Improving Chinese Word Segmentation with Wordhood Memory Networks

Yuanhe Tian[♡*], **Yan Song**[♠†], **Fei Xia**[♡], **Tong Zhang**[♢], **Yonggang Wang**[♠]
[♡]University of Washington, [♠]Sinovation Ventures
[♢]The Hong Kong University of Science and Technology
[♡]{yhtian, fxia}@uw.edu [♠]clksong@gmail.com
[♢]tongzhang@ust.hk [♠]wangyonggang@chuangxin.com

Abstract

Contextual features always play an important role in Chinese word segmentation (CWS). Wordhood information, being one of the contextual features, is proved to be useful in many conventional character-based segmenters. However, this feature receives less attention in recent neural models and it is also challenging to design a framework that can properly integrate wordhood information from different wordhood measures to existing neural frameworks. In this paper, we therefore propose a neural framework, WMSEG, which uses memory networks to incorporate wordhood information with several popular encoder-decoder combinations for CWS. Experimental results on five benchmark datasets indicate the memory mechanism successfully models wordhood information for neural segmenters and helps WMSEG achieve state-of-the-art performance on all those datasets. Further experiments and analyses also demonstrate the robustness of our proposed framework with respect to different wordhood measures and the efficiency of wordhood information in cross-domain experiments.[1]

1 Introduction

Unlike most written languages in the world, the Chinese writing system does not use explicit delimiters (e.g., white space) to separate words in written text. Therefore, Chinese word segmentation (CWS) conventionally serves as the first step in Chinese language processing, especially for many downstream tasks such as text classification (Zeng et al., 2018), question answering (Liu et al., 2018), machine translation (Yang et al., 2018), etc.

In the past two decades, the mainstream methodology of CWS treated CWS as a character-based sequence labeling task (Tseng et al., 2005; Song et al., 2006; Sun and Xu, 2011; Pei et al., 2014; Chen et al., 2015; Zhang et al., 2016; Chen et al., 2017; Ma et al., 2018; Higashiyama et al., 2019; Qiu et al., 2019), where various studies were proposed to effectively extract contextual features to help better predicting segmentation labels for each character (Zhang et al., 2013; Zhou et al., 2017; Higashiyama et al., 2019). Among all the contextual features, the ones measuring wordhood for n-grams illustrate their helpfulness in many non-neural CWS models (Sun et al., 1998; Xue and Shen, 2003; Feng et al., 2004; Song and Xia, 2012).

Later, following the track of the sequence labeling methodology, recent approaches with neural networks are proved to be powerful in this task (Chen et al., 2015; Ma et al., 2018; Higashiyama et al., 2019). However, since neural networks (e.g., LSTM) is considered to be able to provide a good modeling of contextual dependencies, less attention is paid to the idea of explicitly leveraging wordhood information of n-grams in the context as what had previously been done in non-neural models. Although some studies sidestepped the idea by incorporating contextual n-grams (Pei et al., 2014; Zhou et al., 2017) or word attention (Higashiyama et al., 2019) into the sequence labeling process, they are limited in either concatenating word and character embeddings or requiring a well-defined word lexicon. Therefore, it has not been fully explored what would be the best way of representing contextual information such as wordhood features in neural CWS models. Moreover, consider there are various choices of wordhood measures, it is also a challenge to design a framework that can incorporate different wordhood features so that the entire CWS approach can be general while being effective in accommodating the input from any measures.

In this paper, we propose WMSEG, a neural framework with a memory mechanism, to improve

[*]Partially done as an intern at Sinovation Ventures.
[†]Corresponding author.
[1]WMSEG (code and the best performing models) is released at https://github.com/SVAIGBA/WMSeg.

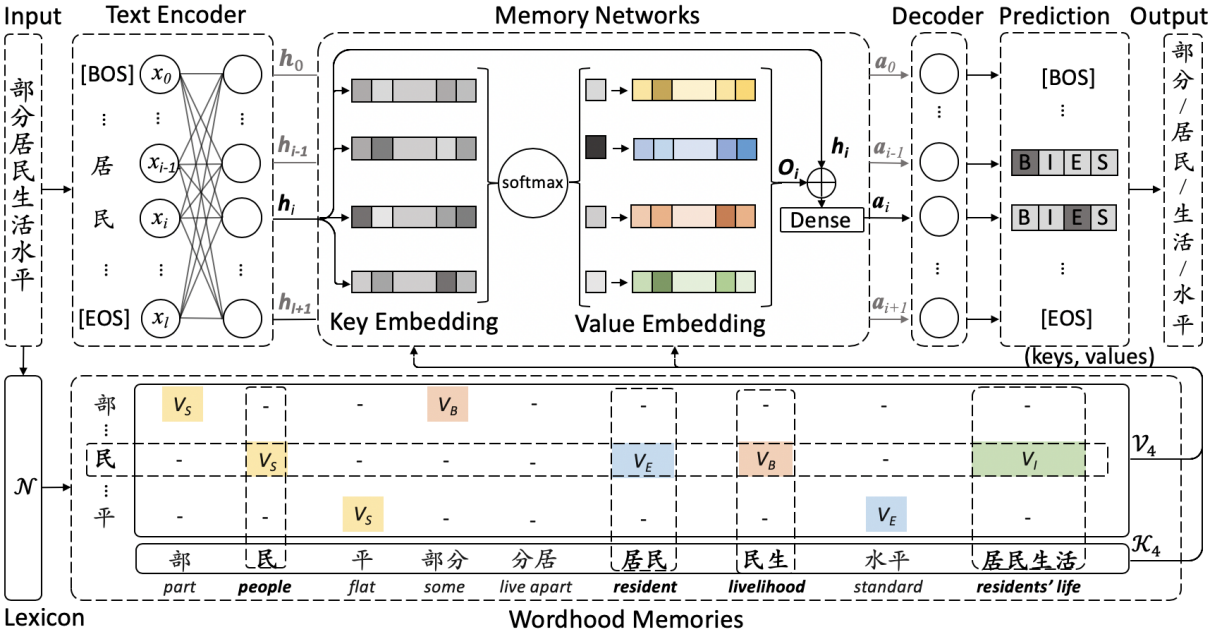

Figure 1: The architecture of WMSEG. "$\mathcal{N}$" denotes a lexicon constructed by wordhood measures. N-grams (keys) appearing in the input sentence "部分居民生活水平" (*some residents' living standard*) and the wordhood information (values) of those n-grams are extracted from the lexicon. Then, together with the output from the text encoder, n-grams (keys) and their wordhood information (values) are fed into the memory module, whose output passes through a decoder to get final predictions of segmentation labels for every character in the input sentence.

CWS by leveraging wordhood information. In detail, we utilize key-value memory networks (Miller et al., 2016) to incorporate character n-grams with their wordhood measurements in a general sequence labeling paradigm, where the memory module can be incorporated with different prevailing encoders (e.g., BiLSTM and BERT) and decoders (e.g., softmax and CRF). For the memory, we map n-grams and their wordhood information to keys and values in it, respectively, and one can use different wordhood measures to generate such information. Then for each input character, the memory module addresses all the n-grams in the key list that contain the character and uses their corresponding values to generate an output vector to enhance the decoder for assigning a segmentation label to the character. Experimental results from five widely used benchmark datasets confirm that WMSEG with wordhood information can improve CWS over powerful baseline segmenters and ourperform previous studies, where state-of-the-art performance is observed on all the datasets. Further experiments and analyses are also performed to investigate different factors affecting WMSEG's performance.

2 The Proposed Framework

Following previous studies, we regard CWS as a character-based sequence labeling task. The architecture of WMSEG is illustrated in Figure 1, where the general sequence labeling paradigm is the top part with a memory module inserted between the encoder and the decoder. The model predicts a tag (e.g., tag *B* for the 1st character in a word) for each character, and the predicted tag sequence is then converted to word boundary in the system output. The bottom part of the figure starts with a lexicon $\mathcal{N}$, which is simply a list of n-grams and can be built by various methods (see Section 2.1). Given an input sentence $\mathcal{X} = x_1 x_2 ... x_i ... x_l$, for each character x_i in $\mathcal{X}$, our approach uses the lexicon $\mathcal{N}$ to generate (keys, values) for x_i and send it to the memory module. In all, the process of WMSEG to perform CWS can be formalized as

$$\widehat{\mathcal{Y}} = \arg\max_{\mathcal{Y} \in \mathcal{T}^l} p(\mathcal{Y} | \mathcal{X}, \mathcal{M}(\mathcal{X}, \mathcal{N})) \quad (1)$$

where $\mathcal{T}$ denotes the set of all types of segmentation labels, and l stands for the length of the input sentence $\mathcal{X}$. The output $\mathcal{Y}$ is the corresponding label sequence for $\mathcal{X}$ with $\widehat{\mathcal{Y}}$ representing the best label sequence according to the model. $\mathcal{M}$ is the memory module proposed in this paper that consumes $\mathcal{X}$ and $\mathcal{N}$ and provides corresponding wordhood information for $\mathcal{X}$ to maximize p.

In the rest of this section, we describe the construction of the n-gram lexicon, the proposed wordhood memory networks, and how it is integrated with different encoders and decoders, respectively.

2.1 Lexicon Construction

To build the wordhood memory networks, the first step is to construct the lexicon $\mathcal{N}$ because the keys in the memory module are built upon $\mathcal{N}$, where each n-gram in $\mathcal{N}$ is stored as a key in it.[2] In this study, $\mathcal{N}$ is simply a list of n-grams, and technically, it can be constructed through many existing resources or automatic methods. Compared to using an off-the-shelf lexicon or the word dictionary from the training data, it is hypothesized that, for the purpose of incorporating wordhood information into the general sequence labeling framework, unsupervised wordhood measures, such as accessor variety (AV) (Feng et al., 2004), pointwise mutual information (PMI) (Sun et al., 1998), and description length gain (DLG) (Kit and Wilks, 1999), would perform better. For example, AV measures the wordhood of an n-gram k by

$$AV(k) = min(L_{av}(k), R_{av}(k)) \quad (2)$$

where $L_{av}(k)$ and $R_{av}(k)$ denote the number of different character types that can precede (left access number) or follow (right access number) the n-gram k. Normally, the higher the AV score is, the more likely the n-gram forms a word.

2.2 Wordhood Memory Networks

To encode both n-grams and the wordhood information they carry, one requires an appropriate framework to do so for CWS. Compared with other network structures that can exploit n-grams such as the attention mechanism, key-value memory networks are more appropriate to model such pairwise knowledge via transforms between keys and values. In the memory, we map n-grams and their wordhood information to keys and values, respectively. Following Miller et al. (2016), we illustrate how our memory module generates and operates the (keys, values) pair for each x_i in this subsection.

N-gram Addressing For each x_i in a training/test instance, normally there are many n-grams in $\mathcal{N}$ that contain x_i. Therefore, the n-gram addressing step is to generate all n-grams from x_i's context (including x_i) and keep only the ones that appear in $\mathcal{N}$, resulting $\mathcal{K}_i = [k_{i,1}, k_{i,2} \cdots, k_{i,j}, \cdots k_{i,m_i}]$ that x_i is a part of $k_{i,j}$. For example, in the input sentence shown in Figure 1, the n-grams that contain the character x_4 =“民” (*people*) form the list $\mathcal{K}_4$ = [“民” (*people*), “居民” (*resident*), “民生” (*livelihood*), “居民生活” (*residents' life*)], which are highlighted in the dashed boxes illustrated at the bottom part of the figure. Then, the memory module activates the corresponding keys in it, addresses their embeddings (which are denoted as $\mathbf{e}^k_{i,j}$ for each $k_{i,j}$), and computes the probability distribution for them with

$$p_{i,j} = \frac{exp(\mathbf{h}_i \cdot \mathbf{e}^k_{i,j})}{\sum_{j=1}^{m_i} exp(\mathbf{h}_i \cdot \mathbf{e}^k_{i,j})} \quad (3)$$

for each key, where $\mathbf{h}_i$ is the vector for x_i which can be generated from any text encoder.

Rule	$v_{i,j}$
x_i is the beginning of the key $k_{i,j}$	V_B
x_i is inside the key $k_{i,j}$	V_I
x_i is the ending of the key $k_{i,j}$	V_E
x_i is the single-character key $k_{i,j}$	V_S

Table 1: The rules for assigning different values to x_i according to its position in a key $k_{i,j}$.

Wordhood Reading Values in the memory represent the wordhood information for a given x_i and $k_{i,j}$ pair, which is not a straightforward mapping because x_i may have different roles in each $k_{i,j}$. For example, $k_{i,j}$ delivers different wordhood information when x_i appears at the beginning or the ending of $k_{i,j}$. Therefore, we set rules in Table 1 to read a value for a key according to different situations of x_i in $k_{i,j}$, where we use a set of values $\{V_B, V_I, V_E, V_S\}$ with embeddings $\{\mathbf{e}_{V_B}, \mathbf{e}_{V_I}, \mathbf{e}_{V_E}, \mathbf{e}_{V_S}\}$ (illustrated in different colors in Figure 1) so that all n-grams should map to one of the values based on x_i's position in $k_{i,j}$. To illustrate that, in the aforementioned example, n-grams in $\mathcal{K}_4$ for x_4 =“民” (*people*) are mapped to a value list $\mathcal{V}_4 = [V_S, V_E, V_B, V_I]$ (see Figure 1). As a result, each K_i for x_i has a list of values denoted by $\mathcal{V}_i = [v_{i,1}, v_{i,2} \cdots, v_{i,j}. \cdots v_{i,m_i}]$. Then the total wordhood memory for x_i is computed from the weighted sum of all keys and values by

$$\mathbf{o}_i = \sum_{j=1}^{m_i} p_{i,j} \mathbf{e}^v_{i,j} \quad (4)$$

where $\mathbf{e}^v_{i,j}$ is the embedding for $v_{i,j}$. Afterwards, $\mathbf{o}_i$ is summed element-wise with $\mathbf{h}_i$ and the result is passed through a fully connected layer by

$$\mathbf{a}_i = \mathbf{W_o} \cdot (\mathbf{h_i} + \mathbf{o_i}) \quad (5)$$

[2] Therefore n-gram and key are equivalent in the memory.

	MSR		PKU		AS		CITYU		CTB6		
	TRAIN	TEST	TRAIN	TEST	TRAIN	TEST	TRAIN	TEST	TRAIN	DEV	TEST
CHAR #	4,050K	184K	1,826K	173K	8,368K	198K	2,403K	68K	1,056K	100K	134K
WORD #	2,368K	107K	1,110K	104K	5,500K	123K	1,456K	41K	641K	60K	82K
CHAR TYPE #	5K	3K	5K	3K	6K	4K	5K	3K	4K	3K	3K
WORD TYPE #	88K	13K	55K	13K	141K	19K	69K	9K	42K	10K	12K
OOV RATE	-	2.7	-	5.8	-	4.3	-	7.2	-	5.4	5.6

Table 2: Statistics of the five benchmark datasets, in terms of the number of character and word tokens and types in each training and test set. Out-of-vocabulary (OOV) rate is the percentage of unseen word tokens in the test set.

where $\mathbf{W_o}$ is a trainable parameter and the output $\mathbf{a}_i \in \mathbb{R}^{|\mathcal{T}|}$ is a weight vector with its each dimension corresponding to a segmentation label.

2.3 Text Encoders and Decoders

To ensure wordhood memory networks functionalize, one requires to generate $\mathbf{h_i}$ for each x_i by

$$[\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_i, ..., \mathbf{h}_l] = Encoder(\mathcal{X}) \quad (6)$$

where the $Encoder$ can be different models, e.g., Bi-LSTM and BERT (Devlin et al., 2019), to represent a sequence of Chinese characters into vectors.

Once all $\mathbf{a_i}$ are generated from the memory for each x_i, a decoder takes them to predict a sequence of segmentation labels $\widehat{\mathcal{Y}} = \widehat{y}_1\widehat{y}_2 \cdots \widehat{y}_l$ for $\mathcal{X}$ by

$$\widehat{\mathcal{Y}} = Decoder(\mathcal{A}) \quad (7)$$

where $\mathcal{A} = \mathbf{a}_1\mathbf{a}_2 \cdots \mathbf{a}_i \cdots \mathbf{a}_l$ is the sequence of output from Eq. 5. The $Decoder$ can be implemented by different algorithms, such as softmax:

$$\widehat{y}_i = \arg\max \frac{exp(a_i^t)}{\sum_{t=1}^{|\mathcal{T}|} exp(a_i^t)} \quad (8)$$

where a_i^t is the value at dimension t in $\mathbf{a_i}$. Or one can use CRF for the $Decoder$:

$$\widehat{y}_i = \arg\max_{y_i \in \mathcal{T}} \frac{exp(\mathbf{W}_c \cdot \mathbf{a}_i + \mathbf{b}_c)}{\sum_{y_{i-1}y_i} exp(\mathbf{W}_c \cdot \mathbf{a}_i) + \mathbf{b}_c} \quad (9)$$

where $\mathbf{W}_c \in \mathbb{R}^{|\mathcal{T}|\times|\mathcal{T}|}$ and $\mathbf{b}_c \in \mathbb{R}^{|\mathcal{T}|}$ are trainable parameters to model the transition for y_{i-1} to y_i.

3 Experimental Settings

3.1 Datasets

We employ five benchmark datasets in our experiments: four of them, namely, MSR, PKU, AS, and CITYU, are from SIGHAN 2005 Bakeoff (Emerson, 2005) and the fifth one is CTB6 (Xue et al., 2005). AS and CITYU are in traditional Chinese characters whereas the other three use simplified ones. Following previous studies (Chen et al., 2015, 2017; Qiu et al., 2019), we convert traditional Chinese characters in AS and CITYU into simplified ones.[3] For MSR, AS, PKU, and CITYU, we follow their official training/test data split. For CTB6, we use the same split as that stated in Yang and Xue (2012); Chen et al. (2015); Higashiyama et al. (2019), and only use its test set for the final experiment. Table 2 show the statistics of all datasets in terms of the number of characters and words and the percentage of out-of-vocabulary (OOV) words in the dev/test sets with respect to the training set.

	BC	BN	MZ	NW	WEB
CHAR #	275K	483K	403K	443K	342K
WORD #	184K	287K	258K	260K	210K
CHAR TYPE #	3K	3K	4K	3K	4K
WORD TYPE #	12K	23K	26K	21K	21K
OOV RATE	3.4	6.0	8.9	5.9	7.1

Table 3: Statistics of CTB7 with respect to five different genres. The OOV rate for each genre is computed based on the vocabulary from all the other four genres.

In addition, we also use CTB7 (LDC2010T07) to perform our cross-domain experiments. There are five genres in CTB7, including broadcast conversation (BC), broadcast news (BN), magazine (MZ), newswire (NW), and weblog (WEB). The statistics of all the genres are reported in Table 3, where the OOV rate for each genre is computed according to the union of all other genres. For example, the OOV rate for BC is computed with respect to the union of BN, MZ, NW, and WEB.

3.2 Wordhood Measures

We experiment with three wordhood measures to construct $\mathcal{N}$. The main experiment adopts the aforementioned AV as the measure to rank all n-grams, because AV was shown to be the most effective wordhood measure in previous CWS studies (Zhao and Kit, 2008). Since AV is sensitive to

[3]The conversion scripts are from https://github.com/skydark/nstools/tree/master/zhtools

	MSR	PKU	AS	CITYU	CTB6
AV	49K	71K	105K	104K	50K
PMI	18K	16K	22K	21K	16K
DLG	32K	22K	32K	27K	16K

Table 4: The size of lexicon $\mathcal{N}$ generated from different wordhood measures under our settings.

corpus size, in our experiments we use different AV thresholds when building the lexicon for each dataset: the threshold is 2 for PKU, CITYU, CTB6 and CTB7, and 5 for MSR and AS.

To test the the robustness of WMSEG, we also try two other wordhood measures, i.e., PMI (Sun et al., 1998) and DLG (Kit and Wilks, 1999). PMI measures pointwise mutual information between two Chinese characters, x' and x'', via

$$PMI(x', x'') = log\frac{p(x'x'')}{p(x')p(x'')} \quad (10)$$

where p computes the probability of an n-gram (i.e., x', x'' and $x'x''$) in a dataset. A high PMI score indicates that the two characters co-occur a lot in the dataset and are likely to form a word. Hence, we use a threshold to determine whether a word boundary delimiter should be inserted between two adjacent characters in the dataset. In our experiments, we set the threshold to 0, PMI score lower than it will result in a segmentation. In other words, for each dataset, we use PMI to perform unsupervised segmentation and collect the segmented words from it to build the n-gram lexicon $\mathcal{N}$.

The other measure, DLG, computes wordhood of an n-gram s according to the change of the description length of a dataset D with and without treating that n-gram as a segment:

$$DLG(s) = DL(\mathcal{D}) - DL(\mathcal{D}[r \rightarrow s] \oplus s) \quad (11)$$

where $\mathcal{D}$ denotes the original dataset and $\mathcal{D}[r \rightarrow s] \oplus s$ represents a new dataset by treating s as a new segment, replacing all the occurrences of s with a new symbol r (which can be seen as an index for newly identified segment s), and then appending s at the end. $DL(\mathcal{D})$ is the Shannon-Fano code length of a dataset $\mathcal{D}$, calculated by

$$DL(\mathcal{D}) = -\sum_{x \in \mathcal{V}} c(x) log\frac{c(x)}{|\mathcal{D}|} \quad (12)$$

where $\mathcal{V}$ refers to the vocabulary of $\mathcal{D}$ and $c(x)$ the count of segment x. We set the threshold for DLG to 0 and use the n-grams whose DLG is higher than it to build lexicon $\mathcal{N}$ for each dataset.

	Bi-LSTM	BERT / ZEN
Word Embedding Size	200	-
Hidden State Size	100	768
Hidden State Layers	1	12
Key Embedding Size	200	768
Value Embedding Size	200	768
Dropout Rate	0.2	0.1

Table 5: The hyper-parameters for our models w.r.t. different encoders, i.e., Bi-LSTM, BERT and ZEN.

All aforementioned measures are conducted on the union of the training and test sets, so that n-grams and their wordhood information are shared in both the learning and prediction phase. We remove all white spaces from the data and use the resulted raw texts to perform these measures. Table 4 shows the sizes of the lexicons created with these wordhood measures on the five datasets.

3.3 Model Implementation

Following previous studies (Sun and Xu, 2011; Chen et al., 2015, 2017; Ma et al., 2018; Qiu et al., 2019), we use four segmentation labels in our experiments, i.e., $\mathcal{T} = \{B, I, E, S\}$. Among them, B, I, and E indicate a character is the beginning, inside, and the ending of a word and S denotes that the character is a single-character word.

Since text representation plays an important role to facilitate many tasks (Conneau et al., 2017; Song et al., 2017, 2018; Sileo et al., 2019), we try two effective and well-known encoders, i.e., Bi-LSTM and BERT[4]. In addition, we test WMSEG on a pre-trained encoder for Chinese language, i.e., ZEN[5] (Diao et al., 2019), which learns n-gram information in its pre-training from large raw corpora and outperforms BERT on many Chinese NLP tasks. Table 5 shows the hyperparameter settings for all the encoders: for the Bi-LSTM encoder, we follow the setting of Chen et al. (2015) and adopt their character embeddings for $\mathbf{e}_i^x$, and for BERT and ZEN encoders, we follow the default settings in their papers (Devlin et al., 2019; Diao et al., 2019).

For the decoders, we use softmax and CRF, and set their loss functions as cross-entropy and negative log-likelihood, respectively. The memory module can be initialized by random or pre-trained word embeddings for keys and values. In our experiments, we use random initialization for them.[6]

[4]We use the Chinese base model from https://s3.amazonaws.com/models.huggingface.co/.

[5]https://github.com/sinovation/ZEN.

[6]We tried different initialization methods, and they did not show a significant difference in CWS performance.

CONFIG		MSR		PKU		AS		CITYU		CTB6	
EN-DN	WM	F	R_{OOV}	F	R_{OOV}	F	R_{OOV}	F	R_{OOV}	F	R_{OOV}
BL-SM	×	95.53	62.96	91.85	48.84	94.52	62.21	93.79	67.26	93.56	67.39
	√	**95.61**	**63.94**	**91.97**	**49.00**	**94.70**	**64.18**	**93.88**	**69.20**	**93.70**	**68.52**
BL-CRF	×	95.80	66.17	92.35	52.04	94.39	61.59	93.96	67.84	93.84	70.81
	√	**95.98**	**68.75**	**92.43**	**56.80**	**95.07**	**68.17**	**94.20**	**69.91**	**94.03**	**71.88**
BT-SM	×	97.84	86.32	96.20	84.43	96.33	77.86	97.51	**86.69**	96.90	**88.46**
	√	**98.16**	**86.50**	**96.47**	**86.34**	**96.52**	**78.67**	**97.77**	86.62	**97.13**	88.30
BT-CRF	×	97.98	85.52	96.32	85.04	96.34	77.75	97.63	86.66	96.98	87.43
	√	**98.28**	**86.67**	**96.51**	**86.76**	**96.58**	**78.48**	**97.80**	**87.57**	**97.16**	**88.00**
ZEN-SM	×	98.35	**85.78**	96.27	84.50	96.38	77.62	97.78	90.69	97.08	86.20
	√	**98.36**	85.30	**96.49**	**84.95**	**96.55**	**78.02**	**97.86**	**90.89**	**97.22**	**86.83**
ZEN-CRF	×	98.36	**86.82**	96.36	84.81	96.39	77.81	97.81	**91.78**	97.13	87.08
	√	**98.40**	84.87	**96.53**	**85.36**	**96.62**	**79.64**	**97.93**	90.15	**97.25**	**88.46**

Table 6: Experimental results of WMSEG on SIGHAN2005 and CTB6 datasets with different configurations. "EN-DN" stands for the text encoders ("BL" for Bi-LSTM and "BT" for BERT) and decoders ("SM" for softmax and "CRF" for CRF). The "WM" column indicates whether the wordhood memories are used (√) or not (×).

4 Results and Analyses

In this section, we firstly report the results of WMSEG with different configurations on five benchmark datasets and its comparison with existing models. Then we explore the effect of using different lexicon $\mathcal{N}$ and different wordhood measures in WMSEG. We also use a cross-domain experiment to illustrate the effectiveness of WMSEG when more OOVs are in the test set. Lastly, a case study is performed to visualize how the wordhood information used in WMSEG helps CWS.

4.1 Results on Benchmark Datasets

In the main experiment, we illustrate the validity of the proposed memory module by comparing WMSEG in different configurations, i.e., with and without the memory in integrating with three encoders, i.e., Bi-LSTM, BERT, and ZEN, and two decoders, i.e., softmax and CRF. The experimental results on the aforementioned five benchmark datasets are shown in Table 6, where the overall F-score and the recall of OOV are reported. With five datasets and six encoder-decoder configurations, the table includes results from 30 pairs of experiments, each pair with or without using the memories.

There are several observations drawn from the results. First, the overall comparison clearly indicates that, WMSEG (i.e., the model with wordhood memories) outperforms the baseline (i.e., the model without wordhood memories) for all 30 pairs in terms of F-scores and for 25 pairs in terms of R_{OOV}. Second, the proposed memory module works smoothly with different encoders and decoders, where some improvement is pretty significant; for instance, when using Bi-LSTM as the encoder and CRF as the decoder, WMSEG improves the F-score on the AS dataset from 94.39 to 95.07 and R_{OOV} from 61.59 to 68.17. With BERT or ZEN as the encoder, even when the baseline system performs very well, the improvement of WMSEG on F-scores is still decent. Third, among the models with ZEN, the ones with the memory module further improve their baselines, although the context information carried by n-grams is already learned in pre-training ZEN. This indicates that wordhood information provides additional cues (besides the contextual features) that can benefit CWS, and our proposed memory module is able to provide further task-specific guidance to an n-gram integrated encoder. Fourth, the wordhood memory shows its robustness with different lexicon size when we consider WMSEG's performance with the lexicon statistics reported in Table 4 together. To summarize, the results in this experiment not only confirm that wordhood information is a simple yet effective source of knowledge to help CWS without requiring external support such as a well-defined dictionary or manually crafted heuristics, but also fully illustrate that the design of our model can effectively integrate this type of knowledge.

To further illustrate the validity and the effectiveness of WMSEG, we compare our best-performing model with the ones in previous studies on the same benchmark datasets. The comparison is presented in Table 7, where WMSEG (both the one with BERT and ZEN) outperforms all existing models with respect to the F-scores and achieves new state-of-the-art performance on all datasets.

	MSR		PKU		AS		CITYU		CTB6	
	F	R_{OOV}	F	R_{OOV}	F	R_{OOV}	F	R_{OOV}	F	R_{OOV}
ZHANG ET AL. (2013)	97.5	-	96.1	73.1	-	-	-	-	-	-
PEI ET AL. (2014)	97.2	-	95.2	-	-	-	-	-	-	-
MA AND HINRICHS (2015)	96.6	**87.2**	95.1	76.0	-	-	-	-	-	-
CHEN ET AL. (2015)	97.4	-	96.5	-	-	-	-	-	96.0	-
XU AND SUN (2016)	96.3	-	96.1	-	-	-	-	-	95.8	-
ZHANG ET AL. (2016)	97.7	-	95.7	-	-	-	-	-	95.95	-
CHEN ET AL. (2017)	96.04	71.60	94.32	72.64	94.75	75.34	95.55	81.40	-	-
WANG AND XU (2017)	98.0	-	96.5	-	-	-	-	-	-	-
ZHOU ET AL. (2017)	97.8	-	96.0	-	-	-	-	-	96.2	-
MA ET AL. (2018)	98.1	80.0	96.1	78.8	96.2	70.7	97.2	87.5	96.7	85.4
GONG ET AL. (2019)	97.78	64.20	96.15	69.88	95.22	77.33	96.22	73.58	-	-
HIGASHIYAMA ET AL. (2019)	97.8	-	-	-	-	-	-	-	96.4	-
QIU ET AL. (2019)	98.05	78.92	96.41	78.91	96.44	76.39	96.91	86.91	-	-
WMSEG (BERT-CRF)	98.28	86.67	96.51	**86.76**	96.58	78.48	97.80	87.57	97.16	88.00
WMSEG (ZEN-CRF)	**98.40**	84.87	**96.53**	85.36	**96.62**	**79.64**	**97.93**	**90.15**	**97.25**	**88.46**

Table 7: Performance (F-score) comparison between WMSEG (BT-CRF and ZEN-CRF with woodhood memory networks) and previous state-of-the-art models on the test set of five benchmark datasets.

4.2 Cross-Domain Performance

As domain variance is always an important factor affecting the performance of NLP systems especially word semgenters (Song et al., 2012; Song and Xia, 2013), in addition to the experiments on benchmark datasets, we also run WMSEG on CTB7 across domains (genres in this case) with and without the memory module. To test on each genre, we use the union of the data from the other four genres to train our segmenter and use AV to extract n-grams from the entire raw text from CTB7 in this experiment. Table 8 reports the results in F-score and OOV recall, which show a similar trend as that in Table 6, where WMSEG outperforms baselines for all five genres. Particularly, for genres with large domain variance (e.g., the ones with high OOV rates such as MZ and WEB), CWS is difficult, and its relatively low F-scores in Table 8 from baseline models confirm that. Yet WMSEG offers a decent way to improve cross-domain CWS performance without any help from external knowledge or complicated model design, which further illustrates the effectiveness of the memory module. The reason could be that many n-grams are shared in both training and test data; these n-grams with their wordhood information present a strong indication to the model on what combinations of characters can be treated as words, even though some of them never appear in the training data.

4.3 Effect of Using Different $\mathcal{N}$

To analyze the robustness of WMSEG with respect to the lexicon, we compare four ways (ID: 2-5 in Table 9) of constructing the lexicon ($\mathcal{N}$): the first one simply uses the vocabulary from the training data (marked as GOLD LABEL in Table 9; ID: 2); the other three ways use AV to extract n-grams from the unsegmented training data only (ID: 3), the test data only (ID: 4), and training + test set (ID: 5), respectively.[7] Table 9 shows the results of running BERT-CRF on the WEB genre of CTB7 without the wordhood memories (ID: 1) and with the memories (ID: 2-5), following the cross-domain setting in §4.2. While the four methods with memories achieve similar results on the F score, indicating the robustness of our proposed framework, the one that builds $\mathcal{N}$ using the raw texts from both training and test sets through unsupervised method (ID: 5) achieves the biggest improvement on R_{OOV}, demonstrating the advantage of including the unlabeled test set by incorporating the results from unsupervised wordhood measures into the models.

4.4 Effect of Different Wordhood Measures

WMSEG provides a general way of integrating wordhood information for CWS, we expect other wordhood measures to play the same role in it. Therefore, we test PMI and DLG in our model and compare them with the previous results from AV (see Table 6). Specifically, we use our best performing BERT-based model, i.e., BERT-CRF, with the n-gram lexicons constructed by the aforementioned three measures and run it on all benchmark datasets. We draw the histograms of the F-scores obtained from WMSEG with each measure (red, green, and blue bars for AV, PMI, and DLG, re-

[7] One could also use an external corpus to build N, which is not considered in this experiment.

Config		BC		BN		MZ		NW		Web	
En-Dn	WM	F	R_{OOV}	F	R_{OOV}	F	R_{OOV}	F	R_{OOV}	F	R_{OOV}
BL-SM	×	93.73	63.39	93.65	68.88	90.55	**66.95**	93.70	69.57	90.81	55.50
	√	**94.04**	**63.53**	**93.91**	**72.32**	**90.76**	65.65	**93.83**	**72.40**	**91.22**	**56.62**
BL-CRF	×	93.95	65.60	93.87	71.89	90.67	67.13	93.87	72.17	91.12	57.51
	√	**94.21**	**66.81**	**94.11**	**74.22**	**90.95**	**67.29**	**93.96**	**74.38**	**91.49**	**58.37**
BT-SM	×	96.27	80.76	96.88	87.90	94.97	84.45	97.08	89.78	94.82	74.00
	√	**96.41**	**81.15**	**97.00**	**89.47**	**95.10**	**85.48**	**97.24**	**91.96**	**95.00**	**75.51**
BT-CRF	×	96.25	79.04	96.87	89.15	94.94	85.27	96.99	91.34	94.79	75.58
	√	**96.43**	**81.29**	**97.09**	**90.29**	**95.11**	**85.32**	**97.21**	**92.48**	**95.03**	**76.30**
ZEN-SM	×	96.39	79.97	96.95	88.93	95.05	85.14	97.17	91.33	94.03	75.33
	√	**96.45**	**81.34**	**97.03**	**89.78**	**95.06**	**85.60**	**97.21**	**91.73**	**95.08**	**75.60**
ZEN-CRF	×	96.30	80.05	96.97	**90.38**	94.93	85.64	97.10	91.03	94.90	74.98
	√	**96.50**	**80.44**	**97.11**	90.29	**95.13**	**85.96**	**97.24**	**91.68**	**95.04**	**75.74**

Table 8: Experimental results on five genres of CTB7. Abbreviations follow the same notation in Table 6.

ID	Train	Test	Gold Label	F	R_{OOV}
1	-	-	-	94.79	75.58
2	×	×	√	+0.22	+0.21
3	√	×	×	+0.21	+0.20
4	×	√	×	+0.23	+0.33
5	√	√	×	**+0.24**	**+0.72**

Table 9: Comparisons of performance gain on the Web genre of CTB7 with respect to the baseline BERT-CRF model when the n-gram lexicon $\mathcal{N}$ for WMSeg is built upon different sources. √ and × refer to if a corresponding data source is used or not, respectively.

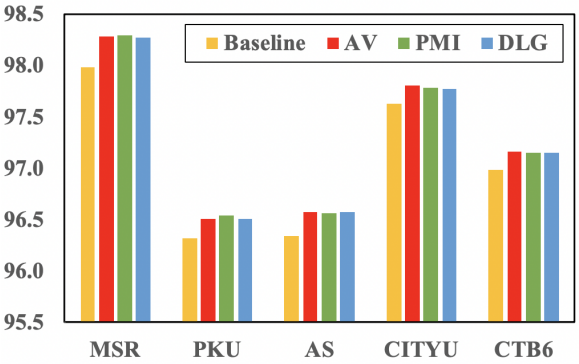

Figure 2: The F-scores of WMSeg (BERT) using three different wordhood measures, namely AV (red), PMI (green), and DLG (blue), on five benchmark datasets.

spectively) in Figure 2, where the F-scores of the baseline model are also presented in orange bars.

As shown in the figure, the performances of using the three measures are very similar, which indicates that WMSeg is able to robustly incorporate the wordhood information from various measures, despite that those measures focus on different aspects of n-grams when determining whether the n-grams should be treated as words. Particularly, consider that the lexicons produced by the three measures are rather different in their sizes (as shown in Table 4), the results in Figure 2 strongly demonstrate the effectiveness of our proposed approach in learning with a limited number of n-grams. This observation also reveals the possibility that many n-grams may be redundant for our model, and WMSeg is thus able to identify the most useful ones from them, which is analyzed in the case study.

4.5 Case Study

To investigate how the memory learns from the wordhood information carried by n-grams, we conduct a case study with an example input sentence "他/从/小/学/电/脑/技/术" (*He learned computer techniques since childhood*). In this sentence, the n-gram "从小学" is ambiguous with two possible interpretations: "从小/学" (*learn since childhood*) and "从/小学" (*from primary school*). Native Chinese speakers can easily choose the first one with the given context but a word segmenter might incorrectly choose the second segmentation.

We feed this case into our BERT-CRF model with the memory module. In Figure 3, we visualize the resulted weights that learned from keys (a) and values (b) of the memory, as well as from the final tagger (c). The heatmaps of all keys and values in the memory with respect to each corresponding input character clearly illustrate that the appropriate n-grams, e.g., "他" (*he*), "学" (*learn*), "从小" (*from childhood*), etc., receive higher weights than others and the corresponding values for them are also emphasized, which further affects final CWS tagging so that the weight distributions from (b) and (c) look alike to each other. Therefore, this visualization explains, to some extent, that the proposed memory mechanism can identify and distinguish important n-grams within a certain context and thus improves CWS performance accordingly.

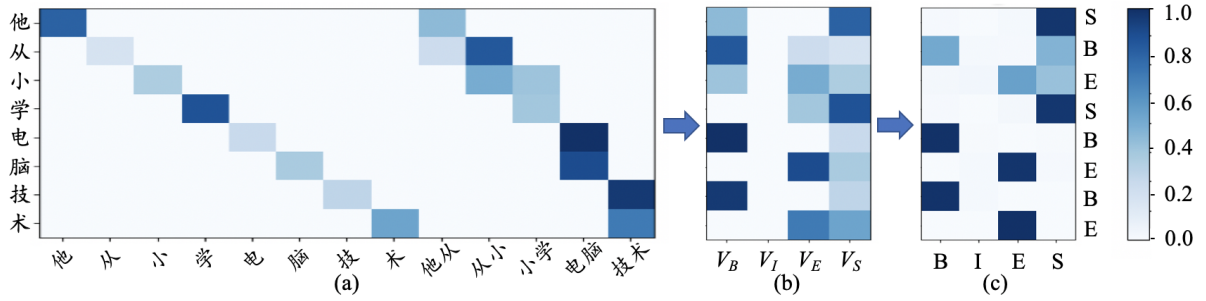

Figure 3: Heatmaps of weights learned for (a) keys and (b) values in the memory, and (c) the tags from the decoder, with respect to each character in an input sentence. Higher weights are visualized with darker colors.

5 Related Work

As one of the most fundamental NLP tasks for Chinese language processing, CWS has been studied for decades, with two steams of methods, i.e., word-based and character-based ones (Xue and Shen, 2003; Peng et al., 2004; Levow, 2006; Zhao et al., 2006; Zhao and Kit, 2008; Li and Sun, 2009; Song et al., 2009a; Li, 2011; Sun and Xu, 2011; Mansur et al., 2013; Zhang et al., 2013; Pei et al., 2014; Chen et al., 2015; Ma and Hinrichs, 2015; Liu et al., 2016; Zhang et al., 2016; Wang and Xu, 2017; Zhou et al., 2017; Chen et al., 2017; Ma et al., 2018; Higashiyama et al., 2019; Gong et al., 2019; Qiu et al., 2019). Among these studies, most of them follow the character-based paradigm to predict segmentation labels for each character in an input sentence; n-grams are used in some of these studies to enhance model performance, which is also observed in many other NLP tasks (Song et al., 2009b; Xiong et al., 2011; Shrestha, 2014; Shi et al., 2016; Diao et al., 2019). Recently, CWS benefits from neural networks and further progress are made with embeddings (Pei et al., 2014; Ma and Hinrichs, 2015; Liu et al., 2016; Zhang et al., 2016; Wang and Xu, 2017; Zhou et al., 2017), recurrent neural models (Chen et al., 2015; Ma et al., 2018; Higashiyama et al., 2019; Gong et al., 2019) and even adversarial learning (Chen et al., 2017). To enhance CWS with neural models, there were studies leverage external information, such as vocabularies from auto-segmented external corpus (Wang and Xu, 2017; Higashiyama et al., 2019), where Higashiyama et al. (2019) introduced a word attention mechanism to learn from large granular texts during the CWS process. In addition, the studies from Chen et al. (2017) and Qiu et al. (2019) try to improve CWS by learning from data annotated through different segmentation criteria. Moreover, there is a study leveraging auto-analyzed syntactic knowledge obtained from off-the-shelf toolkits to help CWS and part-of-speech tagging (Tian et al., 2020). Compare to these studies, WMSEG offers an alternative solution to robustly enhancing neural CWS models without requiring external resources.

6 Conclusion

In this paper, we propose WMSEG, a neural framework for CWS using wordhood memory networks, which maps n-grams and their wordhood information to keys and values in it and appropriately models the values according to the importance of keys in a specific context. The framework follows the sequence labeling paradigm, and the encoders and decoders in it can be implemented by various prevailing models. To the best of our knowledge, this is the first work using key-value memory networks and utilizing wordhood information for neural models in CWS. Experimental results on various widely used benchmark datasets illustrate the effectiveness of WMSEG, where state-of-the-art performance is achieved on all datasets. Further experiments and analyses also demonstrate the robustness of WMSEG in the cross-domain scenario as well as when using different lexicons and wordhood measures.

References

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.

Xinchi Chen, Zhan Shi, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial Multi-Criteria Learning for Chinese Word Segmentation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1193–1203.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark.

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.

Shizhe Diao, Jiaxin Bai, Yan Song, Tong Zhang, and Yonggang Wang. 2019. ZEN: Pre-training Chinese Text Encoder Enhanced by N-gram Representations. *ArXiv*, abs/1911.00720.

Thomas Emerson. 2005. The Second International Chinese Word Segmentation Bakeoff. In *Proceedings of the fourth SIGHAN workshop on Chinese language Processing*, pages 123–133.

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.

Shohei Higashiyama, Masao Utiyama, Eiichiro Sumita, Masao Ideuchi, Yoshiaki Oida, Yohei Sakamoto, and Isaac Okada. 2019. Incorporating Word Attention into Character-Based 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 2699–2709.

Chunyu Kit and Yorick Wilks. 1999. Unsupervised Learning of Word Boundary with Description Length Gain. In *EACL 1999: CoNLL-99 Computational Natural Language Learning*, pages 1–6.

Gina-Anne Levow. 2006. The Third International Chinese Language Processing Bakeoff: Word Segmentation and Named Entity Recognition. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 108–117.

Zhongguo Li. 2011. Parsing the Internal Structure of Words: A New Paradigm for Chinese Word Segmentation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1405–1414, Portland, Oregon, USA.

Zhongguo Li and Maosong Sun. 2009. Punctuation as Implicit Annotations for Chinese Word Segmentation. *Computational Linguistics*, 35(4):505–512.

Yijia Liu, Wanxiang Che, Jiang Guo, Bing Qin, and Ting Liu. 2016. Exploring Segment Representations for Neural Segmentation Models. *arXiv preprint arXiv:1604.05499*.

Ziqing Liu, Enwei Peng, Shixing Yan, Guozheng Li, and Tianyong Hao. 2018. T-Know: a Knowledge Graph-based Question Answering and Infor-mation Retrieval System for Traditional Chinese Medicine. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 15–19, Santa Fe, New Mexico.

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.

Jianqiang Ma and Erhard Hinrichs. 2015. Accurate Linear-Time Chinese Word Segmentation via Embedding Matching. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1733–1743.

Mairgup Mansur, Wenzhe Pei, and Baobao Chang. 2013. Feature-based Neural Language Model and Chinese Word Segmentation. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 1271–1277, Nagoya, Japan.

Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-Value Memory Networks for Directly Reading Documents. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1400–1409.

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.

Fuchun Peng, Fangfang Feng, and Andrew McCallum. 2004. Chinese Segmentation and New Word Detection Using Conditional Random Fields. In *Proceedings of the 20th international conference on Computational Linguistics*, page 562.

Xipeng Qiu, Hengzhi Pei, Hang Yan, and Xuanjing Huang. 2019. Multi-Criteria Chinese Word Segmentation with Transformer. *arXiv preprint arXiv:1906.12035*.

Yangyang Shi, Kaisheng Yao, Le Tian, and Daxin Jiang. 2016. Deep LSTM based Feature Mapping for Query Classification. In *Proceedings of the 2016 Conference of the North American Chapter of*

the Association for Computational Linguistics: Human Language Technologies, pages 1501–1511, San Diego, California.

Prajwol Shrestha. 2014. Incremental N-gram Approach for Language Identification in Code-Switched Text. In *Proceedings of the First Workshop on Computational Approaches to Code Switching*, pages 133–138, Doha, Qatar.

Damien Sileo, Tim Van De Cruys, Camille Pradel, and Philippe Muller. 2019. Mining Discourse Markers for Unsupervised Sentence Representation Learning. 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 3477–3486, Minneapolis, Minnesota.

Yan Song, Dongfeng Cai, Guiping Zhang, and Hai Zhao. 2009a. Approach to Chinese Word Segmentation Based on Character-word Joint Decoding. *Journal of Software*, 20(9):2236–2376.

Yan Song, Jiaqing Guo, and Dongfeng Cai. 2006. Chinese Word Segmentation Based on an Approach of Maximum Entropy Modeling. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 201–204, Sydney, Australia.

Yan Song, Chunyu Kit, and Xiao Chen. 2009b. Transliteration of Name Entity via Improved Statistical Translation on Character Sequences. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration (NEWS 2009)*, pages 57–60, Suntec, Singapore.

Yan Song, Prescott Klassen, Fei Xia, and Chunyu Kit. 2012. Entropy-based Training Data Selection for Domain Adaptation. In *Proceedings of COLING 2012: Posters*, pages 1191–1200, Mumbai, India.

Yan Song, Chia-Jung Lee, and Fei Xia. 2017. Learning Word Representations with Regularization from Prior Knowledge. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 143–152, Vancouver, Canada.

Yan Song, Shuming Shi, Jing Li, and Haisong Zhang. 2018. Directional Skip-Gram: Explicitly Distinguishing Left and Right Context for Word Embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2*, pages 175–180, New Orleans, Louisiana.

Yan Song and Fei Xia. 2012. Using a Goodness Measurement for Domain Adaptation: A Case Study on Chinese Word Segmentation. In *LREC*, pages 3853–3860.

Yan Song and Fei Xia. 2013. A Common Case of Jekyll and Hyde: The Synergistic Effect of Using Divided Source Training Data for Feature Augmentation. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 623–631, Nagoya, Japan.

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, Montreal, Quebec, Canada.

Weiwei Sun and Jia Xu. 2011. Enhancing Chinese Word Segmentation Using Unlabeled Data. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 970–979.

Yuanhe Tian, Yan Song, Xiang Ao, Fei Xia, Xiaojun Quan, Tong Zhang, and Yonggang Wang. 2020. Joint Chinese Word Segmentation and Part-of-speech Tagging via Two-way Attentions of Auto-analyzed Knowledge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8286–8296, Online.

Huihsin Tseng, Pichuan Chang, Galen Andrew, Daniel Jurafsky, and Christopher Manning. 2005. A Conditional Random Field Word Segmenter for Sighan Bakeoff 2005. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*, pages 168–171.

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 (Volume 1: Long Papers)*, pages 163–172, Taipei, Taiwan.

Deyi Xiong, Min Zhang, and Haizhou Li. 2011. Enhancing Language Models in Statistical Machine Translation with Backward N-grams and Mutual Information Triggers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1288–1297, Portland, Oregon, USA.

Jingjing Xu and Xu Sun. 2016. Dependency-based Gated Recursive Neural Network for Chinese Word Segmentation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 567–572, Berlin, Germany.

Naiwen Xue, Fei Xia, Fu-Dong Chiou, and Marta Palmer. 2005. The Penn Chinese Treebank: Phrase structure annotation of a large corpus. *Natural language engineering*, 11(2):207–238.

Nianwen Xue and Libin Shen. 2003. Chinese Word Segmentation as LMR Tagging. In *Proceedings of the second SIGHAN workshop on Chinese language processing-Volume 17*, pages 176–179.

Yaqin Yang and Nianwen Xue. 2012. Chinese Comma Disambiguation for Discourse Analysis. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 786–794.

Zhen Yang, Wei Chen, Feng Wang, and Bo Xu. 2018. Improving Neural Machine Translation with Conditional Sequence Generative Adversarial Nets. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1346–1355, New Orleans, Louisiana.

Jichuan Zeng, Jing Li, Yan Song, Cuiyun Gao, Michael R. Lyu, and Irwin King. 2018. Topic Memory Networks for Short Text Classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3120–3131, Brussels, Belgium.

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.

Meishan Zhang, Yue Zhang, and Guohong Fu. 2016. Transition-Based Neural Word Segmentation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 421–431.

Hai Zhao, Chang-Ning Huang, and Mu Li. 2006. An Improved Chinese Word Segmentation System with Conditional Random Field. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 162–165, Sydney, Australia.

Hai Zhao and Chunyu Kit. 2008. An Empirical Comparison of Goodness Measures for Unsupervised Chinese Word Segmentation with a Unified Framework. In *Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-I*, pages 9–16.

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.