Advancing Multi-Criteria Chinese Word Segmentation Through Criterion Classification and Denoising

Anonymous ACL submission

Abstract

Recent research on multi-criteria Chinese word segmentation (MCCWS) mainly focuses on building complex private structures, adding more handcrafted features, or introducing complex optimization processes. In this work, we show that through a simple yet elegant input-hint-based MCCWS model, we can achieve state-of-the-art (SoTA) performances on several datasets simultaneously. We further propose a novel criterion-denoising objective that hurts slightly on F1 score but acheives SoTA recall on out-of-vocabulary words. Our result establishes a simple yet strong baseline for future MCCWS research.

Dataset	Samples	Labels
PKU	江-泽民	S-BE
MSRA	江泽民	BME
AS	何-樂-而-不-為	S-S-S-S-S
CITYU	何樂而不為	BMMME

Table 1: Actual samples from SIGHAN bakeoff 2005 datasets (Emerson, 2005) demonstrating labeling inconsistency. The hyphen "-" denotes segmentation. Labels are defined in Section 3.1. In the first two rows, the human name 江泽民 (Jiang Zemin) in PKU dataset is segmented into the last name 江 (Jiang) and the first name 泽民 (Zemin), but not in MSRA dataset. In the last two rows, the idiom 何樂而不為 (Why not do something?) is segmented in AS dataset but not in CITYU dataset. More examples can be found in these datasets.

1 Introduction

Chinese word segmentation (CWS) is a preliminary step for performing Chinese NLP tasks. Researchers have proposed many CWS datasets to enhance word segmentation performance in different text domains. However, due to the divergence in linguistic perspectives, the same text passage can be segmented in entirely different ways across datasets. For example, in their written forms, Chinese human names have no spaces in between. Some datasets segment human names into last and first names, while others leave human names as a whole (see Table 1). The simplest way to address such an issue is through single-criterion CWS (SCCWS) model, i.e., to train different models for different datasets. But the cost of maintaining multiple versions of the same model becomes cumbersome as recent deep learning models get deeper and larger. Thus, recent CWS works started to shift their focuses to multi-criterion Chinese word segmentation (MCCWS), which aims to fit one model for all CWS datasets (Chen et al., 2017; He et al., 2019; Gong et al., 2019; Huang et al., 2020b,a; Ke et al., 2020; Qiu et al., 2020; Ke et al., 2021).

MCCWS can be seen as a multi-task learning problem (Chen et al., 2017) that benefits from leveraging large amounts of heterogeneous data, meanwhile dealing with subtle linguistic divergence. Prior works are mainly divided into private-structure-based and input-hint-based models. In a typical SCCWS workflow, an input character sequence is first converted to character embeddings and fed to an encoder to get contextualized representation. The encoder output is then passed to a decoder to generate the final prediction (see Figure 1(a)). In private-structure-based MCCWS, an encoder-decoder pair is created for each dataset, but an additional encoder is shared across datasets to better leverage general knowledge (see Figure 1(b)). In input-hint-based MCCWS, instead of creating private structures for each dataset, all datasets share one encoder-decoder pair, and a criterion-specific hint is given as part of the input (see Figure 1(c)). Despite its simplicity, input-hint-based MCCWS models outperform private-structure-based MCCWS models.

Proven to be simple and effective, the input-hint-based approach has become the most popular choice of recent MCCWS works (He et al., 2019; Gong et al., 2019; Huang et al., 2020a; Ke et al., 2020; Qiu et al., 2020; Ke et al., 2021). While

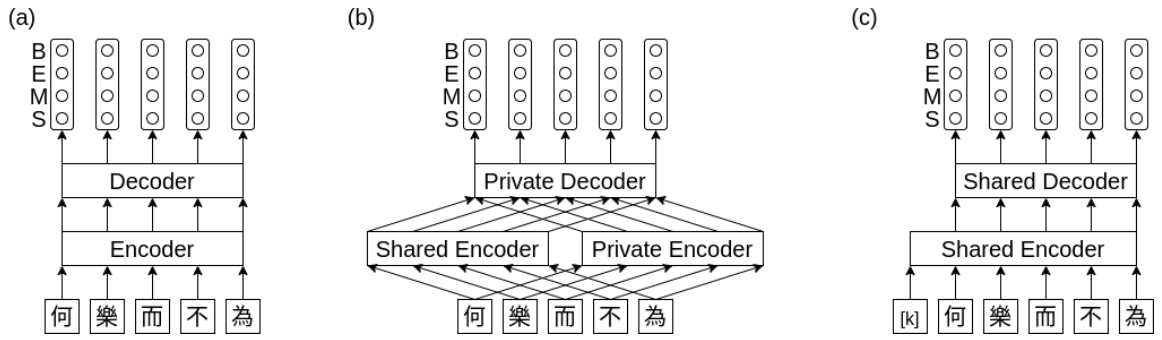

Figure 1: (a) Typical SCCWS model, (b) private-structure-based MCCWS model, and (c) input-hint-based MCCWS model. All three types of models share similar workflows. $\mathrm{B}, \mathrm{E}, \mathrm{M}, \mathrm{S}$ are collectively defined as the output tagset of a CWS model (see Section 3.1). The character sequence "何樂而不為" (Why not do something?) is used as an input demonstration. $[\mathrm{k}]$ represents the criterion of the k-th dataset and is served as an input hint. SCCWS and input-hint-based MCCWS models are nearly identical with the input being the only difference.

existing works kept adding complex features and structures, we show that without such complexity, we can still achieve state-of-the-art (SoTA) results across 10 CWS datasets. We do this by jointly training MCCWS with a criterion classification objective on a simple model. In particular, we used a pre-trained Chinese BERT (Devlin et al., 2019) as our encoder and a softmax decoder. Neither handcrafted features nor complex non-greedy decoding algorithms were used.

One problem remains for input-hint-based MCCWS models. When fitting on a training set or evaluating a test set, each character sequence is sampled from a particular dataset, so one would always know which criterion-specific hint was given as input. However, when performing inference, one would not know the source of a given character sequence. Therefore, one has to choose the criterion in such cases manually. With hundreds of linguistic rules (Emerson, 2005), it is difficult for non-linguists to determine which criterion to use. Thus, inspired by the masked language model, we proposed a novel criterion-denoising objective to make our MCCWS model automatically choose a suitable criterion for each input. We show that adding such a denoising objective surprisingly retains near SoTA performance on the F1-score, and even outperforms SoTA performance on the recall of out-of-vocabulary (OOV) words.

2 Related Works

After Xue (2003) proposed to treat CWS as a character tagging problem, many works followed the same problem formulation to address CWS. Chen et al. (2017) is the first to propose a multi-criteria learning framework for CWS. They proposed multiple private-structure-based MCCWS models and trained them in an adversarial setting. A criterion discriminator was used in their adversarial training so that common knowledge across datasets could be shared through different private structures. But the nature of adversarial training forces their criterion discriminator to predict each criterion with equal probability (Goodfellow et al., 2014; Chen et al., 2017). Thus their criterion discriminator failed to provide accurate criterion prediction and cannot be used to choose a suitable criterion for each input.

Inspired by the success of the BiLSTM-based SCCWS model (Ma et al., 2018) and input-hint-based multilingual neural machine translation system (Johnson et al., 2017), He et al. (2019) proposed to build an input-hint-based MCCWS on top of the BiLSTM. They added two artificial tokens representing a criterion and put them at the beginning and the end of an input sentence. Such a simple idea advanced the SoTA performance on seven datasets simultaneously. Gong et al. (2019) proposed switch-LSTMs, which can dynamically route between multiple BiLSTMs to encode criterion-specific features when given different input hints. Their work set the SoTA limit that can be achieved via LSTM architecture.

After the remarkable effectiveness of pre-trained language models was found, MCCWS works started to replace LSTM encoders with Transformer encoders (Vaswani et al., 2017). Huang et al. (2020a) used RoBERTa (Liu et al., 2019) to build an input-hint-based MCCWS model, which advanced SoTA performance. Huang et al. (2020b)

shows that adding private structures on top of a large pre-trained model can push SoTA even further. Ke et al. (2021) pre-trained an input-hint-based MCCWS on BERT (Devlin et al., 2019) with meta-learning (Finn et al., 2017), but only after fine-tuning did they become the new SoTA on SCCWS models.

Ke et al. (2020) and Qiu et al. (2020) are the most similar to ours among many MCCWS works. We use a nearly identical input-hint-based model as in Qiu et al. (2020). However, like all the works mentioned before, they do not include a criterion classification objective, and therefore fail to provide a way to choose criteria automatically. Ke et al. (2020) is the only work using criterion classification objective, but we further simplified its model structure, which outperforms their models on average F1-score. We further proposed a novel criterion-denoising objective that helps choose criteria automatically. By trading off 0.06% F1-score on average, we achieved the new SoTA on the OOV recall, which improved by a large margin compared to the previous SoTA (1.75%).

In summary, previous research on MCCWS either did not provide a way to choose a criterion or always manually chose a criterion. In our work, we proposed a simple yet elegant way to make our MCCWS model automatically choose a suitable criterion for the given character sequence. Comparing our works to others, we find that (1) our model has the simplest structure and is the easiest to implement among other works; (2) we achieved MCCWS SoTA performance on several CWS datasets and on average F1-score over 10 datasets; (3) we improved SoTA OOV recall by a large margin.

3 MCCWS

In this section, we describe the detail of our methodology. We first give a formal definition of input-hint-based MCCWS (Section 3.1). Then we introduce our MCCWS model (Section 3.2). Finally, we formally define our criterion-denoising objective and describe how to jointly train our MCCWS on top of the proposed denoising objective (Section 3.3).

3.1 Problem Definition

Let x be a character sequence. Denote the i-th character of sequence x as x_i, and the i-th output corresponds to x as y_i. Each y_i belongs to a tagset $\mathcal{T} = \{\mathrm{B}, \mathrm{M}, \mathrm{E}, \mathrm{S}\}$ where $\mathrm{B}, \mathrm{M}, \mathrm{E}$ represent the beginning, the middle, and the end of a word, and S represents a word with a single character. When receiving a character sequence x, a SCCWS model will pass x to its encoder (with parameter θ_{enc}) to generate the contextualized representation of x, then feed the encoder output to its decoder (with parameter θ_{dec}) to generate prediction y based on x, following the constraint of the tagset $\mathcal{T}$ (see Figure 1(a)). Typically, a decoder such as the conditional random field (CRF) (Lafferty et al., 2001) will search through all possible combinations and return the combination with the highest probability:

$$y^* = \arg\max_{y \in \mathcal{T}^{|x|}} \Pr(y \mid x; \theta_{\text{enc}}, \theta_{\text{dec}}), \qquad (1)$$

where $|x|$ denotes the number of characters of x. The goal of a SCCWS model with parameters θ_{enc} and θ_{dec} is to maximize the probability of y given x over all pairs of (x, y) in a CWS dataset $\mathcal{D}$. One can achieve this by minimizing the negative log-likelihood $\mathcal{L}$ over dataset $\mathcal{D}$:

$$\begin{aligned}&\mathcal{L}(\mathcal{D}, \theta_{\text{enc}}, \theta_{\text{dec}}) \\ &\quad = \min -\sum_{(x,y)\in\mathcal{D}} \log \Pr(y \,|\, x; \theta_{\text{enc}}; \theta_{\text{dec}}). \qquad (2)\end{aligned}$$

Now suppose there are K different CWS datasets $\{\mathcal{D}^k\}_{k=1}^K$. When receiving a character sequence x from the k-th dataset $\mathcal{D}^k$, an input-hint-based MCCWS model will combine x with the k-th criterion token $[\mathtt{k}]$ to form a new sequence (see Figure 1(c)). The new sequence is then processed as in Equation (1). Therefore, we can rewrite Equation (2) to define the minimization objective of an input-hint-based MCCWS model with parameters θ_{enc} and θ_{dec}:

$$\begin{aligned}&\mathcal{L}(\{\mathcal{D}\}_{k=1}^K, \theta_{\text{enc}}, \theta_{\text{dec}}) \\ &= \min -\sum_{k=1}^{K} \sum_{(x,y)\in\mathcal{D}^k} \log \Pr(y \,|\, x, [\mathtt{k}]; \theta_{\text{enc}}; \theta_{\text{dec}}).\end{aligned} \qquad (3)$$

Observe that the negative log-likelihood of y is conditioned on both x and $[\mathtt{k}]$, and the minimization is performed on all K datasets simultaneously instead of a single dataset.

3.2 Model Definition

Input Format. For each dataset $\mathcal{D}^k$ and each character sequence $x \in \mathcal{D}^k$, let

$$\mathbf{x} = [[\mathtt{CLS}]; [\mathtt{k}]; x; [\mathtt{SEP}]] \qquad (4)$$

be the new sequence formed by concatenating the [CLS] token, the k-th criterion token [k], character sequence x, and the [SEP] token. $\mathbf{x}$ is treated as a sequence with $3 + |x|$ characters and fed into our MCCWS encoder.

Encoder. We used a pre-trained Chinese BERT[1] as our encoder, and we denote the output of BERT as $\mathbf{h}$:

$$\mathbf{h} = \text{BERT}(\mathbf{x}; \theta_{\text{enc}}) \in \mathbb{R}^{(3+|x|)\times d_{\text{model}}}, \quad (5)$$

where d_{model} is the hidden dimension of BERT. Devlin et al. (2019) includes all details of BERT. Both [CLS] and [SEP] tokens are only used to follow the BERT input format with no further computations done on both tokens. We note that we neither use any private structures nor handcrafted features. Thus, our encoder architecture can be considered as the simplest among other MCCWS works.

Decoder. To keep our model simple, we choose a greedy decoding algorithm over a non-greedy one. We use one linear layer followed by a softmax normalization as our decoder. The output of BERT encoder $\mathbf{h}$, with starting index 3, is fed directly into our decoder:

$$\mathbf{y}_{i-2} = \text{softmax}(W^h \cdot \mathbf{h}_i + b^h) \in \mathbb{R}^4 \quad \text{for all } i \in \{3, \ldots, |x| + 2\}. \quad (6)$$

$W^h \in \mathbb{R}^{4\times d_{\text{model}}}$ and $b^h \in \mathbb{R}^4$ are trainable parameters, and 4 is the size of tagset $\mathcal{T}$. Our decoder will generate a sequence of probability vectors $\mathbf{y} = (\mathbf{y}_1, \ldots, \mathbf{y}_{|x|}) \in \mathbb{R}^{|x|\times 4}$. Since we use greedy decoding, we optimize our input-hint-based MCCWS model with cross-entropy loss instead of negative log-likelihood. So we change Equation (3) as follows:

$$\mathcal{L}(\{\mathcal{D}\}_{k=1}^{K}, \theta_{\text{enc}}, \theta_{\text{dec}}) = \min -\sum_{k=1}^{K} \sum_{(x,y)\in\mathcal{D}^k} \sum_{i=1}^{|x|} \mathbf{1}_{y_i} \odot \log \mathbf{y}_i, \quad (7)$$

where $\mathbf{1}_{y_i}$ denotes the one-hot encoding corresponding to y_i, $\odot$ denotes the Hadamard product, and $\log \mathbf{y}_i$ denotes performing log operation on probability vector $\mathbf{y}_i$ in an element-wise fashion.

[1] Pre-trained model checkpoint is available at https://huggingface.co/bert-base-chinese.

Criterion Classification To make our model remember the meaning of criterion hint [k] during the forward pass, we introduce a criterion classification task. We let our model predict which criterion hint it received. So we pick $\mathbf{h}_2$, the output of BERT that corresponds to the criterion token [k], and feed it into a criterion classifier which consists of one linear layer (different from our decoder) following a softmax normalization:

$$\mathbf{c} = \text{softmax}(W^c \cdot \mathbf{h}_2 + b^c) \in \mathbb{R}^K. \quad (8)$$

Both $W^c \in \mathbb{R}^{K\times d_{\text{model}}}$ and $b^c \in \mathbb{R}^K$ are trainable parameters. Our criterion classifier is set to minimize cross-entropy loss, just like Equation (7):

$$\mathcal{L}_{\mathbf{c}}(\{\mathcal{D}\}_{k=1}^{K}, \theta_{\text{enc}}, \theta_{\text{dec}}) = \min -\sum_{k=1}^{K} \sum_{(x,y)\in\mathcal{D}^k} \mathbf{1}_{[\text{k}]} \odot \log \mathbf{c}, \quad (9)$$

where $\mathbf{1}_{[\text{k}]}$ denotes the one-hot encoding that corresponds to [k] and $\log \mathbf{c}$ denotes the element-wise log operation on the probability vector $\mathbf{c}$.

Total Loss Combining Equations (7) and (9), we get our final loss $\mathcal{L}_{\text{final}}$:

$$\mathcal{L}_{\text{final}}(\{\mathcal{D}\}_{k=1}^{K}, \theta_{\text{enc}}, \theta_{\text{dec}}) = \mathcal{L}(\{\mathcal{D}\}_{k=1}^{K}, \theta_{\text{enc}}, \theta_{\text{dec}}) + \mathcal{L}_{\mathbf{c}}(\{\mathcal{D}\}_{k=1}^{K}, \theta_{\text{enc}}, \theta_{\text{dec}}). \quad (10)$$

We jointly train both objectives on our input-hint-based MCCWS model. Surprisingly, this joint objective gives us SoTA performance on several datasets.

3.3 Criterion Denoising

To avoid manually giving criterion tokens, we design a criterion-denoising objective to make our model choose the suitable criterion for each input. We define a token [UNC], which stands for "unknown criterion," and we randomly replace each pairing criterion [k] with [UNC]. In this situation, the goal of our criterion classifier (see Equation (8)) is to find the best fitting criterion for the given input x. So Equation (9) becomes a denoising objective, in a similar way to the masked language model objective used in BERT. After training with [UNC], the model can choose a suitable criterion for x and perform CWS simultaneously, all in just a single forward pass. We show that such an auto mechanism does not harm the performance, making our model effective and practical.

4 Experiments

4.1 Datasets

We perform experiments on 10 CWS datasets (this means $K = 10$). Four datasets are from the SIGHAN2005 bakeoff (Emerson, 2005), including AS, CITYU, PKU, and MSRA; SXU is from the SIGHAN2008 bakeoff (Jin and Chen, 2008); the rest are CNC[2], CTB6 (Xue et al., 2005), UD (Zeman et al., 2018), WTB (Wang et al., 2014) and ZX (Zhang et al., 2014). Following Emerson (2005), we report the F1-score and OOV recall.

Our preprocessing mainly follows the works of He et al. (2019) and Chen et al. (2017), as done by others. We first convert all full-width characters into half-width. Then, we replace different consecutive digits into one token (we do the same for alphabets). Unlike others who set the maximum sentence length to 128 or lower to speed up the training process, we decide to utilize the full computing power of BERT and include as many characters in the same context as possible. So we set the maximum sentence length to 512. For sentences longer than 512, we try to find the nearest punctuation as our delimiter, otherwise, we split on the 512th character. The statistics for all datasets can be found in Appendix A.

4.2 Hyperparameters

We use PyTorch (Paszke et al., 2019) to implement our model. We fine-tune BERT with AdamW (Loshchilov and Hutter, 2019) on the pre-trained checkpoint `bert-base-chinese` provided by huggingface (Wolf et al., 2019) (this means $d_{\text{model}} = 768$ and the number of parameters is around 110M). Moving average coefficients (β_1, β_2) of AdamW are set to $(0.9, 0.999)$. The learning rate is set to 2×10^{-5}, and the weight decay coefficient is set to 0.01. We schedule the learning rate with linear warmup and linear decay. The warmup ratio is set to 0.1, and the total training step is set to 170000. Dropout (Srivastava et al., 2014) is applied with a probability of 0.1. We set the batch size to 32, and use gradient accumulation with two steps (this is almost equivalent to setting the batch size to 64). We use label smoothing only on the decoder but not on the criterion classifier, and we set the smoothing value to 0.1. We pick the checkpoint with the highest F1 on the development set to calculate test set F1. For each experiment reported later, we ran each over 5 random seeds and reported only the best result. The results of all trials are listed in Appendix A. All experiments were run on a single Intel Xeon Silver 4216 CPU and an Nvidia RTX 3090 GPU.

[2] `http://corpus.zhonghuayuwen.org/`

4.3 Main Results

SoTA F1-score. Table 2 shows our results on F1 over 10 CWS datasets. Our MCCWS model (denoted as "Ours") achieves SoTA results on 5 out of 10 datasets. Since not all works performed experiments on all the same 10 datasets, we also report average results on the most common 4 (denoted as Avg.4) and 6 (denoted as Avg.6) datasets. Results show that our model is ranked 2nd under Avg.4 and Avg.6, which is only 0.14% and 0.05% less than the best-performing model respectively. We note that Huang et al. (2020b) used a private-structure-based MCCWS with CRF decoder, therefore, has way more parameters than our proposed model. Nevertheless, our model achieves the SoTA performance on average over 10 datasets (denoted as Avg.10). Therefore, despite the simplicity, our model still performs well against strong baselines.

Noisy but near SoTA. In Section 3.3, we proposed a criterion-denoising objective. We randomly select 10% criterion tokens for each mini-batch and replace them with [`UNC`]. Table 2 shows the performance of our criterion denoising MCCWS model (denoted as ours+10%[`UNC`]). We see that the denoising version of our model beats the previous SoTA on Avg.10 and even achieved the new SoTA on two datasets. This shows that our criterion-denoising objective does not hinder the performance, but helps our model advance to near SoTA results.

SoTA OOV Recall. Table 3 shows our results on OOV recall over 10 CWS datasets. Our models achieve SoTA results on 8 out of 10 datasets with or without criterion-denoising objective. CWS task is challenging when the word boundary is ambiguous, which can only be eased by giving enough context. Thus, we attribute the remarkable OOV recall improvement to our preprocessing step, for which we set the maximum input length to 512, giving our model enough context to identify unseen words. We will further discuss this result in Section 4.4. But with the help of our criterion-denoising objective, we see that OOV recall is boosted even higher, showing the effectiveness of our criterion-denoising objective.

MCCWS Models	AS	CITYU	CNC	CTB6	MSRA	PKU	SXU	UD	WTB	ZX	Avg.4	Avg.6	Avg.10
Model-I+ADV[a]	94.64	95.55	-	96.18	96.04	94.32	96.04	-	-	-	95.14	95.46	-
BiLSTM+CRF-4[b]	95.40	96.20	-	-	97.40	95.90	-	-	-	-	96.26	-	-
BiLSTM+CRF-8[b]	95.47	95.60	-	95.84	97.35	95.78	96.49	-	-	-	96.05	96.09	-
Switch-LSTMs[c]	95.22	96.22	-	97.62	97.78	96.15	97.25	-	-	-	96.34	96.71	-
RoBERTa+softmax[d]	-	-	97.19	97.56	98.29	96.85	97.56	97.69	-	96.46	-	-	-
BERT+CRF[e]	97.00	97.80	97.30	97.80	98.50	**97.3**	97.50	97.80	93.20	97.10	**97.65**	**97.65**	97.13
Transformer+CRF[f]	96.44	96.91	-	96.99	98.05	96.41	97.61	-	-	-	96.95	97.07	-
Unified BiLSTM[g]	95.47	95.60	-	95.84	97.35	95.78	96.49	-	-	-	96.05	96.09	-
Unified BERT[g]	96.90	97.07	-	97.20	**98.45**	96.89	**97.81**	-	-	-	97.33	97.39	-
METASEG[h]	**97.04**	98.12	97.25	97.87	98.02	96.76	97.51	83.84	89.53	88.48	97.49	97.55	-
Ours	96.65	**98.15**	**97.43**	97.84	98.36	96.86	97.73	**98.28**	**93.94**	**97.14**	97.51	97.60	**97.24**
Ours+10%[UNC]	96.66	98.08	97.35	**97.93**	98.21	96.89	97.61	98.07	93.85	**97.14**	97.46	97.56	97.18
Ours+10%[UNC]+auto	96.64	97.30	97.01	96.89	92.78	95.08	93.43	97.98	92.26	96.05	95.45	95.35	95.54

Table 2: The F1-score (in percentage) on all 10 datasets. The F1-scores other than ours are directly recorded from their papers. Avg.4: Average over AS, CITYU, MSRA, and PKU; Avg.6: Average over AS, CITYU, CTB6, MSRA, PKU, and SXU; Avg.10: Average over 10 datasets; *a*: (Chen et al., 2017); *b*: (He et al., 2019); *c*: (Gong et al., 2019); *d*: (Huang et al., 2020a); *e*: (Huang et al., 2020b); *f*: (Qiu et al., 2020); *g*: (Ke et al., 2020); *h*: (Ke et al., 2021); Ours: Our model without criterion-denoising objective; Ours+10%[UNC]: Our model with criterion-denoising objective and randomly replacing 10% of criterion with [UNC]; Ours+10%[UNC]+auto: Same as Our+10%[UNC] but use [UNC] token to perform evaluation.

MCCWS Models	AS	CITYU	CNC	CTB6	MSRA	PKU	SXU	UD	WTB	ZX	Avg.4	Avg.6	Avg.10
Model-II+ADV[a]	75.37	81.05	-	82.19	72.76	73.13	76.88	-	-	-	75.578	76.897	-
Switch-LSTMs[b]	77.33	73.58	-	83.89	64.20	69.88	78.69	-	-	-	71.248	74.595	-
RoBERTa+softmax[c]	-	-	59.44	88.02	81.75	82.35	85.73	91.40	-	82.51	-	-	-
Transformer+CRF[d]	76.39	86.91	-	87.00	78.92	78.91	85.08	-	-	-	80.283	82.202	-
Unified BERT[e]	79.26	87.27	-	87.77	83.35	79.71	86.05	-	-	-	82.398	83.902	-
METASEG[f]	**80.89**	90.66	61.90	89.21	83.03	80.90	85.98	93.59	85.00	87.33	83.870	85.112	83.849
Ours	79.07	91.61	66.15	91.40	**88.82**	82.87	87.27	**93.75**	85.63	87.20	85.593	86.840	85.377
Ours+10%[UNC]	79.60	**92.28**	**66.28**	**91.66**	88.00	**83.44**	**87.60**	92.74	**86.24**	**88.14**	**85.830**	**87.097**	**85.598**
Ours+10%[UNC]+auto	80.12	91.26	65.64	89.83	64.24	78.28	80.57	93.07	83.49	85.94	78.475	80.717	81.244

Table 3: The OOV recall (in percentage) on all 10 CWS datasets. The OOV recalls other than ours are directly recorded from their papers. Avg.4: Average over AS, CITYU, MSRA, and PKU; Avg.6: Average over AS, CITYU, CTB6, MSRA, PKU, and SXU; Avg.10: Average over 10 datasets; *a*: (Chen et al., 2017); *b*: (Gong et al., 2019); *c*: (Huang et al., 2020a); *d*: (Qiu et al., 2020); *e*: (Ke et al., 2020); *f*: (Ke et al., 2021); Ours: Our model without criterion-denoising objective; Ours+10%[UNC]: Our model with criterion-denoising objective and randomly replacing 10% of criterion with [UNC]; Ours+10%[UNC]+auto: Same as Our+10%[UNC] but use [UNC] token to perform evaluation.

Auto Mechanism In Section 3.3, we claimed that our criterion-denoising objective could be used for choosing criteria automatically. We do this by pairing each input sequence on the test set with [UNC] and performing the evaluation. Table 2 shows that most datasets maintain their performances almost on par with the original even when using [UNC], and the average F1-score remains competitive with other baselines. This suggests that some common knowledge is shared throughout the 10 heterogeneous datasets, and our model can learn and leverage this knowledge.

4.4 Ablation Study

Increase Criterion Denoising Rate. This section studies what happens when the criterion denoising rate increases. Figure 2 shows that both the average F1-score and the average OOV recall decrease as criterion noise increases. This is expected as in the masked language model experiment of BERT, where increasing the masked rate results in fine-tune performance drop. However, as shown in Figure 2, using [UNC] to perform inference only gets affected slightly by different denoising rates. This suggests that when using criterion-denoising objective, our model learns to segment on the most common patterns showed across datasets. Thus, our model is robust to diverse inputs, which proven itself to be a "general CWS model" that shares knowledge across different CWS datasets.

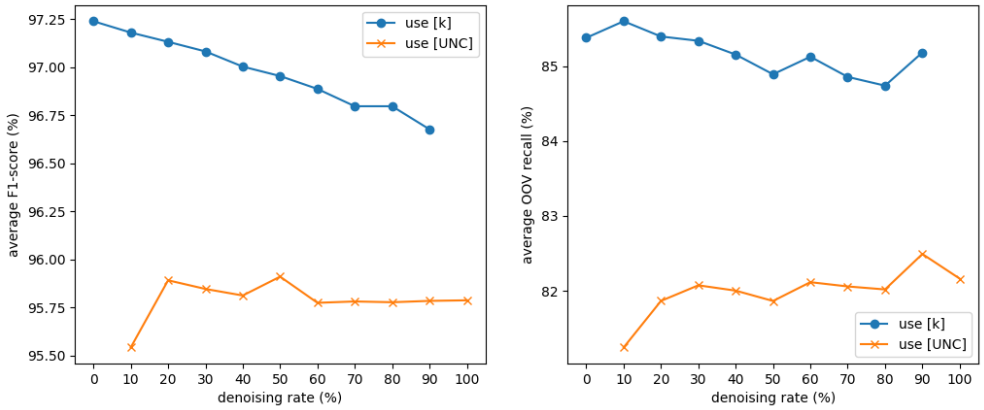

Figure 2: Left: Trade-off between denoising rate and the average F1-score. Right: Trade-off between denoising rate and the average OOV recall. use [k]: Use criterion-specific token [k] to perform inference; use [UNC]: Use [UNC] to perform inference.

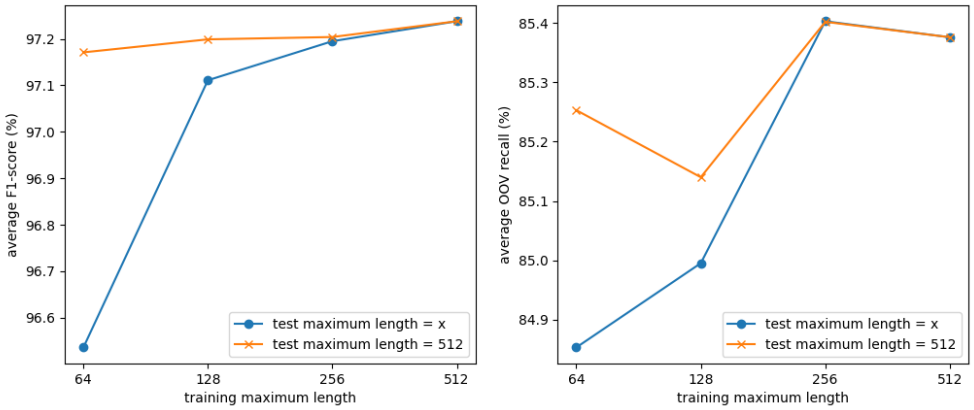

Figure 3: Left: Trade-off between the maximum sentence length constraint used in training and the average F1-score. Right: Trade-off between the maximum sentence length constraint used in training and the average OOV recall. test maximum length = x: Use the same maximum length constraint to perform inference. test maximum length = 512: Ignore the maximum length constraint and use up to 512 characters to perform inference.

Reduce Maximum Sentence Length. As shown in Table 3, our model's OOV recall outperformed others by a large margin. We suspect that it is due to our preprocessing step, which allows our model to take inputs up to 512 characters. Figure 3 shows that the longer a model's character sequence is allowed to take, the better the performance on the average F1-score and the average OOV recall. Performance on input length longer than 256 stays mostly the same since only a few sequences have their length longer than 256 (the average sentence length on all 10 datasets is 37.09, see Appendix A). However, we found an easy fix for models trained on shorter sentences: That is, allow input sequence length up to 512. Despite not being trained on such a long sequence, we found that all models' performance increased after feeding longer input. This is consistent with the common sense that longer input reduces the chance of ambiguity and thus performs better on CWS.

Case Study We provide examples to demonstrate our MCCWS model's capability of segmenting differently when given different criterion tokens. Table 4 shows that in some cases, one sentence can be segmented in at least five different ways, which proves that our model can perform CWS based on various criteria. Table 5 shows that in some other cases, most criteria agree with each other, which proves that our model can leverage the common knowledge shared across datasets. We leave more examples in Appendix A for interested readers.

Original Sentence	也是言之有據
AS-gold	也-是-言-之-有-據
CITYU-gold	也是-言之有據
AS-infer	也-是-言-之-有-據
CITYU-infer	也是-言之有據
CNC-infer	也是-言之有據
CTB6-infer	也-是-言之有據
MSRA-infer	也是-言之有據
PKU-infer	也-是-言之有據
SXU-infer	也-是-言之有據
UD-infer	也是-言-之有據
WTB-infer	也是-言之有據
ZX-infer	也-是-言-之-有據
[UNC]-infer	也是-言之有據

Table 4: Examples showcasing that one sentence can have multiple segmentation criteria, and our MCCWS model can deal with these linguistic divergences. We found five different ways to segment the same sentence "也是言之有據" (Claims are justified). $\mathcal{D}^k$-gold: Ground truth segmentation labeled in dataset $\mathcal{D}^k$. $\mathcal{D}^k$-infer: Inference result of our MCCWS model with criterion token [k]. [UNC]-infer: Inference result of our MCCWS model with unknown criterion token [UNC]. The hyphen "-" denotes segmentation.

Original Sentence	江泽民总书记
MSRA-gold	江泽民-总书记
PKU-gold	江-泽民-总书记
AS-infer	江泽民-总书记
CITYU-infer	江泽民-总书记
CNC-infer	江泽民-总书记
CTB6-infer	江泽民-总书记
MSRA-infer	江泽民-总书记
PKU-infer	江-泽民-总书记
SXU-infer	江泽民-总书记
UD-infer	江-泽民-总-书记
WTB-infer	江泽民-总书记
ZX-infer	江泽民-总书记
[UNC]-infer	江泽民-总书记

Table 5: Examples showcasing that our model can leverage shared common knowledge across datasets. We found three different ways to segment the same sentence "江泽民总书记" (General Secretary Jiang Zemin). We define symbols in the same way as in Table 4.

5 Conclusion

In this paper, we proposed a simple yet effective input-hint-based MCCWS model which achieves several SoTA results across 10 CWS datasets. We also proposed a novel criterion-denoising objective which makes our model capable of choosing criterion automatically for each character sequence. Experiment results show that our novel denoising objective does not suffer dramatic performance loss but helps our MCCWS model retain near SoTA performance and even outperform previous work on OOV recall by a large margin. Our model can serve as a simple and robust baseline for MCCWS work or as the starting point to further fine-tune into SC-CWS models. In the future, we will try to gather more CWS datasets and perform more extensive experiments on more datasets.

Limitations

Unfortunately, we cannot access most SIGHAN2008 bakeoff datasets, which were proprietary but used by many previous works. This makes the comparison in Table 2 a little unfair. We argue that we replaced these non-accessible datasets with the ones publicly accessible (including UD, WTB, and ZX). We note that Huang et al. (2020b) faced the same limitation as us. Thus they also replaced datasets just as we did, which makes them the only directly comparable work to ours.

References

Xinchi Chen, Zhan Shi, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial multi-criteria learning for Chinese word segmentation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1193–1203, Vancouver, Canada. Association for Computational Linguistics.

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

Thomas Emerson. 2005. The second international Chinese word segmentation bakeoff. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135. PMLR.

Jingjing Gong, Xinchi Chen, Tao Gui, and Xipeng Qiu. 2019. Switch-lstms for multi-criteria chinese word

segmentation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6457–6464.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Han He, Lei Wu, Hua Yan, Zhimin Gao, Yi Feng, and George Townsend. 2019. Effective neural solution for multi-criteria word segmentation. In *Smart Intelligent Computing and Applications*, pages 133–142, Singapore. Springer Singapore.

Kaiyu Huang, Degen Huang, Zhuang Liu, and Fengran Mo. 2020a. A joint multiple criteria model in transfer learning for cross-domain Chinese word segmentation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3873–3882, Online. Association for Computational Linguistics.

Weipeng Huang, Xingyi Cheng, Kunlong Chen, Taifeng Wang, and Wei Chu. 2020b. Towards fast and accurate neural Chinese word segmentation with multi-criteria learning. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2062–2072, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Guangjin Jin and Xiao Chen. 2008. The fourth international Chinese language processing bakeoff: Chinese word segmentation, named entity recognition and Chinese POS tagging. In *Proceedings of the Sixth SIGHAN Workshop on Chinese Language Processing*.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Zhen Ke, Liang Shi, Erli Meng, Bin Wang, Xipeng Qiu, and Xuanjing Huang. 2020. Unified multi-criteria chinese word segmentation with bert. *arXiv preprint arXiv:2004.05808*.

Zhen Ke, Liang Shi, Songtao Sun, Erli Meng, Bin Wang, and Xipeng Qiu. 2021. Pre-training with meta learning for Chinese word segmentation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5514–5523, Online. Association for Computational Linguistics.

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*, ICML '01, page 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

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.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Ji Ma, Kuzman Ganchev, and David Weiss. 2018. State-of-the-art Chinese word segmentation with Bi-LSTMs. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4902–4908, Brussels, Belgium. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Xipeng Qiu, Hengzhi Pei, Hang Yan, and Xuanjing Huang. 2020. A concise model for multi-criteria Chinese word segmentation with transformer encoder. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2887–2897, Online. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

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 *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

William Yang Wang, Lingpeng Kong, Kathryn Mazaitis, and William W. Cohen. 2014. Dependency parsing for Weibo: An efficient probabilistic logic programming approach. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1152–1158, Doha, Qatar. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Naiwen Xue, Fei Xis, Fu-Dong Chiou, and Marta Palmer. 2005. The penn chinese treebank: Phrase structure annotation of a large corpus. *Natural Language Engineering*, 11(2):207–238.

Nianwen Xue. 2003. Chinese word segmentation as character tagging. In *International Journal of Computational Linguistics & Chinese Language Processing, Volume 8, Number 1, February 2003: Special Issue on Word Formation and Chinese Language Processing*, pages 29–48.

Daniel Zeman, Jan Hajič, Martin Popel, Martin Potthast, Milan Straka, Filip Ginter, Joakim Nivre, and Slav Petrov. 2018. CoNLL 2018 shared task: Multilingual parsing from raw text to Universal Dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–21, Brussels, Belgium. Association for Computational Linguistics.

Meishan Zhang, Yue Zhang, Wanxiang Che, and Ting Liu. 2014. Type-supervised domain adaptation for joint segmentation and POS-tagging. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 588–597, Gothenburg, Sweden. Association for Computational Linguistics.

A Appendix

We list the preprocessing statistics in Table 6. The datasets' description and preprocessing steps can be found in Section 4.1. All datasets' licenses can be found in Table 7. Experiments on multiple trials can be found in Tables 8 and Table 9. Tables 10,11,12,13 give more examples to demonstrate our input-hint-based MCCWS model's capability of segmenting Chinese words with multiple criteria.

Dataset	Split	#C	#S	#W	#UC	#UW	OOV%	Avg.SL
AS	train	7,453,690	638,058	4,898,372	5,957	124,512	0	11.68
	dev	805,692	70,895	551,209	4,353	32,000	1.86	11.36
	test	193,723	14,429	122,610	3,579	18,093	3.73	13.43
CITYU	train	2,132,370	47,718	1,317,626	4,799	60,650	0	44.69
	dev	220,243	5,301	138,004	3,234	16,372	3.79	41.55
	test	66,353	1,492	40,936	2,643	8,633	7.38	44.47
CNC	train	8,908,376	207,001	5,841,321	6,643	113,223	0	43.04
	dev	1,109,292	25,875	727,783	5,109	47,773	0.76	42.87
	test	1,107,772	25,876	726,038	5,154	47,268	0.75	42.81
CTB6	train	1,108,461	24,416	678,811	4,201	42,086	0	45.40
	dev	82,765	1,904	51,229	2,491	8,639	4.89	43.47
	test	86,157	1,975	52,861	2,538	8,747	5.17	43.62
MSRA	train	3,615,524	78,227	2,144,776	5,023	71,399	0	46.22
	dev	363,425	8,691	223,615	3,676	22,515	2.57	41.82
	test	180,988	3,985	106,873	2,805	11,858	2.12	45.42
PKU	train	1,616,528	17,255	1,004,155	4,569	48,758	0	93.68
	dev	170,803	1,917	105,792	3,019	13,613	3.15	89.10
	test	168,992	1,949	104,372	2,881	12,456	3.31	86.71
SXU	train	744,162	15,407	474,758	4,026	28,207	0	48.30
	dev	85,470	1,711	53,480	2,206	6,460	6.23	49.95
	test	179,688	3,654	113,527	2,776	11,600	4.93	49.18
UD	train	147,295	3,997	98,608	3,390	15,930	0	36.85
	dev	19,027	500	12,663	1,922	4,040	10.95	38.05
	test	18,080	500	12,012	1,806	3,748	11.05	36.16
WTB	train	22,512	813	14,774	1,635	3,045	0	27.69
	dev	2,875	95	1,843	770	837	18.39	30.26
	test	2,838	92	1,860	733	731	15.05	30.85
ZX	train	96,647	2,373	67,648	2,289	6,770	0	40.73
	dev	28,309	788	20,393	1,651	3,184	7.85	35.93
	test	47,992	1,394	34,355	1,787	4,126	6.45	34.43
All	train	25,845,565	1,035,265	16,540,849	9,286	310,538	0	24.97
	dev	2,887,901	117,677	1,886,011	7,134	95,398	1.30	24.54
	test	2,052,583	55,346	1,315,444	6,789	77,145	1.21	37.09

Table 6: Dataset statistics (after preprocessing) for training, development, and test sets. #C: Number of characters. #S: Number of sentences. #W: Number of words. #UC: Number of unique characters. #UW: Number of unique words. OOV%: Out-of-vocabulary words rate. Avg.SL: Average sentence length.

Dataset	Provider	License
AS	SIGHAN2005	Research Purpose
CITYU	SIGHAN2005	Research Purpose
CNC	CNCorpus	Research Purpose
CTB6	StanfordCoreNLP	Apache License
MSRA	SIGHAN2005	Research Purpose
PKU	SIGHAN2005	Research Purpose
SXU	Shan Xi University	Research Purpose
UD	UD Project	BY-NC-SA 4.0
WTB	Wang et al. (2014)	Research Purpose
ZX	Zhang et al. (2014)	Research Purpose

Table 7: All datasets' licenses.

Experiments	Seeds	**AS**	**CITYU**	**CNC**	**CTB6**	**MSRA**	**PKU**	**SXU**	**UD**	**WTB**	**ZX**	**Avg.10**
Ours	927	96.65	98.15	97.43	97.84	98.36	96.86	97.73	98.28	93.94	97.14	97.238
	4332	96.66	98.10	97.44	97.96	98.47	96.95	97.70	98.19	93.69	97.00	97.216
	6664	96.58	98.05	97.44	97.84	98.41	96.91	97.72	98.23	93.42	97.20	97.180
	7155	96.73	98.02	97.45	97.91	98.37	96.90	97.79	98.30	93.56	97.03	97.206
	8384	96.68	98.05	97.44	97.83	98.37	96.89	97.65	98.21	93.55	97.04	97.171
	Avg.5	96.660	98.074	97.440	97.876	98.396	96.902	97.718	98.242	93.632	97.082	97.202
	Std.5	0.049	0.046	0.006	0.051	0.041	0.029	0.045	0.042	0.176	0.075	-
Ours+10%[UNC]	927	96.66	98.16	97.39	97.88	98.28	96.85	97.67	98.04	93.65	97.07	97.165
	4332	96.65	97.99	97.37	97.90	98.26	96.88	97.63	97.93	93.32	97.04	97.097
	6664	96.66	98.08	97.35	97.93	98.21	96.89	97.61	98.07	93.85	97.14	97.179
	7155	96.77	98.00	97.36	97.93	98.27	96.83	97.64	98.11	93.54	97.03	97.148
	8384	96.65	98.00	97.38	97.93	98.29	96.87	97.61	98.30	93.63	96.94	97.160
	Avg.5	96.678	98.046	97.370	97.914	98.262	96.864	97.632	98.090	93.598	97.044	97.150
	Std.5	0.046	0.066	0.014	0.021	0.028	0.022	0.022	0.121	0.172	0.065	-
Ours+10%[UNC]+auto	927	96.63	97.26	96.92	96.87	95.35	95.35	92.94	97.94	92.45	96.29	95.800
	4332	96.60	97.22	96.92	96.84	95.19	95.50	93.54	97.92	92.72	96.39	95.884
	6664	96.64	97.30	97.01	96.89	92.78	95.08	93.43	97.98	92.26	96.05	95.542
	7155	96.70	97.34	96.91	96.83	95.12	95.49	93.53	97.94	92.48	96.05	95.839
	8384	96.64	97.17	96.86	96.88	95.52	95.44	93.24	98.06	92.48	96.23	95.852
	Avg.5	96.642	97.258	96.924	96.862	94.792	95.372	93.336	97.968	92.478	96.202	95.783
	Std.5	0.032	0.059	0.048	0.023	1.015	0.155	0.225	0.050	0.146	0.134	-

Table 8: F1 results of 5 different trials. Experiment names are the same as in Table 2. Seed: Random seed set in an experiment. **Avg.10**: Average over 10 datasets. **Avg.5**: Average over 5 trials. **Std.5**: Standard deviation over 5 trials.

Experiments	Seeds	**AS**	**CITYU**	**CNC**	**CTB6**	**MSRA**	**PKU**	**SXU**	**UD**	**WTB**	**ZX**	**Avg.10**
Ours	927	79.07	91.61	66.15	91.40	88.82	82.87	87.27	93.75	85.63	87.20	85.377
	4332	79.52	91.77	66.05	91.78	88.34	83.80	87.29	93.68	85.63	87.74	85.560
	6664	78.45	91.48	66.57	91.69	88.24	83.39	87.17	93.68	86.54	88.05	85.526
	7155	80.52	91.16	66.17	91.86	88.34	83.23	87.55	93.41	85.63	87.56	85.543
	8384	79.88	91.26	66.13	91.02	89.06	83.00	87.00	93.07	84.40	87.60	85.242
	Avg.5	79.488	91.456	66.214	91.550	88.560	83.258	87.256	93.518	85.566	87.630	85.450
	Std.5	0.703	0.223	0.183	0.307	0.321	0.325	0.179	0.252	0.681	0.275	-
Ours+10%[UNC]	927	79.26	92.09	66.82	91.60	88.41	83.31	87.15	93.07	85.32	87.60	85.463
	4332	79.07	91.03	65.96	91.40	87.73	83.39	86.76	93.07	83.49	87.78	84.968
	6664	79.60	92.28	66.28	91.66	88.00	83.44	87.60	92.74	86.24	88.14	85.598
	7155	80.63	91.48	65.71	91.80	88.62	83.67	87.41	93.07	85.63	87.83	85.585
	8384	79.07	91.38	66.98	91.75	88.48	82.92	87.60	94.01	85.63	87.65	85.547
	Avg.5	79.525	91.652	66.350	91.642	88.248	83.346	87.304	93.192	85.262	87.800	85.432
	Std.5	0.585	0.464	0.487	0.139	0.331	0.244	0.318	0.429	0.935	0.189	-
Ours+10%[UNC]+auto	927	79.50	90.62	65.44	89.86	74.94	79.29	77.58	92.94	83.18	86.66	82.001
	4332	79.11	90.24	64.77	89.78	74.01	79.57	79.14	93.00	81.35	87.11	81.808
	6664	80.12	91.26	65.64	89.83	64.24	78.28	80.57	93.07	83.49	85.94	81.244
	7155	80.44	90.71	64.62	89.89	73.71	79.65	79.48	92.94	84.71	85.98	82.213
	8384	79.67	90.20	66.07	90.10	76.79	79.03	78.57	93.07	84.71	87.29	82.550
	Avg.5	79.768	90.606	65.308	89.892	72.738	79.164	79.068	93.004	83.487	86.596	81.963
	Std.5	0.467	0.384	0.542	0.110	4.383	0.493	0.989	0.058	1.237	0.559	-

Table 9: OOV recalls of 5 different trials. Experiment names are the same as in Table 3. Seed: Random seed set in an experiment. **Avg.10**: Average over 10 datasets. **Avg.5**: Average over 5 trials. **Std.5**: Standard deviation over 5 trials.

Original Sentence	何樂而不為
AS-gold	何-樂-而-不-為
CITYU-gold	何樂而不為
AS-infer	何-樂-而-不-為
CITYU-infer	何樂而不為
CNC-infer	何-樂-而-不-為
CTB6-infer	何-樂而-不為
MSRA-infer	何樂而不為
PKU-infer	何樂而不為
SXU-infer	何樂而不為
UD-infer	何-樂-而-不-為
WTB-infer	何樂而不為
ZX-infer	何-樂-而-不-為
[UNC]-infer	何樂而不為

Table 10: More examples showcase the capability of our input-hint-based MCCWS model. This example is the same one used in Table 1. We found three different ways to segment the same sentence "何樂而不為" (Why not do something?). We define symbols in the same way as in Table 4.

Original Sentence	一去不復返
AS-gold	一-去-不復-返
CITYU-gold	一去不復返
CNC-gold	一去不復返
MSRA-gold	一去不復返
PKU-gold	一去不復返
AS-infer	一-去-不復-返
CITYU-infer	一去不復返
CNC-infer	一去不復返
CTB6-infer	一-去-不復-返
MSRA-infer	一去不復返
PKU-infer	一去不復返
SXU-infer	一去不復返
UD-infer	一-去-不復-返
WTB-infer	一去-不復返
ZX-infer	一-去-不-復-返
[UNC]-infer	一去-不復返

Table 11: More examples showcase the capability of our input-hint-based MCCWS model. We found four different ways to segment the same sentence "一去不復返" (Once gone is gone). We define symbols in the same way as in Table 4.

Original Sentence	四月二十六日
AS-gold	四月-二十六日
CITYU-gold	四月-二十六-日
CNC-gold	四-月-二十六-日
MSRA-gold	四月二十六日
AS-infer	四月-二十六日
CITYU-infer	四月-二十六-日
CNC-infer	四-月-二十六-日
CTB6-infer	四月-二十六日
MSRA-infer	四月二十六日
PKU-infer	四月-二十六日
SXU-infer	四-月-二十六-日
UD-infer	四-月-二十六-日
WTB-infer	四月-二十六日
ZX-infer	四月-二十六日
[UNC]-infer	四月-二十六-日

Table 12: More examples showcase the capability of our input-hint-based MCCWS model. We found four different ways to segment the same sentence "四月二十六日" (April 26). We define symbols in the same way as in Table 4.

Original Sentence	並不足以
AS-gold	並-不-足以
CITYU-gold	並-不足以
CNC-gold	並不-足以
AS-infer	並-不-足以
CITYU-infer	並-不足以
CNC-infer	並不-足以
CTB6-infer	並不-足以
MSRA-infer	並不-足以
PKU-infer	並-不足以
SXU-infer	並-不足以
UD-infer	並-不-足-以
WTB-infer	並不-足以
ZX-infer	並-不-足以
[UNC]-infer	並-不-足以

Table 13: More examples showcase the capability of our input-hint-based MCCWS model. We found four different ways to segment the same sentence "並不足以" (Not enough). We define symbols in the same way as in Table 4.