Weighted self Distillation for Chinese word segmentation

Rian He[1], Shubin Cai[*2], Zhong Ming[*3], Jialei Zhang[4]
National Engineering Laboratory for Big Data System Computing Technology
College of Computer Science and Software Engineering
Shenzhen University, Shenzhen 518060, China
[1]herian2020@email.szu.edu.cn [2]shubin@szu.edu.cn
[3]mingz@szu.edu.cn [4]zhangjialei2021@email.szu.edu.cn

Abstract

Recent researches show that multi-criteria resources and n-gram features are beneficial to Chinese Word Segmentation (CWS). However, these methods rely heavily on such additional information mentioned above and focus less on the model itself. We thus propose a novel neural framework, named Weighted self Distillation for Chinese word segmentation (WeiDC). The framework, which only requires unigram features, adopts self-distillation technology with four hand-crafted weight modules and two teacher models configurations. Experiment results show that WeiDC can make use of character features to learn contextual knowledge and successfully achieve state-of-the-art or competitive performance in terms of strictly closed test settings on SIGHAN Bakeoff benchmark datasets. Moreover, further experiments and analyses also demonstrate the robustness of WeiDC. Source codes of this paper are available on Github[1].

1 Introduction

Chinese is written without explicit word delimiters, while numerous Natural Language Processing (NLP) applications are word-based. Moreover, CWS is always a fundamental and essential step for processing most language tasks.

Following the pace of many researchers (Sun and Xu, 2011; Chen et al., 2015; Ke et al., 2021), we also choose [B, I/M, E, S] tags (Beginning, Inside/Middle, End, Single character), which represent the precise position of a character in one word. Figure 1 gives a simple example.

Char:	我	喜	欢	大	自	然	。
Tag:	S	B	E	B	I	E	S

Figure 1: The [B, I, E, S] tagging scheme. "我喜欢大自然。" ("I love nature.")

Generally, a CWS task usually consists of three important parts: Embedding, Encoder and Decoder. Google published two papers, Mikolov et al. (2013a) and Mikolov et al. (2013b), and distributed representation has been widely used in NLP due to its low dimensions and efficiency in semantic similarity. Most researchers keep a close eye to the encoder part which includes Maximum Entropy (ME) (Berger et al., 1996), feed-forward neural network (Zheng et al., 2013), recursive neural network (Wang and Xu, 2017) , long-short-term memory (LSTM) (Chen et al., 2015), Pre-training of Deep Bidirectional Transformers such as BERT (Tian et al., 2020) and other models. As for the decoder part, in addition to softmax, Conditional Random Fields (CRF) (Lafferty et al., 2001) usually plays a vital role because it can use the rich contextual feature in the annotation process.

With the prevalence of pre-training and fine-tuning, transformer-based pre-trained models have dominated the field of CWS in recent years. Given sufficient training data, the pre-trained models (Nakkiran et al., 2020; Xu et al., 2020) have achieved remarkable results. However, these works may suffer from poor predicting accuracy when rare words or OOV (out-of-vocab) words exist. What's more, Huang and Zhao (2007) confirm that the loss of word segmentation accuracy, caused by OOV words, is at least 5 times greater than word segmentation ambiguity. We believe that improving the accuracy of the OOV words is worthy of further exploration.

Unlike traditional Knowledge Distillation (KD) methods, self distillation teaches a student network by itself instead of a separate teacher (Xu and Liu, 2019; Zhang et al., 2019) . Specifically, during one training epoch, the best student model or the student model from the last iteration will be saved as the teacher model for the next training epoch to teach the student itself.

Moreover, we believe that the student model

[1] Our code implementation. https://github.com/Anzi20/WeiDC

Findings of the Association for Computational Linguistics: ACL 2022, pages 1757 - 1770
May 22-27, 2022 ©2022 Association for Computational Linguistics

should study knowledge selectively according to the importance of information, so it is a practical solution to add an weight matrix to the training process. Different from the temperature distillation technology proposed by Hinton et al. (2015), we subtly utilize the information gap between pseudo labels, predicted by the teacher model or student model, and real labels to obtain the hand-crafted weight matrix. From another perspective, the process of acquiring weight matrices can also be seen as a kind of communication between teachers and students. Finally, to more precisely demonstrate the impact of WeiDC, we will temporarily ignore all external information.

Our contributions are summarized below. We proposed WeiDC, which only requires unigram features and adopts self-distillation technology with four hand-crafted weight modules and two teacher models configurations. Considering there are few choices of weight measures, it is also a challenge to design a feasible method to obtain a rational weight value. We also performed various experiments, such as testing its robustness in some low-resource settings, and explored the efficiency of our framework by combining different encoders and decoders. Experimental results from four widely used benchmark datasets confirm that WeiDC can achieve state-of-the-art or competitive performance, especially in OOV recall.

2 Related Work

Xue and Converse (2002) first treat CWS as a sequence labeling task and use a maximum entropy tagger to train the data set. Xu (2003) shows a unique charm of the sequential labeling method in the CWS bakeoffs (Sproat and Emerson, 2003), especially its results on R$_{OOV}$ (Recall of Out Of Vocabulary). People thus turn their attention to the research of sequence labeling method (Peng et al., 2004; Zhao et al., 2006; Zhao and Kit, 2008). And Huang and Zhao (2007) conclude that treating the word segmentation process as a character labeling problem can balance the recognition of vocabulary words and unregistered words, because all words are realized through one unified character marking process. In general, our research is related to the following works.

Pre-trained Frameworks Transformer-based pre-trained models, such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) , and ZEN (Diao et al., 2020), have demonstrated excellent performance in CWS tasks. Qiu et al. (2020) propose one unified model for multi-criteria CWS by leveraging the powerful ability of the Transformer encoder. Huang et al. (2020) also use BERT to capture various annotation criteria among datasets. Ke et al. (2021) propose a CWS-specific pre-trained model METASEG. Tian et al. (2020) and Liu et al. (2021) consider the combination of lexicon features and BERT for CWS. Huang et al. (2021) propose a semi-supervised neural method based on RoBERTa encoder through pseudo labels.

Knowledge Distillation Hinton et al. (2015) first propose knowledge distillation, using a larger network to teach a smaller network. Tang et al. (2019) choose to distill knowledge from BERT, a state-of-the-art language representation model, into a simple heterogeneous model. Huang et al. (2020) also extract knowledge from BERT to a truncated (3 or 6 layers) BERT to balance computational cost and segmentation accuracy on CWS tasks. Jiao et al. (2020) adopt multiple distilling strategies to reduce the number of parameters of the pre-trained language models. Huang et al. (2021) collect massive unlabeled data and distill knowledge from the teacher model to the student model by generating pseudo labels. Zhang et al. (2019) put forward self-distillation, which has recently been used in computer vision, but not commonly used in NLP.

To summarize, for further improving word segmentation accuracy, many researchers make use of lexicon information (Tian et al., 2020; Liu et al., 2021), multi-criteria label data (Chen et al., 2017; Huang et al., 2020; Qiu et al., 2020; Ke et al., 2020) and even unlabeled data (Sun and Xu, 2011; Zhang et al., 2013; Huang et al., 2021).

3 The WeiDC Framework

Huang and Zhao (2007) point out that CWS is the first step of most Chinese information processing systems, which usually relies on the shallow information of the text content, such as character features, which is distinct from the idea, "understand first and then segment words". As shown in Figure 2, we adopted the traditional word segmentation scheme, but added self distillation and weight modules to the training phase.

3.1 The Sequential Part

The traditional word segmentation scheme consists of the Embedding layer, Encoder layer, and Decoder layer. Formally, x is always seen as all

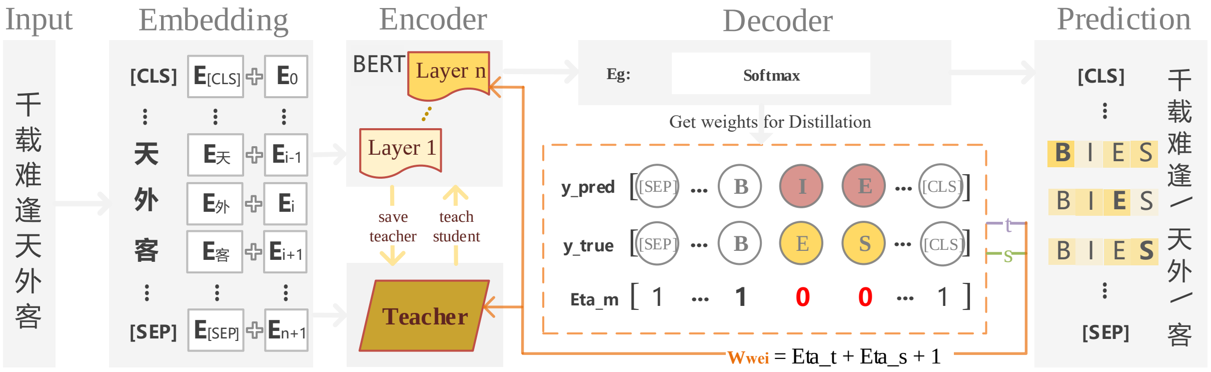


Figure 2: The WeiDC framework. The sentence, "千载难逢天外客" ("A once-in-a-lifetime visitor from outside the sky"), is from the MSR testing corpus. And it's difficult to split "天外客" ("A visitor from outside the sky").

marked data sequences and $x = [x_1, x_2, ..., x_n]$, and y is over corresponding label sequences and $y = [y_1, y_2, ..., y_n]$. We choose the BERT model to get character embeddings and encode these embeddings. After that, the encoder's outputs are fed into the decoder layer to obtain predicted tags.

Embedding layer We use BertTokenizer to obtain our input embeddings. Each character embedding consists of token embedding and position embedding. We don't need to consider the Next Sentence Prediction problem and remove token_type embedding. Additionally, to easily explore various weight mechanisms, WeiDC ignores unlabeled data or n-gram features.

Encoder layer Once obtaining character embeddings, they will be fed into an encoder, such as BERT or its derivative models. We choose `bert-base-chinese`[2] version and only need `config.json`, `pytorch_model.bin`, and `vocab.txt` to train linguistic data. Vaswani et al. (2017) give BERT, based on Transformer, an abundant description. We decide to omit its background description here. Furthermore, we also take RoBERTa[3] as our encoder to explore the impact of various pre-trained models on the CWS experiments.

Decoder layer Compared with Hidden Markov Models, Lafferty et al. (2001) present CRF for building probabilistic models to mark and segment the sequence data with weak independence assumptions.

$$p(y_i|x_i) = \frac{\exp(W_c \cdot z_i + b_c)}{\sum_{y_{i-1}y_i} \exp(W_c \cdot z_i + b_c)} \qquad (1)$$

In addition, softmax is also a frequent decoder, which can efficiently convert logit to probability regardless of intrinsic correlation.

$$p(y_i|x_i) = \log \frac{\exp(z_i^d)}{\sum_d^{\mathcal{D}} \exp(z_i^d)} \qquad (2)$$

where $z_i \in \mathbb{R}^{|\mathcal{D}|}$ is logits and z_i^d is the value at dimension d in z_i. $p(y_i|x_i)$ is the corresponding probability value. $W_c \in \mathbb{R}^{|\mathcal{D}|\times|\mathcal{D}|}$ and $b_c \in \mathbb{R}^{|\mathcal{D}|}$ are trainable parameters of CRF. $y_{i-1}y_i$ models the state from y_{i-1} to y_i.

We continue to operate on the probability $(p(y|x))$ to get the predicted label $(\hat{y})$.

$$\hat{y} = argmax\ p(y|x) \qquad (3)$$

Through comparative experiments, Qiu et al. (2020) conclude that with or without CRF does not make much difference. Since CRF is more complex and the training cost is higher, we mainly try softmax to decode logits to make full use of computing resources.

3.2 Weight Mechanism

During one training epoch, the pseudo labels $(\hat{y})$ from t or s are compared with corresponding true labels (y), which can be expressed by formula 4. t and s indicate that $\hat{y}$ come from the teacher model or student model, respectively. η refers to the information difference between $\hat{y}$ and y.

$$\eta_m = |\hat{y}_m - y|, m = t, s \qquad (4)$$

In the process of executing equation 4, we use absolute value operations. When one pseudo label is equal to the corresponding true label, we get 0, otherwise we get a positive number. Since the

[2] `https://huggingface.co/bert-base-chinese/tree/main`

[3] `https://github.com/brightmart/roberta_zh` (RoBERTa_zh_L12 PyTorch)

result is the opposite of what we want, we have to perform 5 and 6.

$$F(j) = \begin{cases} 0, & j = 0 \\ 1, & j \neq 0 \end{cases} \tag{5}$$

$F(j)$ converts all positive numbers to 1 and j is a variable symbol. Then, the intermediate value is processed by equation 6 to get the final result.

$$\eta_m = 1 - F(\eta_m) \tag{6}$$

We hope there will be enough communication between the teacher and student to obtain a reasonable weight value, so we designed equation 7. w_{wei}^1 is the first type of weight vector.

$$w_{wei}^1 = \eta_t + \eta_s + 1 \tag{7}$$

The meaning of equation 7 is very concise. During distillation, samples with higher accuracy are given more attention, while samples with lower accuracy are given less attention. Moreover, to avoid losing the basic information carried by each sample, we need to make sure that the minimum value of w_{wei}^1 is 1, we thus add 1.

We also notice that η_t and η_s may contain various amounts of knowledge. Therefore, we multiply η_t or η_s by 2 to get equations 8 and 9, respectively. Certainly, other coefficients can also be selected according to actual needs.

$$w_{wei}^2 = 2 \cdot \eta_t + \eta_s + 1 \tag{8}$$

$$w_{wei}^3 = \eta_t + 2 \cdot \eta_s + 1 \tag{9}$$

From another perspective, if the teacher model is correct and the student model is wrong, this kind of knowledge should be more valuable. We thus get another calculation method, which is described in equation 10, to obtain the weight vector.

$$w_{wei}^4 = 2 \cdot \eta_t - \eta_s + 2 \tag{10}$$

We must add 2 to ensure that the minimum value of w_{wei}^4 is 1.

Finally, according to different weight modules, all possible values of a single character (marked as k) are shown in Table 1. The above four weight mechanisms show that different key factors affect the weight value. In other words, for the same pseudo label, different reference factors will lead to various weight values.

η_{t_k}	η_{s_k}	$w_{wei_k}^1$	$w_{wei_k}^2$	$w_{wei_k}^3$	$w_{wei_k}^4$
1	1	3	4	4	3
1	0	2	3	2	4
0	1	2	2	3	1
0	0	1	1	1	2

Table 1: All possible weight values of character k.

For example, if we consider that words with low frequency can better reflect the models' performance, we can increase their weights to penalize the loss of misclassifying these words. As a result, the student model will pay more attention to low-frequency words.

According to different distillation scenarios or learning needs, it is necessary to choose appropriate reference factors to design weight calculation methods. Here, we take the segmentation difficulty of words as a reference standard.

3.3 Distillation

Unlike self-training, self-distillation takes a fully supervised way to dig the potential of the model itself, requiring no auxiliary models or data. In this paper, the teacher model comes from two sources, either the student model from the last iteration (D_{last}) or the student model with the best historical performance (D_{best}) .

The student also learns from two sources of information, predicted probabilities from the teacher and one-hot ground-truth label. Hence, the final loss ($\mathcal{L}_{KD}$) consists of two parts, cross-entropy loss ($\mathcal{L}_{CE}$) and distillation loss ($\mathcal{L}_{Distill}$) :

$$\mathcal{L}_{KD} = (1 - \alpha) \cdot \mathcal{L}_{CE} + \alpha \cdot \mathcal{L}_{Distill} \tag{11}$$

To balance the above two losses, we need a coefficient α, which is also set to a fixed value during the training phase.

$\mathcal{L}_{CE}$ is to penalize the cross-entropy loss between the predicted label ($\hat{y}$) against the true label (y):

$$\mathcal{L}_{CE} = -\sum_{x} y \log \hat{y}_{(x)} \tag{12}$$

$\mathcal{L}_{Distill}$ is to reduce the mean-squared-error loss between the teacher's logits ($z^{(T)}$) and the student's logits ($z^{(S)}$), and w_{wei} can be any of the above four weight types.

$$\mathcal{L}_{Distill} = ||w_{wei} \cdot z^{(T)} - w_{wei} \cdot z^{(S)}||_2^2 \tag{13}$$

To better verify the effect of WeiDC, the temperature distillation technology is not considered here.

Dataset	MSR		PKU		AS		CITYU	
	train	test	train	test	train	test	train	test
Char	4,050K	184K	1,826K	173K	8,368K	198K	2,403K	68K
Word	2,368K	107K	1,110K	104K	5,450K	123K	1,456K	41K
Char types	5,168	2,838	4,698	2,934	5,979	3,628	4,832	2,663
Word types	88,119	12,923	55,303	13,148	141,339	18,759	69,085	8,993
OOV Rate	-	2.7	-	5.8	-	4.3	-	7.2

Table 2: Corpus details of four CWS datasets

Distinct from previous studies on knowledge distillation, our framework adds the weight mechanism, allowing the teacher and the student to communicate fully to focus on more valuable knowledge. Furthermore, the teacher is not a static model but dynamically evolves as training proceeds. Hence, the weight vector will also alter as the teacher model changes so that the student model can learn richer knowledge.

4 Experiments

4.1 Dataset and Evaluation Metric

The second SIGHAN international Chinese word segmentation bakeoff (Emerson, 2005), which includes **MSR**, **PKU**, **AS** and **CITYU** datasets, is frequently used in CWS tasks. Since AS and CITYU are traditional Chinese characters, we convert these data into simplified ones by following previous studies (Chen et al., 2015; Qiu et al., 2020; Tian et al., 2020) . We will use these datasets in the following experiments and corpus details are listed in Table 2.

We also choose precision (P), recall (R), F-score, and R_{OOV}, which is the recall for out-of-vocabulary (OOV) words, to evaluate segmentation performance. Specifically, we first record the word information in the complete training corpus and then divide the corpus into a training set and validation set. Besides, we take no extra resources but only training corpus to train our model.

4.2 Baselines

According to whether to use a pre-trained model such as BERT as the encoder, we have selected two types of baselines, Non-pretrained Models and Pre-trained Models.

Non-pretrained Models Chen et al. (2017) propose adversarial multi-criteria learning for CWS tasks by exploiting the underlying shared knowledge across multiple heterogeneous criteria. Ma et al. (2018) also point out that using external knowledge can improve the CWS accuracy. Gong et al. (2019) provide a more flexible solution to transfer the learned information to new criteria. They all use the bidirectional LSTM encoder. Qiu et al. (2020) propose one unified model for multi-criteria CWS based on the Transformer encoder. Through the Gaussian-masked Directional (GD) Transformer, Duan and Zhao (2020) try to further strengthen the model itself to perfect CWS tasks.

Pre-trained Models Huang et al. (2020) propose a domain adaptive segmenter to exploit various open-domain knowledge. Tian et al. (2020) use key-value memory networks to incorporate wordhood information with BERT or ZEN as the encoder. Ke et al. (2021) put forward a CWS-specific pre-trained model to alleviate the discrepancy between pre-trained models and downstream CWS tasks. Nguyen et al. (2021) propose a span labeling approach to model n-gram features for word segmentation.

4.3 Training Details

All experiments are implemented on the hardware with Intel(R) Xeon(R) Gold 6240 CPU @ 2.60GHz and NVIDIA Tesla V100. Following previous works (Ma et al., 2018; Qiu et al., 2020), we randomly select 10% training data for development and only use its testing set at the end of the training phase. Similar to the previous work (Tian et al., 2020), we performed other preprocessing measures on all data sets.

During fine-tuning, we use Adam with the learning rate of 2e-5. Both train_batch_size and eval_batch_size are 16. As for the trade-off hyperparameter (α), we randomly select 1% of the training set to explore the influence of various α on WeiDC. We observe that when α is 0.3, WeiDC performs better.

Besides, we train all models up to 50 with some early stopping strategies, such as "patient epochs"

Model	MSR		PKU		AS		CITYU		AVG	
	F	R$_{OOV}$	F	R$_{OOV}$	F	R$_{OOV}$	F	R$_{OOV}$	F	R$_{OOV}$
Chen et al. (2017) ⋆	96.04	71.6	94.32	72.67	94.75	75.37	95.55	81.4	95.17	75.26
Ma et al. (2018) †	98.1	80.0	96.1	78.8	96.2	70.7	97.2	87.5	96.9	79.25
Gong et al. (2019) ⋆	97.78	64.2	96.15	69.88	95.22	77.33	96.22	73.58	96.34	77.82
Qiu et al. (2020) ⋆†	98.05	78.92	96.41	78.91	96.44	76.39	96.91	86.91	96.95	80.28
Duan and Zhao (2020)	97.6	-	95.5	-	95.7	-	95.4	-	96.05	-
Huang et al. (2020) ⋆	97.9	84.0	96.7	81.6	96.7	77.3	97.6	90.1	97.23	83.25
Tian et al. (2020) † (BERT)	98.28	86.67	96.51	86.76	96.58	78.48	97.8	87.57	97.29	84.87
Tian et al. (2020) † (ZEN)	98.4	84.87	96.53	85.36	96.62	79.64	97.93	90.15	97.37	85.0
Ke et al. (2021) ⋆‡	**98.5**	83.03	**96.92**	80.9	**97.01**	**80.89**	**98.2**	**90.66**	**97.66**	83.87
Nguyen et al. (2021) †	98.31	85.32	96.56	85.83	96.62	79.36	97.74	87.45	97.31	84.49
WeiDC (BERT)	98.28	86.39	96.59	87.21	96.76	80.23	97.79	87.58	97.36	85.35
WeiDC (RoBERTa)	98.43	**87.17**	96.74	**87.48**	96.59	79.26	97.95	89.93	97.43	**85.96**

Table 3: First two blocks record different baselines, namely Non-pre and Pre. The last block is our scores. ⋆ uses a multi-criteria learning framework, which means that the marked training data are different from the rest. † uses lexicons or n-gram features. ‡ uses a CWS-specific pre-trained model.

of 3 and "minimum F value" of 0.0001. Specifically, when the gap between the current F value and the optimal F value is less than 0.0001, we will not replace our saved model to avoid frequently updating the teacher model. Table 4 summarizes all the vital parameters.

mininum F value	1e-4	train_batch_size	16
num_train_epochs	50	eval_batch_size	16
patient epochs	3	learning_rate	2e-5
train : eval	9 : 1	alpha (α)	0.3

Table 4: Hyper parameters of WeiDC.

We take [B, I, E, S] tagging scheme in our experiments. To explore the influence of diverse weight modules on CWS, we will only try BERT and RoBERTa as our encoder. As for BERT, we follow the default settings in their paper (Devlin et al., 2019). In addition to combining four weight modules and two types of teacher models, we also plan to conduct some exploratory experiments, such as testing the performance of WeiDC on a small amount of training data.

5 Results and Analysis

In this section, we firstly report the results of WeiDC and its comparison with the state-of-the-art works available. Then we explore the robustness of WeiDC through lots of experiments in different low-resource settings. We also analyze the impact of OOV words on the model. Finally, we perform various NER tasks to test WeiDC's effectiveness.

5.1 Main Results

Several observations are drawn from Table 3 and Table 5, where the overall F-score and OOV recall are all reported.

First, Table 3 demonstrates that pre-trained models, with lots of prior knowledge, perform better than non-pretrained models, especially in OOV recall. Compared with baselines listed in Table 3, the results in these experiments not only confirm that self distillation and weight mechanism are effective methods to benefit CWS without any auxiliary data or CWS-specific pre-trained models, but also fully illustrate that the design of WeiDC can enhance the model learning ability.

Second, as shown in Table 5, WeiDC achieved exciting results on R$_{OOV}$ with maintaining competitive performance on F-score. For instance, when we took BERT as our encoder, WeiDC improved the F-score by 0.16% on average, from 97.2% to 97.36%, and the R$_{OOV}$ score by 1.71% on average, from 83.64% to 85.35%.

Third, in most cases, D_{best} outperforms D_{last}, and we speculate that updating the teacher model too frequently will be detrimental to the learning process of the student model. Besides, different CWS tasks need various weight modules, so it is essential to choose reasonable weight mechanisms according to the characteristics of datasets.

Fourth, with BERT as the encoder and softmax as the decoder, our base model is powerful, but the improvement of WeiDC on R$_{OOV}$ scores is still very decent. Specifically, under the current

Model	MSR		PKU		AS		CITYU		AVG	
	F	R$_{OOV}$	F	R$_{OOV}$	F	R$_{OOV}$	F	R$_{OOV}$	F	R$_{OOV}$
BERT(base)	98.22	85.22	96.5	85.6	96.44	77.37	97.63	86.35	97.2	83.64
$+D_{best}$	98.22	85.58	**96.59**	87.04	96.64	79.51	97.68	86.52	97.28	84.66
$+D_{best} + w^2_{wei}$	98.17	86.07	96.53	**88.03**	96.71	**80.57**	97.6	85.4	97.25	85.02
$+D_{best} + w^4_{wei}$	**98.28**	**86.39**	**96.59**	87.21	**96.76**	80.23	**97.79**	**87.58**	**97.36**	**85.35**
RoBERTa(base)	98.33	86.74	96.58	87.04	96.34	76.14	97.8	88.8	97.26	84.68
$+D_{best}$	98.43	86.67	96.56	86.34	96.52	78.47	97.84	89.38	97.34	85.22
$+D_{best} + w^2_{wei}$	98.33	86.21	**96.79**	**88.34**	**96.6**	**79.26**	**97.96**	**90.33**	97.42	**86.04**
$+D_{best} + w^4_{wei}$	**98.43**	**87.17**	96.74	87.48	96.59	**79.26**	97.95	89.93	**97.43**	85.96

Table 5: Ablation studies combining self distillation and four weight modules. Complete results can be found in the Appendix Tables 10 and 11.

Sampling Rates	1%		5%		10%		20%		50%		80%		100%		AVG	
	F	R$_{OOV}$	F	R$_{OOV}$	F	R$_{OOV}$	F	R$_{OOV}$	F	R$_{OOV}$	F	R$_{OOV}$	F	R$_{OOV}$	F	R$_{OOV}$
BERT(base)	**93.92**	**83.38**	94.37	77.65	94.72	76.74	95.83	83.46	96.15	85.13	96.33	84.18	96.5	85.6	95.4	82.31
$+D_{best}$	93.7	82.95	95	82.33	**95.79**	86.56	**95.98**	85.63	96.34	85.6	96.36	84.91	**96.59**	87.04	**95.68**	85.0
$+D_{best} + w^2_{wei}$	93.29	83.3	**95.37**	**87.86**	95.69	**87.36**	95.82	**86.39**	**96.35**	**87.96**	**96.56**	**87.73**	96.53	**88.03**	95.66	**86.95**

Table 6: Scores on PKU test set in low-resource settings.

experimental conditions (listed in table 4), w^4_{wei} has the best overall performance on all data sets, while w^3_{wei} has the worst performance.

Last, RoBERTa outperforms BERT when we deal with the CWS task. If CRF is used as the decoder, the CWS model seems to be more prone to overfitting, resulting in worse word segmentation.

5.2 Low-Resource Settings

In real life, the training corpus is usually insufficient, and it is valuable to measure the performance of CWS models in some low-resource settings. The partition criterion of our training sets follows Ke et al. (2021), whose sampling rates are 0.01, 0.05, 0.1, 0.2, 0.5, 0.8, and 1.0. For easy operation, we will obtain the above training datasets after randomizing the original training dataset but finally test on the same original testing dataset.

We decided to perform the above experiment on PKU without changing any parameters in Table 4. We first took BERT as the base model and gradually added D_{best} and w^2_{wei}. Related results of the experiment are shown in Table 6.

We notice that the performance of all models is greatly affected by sampling rates, especially at a low ratio such as 1% and 5%. In addition, self distillation can significantly improve the effect of CWS, and weight mechanisms can further increase the R$_{OOV}$ scores.

Specifically, when the sampling rate drops from 100% to 5%, "$BERT + D_{best}$" and "$BERT + D_{best} + w^2_{wei}$" have better F1 scores than "$BERT$". For R$_{OOV}$ scores, "$BERT$" decreases by 7.95% while that of "$BERT + D_{best}$" only decreases by 4.71%. Surprisingly, "$BERT + D_{best} + w^2_{wei}$" almost always maintains high R$_{OOV}$ scores, fluctuating between 87% and 88%. We do not pay too much attention to 1%, because the sample size may be too small to reflect the real performance of the model.

Generally speaking, the above results confirm that WeiDC has strong robustness when manual annotation resources are insufficient.

5.3 OOV Words

From the above experiments, WeiDC worked well in R$_{OOV}$. To verify the performance of each model on OOV words, we operated the PKU training corpus to train all models but took other testing data sets to evaluate these models.

We first digitized the discrepancy between the training set of PKU and the test sets of MSR, AS and CITYU. For visual comparison, we also listed the distribution of OOV words in the PKU test set. See Table 7 for more details. It should not be ignored that both AS and CITYU are traditional Chinese datasets, where words may be slightly different, such as "铁公路" ("iron road") on CITYU while "铁路" ("railway") on PKU.

As shown in Table 8, WeiDC almost performs better than the base model on all three testing tasks, especially in R$_{OOV}$. According to table 7 and Table 8, the effect of WeiDC on the test set with a higher

OOV$_{word}$	**PKU**		**MSR**		**AS**		**CITYU**	
	Type	Freq	Type	Freq	Type	Freq	Type	Freq
NotInPKU_Train	2863	6006	4100	8110	8386	18006	3099	6726
All Test Word	13148	104372	12923	106873	18759	122610	8993	40936
OOV Rate	21.78	5.75	31.73	7.60	44.70	14.69	34.46	16.43

Table 7: OOV words for the four CWS test sets. "NotInPKU_Train" represents words that appear in the test set while not in the PKU training set. Column "Type" only includes the type of OOV word, but column "Freq" considers the frequency.

Model	MSR		AS		CITYU	
	F	R$_{OOV}$	F	R$_{OOV}$	F	R$_{OOV}$
BERT(base)	86.95	20.51	90.05	71.82	90.77	73.51
$+D_{best}$	+0.0	**+0.88**	+0.45	+2.38	**+0.52**	+2.2
$+D_{best} + w_{wei}^2$	-0.08	+0.81	**+0.47**	**+2.41**	+0.51	**+3.06**

Table 8: Train on PKU, but test on other three datasets.

frequency of OOV words is more distinct. However, the number of types of OOV words seems to be less beneficial.

We finally checked the PKU and MSR datasets to find out why all models performed poorly on MSR. The word segmentation standards of the above two corpora are very different, such as "最大" ("biggest") on MSR while "最 大" ("most" and "big") on PKU, which directly causes all models to perform better on AS and CITYU, but poorly on MSR.

5.4 NER Tasks

Similar to CWS tasks, Named Entity Recognition (NER) tasks can also be performed in the form of sequence annotations. To further explore the effectiveness of the weight mechanism and compare which weight mechanism performs better, we conduct some NER experiments. All hyperparameters are the same as the CWS task. The relevant results are shown in Appendix Table 13.

We can get the following suggestions. First, the hand-crafted weight module can improve sequence labeling tasks, whether CWS or NER. Second, w_{wei}^4 has the best overall performance among all weight mechanisms and is also a good choice when the features of the training dataset are unclear.

Moreover, the labeling rules of various datasets vary widely, so it is almost impossible to design a general weight mechanism. This also explains that our chosen parameters do not always yield the best results. To focus our attention on experimental exploration, we did not spend much time on parameter tuning.

6 Case Study

For CWS tasks, it is very hard to get the right segmentations if two adjacent words, such as "天外" ("outside the sky") and "客" ("guest"), both appear for the first time, as shown in Table 9. Unfortunately, WeiDC can't handle this problem properly either. However, we find that if we add some valuable context, our model can still get rational results.

Gold	千载难逢 天外 客
Original	$text$: 千载难逢天外客 $BERT$: 千载难逢 **天外客** $+D_{best} + w_{wei}^2$: 千载难逢**天外客**
Replace 1	天外的人，千载难逢天外客 天 外 的 人，千载难逢 **天外 客** 天外 的 人 , 千载难逢 **天外客**
Replace 2	天外的客，千载难逢天外客 天外 的 客 ， 千载难逢 **天外 客** 天外 的 客 ， 千载难逢 **天外 客**
Replace 3	天外的流星，来做客，千载难逢天外客 天外 的 流星 ， 来 做客 ， 千载难逢 **天外 客** 天外 的 流星 ， 来 做客， 千载难逢 **天外客**
Replace 4	客人说，见到了天外来的流星，千载难逢天外客 客人 说 ， 见 到 了 天 外 来 的 流星 ， 千载难逢 **天外 客** 客人 说 ， 见 到 了 天外 来 的 流星 ， 千载难逢 **天外 客**

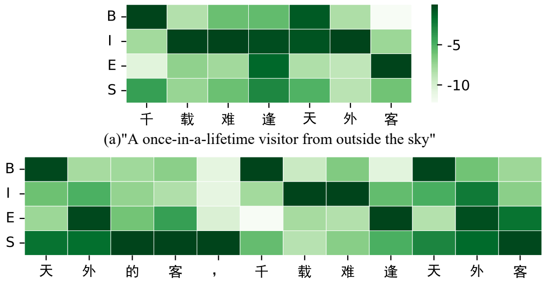
Table 9: "千载难逢天外客" ("A once-in-a-lifetime visitor from outside the sky"). In each block, the first line is a raw text, and the last two lines are segmentation results of BERT and WeiDC, respectively. Both models are trained on PKU.

Although in some cases both "天外客" ("A visitor from outside the sky") and "天外 客" ("outside the sky" and "guest") are rational representations, here we assume that "天外 客" ("outside the sky" and "guest") is correct one and let these models learn it by enhancing the semantic environment.

First, according to "Replace 1" and "Replace 4", if only "天外" ("outside the sky") appears in the previous text, BERT obtains "天外 客" ("outside the sky" and "guest") at the cost of inconsistent segmentation criteria in "天外" ("outside the sky"). For WeiDC, "天外客" ("A visitor from outside the sky") is regarded as a derivative of "天外" ("outside the sky"), as shown in "Replace 1". After semantic information gets enriched, the possibility of "天外" ("outside the sky") becoming an independent word increases, so the correct result is obtained. We also notice that when text content is rich, WeiDC will get desired results even if there is interference information such as "外来" ("outside") in the added semantic knowledge.

Second, from "Replace 2", when "的" ("of") locates between "天外" ("outside the sky") and "客" ("guest"), both BERT and WeiDC learn the right segmentation position by treating "的" ("of") as a single word. We analyzed the PKU training set for further exploration and found that "的" ("of") is a high-frequency single-character word. When we blur the semantic information, as shown in "Replace 3", WeiDC treats "天外客" ("A visitor from outside the sky") as a word, while BERT can still obtain the correct segmentation. We speculate that the added interference information hurts the small text content. From another perspective, WeiDC has a strong ability to learn contextual knowledge from different semantic environments to assist CWS tasks.

Last but not least, we make heatmaps to visualize the word segmentation process in Figure 3.



(a)"A once-in-a-lifetime visitor from outside the sky"

(b) "A visitor from outside the sky, a once-in-a-lifetime visitor from outside the sky"

Figure 3: Heatmaps of the label probability.

7 Conclusion

In this paper, we proposed a novel framework named WeiDC, which could make good use of the knowledge in teacher models through self-distillation. The framework also follows the sequence labeling paradigm but first applies self distillation and weight mechanism to CWS, combining four hand-crafted weight modules and two types of teacher models. Experimental results show that WeiDC could achieve higher performance on four CWS datasets, with the average F-score ranking second and the average R$_{OOV}$ score ranking first.

However, for non-sequential labeling problems, such as text classification, a paragraph only corresponds to one tag, so the number of labels is too small, which may render the method in this paper ineffective. How to solve such a dilemma deserves more exploration. Besides, it is also promising to consider whether more efficient weight methods exist.

8 Acknowledgments

We thank the anonymous reviewers for constructive and expert comments, and the support of National Natural Science Foundation of China No. 61836005.

References

Adam L. Berger, Stephen Della Pietra, and Vincent J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.

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, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1197–1206. Association for Computational Linguistics.

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, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1193–1203. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Shizhe Diao, Jiaxin Bai, Yan Song, Tong Zhang, and Yonggang Wang. 2020. ZEN: pre-training chinese text encoder enhanced by n-gram representations. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 4729–4740. 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 2020, Online, November 16-20, 2020*, pages 3862–3872. 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, SIGHAN@IJCNLP 2005, Jeju Island, Korea, 14-15, 2005*. Association for Computational Linguistics.

Jingjing Gong, Xinchi Chen, Tao Gui, and Xipeng Qiu. 2019. Switch-lstms for multi-criteria chinese word segmentation. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6457–6464. AAAI Press.

Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. *stat*, 1050:1–9.

Changning Huang and Hai Zhao. 2007. Chinese word segmentation: A decade review. *Journal of Chinese information processing*, 27:8–19.

Kaiyu Huang, Junpeng Liu, Degen Huang, Deyi Xiong, Zhuang Liu, and Jinsong Su. 2021. Enhancing chinese word segmentation via pseudo labels for practicability. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 4369–4381. Association for Computational Linguistics.

Weipeng Huang, Xingyi Cheng, Kunlong Chen, Taifeng Wang, and Wei Chu. 2020. Towards fast and accurate neural chinese word segmentation with multi-criteria learning. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 2062–2072. International Committee on Computational Linguistics.

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, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 4163–4174. Association for Computational Linguistics.

Zhen Ke, Liang Shi, Erli Meng, Bin Wang, Xipeng Qiu, and Xuanjing Huang. 2020. Unified multi-criteria chinese word segmentation with BERT. *CoRR*, abs/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, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 5514–5523. 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 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001*, pages 282–289. Morgan Kaufmann.

Gina-Anne Levow. 2006. The third international chinese language processing bakeoff: Word segmentation and named entity recognition. In *Proceedings of the Fifth Workshop on Chinese Language Processing, SIGHAN@COLING/ACL 2006, Sydney, Australia, July 22-23, 2006*, pages 108–117. Association for Computational Linguistics.

Wei Liu, Xiyan Fu, Yue Zhang, and Wenming Xiao. 2021. Lexicon enhanced chinese sequence labeling using BERT adapter. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 5847–5858. Association for Computational Linguistics.

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. *CoRR*, abs/1907.11692.

Ji Ma, Kuzman Ganchev, and David Weiss. 2018. State-of-the-art chinese word segmentation with bilstms. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 4902–4908. Association for Computational Linguistics.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Tomás Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013,* *Lake Tahoe, Nevada, United States*, pages 3111–3119.

Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. 2020. Deep double descent: Where bigger models and more data hurt. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Duc-Vu Nguyen, Linh-Bao Vo, Dang Van Thin, and Ngan Luu-Thuy Nguyen. 2021. Span labeling approach for vietnamese and chinese word segmentation. In *PRICAI 2021: Trends in Artificial Intelligence - 18th Pacific Rim International Conference on Artificial Intelligence, PRICAI 2021, Hanoi, Vietnam, November 8-12, 2021, Proceedings, Part II*, volume 13032 of *Lecture Notes in Computer Science*, pages 244–258. Springer.

Fuchun Peng, Fangfang Feng, and Andrew McCallum. 2004. Chinese segmentation and new word detection using conditional random fields. In *COLING 2004, 20th International Conference on Computational Linguistics, Proceedings of the Conference, 23-27 August 2004, Geneva, Switzerland*, pages 562–568. COLING.

Nanyun Peng and Mark Dredze. 2015. Named entity recognition for chinese social media with jointly trained embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 548–554. The 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, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 2887–2897. Association for Computational Linguistics.

Richard Sproat and Thomas Emerson. 2003. The first international chinese word segmentation bakeoff. In *Proceedings of the Second Workshop on Chinese Language Processing, SIGHAN 2003, Sapporo, Japan, July 11-12, 2003*, pages 133–143.

Weiwei Sun and Jia Xu. 2011. Enhancing chinese word segmentation using unlabeled data. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 970–979. Association for Computational Linguistics.

Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. 2019. Distilling task-specific knowledge from BERT into simple neural networks. *CoRR*, abs/1903.12136.

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, ACL 2020, Online, July 5-10, 2020*, pages 8274–8285. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Chunqi Wang and Bo Xu. 2017. Convolutional neural network with word embeddings for chinese word segmentation. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing, IJCNLP 2017, Taipei, Taiwan, November 27 - December 1, 2017 - Volume 1: Long Papers*, pages 163–172. Asian Federation of Natural Language Processing.

Canwen Xu, Wangchunshu Zhou, Tao Ge, Furu Wei, and Ming Zhou. 2020. Bert-of-theseus: Compressing BERT by progressive module replacing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 7859–7869. Association for Computational Linguistics.

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

Ting-Bing Xu and Cheng-Lin Liu. 2019. Data-distortion guided self-distillation for deep neural networks. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 5565–5572. AAAI Press.

Nianwen Xue and Susan P. Converse. 2002. Combining classifiers for chinese word segmentation. In *The First Workshop on Chinese Language Processing, SIGHAN@COLING 2002, Taipei, Taiwan, August 24 - September 1, 2002*, pages 1–7.

Linfeng Zhang, Jiebo Song, Anni Gao, Jingwei Chen, Chenglong Bao, and Kaisheng Ma. 2019. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 3712–3721. IEEE.

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, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 311–321. Association for Computational Linguistics.

Yue Zhang and Jie Yang. 2018. Chinese NER using lattice LSTM. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 1554–1564. Association for Computational Linguistics.

Hai Zhao, Changning Huang, and Mu Li. 2006. An improved chinese word segmentation system with conditional random field. In *Proceedings of the Fifth Workshop on Chinese Language Processing, SIGHAN@COLING/ACL 2006, Sydney, Australia, July 22-23, 2006*, pages 162–165. Association for Computational Linguistics.

Hai Zhao and Chunyu Kit. 2008. An empirical comparison of goodness measures for unsupervised chinese word segmentation with a unified framework. In *Third International Joint Conference on Natural Language Processing, IJCNLP 2008, Hyderabad, India, January 7-12, 2008*, pages 9–16. Association for Computer Linguistics.

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 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 647–657. Association for Computational Linguistics.

Model	MSR		PKU		AS		CITYU		AVG	
	F	R_{OOV}	F	R_{OOV}	F	R_{OOV}	F	R_{OOV}	F	R_{OOV}
BERT(base)	98.22	85.22	96.5	85.6	96.44	77.37	97.63	86.35	97.2	83.64
$+D_{best}$	98.22	85.58	96.59	87.04	96.64	79.51	97.68	86.52	97.28	84.66
$+D_{best} + w_{wei}^1$	98.16	85.75	96.63	87.29	96.68	80.62	97.78	86.52	97.31	85.05
$+D_{best} + w_{wei}^2$	98.17	86.07	96.53	**88.03**	96.71	80.57	97.6	85.4	97.25	85.02
$+D_{best} + w_{wei}^3$	98.11	85.61	96.5	86.33	96.67	80.57	97.68	86.59	97.24	84.78
$+D_{best} + w_{wei}^4$	**98.28**	86.39	96.59	87.21	96.76	80.23	**97.79**	**87.58**	**97.36**	**85.35**
$+D_{last}$	98.16	**86.43**	**96.64**	86.93	96.51	78.22	97.63	86.04	97.24	84.41
$+D_{last} + w_{wei}^2$	97.82	86.07	96.53	87.08	96.67	80.51	97.77	87.3	97.2	85.24
$+D_{last} + w_{wei}^4$	98.16	86.21	96.58	87.81	96.68	80.11	97.68	86.76	97.28	85.22
$+D_{best} + w_{wei}^2 + CRF$	98.17	85.37	96.37	85.26	96.75	80.96	**97.79**	86.86	97.27	84.61
$+D_{best} + w_{wei}^4 + CRF$	98.16	85.61	96.48	86.59	**96.77**	**81.63**	97.63	85.81	97.26	84.91

Table 10: Take BERT as the base model.

Model	MSR		PKU		AS		CITYU		AVG	
	F	R_{OOV}	F	R_{OOV}	F	R_{OOV}	F	R_{OOV}	F	R_{OOV}
RoBERTa(base)	98.33	86.74	96.58	87.04	96.34	76.14	97.8	88.8	97.26	84.68
$+D_{best}$	98.43	86.67	96.56	86.34	96.52	78.47	97.84	89.38	97.34	85.22
$+D_{best} + w_{wei}^1$	98.35	**88.55**	96.64	87.39	96.53	78.58	97.95	90.03	97.37	86.14
$+D_{best} + w_{wei}^2$	98.33	86.21	**96.79**	88.34	96.6	79.26	97.96	**90.33**	97.42	86.04
$+D_{best} + w_{wei}^3$	98.25	87.88	96.57	87.23	96.6	79.41	97.9	89.58	97.33	86.03
$+D_{best} + w_{wei}^4$	**98.43**	87.17	96.74	87.48	96.59	79.26	97.95	89.93	**97.43**	85.96
$+D_{last}$	98.4	87.45	96.53	87.19	96.48	78.36	97.89	89.93	97.33	85.73
$+D_{last} + w_{wei}^2$	98.15	86.89	96.7	**88.39**	96.54	79.21	97.94	90.2	97.33	86.17
$+D_{last} + w_{wei}^4$	98.23	87.88	96.67	88.09	**96.67**	**79.81**	**97.98**	89.82	97.39	**86.4**
$+D_{best} + w_{wei}^2 + CRF$	98.41	87.0	96.63	86.86	96.55	79.09	97.9	89.28	97.37	85.56

Table 11: Take RoBERTa as the base model.

A CWS Appendix

Combining two encoders and two decoders, the final results on the four datasets are included in Tables 10 and 11. All experiments adopted the same hyperparameters, as shown in Table 4.

We speculate that RoBERTa benefits from longer training time and larger batches of training data than BERT. In addition, some training tricks used in RoBERTa may also improve the performance of the pre-trained model, such as removing the next sentence prediction target, training longer sequences, and dynamically changing the mask pattern to be applied to the training data.

To our surprise, if CRF is used as the decoder, the CWS model seems to be more prone to overfitting, resulting in worse word segmentation. However, we also notice that CRF performs well on the AS dataset when using BERT as the encoder, suggesting that Softmax may not really outperform CRF. We consider that the current parameters are more suitable for Softmax. More detailed analysis is available from Section 5.

Dataset	**WEIBO**			**RESUME**			**MSRA**		
	train	test	dev	train	test	dev	train	test	dev
Sentences	1.4k	0.27k	0.27k	3.8k	0.48k	0.46k	46.4k	4.4k	-
Chars	73.8k	14.8k	14.5k	124.1k	15.1k	13.9k	2169.9k	172.6k	-
Entities	1.89k	0.42k	0.39k	1.34k	0.15k	0.16k	74.8k	6.2k	-

Table 12: Corpus details of three NER datasets

Model	**WEIBO**			**RESUME**			**MSRA**			**AVG**		
	P	R	F	P	R	F	P	R	F	P	R	F
BERT(base)	68.01	66.27	67.15	94.58	95.34	94.96	95.66	94.03	94.84	86.08	85.21	85.65
$+D_{best}$	68.83	66.03	67.4	94.34	96.07	95.2	94.84	**94.87**	94.86	86.0	85.66	85.82
$+D_{best} + w_{wei}^1$	70.12	69.62	69.87	95.21	**96.32**	**95.76**	95.09	94.27	94.68	86.81	86.74	86.77
$+D_{best} + w_{wei}^2$	70.1	66.75	68.38	**95.52**	95.46	95.49	95.39	94.74	95.06	87.0	85.65	86.31
$+D_{best} + w_{wei}^3$	69.93	70.1	70	95.32	96.2	**95.76**	95.48	94.73	95.1	86.91	**87.01**	86.95
$+D_{best} + w_{wei}^4$	**71.08**	**70.57**	**70.83**	94.8	95.15	94.98	**95.84**	94.64	**95.24**	**87.24**	86.79	**87.02**

Table 13: NER tasks. Take BERT as the base model.

B NER Appendix

Corpus details of MSRA (Levow, 2006), WEIBO (Peng and Dredze, 2015), and RESUME (Zhang and Yang, 2018) are summarized in Table 12. We have no access to OntoNote 4, so didn't test it. All experiments adopted the same hyperparameters, as shown in Table 4. We did not list the latest performance of existing NER tasks, as we only explored whether WeiDC works for NER tasks and which weight mechanism is more robust.

As shown in Table 13, w_{wei}^4 performs the best on the WEIBO and MSRA datasets, while the worst on the RESUME dataset, indicating that it is difficult, if not impossible, to design a general weight mechanism. The overall performance of w_{wei}^4 is still higher than other weight mechanisms. How to more naturally integrate weight mechanisms and knowledge distillation into NER tasks requires more exploration and research.

In addition to such NER tasks, non-sequence annotation tasks, such as text classification, usually have only one label per sentence, which may limit the application of WeiDC.