Enhancing Chinese Word Segmentation via Pseudo Labels for Practicability

Kaiyu Huang[1] **Junpeng Liu**[1] **Degen Huang**[1]*
Deyi Xiong[2,3] **Zhuang Liu**[4] **Jinsong Su**[5]
[1]Dalian University of Technology [2]Tianjin University
[3]Global Tone Communication Technology Co., Ltd.
[4]Dongbei University of Finance and Economics [5]Xiamen University
{kaiyuhuang,liujunpeng_nlp}@mail.dlut.edu.cn
huangdg@dlut.edu.cn dyxiong@tju.edu.cn
liuzhuang@dufe.edu.cn jssu@xmu.edu.cn

Abstract

Pre-trained language models (e.g., BERT) significantly alleviate two traditional challenging problems for Chinese word segmentation (CWS): segmentation ambiguity and out-of-vocabulary (OOV) words. However, such improvements are usually achieved on traditional benchmark datasets and not close to an important goal of CWS: practicability (i.e., low complexity as a standalone task and high beneficiality to downstream tasks). To make a trade-off between traditional evaluation and practicability for CWS, we propose a semi-supervised neural method via pseudo labels. The neural method consists of a teacher model and a student model, which distills knowledge from unlabeled data to the student model so as to improve both in-domain and out-of-domain CWS. Experiments show that our proposed method can not only keep the practicability of the lightweight student model but also improve the performance of segmentation effectively. We also evaluate a range of heterogeneous neural architectures of CWS on downstream Chinese NLP tasks. Results of further experiments demonstrate that our proposed segmenter is reliable and practical as a pre-processing step of the downstream NLP tasks at the minimum cost.[1]

1 Introduction

Natural language processing (NLP) tasks often leverage word-level features to exploit lexical knowledge. Segmenting a sentence into a sequence of words, especially for languages without explicit word boundaries (e.g., Chinese) not only extracts lexical features, but also shortens the length of the sentence to be processed. Thus, word segmentation, detecting word boundaries, is a crucial pre-processing task for many NLP tasks. In this aspect, Chinese word segmentation (CWS) is widely acknowledged as an essential task for Chinese NLP.

CWS has made substantial progress in recent studies on several benchmarks, which is reported by Huang and Zhao (2007) and Zhao et al. (2019). In particular, pretrained language models (PLMs), like BERT (Devlin et al., 2019), have established new state-of-the-art in sequence labeling (Meng et al., 2019). Various fine-tuning methods have been proposed to improve the performance of in-domain and cross-domain CWS based on PLMs (Huang et al., 2020; Tian et al., 2020). The two challenging problems in CWS, segmentation ambiguity and out-of-vocabulary (OOV) words, have been significantly mitigated by PLM-based methods that are fine-tuned on large-scale annotated CWS corpora. Such methods are even reaching human performance on benchmarks. Nevertheless, CWS is more valuable as a prelude for downstream NLP tasks than as a standalone task. Intrinsic evaluation of CWS on benchmark datasets only examines the effectiveness of current neural methods on word boundary detection. To better apply CWS in downstream NLP tasks, we should comprehensively re-think CWS from the perspective of practicability. In this paper, we define the practicability of CWS with two aspects: low complexity as a standalone task and high beneficiality to downstream tasks.

The complexity is twofold: 1) complexity of implementation and 2) time and space complexity of a CWS algorithm. Previous neural methods usually require additional resources (Zhou et al., 2017; Ma et al., 2018; Zhang et al., 2018b; Zhao et al., 2018; Yang et al., 2019; Qiu et al., 2020), such as external pre-trained embeddings. The complexity of implementation is reflected in the difficulty of acquiring external resources. External resources

*Corresponding author

[1]Our code is available at https://github.com/koukaiu/dlut-nihao

Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 4369–4381
August 1–6, 2021.

vary in quality and the length of time for computation, For example, it is time-consuming to obtain effective pre-trained embeddings as they are trained on a huge amount of data. Generally, it is difficult to maintain high CWS performance for many previous neural methods in a low-resource environment. Neural methods with external resources achieve high CWS performance but at the cost of a high complexity of implementation. On the other hand, for training and inference, PLM-based CWS methods also consume large memory to store a huge number of parameters of their models. The speed of inference is usually slow. The huge memory consumption and slow inference prevent PLM-based CWS models from being deployed in small-scale smart devices. And, as CWS is often used with downstream models, this even weakens the applicability on smart devices as CWS is not supposed to take too much overhead in this situation.

The second is the beneficiality to downstream tasks. CWS is rarely used as a standalone task in industry. Existing CWS evaluations only rely on benchmarks and analyze the behavior of segmentation methods in a static scenario. Some well-known benchmarks are quite old (e.g., Bakeoff-2005) and not challenging for neural CWS anymore. Such evaluations are intrinsic, which are not associated with downstream NLP tasks. High CWS performance (e.g., Precision and F_1) does not mean that segmentation results are beneficial to downstream processing. Additionally, benchmark datasets have a plenty of segmentation noises that affect CWS training and evaluation. For instance, although the structure of "副" (vice) + "X" is segmented as two words: "副" (vice) and "X" in training data and never unified as a single word, "副校长" (vice-president) appears as one word in test data, note that: X presents any job titles, e.g., "总统" (president) and "经理" (manager). There are also many obvious errors due to annotation inconsistency in data. We have found, in one benchmark dataset, the word "操作系统" (operating system) is regarded as two words ["操作" (operate) + "系统" (system)] 6 times and appears as one word 14 times, respectively. Therefore, to measure and improve the beneficiality of CWS to downstream tasks, intrinsic evaluations on CWS benchmark datasets are not sufficient. We should perform extrinsic evaluations with downstream tasks.

To address the aforementioned practicability issue of CWS, we propose a semi-supervised neural method via pseudo labels. The method consists of two parts: a teacher model and a student model. First, we use a fine-tuned CWS model that is trained on the annotated CWS data as the teacher model, which can achieve competitive performance in traditional perspective for CWS. Then we collect massive unlabeled data and distill knowledge from the teacher model to the student model by generating pseudo labels. We filter out noisy pseudo labels to provide reliable knowledge for training the student model. The unlabeled data is easier to obtain than other external resources (e.g., lexicon and pre-trained embeddings) and can be updated anytime at a low cost. And we use the lightweight student model for inference, hence significantly reducing the memory consumption and inference time complexity. The practicability of our proposed method is competitive.

To sum up, the contributions of this work are as follows:

- Our proposed method distills knowledge from the teacher model via unlabeled data to coach the lightweight student model. The proposed method achieves a noticeable improvement over strong baselines for CWS by the traditional intrinsic evaluation.
- The lightweight student can be deployed on a small-scale device, even in a non-GPU environment. We abandon the PLM neural architectures (teacher model) during decoding. The speed of decoding is thus fast for practical application. Our method reduces the complexity of implementation, inference time, and memory consumption.
- We empirically investigate the effectiveness of the proposed method to downstream Chinese NLP tasks and analyze the impact of segmentation results on them via extrinsic evaluations.

2 Related Work

Since Xue (2003) formalizes CWS as a sequence labeling problem, many traditional statistical methods have achieved high performance for CWS on several benchmarks (Emerson, 2005). According to (Huang and Zhao, 2007) and (Zhao et al., 2019), CRF-based models (Tseng et al., 2005; Zhao and Kit, 2008; Zhao et al., 2010; Sun et al., 2012; Zhang et al., 2013) and neural methods (Zheng et al., 2013;

Pei et al., 2014; Chen et al., 2015; Cai and Zhao, 2016; Cai et al., 2017) have been reported to outperform traditional methods with high F_1 scores (0.95-0.97). In particular, Long Short-Term Memory Networks (LSTM) are the main backbone networks being used in these methods (Huang et al., 2015; Ma et al., 2018; Yang et al., 2019). Except for using LSTM, self-attention networks have been also employed for CWS (Duan and Zhao, 2020).

The OOV problem has been a long-standing challenge for CWS and it is particularly serious in the cross-domain scenario. To relieve this issue, many studies incorporate a variety of pre-trained word embeddings and external resources into CWS models (Zhou et al., 2017; Zhang et al., 2018b,a; Yang et al., 2019). Recently, with the development of PLMs (Devlin et al., 2018; Liu et al., 2019), fine-tuning methods benefit from a huge amount of the pre-trained knowledge for alleviating the OOV problem for CWS (Meng et al., 2019; Tian et al., 2020; Huang et al., 2020; Qiu et al., 2020). Such methods are nearly reaching human-level performance.

Nevertheless, external resources and PLMs result in additional costs in memory consumption and inference time. Knowledge distillation has been proposed to alleviate this additional cost issue (Ba and Caruana, 2014; Hinton et al., 2015). Kim and Rush (2016) propose to use knowledge distillation for neural machine translation while Mukherjee and Awadallah (2019) study several aspects of distillation to match the student model for sentiment classification. Jiao et al. (2020) adopt multiple distilling strategies to minimize the number of the parameters of the pre-trained language model. Different from these previous studies, our proposed method utilizes unified pseudo labels to improve the performance of the lightweight model. The model can provide positive influence as a preprocessing step to downstream tasks, compared with previous state-of-the-art methods.

3 Proposed Framework

Aiming at not only keeping competitive performance on benchmarks but also reducing the complexity of the CWS methods, our proposed framework consists of two essential modules: a student model and a teacher model, as shown in Figure 1. There is an obvious performance gap between the model based on PLMs (Huang et al., 2020) and the lightweight model (Duan and Zhao, 2020). The

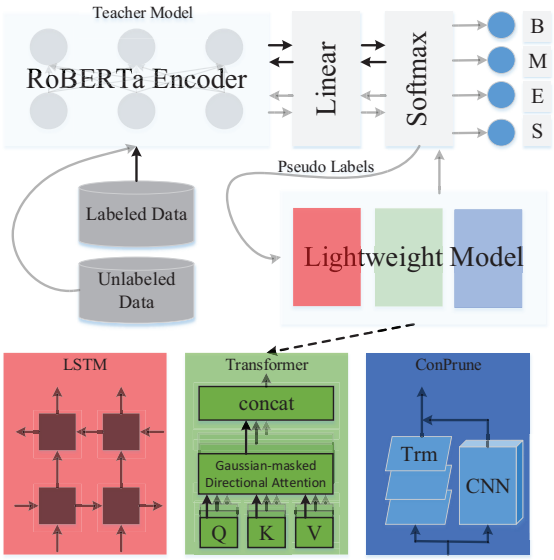

Figure 1: The illustration of our proposed framework. The red, green and blue blocks indicate the heterogeneous student models.

OOV issue is the main reason for the gap. Since the teacher model based on fine-tuned PLMs with high complexity can alleviate the OOV issue effectively, we use a combination of PLM-based teacher and lightweight student. First, the teacher model transfers pre-trained knowledge into a specific data distribution by annotating unlabeled data. Then we utilize a huge amount of such annotated data to distill knowledge from the teacher model to the lightweight student model. The pseudo labels provided by the teacher model can help the lightweight model to alleviate the OOV issue of CWS.

3.1 Teacher Model

Recently, there are several PLMs (e.g., BERT and RoBERTa) that have shown competitive performance for many NLP tasks. In particular, a modified RoBERTa model has been built for Chinese NLP tasks (Cui et al., 2019). Inspired by the previous success of PLM-based models on CWS (Huang et al., 2020), we use the RoBERTa-WWM PLM as the teacher model.

Normally, PLMs are trained for predicting words in general. To adapt PLMs and transfer their knowledge to CWS, we need to fine-tune PLMs on the annotated data of CWS. Let X denote the inputs, which are converted into a sequence of embeddings. For consistency, two tags ("[CLS]" and "[SEP]") are added to the beginning and end of each sentence, respectively. A Linear transfer layer with

$W^{(t)} \in R^{d_{model}*N}$ replaces the original component, where d_{model} is the same as the size of dimensions of the pre-trained model and N presents number of tags in CWS annotated data ($N = 4$). We convert CWS annotations into annotations with a 4-tag set $T = \{B, M, E, S\}$ that indicates the **B**egin, **M**iddle, **E**nd of a word, or a **S**ingle character forming a word. After linear mapping, the teacher model adopts the function of $\mathrm{Softmax}$ and the greedy search for decoding.

$$p^{(t)}(x) = \mathrm{Softmax}(h_t(x) \cdot W^{(t)} + b^{(t)}) \quad (1)$$

where $h_t(x)$ represents the hidden states of the teacher model. Complex algorithms (e.g., CRF or beam search) for decoding are abandoned in order to reduce the complexity. In addition, these complex algorithms only obtain a slight improvement for CWS. CRF increases the time complexity by n times and beam search requires more search time varying with the beam size, compared with the greedy search.

$$GreedySearch \rightarrow O(Mn)$$
$$BeamSearch \rightarrow O(Mnb^2)$$
$$CRF \rightarrow O(Mn^2)$$

where M is a constant representing other factors in the model complexity, n is the size of the sentence, and b is the width of beam.

The training of the teacher model is to minimize the errors by solving the following optimization function:

$$\min_{W^{(t)}} J_{seg}(y(x)|p^{(t)}(x, W^{(t)})) \quad (2)$$

where the loss function J_{seg} is computed by:

$$J_{seg}(y(x)|p^{(t)}(x)) = -\sum_{x} y(x)\log p^{(t)}(x) \quad (3)$$

3.2 Student Model

To improve the practicability of CWS, our proposed framework rediscovers the potential of the lightweight models. The lightweight model suffers from the OOV problem compared with the teacher model. However, the lightweight model can help us to solve the practicability issue of CWS. We propose multiple lightweight models as the student model, as shown in Figure 1.

-ConPrune. This is a pruned PLM model, where three quarters of the PLM's layers are discarded. Particularly, we only use the first top 3 layers of the entire 12 layers. We also incorporate a Convolutional Neural Network (CNN) encoder to capture the local features of the sequence.

-LSTM. LSTM is the most popular architecture for sequence labeling tasks (Ma et al., 2018). As shown in Figure 1, for each input character c_i, the corresponding character uni-gram embedding and bi-gram embedding are represented as e_{c_i} and $e_{c_i c_{i+1}}$, respectively. The LSTM model is fed with the two types of character embeddings by concatenation operation, $w_i = e_{c_i} \oplus e_{c_i c_{i+1}}$. The loss function and the decoding are the same as the teacher model.

-Transformer. The Transformer is usually not working as well as LSTM for sequence labeling tasks despite its success on other tasks. We propose a new Transformer variant that is inspired by Duan and Zhao (2020). The modified Transformer utilizes the Gaussian directional mask to encode unigram features.

-CRF. Although CRF is not a dominant model for CWS, it still has great significance for practicability. We only utilize uni-gram and bi-gram features for CRF, keeping the same as neural methods for a fair comparison. It does not rely on any auxiliary features, e.g., accessor variety (AV) (Feng et al., 2004) or pointwise mutual information (PMI) (Sun et al., 1998).

All formulations and details of the student models are shown in Appendix A.

3.3 Pseudo Labels

Neural networks typically predict the probability of each class by $\mathrm{Softmax}$. In the form of distillation, knowledge is transferred to the distilled model by using a distribution that is produced by the teacher model with a temperature in its $\mathrm{Softmax}$. However, the architectures of the student models are completely different from the teacher model, as shown in the last section. Unlike previous studies on distilling knowledge, the process of knowledge distillation in our framework is essentially the same as the original CWS task. Particularly, our proposed method distills the knowledge from the teacher model to the student model by using a huge amount of unlabeled data as the knowledge container. It is easy to obtain unlabeled data from the Internet. The pseudo labels are generated together with noisy labels and we reduce the impact of noisy labels. Due to the high correlation between training data and unlabeled data, we directly distill knowl-

	MSR		PKU		AS		CITYU		CTB6	
	TRAIN	TEST	TRAIN	TEST	TRAIN	TEST	TRAIN	TEST	TRAIN	TEST
# CHAR	4,050K	184K	1,826K	173K	8,368K	198K	2,403K	68K	1,156K	134K
# WORD	2,368K	107K	1,110K	104K	5,500K	123K	1,456K	41K	701K	82K
	NER-PKU		WMT-18				UNLABELED			
	TRAIN	TEST	TRAIN		DEV		TEST		PEOPLE'S DAILY	
# SEN	17,546	1,714	24,752,392		2,002		2,001		115,856	

Table 1: Statistics of our datasets. The upper part of the table shows the details of the CWS benchmarks and the bottom for the downstream NLP tasks. All datasets come from the official websites.

edge from the teacher model to the student model, and the final loss is shown as follow:

$$J_u(y(u)|p^{(s)}(u)) = -\sum_{x_u} y_u \log y^*(x_u) \quad (4)$$

$$J(\Theta_s, \Theta_u) = J_{seg}(\Theta_s) + \alpha J_u(\Theta_u), \quad \Theta_s : y(x)|p^{(s)}(x), \Theta_u : y(u)|p^{(s)}(u) \quad (5)$$

where s denotes the student model, α is a weight to balance the losses on the labeled data and unlabeled data ($\alpha = 0.5$ in our experiments). The loss function is calculated with two parts. One is from the labeled data, the other is from the unlabeled data x_u. Hard prediction of the teacher model on the unlabeled data produces noisy labels y_u. And the prediction of the student model on unlabeled data is y^*. To reduce the redundant computation, pseudo labels are mix-sampled according to a regular interval. The sampling strategy chooses the different n-gram features with the annotated data, which makes the distribution of unlabeled sentences different from the annotated data. Instead of optimizing the loss function jointly, we adopt a two-stage optimizing method. The first stage trains student models on the large-scale annotated data. In the second stage, the student model is continued to be trained on the data with labels predicted from the teacher model. Since the teacher model is also fine-tuned on the annotated data, the two-stage training does not suffer from the catastrophic forgetting issue.

4 Experiments

4.1 Datasets and Settings

To examine the advantage of distilling knowledge and the complexity of our proposed framework via pseudo labels, we conducted experiments on five benchmarks (Bakeoff-2005[2] and CTB6). The statistics of the benchmarks are shown in Table 1. We randomly picked 10% sentences from the training data as the development data for tuning. The unlabeled data were collected from the People's Daily website. We crawled 5,000 articles. For consistency, we pre-processed unsegmented sentences, which is similar to previous work (Cai et al., 2017). In addition, to empirically validate the beneficiality of the proposed CWS method to downstream tasks, we carried out comprehensive experiments on named entity recognition (NER) and machine translation (MT). The details of datasets for these two tasks are shown in Table 1. We used F_1 as the evaluation metric for NER and BLEU (Papineni et al., 2002) for MT.

Parameters	Teacher	Student		
		PRUNE	LSTM	TRANS
Hidden states	768	768	256	256
uni-gram embeds	768	768	50	256
bi-gram embeds	-	-	50	-
learning rate	2e-5	1e-5	1e-3	0.1
batch size	64	256	64	4096
dropout	0.1	0.1	0.2	0.1
hidden layers	12	3	3	6
epochs	10	20	30	200

Table 2: The hyper-parameters. "PRUNE" represents the ConPrune and "TRANS" represents the modified Transformer.

To fine-tune the teacher model (i.e., RoBERTa-WWM), we adjusted a few crucial hyper-parameters for it, as shown in Table 2. The hyper-parameters of the student model were tuned with the development sets. We evaluated inference speed for all models on the same hardware configuration (Non-GPU environment: Intel(R) Core(TM) i9-10900KF CPU @ 3.70GHz //GPU environment: Nvidia GeForce RTX 3090). All other hyper-parameters and search ranges are shown in Appendix B.

[2] http://sighan.cs.uchicago.edu/bakeoff2005/

MODELS	AU	PKU	MSR	AS	CITYU	CTB6	S-CPU (ms.)	S-GPU (ms.)
TEACHER	-	96.68	98.14	96.62	97.92	97.55	119.24	3.61
CRF	×	95.02	96.72	95.40	94.25	95.65	**3.54**	-
	√	**96.19**	**97.33**	**95.70**	**96.18**	**96.79**		
LSTM	×	95.15	97.26	95.29	95.13	94.98	21.66	**1.03**
	√	**96.25**	**97.58**	**95.81**	**96.30**	**96.66**		
GREEDY LSTM	×	95.37	96.83	95.22	95.54	96.06	11.62	-
	√	**96.43**	**97.17**	**95.43**	**96.44**	**97.04**		
TRANSFOMER	×	95.46	97.59	95.96	95.26	96.10	60.41	1.70
	√	**96.43**	**97.79**	**96.25**	**96.91**	**97.16**		
CONPRUNE	×	96.34	97.93	96.31	97.28	97.19	30.17	**1.29**
	√	**96.76**	**98.07**	**96.47**	**97.58**	**97.45**		

Table 3: Results on the Bakeoff-2005 dataset. "AU" indicates whether the student model utilizes unlabeled data. GREEDY LSTM follows Cai et al. (2017), which is a LSTM model adapted to the CPU environment. S-CPU/GPU denotes the inference speed (ms per sentence) on the CPU/GPU environment.

MODELS	PKU	MSR	AS	CITYU	CTB6
(CHEN ET AL., 2015)	96.5	97.4	-	-	96.0
(ZHOU ET AL., 2017)	96.0	97.8	-	-	96.2
(MA ET AL., 2018)	96.1	97.4	96.2	97.2	96.7
(GONG ET AL., 2019)	96.7	96.5	94.5	93.7	-
(DUAN AND ZHAO, 2020)	95.5	97.7	95.7	96.4	-
(TIAN ET AL., 2020)	96.5	98.4	96.6	97.9	97.3
(HUANG ET AL., 2020)	96.7	98.1	-	-	97.6
TRANSFORMER	96.4	97.8	96.3	96.9	97.2
CONPRUNE	**96.8**	**98.1**	**96.5**	**97.6**	**97.5**

Table 4: Experiment results on the Bakeoff-2005 datasets. The best results obtained by non-PLM models. Our results are significantly better ($p < 0.05$ bootstrap resampling) than all previous state-of-the-art results.

4.2 Results of Intrinsic Evaluation

As shown in Table 3, we investigated the effect of the proposed method on the benchmark Bakeoff-2005, which is the most widely-used dataset for CWS. "TEACHER" denotes the teacher model as introduced in section 3.1. It achieves competitive performance as it is based on a state-of-the-art pretrained model. Other models are the student models.

Experimental results in Table 4 show that our proposed semi-supervised method significantly improves the performance on all 5 benchmark datasets, compared with the pure student model. Surprisingly, results of the proposed semi-supervised method are even close to those of the teacher model. We also compared our proposed method against previous SOTA models, as shown in Table 4. In particular, Tian et al. (2020) and Huang et al. (2020) utilize PLMs which are slower in inference than non-PLM CWS models. This paper focuses on the methods with low complexity. These results demonstrate that our proposed method achieves state-of-the-art performance compared with non-PLM methods. Although there is a small performance gap between our proposed method and fine-tuned PLM methods, the advantage of our method over PLMs is that our method is much faster in both CPU and GPU environments, as displayed in Table 3, which is the key interest of our work. From this perspective, our method is more readily to be used in downstream tasks than previous state-of-the-art PLM methods. In addition, our proposed method not only maintains the advantages of the basic neural methods but also has a low complexity for practicability. Meanwhile, the method leverages easily available unlabeled data to make up for the insufficiency of the student model.

4.3 Results of Extrinsic Evaluation

The performance of models on various CWS benchmarks only demonstrates the merits of models themselves. However, CWS results of different methods that achieve good performance on benchmarks are not necessarily beneficial for specific downstream tasks. We therefore investigated the effect of using different CWS results on the two popular downstream Chinese NLP tasks (NER and MT) to analyze the beneficiality of CWS methods to other tasks. The benchmarks of these two tasks that we adopted are both widely acknowledged in the literature of NER and MT. Particularly, we used

MT (BLEU) ZH→EN			CWS (F_1) SEG. TRAIN	CWS (F_1) SEG. TEST	NER (F_1) NR	NER (F_1) NP	NER (F_1) NT
CHAR	21.16	GOLD	1.000	1.000	.895	.880	.864
TEACHER	23.51 +2.35		.991 -.009	.993 -.007	.897 +.001	.875 -.005	.862 -.002
CRF	23.37 +2.21		.990 -.010	.992 -.008	.907 +.012	.880 -.000	**.877 +.013**
CRF†	23.68 +2.52		.990 -.010	.992 -.008	**.915 +.020**	.881 +.001	.862 -.002
CONPRUNE	23.71 +2.55		.990 -.010	.992 -.008	.903 +.008	.879 -.001	.863 -.001
CONPRUNE†	**23.73 +2.57**		.988 -.012	.991 -.009	**.915 +.020**	**.884 +.004**	**.877 +.013**

Table 5: Results on NER and MT. "NR, NP AND NT" represent entities of person, place and organization. "GOLD" denotes gold-standard word segmentations for NER and "CHAR" indicates the character-level neural model based on Transformer for baseline comparison. † indicates that the corresponding model utilizes the proposed semi-supervised method.

the "PKU" open resources for NER evaluation and a Chinese-to-English machine translation task from WMT-18[3] for MT evaluation. The model for NER employs word-based LSTM to extract context information and applies a CRF layer stacked over LSTM for decoding. It utilizes random word-level embeddings which can be further fine-tuned later. The evaluation of this task is the same as CWS (F_1). The MT model is based on the Transformer (Vaswani et al., 2017) neural network. We used Byte Pair Encoding (BPE) for alleviating the issue of rare words. We kept all other hyper-parameters of the NER and MT models as those widely used. We then fed CWS results produced by different models into the NER and MT models. Results are shown in Table 5.

Clearly, our proposed method can provide word segmentations that are beneficial for the two downstream tasks. The performance of NER using segmentation results yielded by our proposed method is better than others, even ground-truth word segmentations. All segmentation systems achieve good performance with no evidence of OOV. However, there are still some distinctions between two CWS methods, which will be analyzed in the case study section. Except for the quality of word segmentations, the speed of our proposed method is fast enough to support specific downstream tasks. Surprisingly, we find that word segmentations with high F_1 scores on CWS benchmarks do not necessarily indicate high performance on downstream tasks. Especially, the optimal performance of segmentation results ("Seg. train" and "Seg. test") does not suggest the highest performance on NER ("NR", "NP" and "NT"), as shown in Table 5. This might be due to two reasons. First, gold results in NER have noises, which is similar to CWS. Our proposed method has a strong robustness to deal with noisy labels. Second, word segmentation errors do not necessarily cause error propagation.

[3] http://data.statmt.org/wmt18/translation-task/

4.4 Case Study

To make further progress on CWS, it is important to understand errors that CWS methods are making. Hence we randomly selected typical errors from the PKU test set and manually analyzed them.

The segmentation errors can be roughly divided into two categories. One is the type of errors with OOV words. The proposed semi-supervised method can alleviate the issue of OOV words effectively. For instance, "威尔第" (Verdi) is segmented into two words incorrectly by the pure student model. This split frequently occurs in the unlabeled data, and such knowledge is distilled from the teacher model. The semi-supervised method can revise these OOV words.

Except for the type of errors of OOV words, the rest of errors are mainly caused by segmentation inconsistency. For example, the word "人" (person) should be regarded as a suffix word behind some words, e.g., "中国+人" (Chinese) and "代理+人" (agent). "人" (person) also exists as part of other words, e.g., "关系人" and "继承人". Simply training neural model on such inconsistent segmentation data may be insufficient to solve these segmentation errors without further efforts in data processing. This situation naturally raises a question: do the errors caused by segmentation inconsistency really influence the performance of downstream NLP tasks?

To answer this question, we conducted additional experiments on the two downstream NLP tasks. In NER, segmentation results of non-entity

words hardly affect the performance of NER. For instance, the word "不懈奋斗" (untiringly struggle) is regarded as a word according to the criterion of "PKU". Previous state-of-the-art methods that achieve high F_1 scores for CWS can segment it correctly. While our proposed method splits this unit into two independent words "不懈" (untiringly) and "奋斗" (unremitting). However, these two words do not belong to any entities. In other words, the better performance for segmenting non-entity words does not necessarily indicate better performance of NER. In addition, segmentation results of entity words directly affect the veracity of NER. There is a phrase "西方七国集团" (the Group of Seven, abbreviations: G7) in a sentence. This segment is an organizational entity. In word segmentation, it is regarded as two words "西方" (western) and "七国集团" (the group of seven countries) according to ground-truth segmentation results. Previous state-of-the-art methods are usually able to segment it correctly. By contrast, our proposed method segments it into three words "西方" (western), "七国" (seven countries) and "集团" (group). Surprisingly, the final result of NER is out of expectation. The entity with incorrectly segmented words by our method is correctly recognized. Gold segmentation does not achieve a better result on this entity. The vague boundary of a word may increase the uncertainty and difficulty of downstream Chinese NLP tasks. There are many prefix and suffix words in Chinese. Sometimes, it is hard to determine whether these words are a single word or not. For this reason, high performance of CWS is not equal to high performance of Chinese NER.

In MT, due to the technique of BPE, rare words are segmented into sub-words. The issue of unknown words can thus be alleviated effectively. Even if a word as simple as "日本" (Japan) is segmented into two words incorrectly, NMT models are able to prevent the error propagation of segmentation in the training step. Thus, a faster segmentation system, rather than a high-performance segmentation system, is more practical for NMT. To analyze NMT translation differences between two sentences with different segmentation results, we also supply the additional analysis in Appendix C. We find that segmentation results with slight differences make translation results varying. The boundary of words may lead to these differences. But we also conjecture that this is more due to the robustness of NMT models.

5 Conclusion

To bring a positive impact of CWS to downstream NLP tasks, this paper makes a trade-off between the traditional evaluation and the complexity (e.g., implementation and decoding speed), which makes the segmenter more practical. We propose a semi-supervised method that distills knowledge via pseudo labels into the lightweight student model. The method is low coupling, which significantly improves the performance of multiple heterogeneous tiny neural architectures. The proposed framework can achieve competitive performance on CWS benchmarks and the speed of the student model also satisfies the practical requirement. In summary, the advantages of our model are twofold. First, the inference speed of the method is much faster than PLM methods. It can run under low resource conditions, even on CPUs. Second, the model provides efficient segmentation results for downstream NLP tasks.

Acknowledgments

We sincerely thank the reviewers for their insightful comments and suggestions to improve the quality of the paper. The authors gratefully acknowledge the financial support provided by the National Key Research and Development Program of China (2020AAA0108004) and the National Natural Science Foundation of China under(No.U1936109). Deyi Xiong is partially supported by the joint research center between GTCOM and Tianjin University.

References

Jimmy Ba and Rich Caruana. 2014. Do deep nets really need to be deep? In *Advances in neural information processing systems*, pages 2654–2662.

Deng Cai and Hai Zhao. 2016. Neural word segmentation learning for chinese. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 409–420, Berlin, Germany. Association for Computational Linguistics.

Deng Cai, Hai Zhao, Zhisong Zhang, Yuan Xin, Yongjian Wu, and Feiyue Huang. 2017. Fast and accurate neural word segmentation for chinese. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 608–615, Vancouver, Canada. Association for Computational Linguistics.

Xinchi Chen, Xipeng Qiu, Chenxi Zhu, Pengfei Liu, and Xuanjing Huang. 2015. Long short-term memory neural networks for chinese word segmentation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1197–1206, Lisbon, Portugal. Association for Computational Linguistics.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Ziqing Yang, Shijin Wang, and Guoping Hu. 2019. Pre-training with whole word masking for chinese bert. *arXiv preprint arXiv:1906.08101*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

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.

Sufeng Duan and Hai Zhao. 2020. Attention is all you need for Chinese word segmentation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3862–3872, Online. 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*, pages 123–133, Jeju Island, Korea.

Haodi Feng, Kang Chen, Xiaotie Deng, and Weimin Zheng. 2004. Accessor variety criteria for chinese word extraction. *Computational Linguistics*, 30(1):75–93.

Jingjing Gong, Xinchi Chen, Tao Gui, and Xipeng Qiu. 2019. Switch-lstms for multi-criteria chinese word segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6457–6464.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Chang-Ning Huang and Hai Zhao. 2007. Chinese word segmentation: a decade review. *Journal of Chinese Information Processing*, 21(3):8–19.

Kaiyu Huang, Degen Huang, Zhuang Liu, and Fengran Mo. 2020. 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.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. TinyBERT: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174, Online. Association for Computational Linguistics.

Yoon Kim and Alexander M Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327.

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*.

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.

Yuxian Meng, Wei Wu, Fei Wang, Xiaoya Li, Ping Nie, Fan Yin, Muyu Li, Qinghong Han, Xiaofei Sun, and Jiwei Li. 2019. Glyce: Glyph-vectors for chinese character representations. In *Advances in Neural Information Processing Systems*, pages 2742–2753.

Subhabrata Mukherjee and Ahmed Hassan Awadallah. 2019. Distilling transformers into simple neural networks with unlabeled transfer data. *arXiv preprint arXiv:1910.01769*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Wenzhe Pei, Tao Ge, and Baobao Chang. 2014. Max-margin tensor neural network for chinese word segmentation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 293–303, Baltimore, Maryland, USA. Association for Computational Linguistics.

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.

Maosong Sun, Dayang Shen, and Benjamin K Tsou. 1998. Chinese word segmentation without using lexicon and hand-crafted training data. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 2*, pages 1265–1271.

Xu Sun, Houfeng Wang, and Wenjie Li. 2012. Fast online training with frequency-adaptive learning rates for chinese word segmentation and new word detection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 253–262, Jeju, Republic of Korea. Association for Computational Linguistics.

Yuanhe Tian, Yan Song, Fei Xia, Tong Zhang, and Yonggang Wang. 2020. Improving Chinese word segmentation with wordhood memory networks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8274–8285, Online. Association for Computational Linguistics.

Huihsin Tseng, Pichuan Chang, Galen Andrew, Daniel Jurafsky, and Christopher Manning. 2005. A conditional random field word segmenter for sighan bakeoff2005. In *Proceedings of the Fourth SIGHAN workshop on Chinese Language Processing*, pages 168–171, Jeju Island, Korea.

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

Nianwen Xue. 2003. Chinese word segmentation as character tagging. *Computational Linguistics and Chinese Language Processing*, 8(1):29–48.

Jie Yang, Yue Zhang, and Shuailong Liang. 2019. Subword encoding in lattice lstm for chinese word segmentation. 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 2720–2725.

Longkai Zhang, Houfeng Wang, Xu Sun, and Mairgup Mansur. 2013. Exploring representations from unlabeled data with co-training for chinese word segmentation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 311–321, Seattle, Washington, USA. Association for Computational Linguistics.

Meishan Zhang, Nan Yu, and Guohong Fu. 2018a. A simple and effective neural model for joint word segmentation and pos tagging. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(9):1528–1538.

Qi Zhang, Xiaoyu Liu, and Jinlan Fu. 2018b. Neural networks incorporating dictionaries for chinese word segmentation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Hai Zhao, Deng Cai, Changning Huang, and Chunyu Kit. 2019. Chinese word segmentation: Another decade review (2007-2017). *arXiv preprint arXiv:1901.06079*.

Hai Zhao, Chang-Ning Huang, Mu Li, and Bao-Liang Lu. 2010. A unified character-based tagging framework for chinese word segmentation. *ACM Transactions on Asian Language Information Processing*, 9(2):1–32.

Hai Zhao and Chunyu Kit. 2008. Unsupervised segmentation helps supervised learning of character tagging for word segmentation and named entity recognition. In *The Sixth SIGHAN Workshop on Chinese Language Processing*, pages 106–111, Hyderabad, India.

Lujun Zhao, Qi Zhang, Peng Wang, and Xiaoyu Liu. 2018. Neural networks incorporating unlabeled and partially-labeled data for cross-domain chinese word segmentation. In *IJCAI*, pages 4602–4608.

Xiaoqing Zheng, Hanyang Chen, and Tianyu Xu. 2013. Deep learning for chinese word segmentation and pos tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 647–657, Seattle, Washington, USA. Association for Computational Linguistics.

Hao Zhou, Zhenting Yu, Yue Zhang, Shujian Huang, Xinyu Dai, and Jiajun Chen. 2017. Word-context character embeddings for chinese word segmentation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 760–766, Copenhagen, Denmark. Association for Computational Linguistics.

Appendix

A Model Architecture

This paper introduces multiple heterogeneous tiny neural architectures as the student model. To better describe each model, all formulations of the student models are shown as follows.

-ConPrune. ConPrune prunes three quarters of 12 layers of the used PLM. To extract local features, we incorporate a Convolutional Neural Network (CNN) encoder into the pruned model. The kernel size determines the distance of scanning. The input sequence is converted into two vector matrices $\mathbf{E}_t$ and $\mathbf{E}_c$. Word positions are also mapped into a feature matrix $\mathbf{E}_p$. The input to the encoder consists of four parts that are token embedding $\mathbf{E}_t$, position embedding $\mathbf{E}_p$, segment embedding $\mathbf{E}_s$ and CNN embedding $\mathbf{E}_c$. Because of the specificity for CWS, all segment embeddings of sequences are regarded as the same mapping matrix $\mathbf{E}_s$. The input of the two components are:

$$\mathbf{E}_{trm} = \mathbf{E}_t + \mathbf{E}_p + \mathbf{E}_s, \mathbf{E}_{cnn} = \mathbf{E}_c \quad (6)$$

The convolutional encoder involves a filter $W \in \mathbb{R}^{hk}$, which is applied to a window of h characters to produce a new feature.

$$W_{cnn} = \text{Relu}\left(W \cdot x_{i:i+h-1} + b\right) \quad (7)$$

where $Relu$ is a type of activation function. $x_{i:i+h-1}$ indicates the matrix of $\mathbf{E}_c$.

The pruned encoder consists of 3 base Transformer encoded layers with multiple multi-head self-attention layers to extract contextual features for each character. The multi-head self-attention layer adopts "Scaled Dot-Product Attention" which is formulated as:

$$\text{Attn}\left(Q, K, V\right) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (8)$$

where Q, K, V represents a query and a set of key-value pairs through a linear transformation respectively, and d_k is the dimension of K.

$$\text{MultiAttn}\left(E_{trm}\right) = \left[head_1, ..., head_k\right]W^O \quad (9)$$

$$head_i = \text{Attn}\left(E_{trm}W_i^Q, E_{trm}W_i^K, E_{trm}W_i^V\right) \quad (10)$$

where W^O, W_i^Q, W_i^K, W_i^V are trainable parameters.

-LSTM. LSTM can be used as a lightweight architecture for sequence labeling tasks. Rigorous tuning can obtain competitive performance for CWS. The model handles sequence features very well. For each input character c_i, the corresponding character uni-gram embedding and bi-gram embedding are represented as e_{c_i} and $e_{c_ic_{i+1}}$, respectively. The LSTM model is fed with the two types of character embeddings by concatenation operation, $w_i = e_{c_i} \oplus e_{c_ic_{i+1}}$. We get the outputs of the student representations from the LSTM model as follow:

$$\overrightarrow{h_s} = \overrightarrow{LSTM}\ (w_1, w_2, \ldots, w_i) \quad (11)$$

$$\overleftarrow{h_s} = \overleftarrow{LSTM}\ (w_1, w_2, \ldots, w_i) \quad (12)$$

$$h_s = \overrightarrow{h_s} \oplus \overleftarrow{h_s} \quad (13)$$

-Transformer. This paper adopts a modified Transformer which follows the previous study by Duan and Zhao (2020). The modified Transformer changes the multi-head self-attention to the multi-head Gaussian directional attention. Given an input sequence, the attention function can be described as mapping a query and a set of key-value pairs. The Gaussian directional attention incorporates the Gaussian directional attention into traditional self-attention module to pay attention to the neighboring characters of each position and capture features between characters as a fix Gaussian weight for attention. The Gaussian weight function and the multi-head Gaussian directional attention are computed as:

$$G_{i,j} = \sqrt{\frac{2}{\sigma\pi}} \int_{-\infty}^{-d_{i,j}} exp(-\frac{x^2}{2\sigma^2})dx \quad (14)$$

where i and j are two adjacent positions, $d_{i,j}$ is the distance between the two characters, σ represents the standard deviation of the function. We set this hyper-parameter as 2.

$$\text{GDA}(Q, K, V) = \text{softmax}(\frac{QK^TG}{\sqrt{d_k}})V \quad (15)$$

where GDA represents the Gaussian directional attention, d_k is the dimension of the matrix K. Q, k and V are vectors which are similar to the base Transformer. The multi-head attention are computed as:

$$\begin{aligned} MultiGDA &= [head_1, head_2, ..., head_k]W^o \\ head_i &= GDA(QW_i^q, KW_i^k, VW_i^v) \end{aligned} \quad (16)$$

CONFIG	Parameters
hidden states	768
optimizer	Bert Adam
learning rate	[3e-5,**2e-5**,1e-5]
batch size	[16,32,64,**256**]
dropout	[**0.1**,0.2,0.4]
epochs	20

Table 6: The hyper-parameters of the teacher model.

CONFIG	Parameters
hidden states	768
optimizer	Bert Adam
learning rate	[3e-5,2e-5,**1e-5**]
batch size	[16,32,64,**256**]
dropout	[**0.1**,0.2,0.4]
epochs	40
kernel size	[2,**3**,4]
char embeds	[768]

Table 7: The hyper-parameters of the conPrune model.

CONFIG	Parameters
char embeds	[**50**,100,200]
bi-gram embeds	[**50**,100,200]
hidden states	[128,**256**,**512**]
optimizer	Adam
learning rate	[0.01,**0.001**,0.002]
batch size	[**16**,**32**,64,256]
dropout	[0.1,**0.2**,0.4]
hidden layers	[1,2,**3**]
epochs	30

Table 8: The hyper-parameters of the LSTM student model.

where W^O, W_i^Q, W_i^K, W_i^V are trainable parameters. The layer normalization is adopted in the end of each multi-head Gaussian directional attention layer.

B Hyper-parameter Setting

To improve the reproducibility, we list all important hyper-parameters of the teacher model (Table 6), the student models (Table 7 and 8), the word-based NER models (Table 9) and the NMT model. We randomly pick 10% sentences from the training data as the development data for the tuning. In addition, we use the original development set of WMT-18 for MT. We utilize the uniform-sample to choose the hyper-parameters. In particular, we use the hyper-parameters of the modified Transformer model and the NMT model following previous studies (Vaswani et al., 2017; Duan and Zhao, 2020).

CONFIG	Parameters
input embeds	[50,100,**200**]
hidden states	[100,**200**,300]
optimizer	Adam
learning rate	[0.01,0.001,**0.005**]
batch size	[**16**,**32**,64,256]
dropout	[0.1,0.2,**0.5**]
hidden layers	[1,2,**3**]
epochs	30

Table 9: The hyper-parameters of the word-based NER model.

Label	Chinese sentence	Pinyin
I	以它为依托，乌镇大道科创集聚区应运而生	yi ta wei yi tuo, wu zhen da dao ke chuang ji ju qu ying yun er sheng.
II	剥蟹壳，吮蟹脚，挑蟹肉，蘸蟹料，食客感受着江南的美好	bao xie ke, shun xie jiao, tiao xie rou, zhan xie liao, shi ke gan shou zhe jiang nan de mei hao.

Figure 2: The Pinyin sequences for two Chinese sentences.

C Case Study

For NMT, it is difficult to analyze translation results as the interpretability of NMT is poor. We start with examples and focus on the differences between two translations with different segmentation results in addition to sentence-level BLEU scores. The Pinyin sequences for the two Chinese sentences are shown in Figure 2 and translations are shown in Table 10. We find that segmentation results with slight differences make translation results varying. The boundary of words may lead to these differences. There is a considerable discrepancy when the neural machine translation system stops training at different steps. That shows the neural machine translation system is unstable. It is full of uncertainty, and it still brings great challenges for the MT model itself and other crucial techniques.

LABEL	MODEL	SEG. RESULT	MT. RESULT	BLEU
I	TEACHER	yi/ ta/ wei/ yi tuo/ ,/ **wu zhen da dao/** ke chuang/ ji ju qu/ ying yun er sheng	with it as its base, the koku district of wuzhen boulevard came into being.	21.79
	CONPRUNE†	yi/ ta/ wei/ yi tuo/ ,/ **wu zhen/ da dao/** ke chuang/ ji ju qu/ ying yun er sheng	to rely on it, wuzhen boulevard science set up the agglomeration area emerged at the moment.	**70.71**
II	TEACHER	bao/ **xie ke/** ,/ shun/ **xie jiao/** ,/ tiao/ **xie rou/** ,/ zhan/ xie/ liao/ ,/ shi ke/ gan shou/ zhe/ shen/ chu/ jiang nan/ de/ mei hao	dipping crab shell, sucking crab foot, picking crab meat, dipping crab material, eating experience in the south of the good.	17.28
	CONPRUNE†	bao/ **xie/ ke/** ,/ shun/ **xie/ jiao/** ,/ tiao/ **xie/ rou/** ,/ zhan/ xie/ liao/ ,/ shi ke/ gan shou/ zhe/ shen/ chu/ jiang nan/ de/ mei hao	peeling crab shell, sucking crab feet, picking crab meat, dipping crab material, customers feel the good in the south of the river.	**27.08**

Table 10: The evaluation and results for MT. SEG. RESULT represents the segmentation results with different CWS methods. In particular, we use "/" to split the words.