output with One Incorrect Case**. We show the predictions made by our proposed method (BOS&EOS) developed on EWT Train/Dev (top) or WSJ Train/Dev (bottom). We can verify that the former extracts the correct SU span, while the latter incorrectly excludes the first prepositional phrase as an NSU.

		Train	Dev	Test
Total SUs		37,447	2,021	7,442
Total NSUs		0	0	0
Word-Level	B-Label	37,447	2,021	7,442
	I-Label	805,387	44,354	163,132
	O-Label	0	0	0
Character-Level	B-Label	37,447	2,021	7,442
	I-Label	4,308,729	236,798	876,461
	O-Label	0	0	0

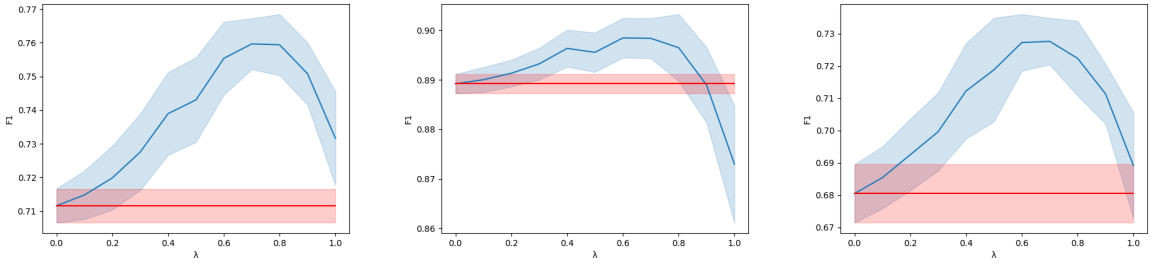
Table 12: WSJ dataset statistics.



(a) EWT Test ($p_{CC}=0.5$), BIO Macro (b) EWT Test ($p_{CC}=0.5$), BIO Weighted (c) EWT Test ($p_{CC}=0.5$), Span



(d) EWT Test ($p_{CC}=0.0$), BIO Macro (e) EWT Test ($p_{CC}=0.0$), BIO Weighted (f) EWT Test ($p_{CC}=0.0$), Span



(g) EWT Test (Postproc.), BIO Macro (h) EWT Test (Postproc.), BIO Weighted (i) EWT Test (Postproc.), Span

Figure 3: **Effect of the Unidirectional Model Interpolation Rate** (Word-Level). We change $\lambda \in [0, 1]$ and report the macro/weighted average F1 of the BIO labeling task and the F1 score of the exact SU span extraction task. Interpolated results are shown in blue and non-interpolated results (i.e. $\lambda = 0$) shown in red. The line shows the mean and the shade shows the standard deviation from the five experimental runs.

Train/Dev Datasets	Model	WSJ Test ($p_{CC}=0.5$)			WSJ Test ($p_{CC}=0$)		
		BIO Macro	BIO Weighted	Span	BIO Macro	BIO Weighted	Span
EWT Train/Dev	EOS-Only	97.4±0.1	99.5±0.0	87.3±0.3	**97.3±0.0**	99.5±0.0	87.2±0.2
	EOS-Only (force last)	**97.6±0.1**	**99.9±0.0**	**87.8±0.3**	**97.3±0.0**	**99.6±0.0**	**87.3±0.2**
	BOS&EOS	97.1±0.2	99.4±0.0	86.7±0.5	97.0±0.1	99.3±0.0	86.5±0.3
WSJ Train/Dev	EOS-Only	98.4±0.6	**99.7±0.1**	92.1±2.9	98.2±0.4	**99.7±0.1**	90.6±1.8
	EOS-Only (force last)	98.4±0.6	**99.7±0.1**	92.1±2.9	98.2±0.4	**99.7±0.1**	90.6±1.8
	EOS-Only (+AUG)	98.2±1.1	99.1±1.0	92.6±2.5	97.3±1.9	99.3±0.8	87.8±6.3
	BOS&EOS	**99.2±0.2**	**99.7±0.3**	**95.5±0.5**	**98.7±0.1**	**99.7±0.2**	**93.1±0.4**
	BOS&EOS (+UNI)	98.5±0.3	98.9±0.5	92.9±1.0	98.1±0.3	98.8±0.5	91.4±0.8
	BOS&EOS (+UNI +AUG)	98.7±0.2	99.3±0.4	94.0±0.7	98.2±0.3	99.1±0.3	91.8±1.1

Table 13: **Overall Results on WSJ Test** (RoBERTa, Word-Level). We report the macro/weighted average F1 of the BIO labeling task and the F1 score of the exact SU span extraction task.