Multiple Character Embeddings for Chinese Word Segmentation

Jingkang Wang[*] **Jianing Zhou**[*] **Jie Zhou** **Gongshen Liu**[†]
The Lab of Information Content Intelligent Analysis, Shanghai, China
School of Cyber Science and Engineering, Shanghai Jiao Tong University
{wangjksjtu,zhjjn1919}@gmail.com, {sanny02,lgshen}@sjtu.edu.cn

Abstract

Chinese word segmentation (CWS) is often regarded as a character-based sequence labeling task in most current works which have achieved great success with the help of powerful neural networks. However, these works neglect an important clue: *Chinese characters incorporate both semantic and phonetic meanings*. In this paper, we introduce multiple character embeddings including *Pinyin Romanization* and *Wubi Input*, both of which are easily accessible and effective in depicting semantics of characters. We propose a novel *shared Bi-LSTM-CRF* model to fuse linguistic features efficiently by sharing the LSTM network during the training procedure. Extensive experiments on five corpora show that extra embeddings help obtain a significant improvement in labeling accuracy. Specifically, we achieve the state-of-the-art performance in AS and CityU corpora with F1 scores of 96.9 and 97.3, respectively without leveraging any external lexical resources.

1 Introduction

Chinese is written without explicit word delimiters so word segmentation (CWS) is a preliminary and essential pre-processing step for most natural language processing (NLP) tasks in Chinese, such as part-of-speech tagging (POS) and named-entity recognition (NER). The representative approaches are treating CWS as a character-based sequence labeling task following Xu (2003) and Peng et al. (2004).

Although not relying on hand-crafted features, most of the neural network models rely heavily on the embeddings of characters. Since Mikolov et al. (2013) proposed word2vec technique, the vector representation of words or characters has become a prerequisite for neural networks to solve NLP tasks in different languages.

However, existing approaches neglect an important fact that Chinese characters contain both semantic and phonetic meanings - there are various representations of characters designed for capturing these features. The most intuitive one is *Pinyin Romanization* (拼音) that keeps many-to-one relationship with Chinese characters - for one character, different meanings in specific context may lead to different pronunciations. This phenomenon called *Polyphony* (and *Polysemy*) in linguistics is very common and crucial to word segmentation task. Apart from Pinyin Romanization, *Wubi Input* (五笔) is another effective representation which absorbs semantic meanings of Chinese characters. Compared to Radical (偏旁) (Sun et al., 2014; Dong et al., 2016; Shao et al., 2017), Wubi includes more comprehensive graphical and structural information that is highly relevant to the semantic meanings and word boundaries, due to plentiful pictographic characters in Chinese and effectiveness of Wubi in embedding the structures.

This paper will thoroughly study how important the extra embeddings are and what scholars can achieve by combining extra embeddings with representative models. To leverage extra phonetic and semantic information efficiently, we propose a shared Bi-LSTMs-CRF model, which feeds embeddings into three stacked LSTM layers with shared parameters and finally scores with CRF layer. We evaluate the proposed approach on five corpora and demonstrate that our method produces state-of-the-art results and is highly efficient as previous single-embedding scheme.

Our contributions are summarized as follows: 1) We firstly propose to leverage both semantic and phonetic features of Chinese characters in NLP tasks by introducing Pinyin Romanization and Wubi Input embeddings, which are easily

[*] Equal contribution (alphabetical order).
[†] Corresponding author.

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop, pages 210–216
Florence, Italy, July 28 - August 2, 2019. ©2019 Association for Computational Linguistics

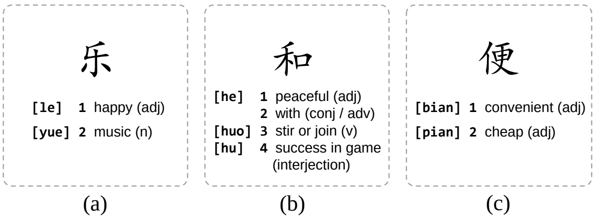


Figure 1: Examples of phono-semantic compound characters and polyphone characters.

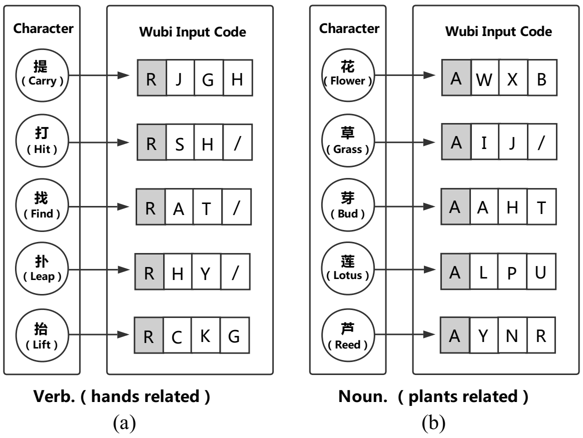


Figure 2: Potential semantic relationships between Chinese characters and Wubi Input. Gray area indicates that these characters have the same first letter in the Wubi Input representation.

accessible and effective in representing semantic and phonetic features; 2) We put forward a *shared Bi-LSTM-CRF* model for efficiently integrating multiple embeddings and sharing useful linguistic features; 3) We evaluate the proposed multi-embedding scheme on Bakeoff2005 and CTB6 corpora. Extensive experiments show that auxiliary embeddings help achieve state-of-the-art performance without external lexical resources.

2 Multiple Embeddings

To fully leverage various properties of Chinese characters, we propose to split the character-level embeddings into three parts: character embeddings for textual features, Pinyin Romanization embeddings for phonetic features and Wubi Input embeddings for structure-level features.

2.1 Chinese Characters

CWS is often regarded as a character-based sequence labeling task, which aims to label every character with $\{B, M, E, S\}$ tagging scheme. Recent studies show that character embeddings are the most fundamental inputs for neural networks (Chen et al., 2015; Cai and Zhao, 2016; Cai et al., 2017). However, Chinese characters are developed to absorb and fuse phonetics, semantics, and hieroglyphology. In this paper, we would like to explore other linguistic features so the characters are the basic inputs with two other presentations (*Pinyin* and *Wubi*) introduced as auxiliary.

2.2 Pinyin Romanization

Pinyin Romanization (拼音) is the official romanization system for standard Chinese characters (ISO 7098:2015, E), representing the pronunciation of Chinese characters like phonogram in English. Moreover, Pinyin is highly relevant to semantics - one character may correspond varied Pinyin code that indicates different semantic meanings. This phenomenon is very common in Asian languages and termed as polyphone.

Figure 1 shows several examples of polyphone characters. For instance, the character ‘乐’ in Figure 1 (a) has two different pronunciations (Pinyin code). When pronounced as ‘yue’, it means ’music’, as a noun. However, with the pronunciation of ’le’, it refers to ’happiness’. Similarly, the character ‘和’ in Figure 1 (b) even has four meanings with three varied Pinyin code.

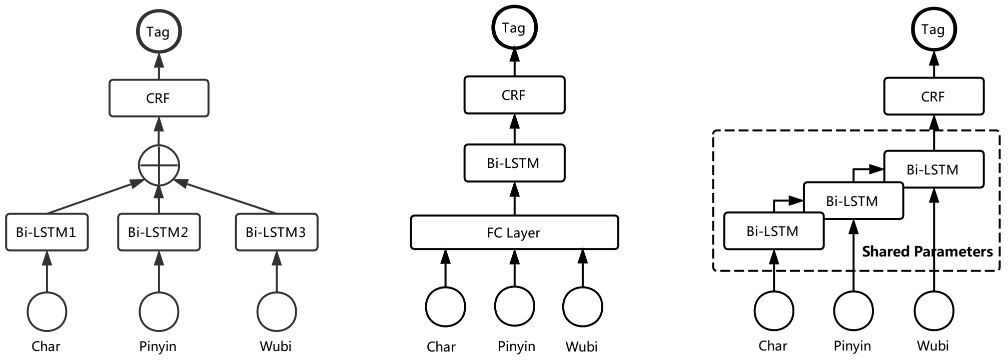
Through Pinyin code, a natural bridge is constructed between the words and their semantics. Now that human could understand the different meanings of characters according to varied pronunciations, the neural networks are also likely to learn the mappings between semantic meanings and Pinyin code automatically.

Obviously, Pinyin provides extra phonetic and semantic information required by some basic tasks such as CWS. It is worthy to notice that Pinyin is a dominant computer input method of Chinese characters, and it is easy to represent characters with Pinyin code as supplementary inputs.

2.3 Wubi Input

Wubi Input (五笔) is based on the structure of characters rather than the pronunciation. Since plentiful Chinese characters are hieroglyphic, Wubi Input can be used to find out the potential semantic relationships as well as the word boundaries. It is beneficial to CWS task mainly in two aspects: 1) Wubi encodes high-level semantic meanings of characters; 2) characters with similar structures (e.g., radicals) are more likely to make up a word, which effects the word boundaries.

To understand its effectiveness in structure description, one has to go through the rules of Wubi



(a) Model-I (b) Model-II (c) Model-III

Figure 3: Network architecture of three multi-embedding models. (a) Model-I: Multi-Bi-LSTMs-CRF Model. (b) Model-II: FC-Layer Bi-LSTMs-CRF Model. (c) Model-III: Shared Bi-LSTMs-CRF Model.

Input method. It is an efficient encoding system which represents each Chinese character with at most four English letters. Specifically, these letters are divided into five regions, each of which represents a type of structure (stroke, 笔画) in Chinese characters.

Figure 2 provides some examples of Chinese characters and their corresponding Wubi code (four letters). For instance, '提' (carry), '打' (hit) and '抬' (lift) in Figure 2 (a) are all verbs related to hands and correspond different spellings in English. On the contrary, in Chinese, these characters are all left-right symbols and have the same radical ('R' in Wubi code). That is to say, Chinese characters that are highly semantically relevant usually have similar structures which could be perfectly captured by Wubi. Besides, characters with similar structures are more likely to make up a word. For example, '花' (flower), '草' (grass) and '芽' (bud) in Figure 2 (b) are nouns and represent different plants. Whereas, they are all up-down symbols and have the same radical ('A' in Wubi code). These words usually make up new words such as '花草' (flowers and grasses) and '花芽' (the buds of flowers).

In addition, the sequence in Wubi code is one approach to interpret the relationships between Chinese characters. In Figure 2, it is easy to find some interesting component rules. For instance, we can conclude: 1) the sequence order implies the order of character components (e.g., 'IA' vs 'AI' and 'IY' vs 'YI'); 2) some code has practical meanings (e.g., 'I' denotes water). Consequently, Wubi is an efficient encoding of Chinese characters so incorporated as a supplementary input like Pinyin in our multi-embedding model.

2.4 Multiple Embeddings

To fully utilize various properties of Chinese characters, we construct the Pinyin and Wubi embeddings as two supplementary character-level features. We firstly pre-process the characters and obtain the basic character embedding following the strategy in Lample et al. (2016); Shao et al. (2017). Then we use the Pypinyin Library[1] to annotate Pinyin code, and an official transformation table[2] to translate characters to Wubi code. Finally, we retrieve multiple embeddings using word2vec tool (Mikolov et al., 2013).

For simplicity, we treat Pinyin and Wubi code as units like characters processed by canonical word2vec, which may discard some semantic affinities. It is worth noticing that the sequence order in Wubi code is an intriguing property considering the fact that structures of characters are encoded by the order of letters (see Sec 2.3). This point merits further study. Finally, we remark that generating Pinyin code relies on the external resources (statistics prior). Nontheless, Wubi code is converted under a transformation table so does not introduce any external resources.

3 Multi-Embedding Model Architecture

We adopt the popular Bi-LSTMs-CRF as our baseline model (Figure 4 without Pinyin and Wubi input), similar to the architectures proposed by Lample et al. (2016) and Dong et al. (2016). To obtain an efficient fusion and sharing mechanism for multiple features, we design three varied architectures (see Figure 3). In what follows, we will provide detailed explanations and analysis.

[1] https://pypi.python.org/pypi/pypinyin
[2] http://wubi.free.fr/index_en.html

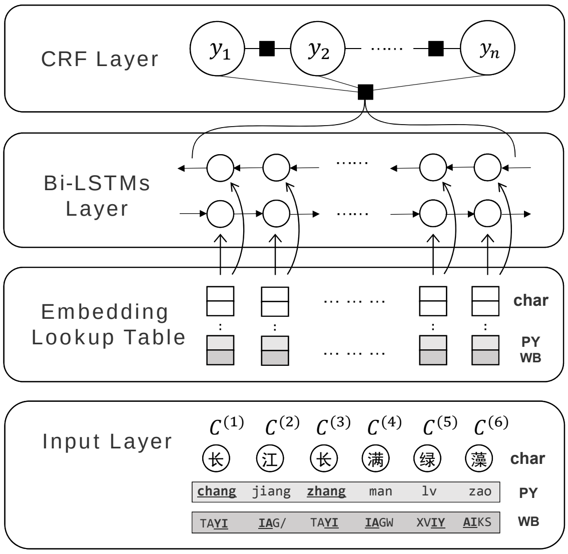

CRF Layer
y_1
y_2
......
y_n
Bi-LSTMs Layer
......
Embedding Lookup Table
char
PY
WB
Input Layer
$C^{(1)}$ $C^{(2)}$ $C^{(3)}$ $C^{(4)}$ $C^{(5)}$ $C^{(6)}$
长 江 长 满 绿 藻
char
chang jiang zhang man lv zao
PY
TAYI IAG/ TAYI IAGW XVIY AIKS
WB

Figure 4: The architecture of Bi-LSTM-CRF network. PY and WB represent *Pinyin Romanization* and *Wubi Input* introduced in this paper.

3.1 Model-I: Multi-Bi-LSTMs-CRF Model

In Model-I (Figure 3a), the input vectors of character, pinyin and wubi embeddings are fed into three independent stacked Bi-LSTMs networks and the output high-level features are fused via addition:

$$\begin{aligned}
\mathbf{h}_{3,c}^{(t)} &= \text{Bi-LSTMs}_1(\mathbf{x}_c^{(t)}, \theta_c), \\
\mathbf{h}_{3,p}^{(t)} &= \text{Bi-LSTMs}_2(\mathbf{x}_p^{(t)}, \theta_p), \\
\mathbf{h}_{3,w}^{(t)} &= \text{Bi-LSTMs}_3(\mathbf{x}_w^{(t)}, \theta_w), \\
\mathbf{h}^{(t)} &= \mathbf{h}_{3,c}^{(t)} + \mathbf{h}_{3,p}^{(t)} + \mathbf{h}_{3,w}^{(t)},
\end{aligned} \tag{1}$$

where θ_c, θ_p and θ_w denote parameters in three Bi-LSTMs networks respectively. The outputs of three-layer Bi-LSTMs are $\mathbf{h}_{3,c}^{(t)}$, $\mathbf{h}_{3,p}^{(t)}$ and $\mathbf{h}_{3,w}^{(t)}$, which form the input of the CRF layer $\mathbf{h}_{(t)}$. Here three LSTM networks maintain independent parameters for multiple features thus leading to a large computation cost during training.

3.2 Model-II: FC-Layer Bi-LSTMs-CRF Model

On the contrary, Model-II (Figure 3b) incorporates multiple raw features directly by inserting one fully-connected (FC) layer to learn a mapping between fused linguistic features and concatenated raw input embeddings. Then the output of this FC layer is fed into the LSTM network:

$$\begin{aligned}
\mathbf{x}_{in}^{(t)} &= [\mathbf{x}_c^{(t)}; \mathbf{x}_p^{(t)}; \mathbf{x}_w^{(t)}], \\
\mathbf{x}^{(t)} &= \sigma(\mathbf{W}_{fc}\mathbf{x}_{in}^{(t)} + \mathbf{b}_{fc}),
\end{aligned} \tag{2}$$

where σ is the logistic sigmoid function; $\mathbf{W}_{fc}$ and $\mathbf{b}_{fc}$ are trainable parameters of fully connected layer; $\mathbf{x}_c^{(t)}$, $\mathbf{x}_p^{(t)}$ and $\mathbf{x}_w^{(t)}$ are the input vectors of character, pinyin and wubi embeddings. The output of the fully connected layer $\mathbf{x}^{(t)}$ forms the input sequence of the Bi-LSTMs-CRF. This architecture benefits from its low computation cost but suffers from insufficient extraction from raw code. Meanwhile, Model-I and Model-II ignore the interactions between different embeddings.

3.3 Model-III: Shared Bi-LSTMs-CRF Model

To address feature dependency while maintaining training efficiency, Model-III (Figure 3c) introduces a sharing mechanism - rather than employing independent Bi-LSTMs networks for Pinyin and Wubi, we let them share the same LSTMs with character embeddings.

In Model-III, we feed character, Pinyin and Wubi embeddings sequentially into a stacked Bi-LSTMs network shared with the same parameters:

$$\begin{aligned}
\begin{bmatrix} \mathbf{h}_{3,c}^{(t)} \\ \mathbf{h}_{3,p}^{(t)} \\ \mathbf{h}_{3,w}^{(t)} \end{bmatrix} &= \text{Bi-LSTMs}(\begin{bmatrix} \mathbf{w}_c^{(t)} \\ \mathbf{w}_p^{(t)} \\ \mathbf{w}_w^{(t)} \end{bmatrix}, \theta), \\
\mathbf{h}^t &= \mathbf{h}_{3,c}^{(t)} + \mathbf{h}_{3,p}^{(t)} + \mathbf{h}_{3,w}^{(t)},
\end{aligned} \tag{3}$$

where θ denotes the shared parameters of Bi-LSTMs. Different from Eqn (1), there is only one shared Bi-LSTMs rather than three independent LSTM networks with more trainable parameters. In consequence, the shared Bi-LSTMs-CRF model can be trained more efficiently compared to Model-I and Model-II (extra FC-Layer expense).

Specifically, at each epoch, the parameters of three networks are updated based on unified sequential character, Pinyin and Wubi embeddings. The second LSTM network will share (or synchronize) the parameters with the first network before it begins the training procedure with Pinyin as inputs. In this way, the second network will take fewer efforts in refining the parameters based on the former correlated embeddings. So does the third network (taking Wubi embedding as inputs).

4 Experimental Evaluations

In this section, we provide empirical results to verify the effectiveness of multiple embeddings for CWS. Besides, our proposed Model-III can be

Models	CTB6			PKU			MSR			AS			CityU		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
baseline	94.1	94.0	94.1	95.8	95.9	95.8	95.3	95.7	95.5	95.6	95.5	95.6	95.9	96.0	96.0
Model-I	94.9	95.0	94.9	95.7	95.7	95.7	96.8	96.6	96.7	96.6	96.5	96.5	96.7	96.5	96.6
Model-II	**95.4**	**95.3**	**95.4**	**96.3**	95.7	96.0	96.6	96.5	96.6	96.8	96.5	96.7	**97.2**	**97.0**	**97.1**
Model-III	**95.4**	95.0	95.2	**96.3**	**96.1**	**96.2**	**97.0**	**96.9**	**97.0**	**96.9**	**96.8**	**96.9**	97.1	**97.0**	**97.1**

Table 1: Comparison of different architectures on five corpora. Bold font signifies the best performance in all given models. Our proposed multiple-embedding models result in a significant improvement compared to vanilla character-embedding baseline model.

Model	PKU	MSR	AS	CityU
(Sun and Wan, 2012)	95.4	97.4	-	-
(Chen et al., 2015)	94.8	95.6	-	-
(Chen et al., 2017)	94.3	96.0	-	94.8
(Ma et al., 2018)	96.1	**97.4**	96.2	97.2
(Zhang et al., 2013)*	96.1	97.4	-	-
(Chen et al., 2015)*	96.5	97.4	-	-
(Cai et al., 2017)*	95.8	97.1	95.6	95.3
(Wang and Xu, 2017)*	96.5	98.0	-	-
(Sun et al., 2017)*	96.0	97.9	96.1	96.9
baseline	95.8	95.5	95.6	96.0
ours (+PY)*	96.0	96.8	96.7	97.0
ours (+WB)	**96.3**	97.2	96.5	**97.3**
ours (+PY+WB)*	96.2	97.0	**96.9**	97.1

Table 2: Comparison with previous state-of-the-art models on all four Bakeoff2005 datasets. The second block (*) represents allowing the use of external resources such as lexicon dictionary or trained embeddings on large-scale external corpora. Note that our WB approach **does not** leverage any external resources.

trained efficiently (slightly costly than baseline) and obtain the state-of-the-art performance.

4.1 Experimental Setup

To make the results comparable and convincing, we evaluate our models on SIGHAN 2005 (Emerson, 2005) and Chinese Treebank 6.0 (CTB6) (Xue et al., 2005) datasets, which are widely used in previous works. We leverage standard word2vec tool to train multiple embeddings. In experiments, we tuned the embedding size following Yao and Huang (2016) and assigned equal size (256) for three types of embedding. The number of Bi-LSTM layers is set as 3.

4.2 Experimental Results

Performance under Different Architectures

We comprehensively conduct the analysis of three architecture proposed in Section 3. As illustrated in Table 1, considerable improvements are obtained by three multi-embedding models compared with our baseline model which only takes character embeddings as inputs. Overall, Model-III (shared Bi-LSTMs-CRF) achieves better performance even with fewer trainable parameters.

Competitive Performance

To demonstrate the effectiveness of supplementary embeddings for CWS, we compare our models with previous state-of-the-art models.

Table 2 shows the comprehensive comparison of performance on all Bakeoff2005 corpora. To the best of our knowledge, we have achieved the best performance on AS and CityU datasets (with F1 score 96.9 and 97.3 respectively) and competitive performance on PKU and MSR even if not leveraging external resources (e.g. pre-trained char/word embeddings, extra dictionaries, labeled or unlabeled corpora). It is worthy to notice that AS and CityU datasets are considered more difficult by researchers due to its larger capacity and higher out of vocabulary rate. It again verifies that Pinyin and Wubi embeddings are capable of decreasing mis-segmentation rate in large-scale data.

Embedding Ablation

We conduct embedding ablation experiments on CTB6 and CityU to explore the effectiveness of Pinyin and Wubi embeddings individually. As shown in Table 3, Pinyin and Wubi result in a considerable improvement on F1-score compared to vanilla single character-embedding model (baseline). Moreover, Wubi-aided model usually leads to a larger improvement than Pinyin-aided one.

Convergence Speed

To further study the additional expense after incorporating Pinyin and Wubi, we record the training time (batch time and convergence time in Table 4) of proposed models on MSR. Compared to

Models	CTB6			CityU		
	P	R	F	P	R	F
baseline	94.1	94.0	94.1	95.9	96.0	96.0
IO + PY	94.6	94.9	94.8	96.8	96.4	96.6
IO + WB	95.3	**95.4**	95.3	**97.3**	**97.3**	**97.3**
Model-II	**95.4**	95.3	**95.4**	97.2	97.0	97.1

Table 3: Feature ablation on CTB6 and CityU. IO + PY and IO + WB denote injecting Pinyin and Wubi embeddings separately under Model-II.

Model	Time (batch)	Time (P-95%)
baseline	1 ×	1 ×
Model-I	2.61 ×	2.51 ×
Model-II	1.03 ×	1.50 ×
Model-III	**1.07** ×	**1.04** ×

Table 4: Relative training time on MSR. (a) averaged training time per batch; (b) convergence time, where above 95% precision is considered as convergence.

the baseline model, it almost takes the same training time (1.07×) per batch and convergence time (1.04×) for Model-III. By contrast, Model-II leads to slower convergence (1.50×) in spite of its lower batch-training cost. In consequence, we recommend Model-III in practice for its high efficiency.

5 Related Work

Since Xu (2003), researchers have mostly treated CWS as a sequence labeling problem. Following this idea, great achievements have been reached in the past few years with the effective embeddings introduced and powerful neural networks armed.

In recent years, there are plentiful works exploiting different neural network architectures in CWS. Among these architectures, there are several models most similar to our model: Bi-LSTM-CRF (Huang et al., 2015), Bi-LSTM-CRF (Lample et al., 2016; Dong et al., 2016), and Bi-LSTM-CNNs-CRF (Ma and Hovy, 2016).

Huang et al. (2015) was the first to adopt Bi-LSTM network for character representations and CRF for label decoding. Lample et al. (2016) and Dong et al. (2016) exploited the Bi-LSTM-CRF model for named entity recognition in western languages and Chinese, respectively. Moreover, Dong et al. (2016) introduced radical-level information that can be regarded as a special case of Wubi code in our model.

Ma and Hovy (2016) proposed to combine Bi-LSTM, CNN and CRF, which results in faster convergence speed and better performance on POS and NER tasks. In addition, their model leverages both the character-level and word-level information.

Our work distinguishes itself by utilizing multiple dimensions of features in Chinese characters. With phonetic and semantic meanings taken into consideration, three proposed models achieve better performance on CWS and can be also adapted to POS and NER tasks. In particular, compared to radical-level information in (Dong et al., 2016), Wubi Input encodes richer structure details and potentially semantic relationships.

Recently, researchers propose to treat CWS as a word-based sequence labeling problem, which also achieves competitive performance (Zhang et al., 2016; Cai and Zhao, 2016; Cai et al., 2017; Yang et al., 2017). Other works try to introduce very deep networks (Wang and Xu, 2017) or treat CWS as a gap-filling problem (Sun et al., 2017). We believe that proposed linguistic features can also be transferred into word-level sequence labeling and correct the error. In a nutshell, multiple embeddings are generic and easily accessible, which can be applied and studied further in these works.

6 Conclusion

In this paper, we firstly propose to leverage phonetic, structured and semantic features of Chinese characters by introducing multiple character embeddings (*Pinyin* and *Wubi*). We conduct a comprehensive analysis on why Pinyin and Wubi embeddings are so essential in CWS task and could be translated to other NLP tasks such as POS and NER. Besides, we design three generic models to fuse the multi-embedding and produce the start-of-the-art performance in five public corpora. In particular, the shared Bi-LSTM-CRF models (Model III in Figure 3) could be trained efficiently and produce the best performance on AS and CityU corpora. In future, the effective ways of leveraging hierarchical linguistic features to other languages, NLP tasks (e.g., POS and NER) and refining mis-labeled sentences merit further study.

Acknowledgement

This research work has been partially funded by the National Natural Science Foundation of China (Grant No.61772337, U1736207), and the National Key Research and Development Program of China NO.2016QY03D0604.

References

Deng Cai and Hai Zhao. 2016. Neural word segmentation learning for chinese. In *ACL (1)*, Berlin, Germany. The Association for Computer Linguistics.

Deng Cai, Hai Zhao, Zhisong Zhang, Yuan Xin, Yongjian Wu, and Feiyue Huang. 2017. Fast and accurate neural word segmentation for chinese. In *ACL (2)*, pages 608–615. Association for Computational Linguistics.

Xinchi Chen, Xipeng Qiu, Chenxi Zhu, Pengfei Liu, and Xuanjing Huang. 2015. Long short-term memory neural networks for chinese word segmentation. In *EMNLP*, pages 1197–1206. The Association for Computational Linguistics.

Xinchi Chen, Zhan Shi, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial multi-criteria learning for chinese word segmentation. In *ACL (1)*, pages 1193–1203. Association for Computational Linguistics.

Chuanhai Dong, Jiajun Zhang, Chengqing Zong, Masanori Hattori, and Hui Di. 2016. Character-based LSTM-CRF with radical-level features for chinese named entity recognition. In *NLPCC/ICCPOL*, volume 10102 of *Lecture Notes in Computer Science*, pages 239–250. Springer.

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.*

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991.

ISO 7098:2015(E). 2015. Information and documentation – Romanization of Chinese. Standard, International Organization for Standardization, Geneva, CH.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *HLT-NAACL*, pages 260–270. The Association for Computational Linguistics.

Ji Ma, Kuzman Ganchev, and David Weiss. 2018. State-of-the-art chinese word segmentation with bi-lstms. In *EMNLP*, pages 4902–4908. Association for Computational Linguistics.

Xuezhe Ma and Eduard H. Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *ACL (1)*. The Association for Computer Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546.

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*, COLING '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yan Shao, Christian Hardmeier, Jörg Tiedemann, and Joakim Nivre. 2017. Character-based joint segmentation and POS tagging for chinese using bidirectional RNN-CRF. In *IJCNLP(1)*, pages 173–183. Asian Federation of Natural Language Processing.

Weiwei Sun and Xiaojun Wan. 2012. Reducing approximation and estimation errors for chinese lexical processing with heterogeneous annotations. In *ACL (1)*, pages 232–241. The Association for Computer Linguistics.

Yaming Sun, Lei Lin, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. 2014. Radical-enhanced chinese character embedding. In *ICONIP (2)*, volume 8835 of *Lecture Notes in Computer Science*, pages 279–286. Springer.

Zhiqing Sun, Gehui Shen, and Zhi-Hong Deng. 2017. A gap-based framework for chinese word segmentation via very deep convolutional networks. *CoRR*, abs/1712.09509.

Chunqi Wang and Bo Xu. 2017. Convolutional neural network with word embeddings for chinese word segmentation. In *IJCNLP(1)*, pages 163–172. Asian Federation of Natural Language Processing.

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

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

Jie Yang, Yue Zhang, and Fei Dong. 2017. Neural word segmentation with rich pretraining. In *ACL (1)*, pages 839–849. Association for Computational Linguistics.

Yushi Yao and Zheng Huang. 2016. Bi-directional LSTM recurrent neural network for chinese word segmentation. In *ICONIP (4)*, volume 9950 of *Lecture Notes in Computer Science*, pages 345–353.

Longkai Zhang, Houfeng Wang, Xu Sun, and Mairgup Mansur. 2013. Exploring representations from unlabeled data with co-training for chinese word segmentation. In *EMNLP*, pages 311–321. ACL.

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, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.