Hyper-Gated Recurrent Neural Networks for Chinese Word Segmentation

Zhan Shi[*], Xinchi Chen , Xipeng Qiu, Xuanjing Huang

Shanghai Key Laboratory of Intelligent Information Processing, Fudan University
School of Computer Science, Fudan University
825 Zhangheng Road, Shanghai, China
{zshi16,xinchichen13,xpqiu,xjhuang}@fudan.edu.cn

Abstract. Recently, recurrent neural networks (RNNs) have been increasingly used for Chinese word segmentation to model the contextual information without the limit of context window. In practice, two kinds of gated RNNs, long short-term memory (LSTM) and gated recurrent unit (GRU), are often used to alleviate the long dependency problem. In this paper, we propose the hyper-gated recurrent neural networks for Chinese word segmentation, which enhance the gates to incorporate the historical information of gates. Experiments on the benchmark datasets show that our model outperforms the baseline models as well as the state-of-the-art methods.

1 Introduction

Unlike English and other western languages, Chinese do not delimit words by white-space. Therefore, Chinese word segmentation (CWS) is a preliminary and important pre-process for Chinese language processing. The popular method is to regard word segmentation task as a sequence labeling problem [17, 15] and has achieved great success. Due to the nature of supervised learning, the performance of these models is greatly affected by the design of features. These features are explicitly represented by the different combinations of context characters, which are based on linguistic intuition and statistical information. However, the number of features could be so large that the result models are too large to use in practice and prone to overfit on training corpus.

Recently, neural network models have been increasingly focused on for their ability to minimize the effort in feature engineering. [20] firstly applied the general neural framework proposed by [5] to Chinese word segmentation. Following this work, many neural models for word segmentation are proposed and achieved a comparable performance to the traditional state-of-the-art methods, such as neural tensor network [14], gated recursive neural network [3]. However, these neural models just concatenate the embeddings of the context characters, and feed them into neural network. Despite of their success, a limitation of them is that their performances are easily affected by the size of the context window. Intuitively, many words are difficult to segment based on the local information only.

[*] Corresponding author.

To alleviate these problem, [4] used an LSTM architecture to capture potential long-distance dependencies. After that, LSTM (or alternative GRU) became the popular model for CWS.

In this paper, we propose the hyper-gated recurrent neural networks to model the complicated combinations of characters, and apply it to Chinese word segmentation task. Specifically, we improve the gated RNNs by enhancing their gates. Since the gates play important roles in gated RNNs to controll the information flow. To better model the combinations of context characters, we add recurrent connections between the gates.

The contributions of this paper can be summarized as follows:

– We conduct extensive experiments on eight CWS corpora, which is by far the largest number of datasets used simultaneously. The experimental results show that our models outperform the baseline methods.

2 Recurrent Neural Networks for Chinese Word Segmentation

Recently, neural networks are widely applied to Chinese word segmentation (CWS) task [14, 12, 16, 18, 2, 19, 20]. In this paper, we focus on character based CWS using recurrent neural networks. Usually, character based Chinese word segmentation task is regarded as a sequence labeling problem. Specifically, each character in a sentence is labeled as one of $\mathcal{L} = \{B, M, E, S\}$, indicating the begin, middle, end of a word, or a word with single character.

Generally, the neural architecture of character based CWS could be characterized by three components: (1) a character embedding layer; (2) Recurrent neural network layer for feature extraction and (3) a CRF layer. Figure 1 illustrates the general architecture of CWS.

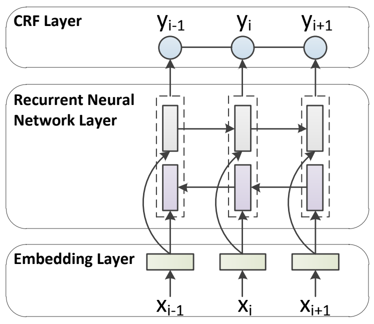
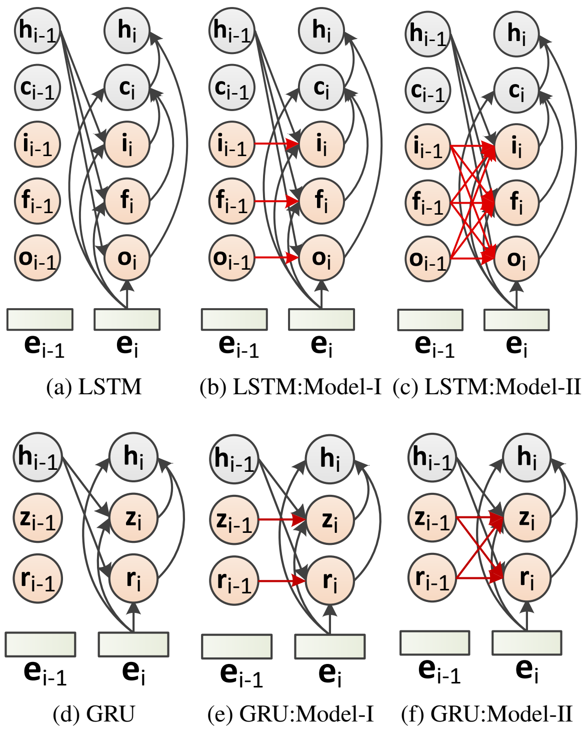


Fig. 1: General neural architecture for Chinese word segmentation.

2.1 Embedding Layer

Regularly, in neural models, the first step is to map discrete language symbols to distributed embedding vectors. Specifically, given a sequence with n characters $X =$



(a) LSTM (b) LSTM:Model-I (c) LSTM:Model-II

(d) GRU (e) GRU:Model-I (f) GRU:Model-II

Fig. 2: Hyper-Gated Recurrent Neural Networks. Red lines shows the difference in information flow.

$\{x_1, \ldots, x_n\}$, we should firstly lookup embedding vector for each character x_i from embedding matrix as $\mathbf{e}_{x_i} \in \mathbb{R}^{d_e}$, where d_e is a hyper-parameter indicating the dimensionality of character embedding.

2.2 Recurrent Neural Network Layer

The role of recurrent neural network layer is to extract features by modeling sequential information of a given unsegmented sentences [4]. In this paper, we employ the bi-direction long short-term memory network and the bi-direction gated recurrent neural network for feature extraction.

Simple Recurrent Neural Network Specifically, simple recurrent neural network (SRN-N) could be formalized as:

$$\mathbf{h}_i = \phi\left(\mathbf{W}\begin{bmatrix}\mathbf{e}_{x_i}\\ \mathbf{h}_{i-1}\end{bmatrix} + \mathbf{b}\right), \tag{1}$$

where $\mathbf{W} \in \mathbb{R}^{d_h \times (d_e + d_h)}$ and $\mathbf{b} \in \mathbb{R}^{d_h}$. d_h is the dimensionality of hidden state of SRNN.

Long Short-Term Memory Neural Network Long short-term memory (LSTM) neural network [7] introduces gate mechanism (input gate $\mathbf{i}$, output gate $\mathbf{o}$, forget gate $\mathbf{f}$) and memory cell (memory cell $\mathbf{c}$) to maintain longer dependency information and avoid gradient vanishing. Specifically, LSTM could be expressed as:

$$\begin{bmatrix} \mathbf{i}_i \\ \mathbf{o}_i \\ \mathbf{f}_i \\ \tilde{\mathbf{c}}_i \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ \sigma \\ \phi \end{bmatrix} \left(\mathbf{W} \begin{bmatrix} \mathbf{e}_{x_i} \\ \mathbf{h}_{i-1} \end{bmatrix} + \mathbf{b} \right), \tag{2}$$

$$\mathbf{c}_i = \mathbf{c}_{i-1} \odot \mathbf{f}_i + \tilde{\mathbf{c}}_i \odot \mathbf{i}_i, \tag{3}$$

$$\mathbf{h}_i = \mathbf{o}_i \odot \phi(\mathbf{c}_i), \tag{4}$$

where $\mathbf{W} \in \mathbb{R}^{4d_h \times (d_e + d_h)}$ and $\mathbf{b} \in \mathbb{R}^{4d_h}$. Function $\sigma(\cdot)$ and $\phi(\cdot)$ are sigmoid and tanh functions respectively.

Gated Recurrent Neural Network The gated recurrent unit (GRU) was proposed by [1] to make gated recurrent neural network (GRU) to adaptively capture dependencies of different time scales. Formally, GRU could be expressed as:

$$\begin{bmatrix} \mathbf{z}_i \\ \mathbf{r}_i \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \end{bmatrix} \left(\mathbf{W}_g \begin{bmatrix} \mathbf{e}_{x_i} \\ \mathbf{h}_{i-1} \end{bmatrix} + \mathbf{b}_g \right), \tag{5}$$

$$\tilde{\mathbf{h}}_i = \phi \left(\mathbf{W} \begin{bmatrix} \mathbf{e}_{x_i} \\ \mathbf{r}_i \odot \mathbf{h}_{i-1} \end{bmatrix} + \mathbf{b} \right), \tag{6}$$

$$\mathbf{h}_i = \tilde{\mathbf{h}}_i \odot \mathbf{z}_i + \mathbf{h}_{i-1} \odot (1 - \mathbf{z}_i), \tag{7}$$

where $\mathbf{z}_i$ and $\mathbf{r}_i$ are update gate and reset gate respectively.

Bidirectional Recurrent Neural Network In order to incorporate information from both sides of sequence, it is common to employ bi-directional recurrent neural network with forward and backward directions. The corresponding Bi-SRNN, Bi-LSTM and Bi-GRU could be derived as:

$$\mathbf{h}_i = \overrightarrow{\mathbf{h}}_i \oplus \overleftarrow{\mathbf{h}}_i, \tag{8}$$

where $\overrightarrow{\mathbf{h}}_i$ and $\overleftarrow{\mathbf{h}}_i$ are the forward and backward hidden states at position i respectively. $\oplus$ is a concatenation operation

2.3 Inference Layer

The objective of inference layer is to figure out the ground truth of labels $Y^* = \{y_1^*, \ldots, y_n^*\}$:

$$Y^* = \underset{Y \in \mathcal{L}^n}{\arg\max}\, p(Y|X), \tag{9}$$

where $\mathcal{L} = \{B, M, E, S\}$.

In this paper, we employ conditional random fields (CRF) [10] layer for tag inference. In CRF layer, $p(Y|X)$ in Eq (9) could be formalized as:

$$p(Y|X) = \frac{\Psi(Y|X)}{\sum_{Y' \in \mathcal{L}^n} \Psi(Y'|X)}. \tag{10}$$

Here, $\Psi(Y|X)$ is the potential function, and we only consider interactions between two successive labels (first order linear chain CRFs):

$$\Psi(Y|X) = \prod_{i=2}^{n} \psi(X, i, y_{i-1}, y_i), \tag{11}$$

$$\psi(\mathbf{x}, i, y', y) = \exp(s(X, i)_y + \mathbf{b}_{y'y}), \tag{12}$$

where $\mathbf{b}_{y'y} \in \mathbf{R}$ is trainable parameters respective to label pair (y', y). Score function $s(X, i) \in \mathbb{R}^{|\mathcal{L}|}$ assigns score for each label on tagging the i-th character:

$$s(X, i) = \mathbf{W}_s^\top \mathbf{h}_i + \mathbf{b}_s, \tag{13}$$

where $\mathbf{h}_i$ is the hidden state of recurrent neural network layer at position i; $\mathbf{W}_s \in \mathbb{R}^{d_h \times |\mathcal{L}|}$ and $\mathbf{b}_s \in \mathbb{R}^{|\mathcal{L}|}$ are trainable parameters.

3 Hyper-Gated Recurrent Neural Networks for Chinese Word Segmentation

Hyper-gated recurrent neural networks enhance the gate mechanism by introducing the sequential information on gates. In this paper, we proposed two types of hyper-gated models for both LSTM and GRU: the gate independent model (Section 3.1) and the gate fusing model (Section 3.2).

3.1 Model-I: Gate Independent Model

In model-I, the gate independent model, we regard the gates independently. Each of them are only related to the current input embedding $\mathbf{e}_{x_i}$, previous hidden state of $\mathbf{h}_{i-1}$, and corresponding previous gate states.

Hyper-Gated LSTMs Hyper-gated long short-term memory (HG-LSTM) neural networks build the input gate, forget gate and output gate via another independent recurrent neural network respectively. Specifically, the gate independent HG-LSTM model could be formalized as:

$$\mathbf{i}_i = \text{SRNN}_\sigma(\mathbf{e}_{x_i}, \mathbf{i}_{i-1}, \mathbf{h}_{i-1}), \tag{14}$$

$$\mathbf{o}_i = \text{SRNN}_\sigma(\mathbf{e}_{x_i}, \mathbf{o}_{i-1}, \mathbf{h}_{i-1}), \tag{15}$$

$$\mathbf{f}_i = \text{SRNN}_\sigma(\mathbf{e}_{x_i}, \mathbf{f}_{i-1}, \mathbf{h}_{i-1}), \tag{16}$$

$$\tilde{\mathbf{c}}_i = \text{SRNN}_\phi(\mathbf{e}_{x_i}, \mathbf{h}_{i-1}), \tag{17}$$

$$\mathbf{c}_i = \mathbf{c}_{i-1} \odot \mathbf{f}_i + \tilde{\mathbf{c}}_i \odot \mathbf{i}_i, \tag{18}$$

$$\mathbf{h}_i = \mathbf{o}_i \odot \phi(\mathbf{c}_i), \tag{19}$$

where $\text{SRNN}_\sigma(\cdot)$ and $\text{SRNN}_\phi(\cdot)$ are regular recurrent networks with sigmoid activation $\sigma(\cdot)$ and tanh activation $\phi(\cdot)$ respectively as Eq. (1).

Hyper-Gated GRUs Hyper-gated GRUs build the update gate and reset gate and output gate via another independent recurrent neural network respectively as well. Specifically, the gate independent HG-GRU model could be formalized as:

$$\mathbf{z}_i = \text{SRNN}_\sigma(\mathbf{e}_{x_i}, \mathbf{z}_{i-1}, \mathbf{h}_{i-1}), \tag{20}$$

$$\mathbf{r}_i = \text{SRNN}_\sigma(\mathbf{e}_{x_i}, \mathbf{r}_{i-1}, \mathbf{h}_{i-1}), \tag{21}$$

$$\tilde{\mathbf{h}}_i = \text{SRNN}_\phi(\mathbf{e}_{x_i}, \mathbf{r}_i \odot \mathbf{h}_{i-1}), \tag{22}$$

$$\mathbf{h}_i = \tilde{\mathbf{h}}_i \odot \mathbf{z}_i + \mathbf{h}_{i-1} \odot (1 - \mathbf{z}_i), \tag{23}$$

where $\text{SRNN}_\sigma(\cdot)$ and $\text{SRNN}_\phi(\cdot)$ are regular recurrent networks with sigmoid activation $\sigma(\cdot)$ and tanh activation $\phi(\cdot)$ respectively as Eq. (1).

3.2 Model-II: Gate Fusing Model

In model-II, the gate fusing model, we additionally take the information interactions of different types of gates into account. Specifically, each type of gate is related to all of other types of gates. Thus, we could derive gate fusing HG-LSTM and gate fusing HG-GRU models as follows.

Hyper-Gated LSTMs Formally, we could formalize the gate fusing HG-LSTM model by only modifying the Eq. (14), Eq. (15), and Eq. (16) as:

$$\mathbf{i}_i = \text{RNN}_\sigma(\mathbf{e}_{x_i}, \mathbf{g}_{i-1}, \mathbf{h}_{i-1}), \tag{24}$$

$$\mathbf{o}_i = \text{RNN}_\sigma(\mathbf{e}_{x_i}, \mathbf{g}_{i-1}, \mathbf{h}_{i-1}), \tag{25}$$

$$\mathbf{f}_i = \text{RNN}_\sigma(\mathbf{e}_{x_i}, \mathbf{g}_{i-1}, \mathbf{h}_{i-1}), \tag{26}$$

where $\mathbf{g}_i$ is the gate fusing state on i-th step, which could be derived by an concatenation operation over gate states (input, output and forget gates) on i-th step:

$$\mathbf{g}_i = \begin{bmatrix} \mathbf{i}_i \\ \mathbf{o}_i \\ \mathbf{f}_i \end{bmatrix} \tag{27}$$

Hyper-Gated GRUs Similarly, we could formalize the gate fusing HG-GRU model by only modifying the Eq. (20) and Eq. (21) as:

$$\mathbf{z}_i = \text{SRNN}_\sigma(\mathbf{e}_{x_i}, \mathbf{g}_{i-1}, \mathbf{h}_{i-1}), \tag{28}$$

$$\mathbf{r}_i = \text{SRNN}_\sigma(\mathbf{e}_{x_i}, \mathbf{g}_{i-1}, \mathbf{h}_{i-1}), \tag{29}$$

where $\mathbf{g}_i$ is the gate fusing state on i-th step, which could be derived by an concatenation operation over gate states (update and reset gates) on i-th step:

$$\mathbf{g}_i = \begin{bmatrix} \mathbf{z}_i \\ \mathbf{r}_i \end{bmatrix} \tag{30}$$

Datasets			N_{w}	N_{c}	$\|\mathcal{D}_w\|$	$\|\mathcal{D}_c\|$	N_s
Sighan05	MSRA	Train	2.4M	4.1M	88.1K	5.2K	86.9K
		Test	0.1M	0.2M	12.9K	2.8K	4.0K
	AS	Train	5.4M	8.4M	141.3K	6.1K	709.0K
		Test	0.1M	0.2M	18.8K	3.7K	14.4K
Sighan08	PKU	Train	1.1M	1.8M	55.2K	4.7K	47.3K
		Test	0.2M	0.3M	17.6K	3.4K	6.4K
	CTB	Train	0.6M	1.1M	42.2K	4.2K	23.4K
		Test	0.1M	0.1M	9.8K	2.6K	2.1K
	CKIP	Train	0.7M	1.1M	48.1K	4.7K	94.2K
		Test	0.1M	0.1M	15.3K	3.5K	10.9K
	CITYU	Train	1.1M	1.8M	43.6K	4.4K	36.2K
		Test	0.2M	0.3M	17.8K	3.4K	6.7K
	NCC	Train	0.5M	0.8M	45.2K	5.0K	18.9K
		Test	0.1M	0.2M	17.5K	3.6K	3.6K
	SXU	Train	0.5M	0.9M	32.5K	4.2K	17.1K
		Test	0.1M	0.2M	12.4K	2.8K	3.7K

Table 1: Details of eight datasets. N_w and N_c indicate numbers of tokens and characters respectively. $\mathcal{D}_w$ and $\mathcal{D}_c$ are the dictionaries of distinguished words and characters respectively. N_s indicates the number of sentences.

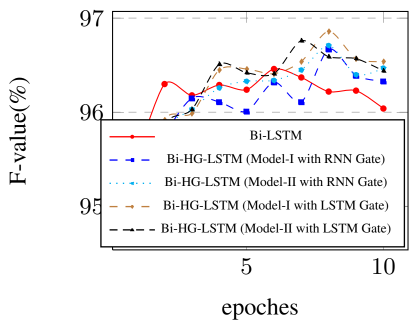
Models	P	R	F	OOV
Baselines				
Bi-LSTM	93.67	92.93	93.3	66.09
HG-LSTM (Model-I) with LSTM gates				
+ LSTM gate o	94.04	93.5	93.77	67.76
+ LSTM gates o & i	94.26	93.44	93.85	68.91
+ LSTM gates o & i & f	94.16	93.75	**93.95**	67.45
Baselines				
Bi-GRU	93.65	92.58	93.11	65.65
HG-GRU (Model-I) with GRU gates				
+ GRU gate z	94.13	92.91	93.52	66.12
+ GRU gates z & r	94.05	93.45	**93.75**	67.51

Table 2: Effects of using hyper gates on the test set of PKU dataset. The maximum F value is highlighted for each main block.

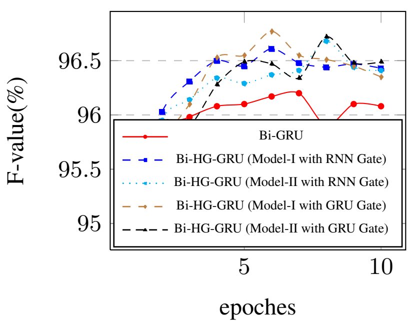
Models		MSRA	AS	PKU	CTB	CKIP	CITYU	NCC	SXU	Avg.
Hyper-Gated LSTM										
Baselines										
Bi-LSTM	P	95.70	93.64	93.67	95.19	92.44	94.00	91.34	94.91	93.86
	R	95.99	94.77	92.93	95.42	93.69	94.15	92.12	95.03	94.26
	F	**95.84**	94.20	93.30	**95.30**	**93.06**	**94.07**	91.73	94.97	**94.06**
	OOV	66.28	70.07	66.09	76.47	72.12	65.79	57.31	71.17	68.16
Stacked Bi-LSTM	P	95.69	93.89	94.10	95.20	92.40	94.13	91.78	94.79	94.00
	R	95.81	94.54	92.66	95.40	93.39	93.99	91.94	95.17	94.11
	F	95.75	**94.22**	**93.37**	**95.30**	92.89	94.06	**91.86**	**94.98**	94.05
	OOV	65.55	71.50	67.92	75.44	70.50	66.35	59.88	69.69	68.35
Hyper-gated LSTM with RNN gates										
Model-I	P	95.81	94.19	94.11	95.39	92.34	93.87	91.47	95.06	94.03
	R	96.21	95.16	93.19	95.24	93.72	94.28	92.27	95.24	94.41
	F	**96.01**	94.67	**93.65**	95.32	93.02	**94.07**	91.87	**95.15**	94.22
	OOV	68.52	72.68	68.44	76.21	71.49	66.50	58.10	70.49	69.05
Model-II	P	95.76	94.34	94.05	95.08	92.90	94.01	91.73	95.28	94.14
	R	95.99	95.13	93.24	95.69	93.68	94.12	92.41	95.01	94.41
	F	95.88	**94.74**	93.64	**95.38**	**93.29**	**94.07**	**92.07**	95.14	**94.28**
	OOV	65.85	71.29	67.39	75.83	73.35	65.88	59.41	71.71	68.84
Hyper-gated LSTM with LSTM gates										
Model-I	P	96.04	94.41	94.02	95.46	92.35	94.24	91.23	95.08	94.10
	R	96.09	95.33	93.20	95.46	93.73	94.15	92.78	95.29	94.50
	F	96.07	**94.87**	93.61	**95.46**	93.04	**94.20**	**92.00**	**95.19**	94.31
	OOV	68.69	70.48	66.57	77.24	70.39	66.59	57.64	70.84	68.56
Model-II	P	96.17	94.29	94.16	95.35	92.64	93.91	91.62	95.13	94.16
	R	96.34	95.23	93.75	95.45	93.78	94.11	92.32	95.14	94.52
	F	**96.26**	94.76	**93.95**	95.40	**93.20**	94.01	91.97	95.14	**94.34**
	OOV	69.20	71.40	67.45	77.21	71.25	65.40	57.96	70.65	68.82

Hyper-Gated GRU										
Baselines										
Bi-GRU	P	94.90	93.10	93.65	95.13	91.73	93.66	91.30	94.74	93.53
	R	95.44	93.99	92.58	95.16	93.24	93.90	91.85	95.04	93.90
	F	95.17	**93.54**	93.11	95.15	92.48	**93.78**	**91.57**	**94.89**	93.71
	OOV	63.82	68.38	65.65	75.91	69.46	65.17	55.67	68.77	66.60
Stacked Bi-GRU	P	95.46	92.86	93.65	95.26	92.07	93.59	90.40	94.54	93.48
	R	95.58	94.17	92.84	95.27	93.15	93.68	92.40	94.99	94.01
	F	**95.52**	93.51	**93.25**	**95.26**	**92.60**	93.63	91.39	94.76	**93.74**
	OOV	65.33	69.68	65.81	77.00	70.98	64.68	54.34	69.41	67.15
Hyper-gated GRU with RNN gates										
Model-I	P	95.72	93.92	94.15	95.41	92.62	93.84	91.10	94.81	93.95
	R	96.04	94.97	93.03	95.30	93.65	94.22	92.47	95.15	94.35
	F	95.88	94.45	**93.58**	**95.35**	93.13	94.03	91.78	94.98	94.15
	OOV	67.09	70.85	68.99	77.33	71.09	65.64	57.05	69.64	68.46
Model-II	P	95.85	94.19	94.08	94.84	92.69	93.86	91.45	94.71	93.96
	R	96.05	95.11	92.70	95.75	93.64	94.39	92.33	95.47	94.43
	F	**95.95**	**94.65**	93.38	95.29	**93.16**	**94.12**	**91.89**	**95.09**	**94.19**
	OOV	68.29	71.50	67.80	75.01	71.37	66.73	58.36	69.06	68.52
Hyper-gated GRU with GRU gates										
Model-I	P	95.78	94.32	94.05	95.56	92.67	94.15	91.31	95.00	94.11
	R	95.86	95.00	93.45	95.45	93.50	94.19	92.58	95.25	94.41
	F	95.82	94.66	**93.75**	**95.50**	93.08	**94.17**	91.94	**95.13**	**94.26**
	OOV	67.65	71.31	67.51	78.93	70.60	67.31	57.58	69.71	68.83
Model-II	P	96.00	94.24	94.10	95.20	92.45	93.88	91.19	94.69	93.97
	R	95.97	95.10	93.27	95.25	93.87	94.29	92.77	95.49	94.50
	F	**95.98**	**94.67**	93.68	95.23	**93.15**	94.09	**91.97**	95.09	94.23
	OOV	69.79	72.09	68.49	76.57	71.10	66.18	57.66	69.03	68.86

Table 3: Results of the proposed models on test sets of eight CWS datasets. P, R, F and OOV indicate precision, recall, F value and out-of-vocabulary recall rate respectively.



(a) HG-LSTM



(b) HG-GRU

Fig. 3: Convergence speed of the proposed HG-LSTM and HG-GRU models on the development set of PKU dataset.

3.3 Hyper-Gated Recurrent Neural Networks with Enhanced Gates

To better integrating the longer dependency information, we further adopt LSTM and GRU to model all types of hyper gates instead of using simple recurrent neural networks (SRNN). Specifically, we could replace all the hyper gates in Eq. (14-16), Eq. (20-21), Eq. (24-26), Eq. (28-29) by LSTM$_\sigma$ or GRU$_\sigma$. LSTM$_\sigma(\cdot)$ is a variation of regular LSTM by using sigmoid activation instead of tanh activation. Similarly, GRU$_\sigma(\cdot)$ is a variation of regular GRU by using sigmoid activation instead of tanh activation.

Character embedding size	$d_e = 50$
Initial learning rate	$\alpha = 0.2$
Loss weight coefficient	$\lambda = 0.05$
LSTM dimensionality	$d_h = 100$
Dropout rate on input layer	$p = 20\%$

Table 4: Configurations of Hyper-parameters.

4 Training

The training object is to maximize the log conditional likelihood of the true labels. The objective function $\mathcal{J}_{seg}(\Theta)$ can be computed as:

$$\mathcal{J}_{seg}(\Theta) = \frac{1}{2m} \sum_{i=1}^{m} \log p(Y_i|X_i; \Theta) + \lambda||\Theta||_2^2, \tag{31}$$

where Θ denotes all the trainable parameters of the proposed model. m denotes the number of training examples. λ is the coefficient of the regularization term.

We use Adam [9] with minibatchs to maximize the objective.

5 Experiments

5.1 Datasets

To evaluate the proposed architecture, we do extensive experiments on eight prevalent CWS datasets from SIGHAN2005 [6] and SIGHAN2008 [8]. The details of the eight datasets are shown in Table 1. Among these datasets, AS, CITYU and CKIP are traditional Chinese, while the remains, MSRA, PKU, CTB, NCC and SXU, are simplified Chinese. We use 10% data of shuffled train set as development set for all datasets.

5.2 Experimental Configurations

Table 4 gives the configurations of the hyper-parameters. Since the scale of each datasets varies, we use different training batch sizes for each corpus. Besides, the batch sizes of AS and MSRA datasets is 512 and 256 respectively, and the batch sizes of the other datasets are 128. To prevant our model from overfiting, we employ dropout strategy after embedding layer with 20% dropout rate (keeping 80% inputs).

For initialization, all parameters is drawn from a uniform distribution $(-0.05, 0.05)$ and the character embedding matrix is pre-trained on Chinese Wikipedia corpus, using word2vec toolkit [13]. Following previous work [4, 14], all experiments including baseline results are using bigram feature, with pre-tarined character embeddings as initialization.

5.3 Overall Results

Table 3 shows the experimental results of the proposed models on test sets of eight CWS datasets, which has two main blocks. Each main block contains three sub-blocks. These two main blocks show the experimental results on LSTM and GRU respectively.

(1) In the first sub-blocks, we could observe that the performance of Bi-LSTM and Bi-GRU cannot be improved by merely increasing the depth of networks. Besides, averagely speaking, Bi-LSTM outperforms Bi-GRU model (the average F value of Bi-LSTM is 94.06, while the average F value of Bi-GRU is only 93.74).

(2) In the second blocks, the performance of proposed hyper-gated LSTM and hyper-gated GRU using RNN gates is boosted significantly. For HG-LSTM with RNN gates, Model-I and Model-II gain 0.16 and 0.22 improvements on averaging F-measure score respectively compared with Bi-LSTM result (94.06%). For HG-GRU with RNN gates, Model-I and Model-II gain 0.41 and 0.45 improvements on averaging F-measure score respectively compared with stacked Bi-GRU result (93.74%). Compared to the baseline results, the proposed models boost the performance with the help of exploiting sequential information of gates.

(3) In the third blocks, we experiment on more sophisticated hyper gates. By introducing LSTM gates and GRU gates, the performances are further boosted. For HG-LSTM with LSTM gates, Model-I and Model-II obtain 94.31 and 94.34 on averaging F-measure score respectively, with 0.06 improvement compared to HG-LSTM with simple RNN gates. For HG-GRU with GRU gates, Model-I and Model-II obtain 94.26 and 94.23 on averaging F-measure score respectively, with 0.07 improvement compared to HG-GRU with simple RNN gates.

In summary, we could observe that (a) Bi-LSTM outperforms Bi-GRU. (b) The performance of Model-I and Model-II is comparable. No significant boost is observed by using gate fusion, which shows that the information interactions between different types of gates have little contribution to the finial performance. (c) By using more sophisticated hyper gates, the performance is further boosted, which shows that sequential information of gates really contribute to the performance much.

5.4 Effects of Hyper Gates

We also investigates the effects of the proposed hyper gates. Figure 2 gives the results of two models, HG-LSTM (Model-I) with LSTM gates and HG-GRU (Model-I) with GRU gates, on the test set of PKU dataset. There are two main block. As we can see, the performance is boosted gradually when we replace more regular gates by the hyper gates in both blocks. It shows that the proposed hyper gate mechanism could better control the information flow by exploiting the sequential information over gates.

5.5 Convergency

Figure 3 shows the learning curve of the proposed HG-LSTM and HG-GRU on the development set of PKU. As we can see, the proposed models convergence as fast as previous plain LSTM and GRU. With the help of hyper gate mechanism, the HG-LSTM and HG-GRU could be convergent to a better results fast.

6 Related Work

Chinese word segmentation has been studied with considerable efforts in the NLP community. The most popular word segmentation method is based on sequence labeling [17]. Recently, researchers have tended to explore neural network based approaches to reduce efforts of the feature engineering. Among these methods, more and more methods adopts RNN-based architecture to model the contextual information [4, 16, 18, 2, 19, 11].

The gates play important roles in gated RNNs to controll the information flow. To better model the combinations of context characters, we enhance the gates by adding recurrent connections between the gates.

7 Conclusion

In this paper, we propose hyper-gated recurrent neural networks to enhance the recurrent connections between the gates for Chinese word segmentation task. Experiments show that our proposed model performances well on eight benchmark datasets.

Despite Chinese word segmentation being a specific case, our model can be easily generalized and applied to other sequence labeling tasks. In future work, we would like to investigate our proposed models on other sequence labeling tasks.

References

[1] Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. ArXiv e-prints (Sep 2014)

[2] Cai, D., Zhao, H.: Neural word segmentation learning for chinese. arXiv preprint arXiv:1606.04300 (2016)

[3] Chen, X., Qiu, X., Zhu, C., Huang, X.: Gated recursive neural network for chinese word segmentation. In: Proceedings of Annual Meeting of the Association for Computational Linguistics. pendency parsing using two heterogeneous gated recursive neural networks. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (2015)

[4] Chen, X., Qiu, X., Zhu, C., Liu, P., Huang, X.: Long short-term memory neural networks for chinese word segmentation. In: EMNLP. pp. 1197–1206 (2015)

[5] Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. The Journal of Machine Learning Research 12, 2493–2537 (2011)

[6] Emerson, T.: The second international chinese word segmentation bakeoff. In: Proceedings of the fourth SIGHAN workshop on Chinese language Processing. vol. 133 (2005)

[7] Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation 9(8), 1735–1780 (1997)

[8] Jin, G., Chen, X.: The fourth international chinese language processing bakeoff: Chinese word segmentation, named entity recognition and chinese pos tagging. In: Sixth SIGHAN Workshop on Chinese Language Processing. p. 69 (2008)

[9] Kingma, D., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)

[10] Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of the Eighteenth International Conference on Machine Learning (2001)

[11] Liu, Y., Che, W., Guo, J., Qin, B., Liu, T.: Exploring segment representations for neural segmentation models. arXiv preprint arXiv:1604.05499 (2016)

[12] Ma, J., Hinrichs, E.W.: Accurate linear-time chinese word segmentation via embedding matching. In: ACL (1). pp. 1733–1743 (2015)

[13] Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)

[14] Pei, W., Ge, T., Baobao, C.: Maxmargin tensor neural network for chinese word segmentation. In: Proceedings of ACL (2014)

[15] Peng, F., Feng, F., McCallum, A.: Chinese segmentation and new word detection using conditional random fields. Proceedings of the 20th international conference on Computational Linguistics (2004)

[16] Xu, J., Sun, X.: Dependency-based gated recursive neural network for chinese word segmentation. In: The 54th Annual Meeting of the Association for Computational Linguistics. p. 567 (2016)

[17] Xue, N.: Chinese word segmentation as character tagging. Computational Linguistics and Chinese Language Processing 8(1), 29–48 (2003)

[18] Yao, Y., Huang, Z.: Bi-directional lstm recurrent neural network for chinese word segmentation. In: International Conference on Neural Information Processing. pp. 345–353. Springer (2016)

[19] Zhang, M., Zhang, Y., Fu, G.: Transition-based neural word segmentation. Proceedings of the 54nd ACL (2016)

[20] Zheng, X., Chen, H., Xu, T.: Deep learning for chinese word segmentation and pos tagging. In: EMNLP. pp. 647–657 (2013)