BiGCNN: Bidirectional Gated Convolutional Neural Network for Chinese Named Entity Recognition

Tianyang Zhao[1], Haoyan Liu[1], Qianhui Wu[2], Changzhi Sun[3], Dongdong Zhan[3], and Zhoujun Li[1](✉)

[1] State Key Lab of Software Development Environment, Beihang University, Beijing, China
{tyzhao,haoyan.liu,lizj}@buaa.edu.cn

[2] Tsinghua University, Beijing, China
wu-qh16@mails.tsinghua.edu.cn

[3] East China Normal University, Shanghai, China
czsun.cs@gmail.com, ahnuzdd@gmail.com

Abstract. Recent advances on Chinese named entity recognition (NER) are mostly based on the recurrent neural network (RNN). Since RNNs are limited in parallel processing, some works apply the convolutional neural network (CNN) to perform NER. However, existing CNN-based models fail to explicitly distinguish the preceding and subsequent contexts, so they are difficult to handle cases that are sensitive to the location of the contexts. Moreover, they pay equal attention to the context within a convolution kernel, while not all the information is useful for semantic understanding. In this paper, we propose a novel CNN-based model, **Bi**directional **G**ated **C**onvolutional **N**eural **N**etwork (BiGCNN), to differentiate the entity-related information between preceding and subsequent contexts and filter out the convolution information adaptively. By incorporating automatic segmentation and glyph information, BiGCNN outperforms state-of-the-art models on four Chinese NER datasets. Additionally, benefiting from the parallelism processing, the proposed method enjoys higher training and testing efficiency, e.g., 12.04 times faster than RNN-based models, while with better performance.

1 Introduction

Named entity recognition (NER) is a task to identify text spans of named entities from text, and to classify them into predefined types like location (`LOC`), person (`PER`), organization (`ORG`), etc. It is an essential component in a variety of NLP applications such as event extraction [4], coreference resolution [9] and relation extraction [33]. And there is increasing interest in the field.

Most existing NER systems are based on recurrent neural network (RNN), especially long-short-term-memory (LSTM) [18,19,21]. But in RNNs, the outputs in each step rely on the previous step which hinders the parallel processing

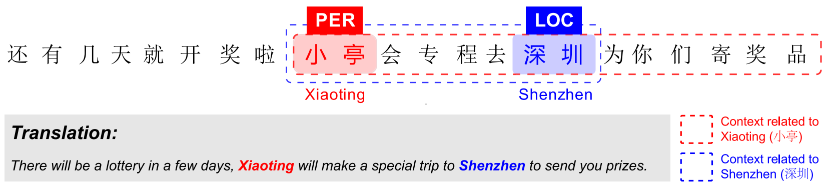

PER
LOC
还 有 几 天 就 开 奖 啦 小 亭 会 专 程 去 深 圳 为 你 们 寄 奖 品
Xiaoting
Shenzhen
Translation:
There will be a lottery in a few days, Xiaoting will make a special trip to Shenzhen to send you prizes.
Context related to Xiaoting (小亭)
Context related to Shenzhen (深圳)

Fig. 1. An example from Weibo dataset where labels of the entities (小亭 $_{PER}$ and 深圳 $_{LOC}$) only depend on either preceding or subsequent context.

over an input sequence, so its computational speed is inevitably constrained. The convolutional neural network (CNN) operates all inputs simultaneously and thus allows parallelization over sequential inputs, which leads to higher efficiency. Therefore, some recent works proposed CNN-based approaches as an alternative to better capture the sequential information. For example, GRN [3] used CNNs with a gated relation structure and gained considerable improvement on English NER. The lastest method LR-CNN [11] adopted CNNs with a lexicon rethinking mechanism and achieved state-of-the-art performance on Chinese NER.

Although recent works adopting CNNs for NER task have achieved great success, these methods still face two limitations. **First**, existing CNN-based models use one convolution to capture both preceding and subsequent context features simultaneously and then combined them together through pooling operations, meaning that the unique information of these two parts cannot be explicitly distinguished. In many cases, the label of an named entity only depends on the preceding or the subsequent context. For example, as shown in the Chinese sentence in Fig. 1, 小亭 (*Xiaoting*) is a person, whose meaning can only be inferred from the the subsequent context "会专程去 $\cdots$ (*will make a special trip to*$\cdots$)", and it is irrelevant to the previous "还有几天 $\cdots$ (*There will be*$\cdots$)". Similarly, the label of 深圳 (*Shenzhen*), `LOC`, only depends on the preceding context "小亭 会专程去 $\cdots$(*Xiaoting will make a special trip to*$\cdots$)" and has little relation with the subsequent "为你们寄奖品 (*to send you prizes*)". Therefore, to better capture the direction-sensitive cases with CNNs, it is necessary to take the bidirectionality into account for CNN structures. **Second**, current CNN-based methods lack an effective mechanism to control the context information. In traditional CNN models, all information within a certain kernel size will be propagated to the next computation stage. However, not all information is truly useful for semantic understanding. So it is difficult to select the meaningful feature based on the plain context. Furthermore, information could easily vanish through transformation. Existing gated linear unit (GLU) [6] only considers the output information to handle this issue but neglects to control the input and other fined-grained features. Hence, a more comprehensive mechanism to filter out context information becomes a pressing need for the CNN structure.

In this paper, we propose a **Bi**directional **G**ated **C**onvolutional **N**eural **N**etwork (BiGCNN), for Chinese NER task. To address the first problem, a novel and effective bidirectional CNN structure is introduced to better differentiate the entity-related information between the preceding and subsequent contexts. Specifically, It employs two independent CNNs to explicitly capture specific features from the two parts. In this manner, contexts of different directions can be modeled separately. To tackle the second problem, we propose a comprehensive gated convolutional unit (GCU) to purify the context information through multiple schemes, i.e.,the convolution gate, the update gate and the reset gate. As a result, the irrelevant information after the convolution update and the useful inputs are integrated to better represent context features. By incorporating the proposed bidirectional and gating mechanism, the CNN structure is capable of extracting the different information of the preceding and subsequent contexts adaptively and flexibly. Additionally, BiGCNN is completely based on CNN structure and thus it is much more efficient than RNN based models, which is quite beneficial to practical applications.

Specifically, we conduct extensive experiments on Chinese NER task. Compared with other languages, Chinese has the following substantial properties. Firstly, there are no explicit word boundaries for Chinese text. To avoid segmentation errors, many Chinese NER systems are based on characters rather than perform word-level NER [20,31]. Recent works further assigned words information to characters to fully use the word-level semantics [11,32]. In particular, the proposed model is character-based and further integrates characters with the automatic segmentation to take advantage of word-level features. Secondly, Chinese characters are logographic-based and the logographs always convey rich semantic meanings. For example, 河 (*river*), 湖 (*lake*), and 洋 (*ocean*) all include the radical 氵 (*water*). Hereby, Dong et al. [8] proposed to use radical sequences to model Chinese characters. However, this method ignores the case that completely different characters may share the same radical sequence. For instance, three characters 困 (*tired*), 呆 (*nerd*), and 杏 (*apricot*) can be only split into two radical sequences, i.e., "口, 木" and "木, 口" . Recently, Glyce [29] collected various historical scripts and writing styles of characters and use the ensembles to encode Chinese characters. Accordingly, we also use the glyph information. But different from Glyce which uses the glyph to pre-train character representations, we treat it as an additional feature to enhance Chinese structural property. Moreover, without any manual work, we use an automatic and simple way to model the glyph feature effectively.

To summarize, the main contributions of this work are:

- We propose a novel and effective bidirectional CNN structure to distinguish the entity-related information between the preceding and subsequent contexts, which is beneficial to direction-sensitive NER cases.
- We introduce a comprehensive gating mechanism for CNN structure through multiple control schemes, to better filter out irrelevant information and retain the useful ones.
- Enhanced with segmentation and glyph features, BiGCNN achieves state-of-the-art performance on four Chinese NER datasets, and accelerates up to 12.04 times over RNN-based models because of the parallelism processing.

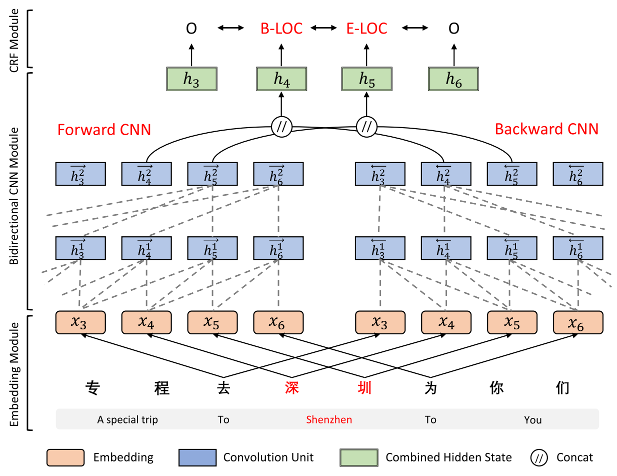


Fig. 2. Framework of BiGCNN. We take the processing of characters "深" and "圳" as an example to illustrate the architecture.

2 Model

In this section, we introduce the overall architecture of BiGCNN in detail. Formally, for an input sentence with T characters $\boldsymbol{s} = c_1, c_2, \ldots, c_T$, where c_i is the i-th character. The NER task is to predict a label sequence $\boldsymbol{y} = y_1, y_2, \ldots, y_T$, where y_i is the entity type of c_i. Suppose a sequence of characters $c_i^j = c_i, \ldots, c_j$ forms a word w, its corresponding segmentation sequence can be formulated as $w_i^j = w_i, \ldots, w_j$ where $w_i, \ldots, w_j = w, \ldots, w$. Take the sequence in Fig. 2 as an example, the characters c_4 = "深" and c_5 = "圳" compose a word "深 圳 (Shenzhen)", and then it comes to w_4 = "深圳" and w_5 = "深圳"..

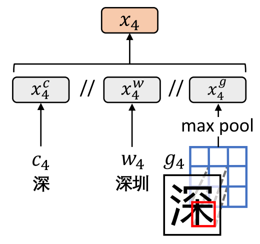
As illustrated in Fig. 2, BiGCNN consists of three modules: the embedding module, the bidirectional CNN module, and the CRF module. We will elaborate on each of them in the following subsections.

2.1 Embedding Module

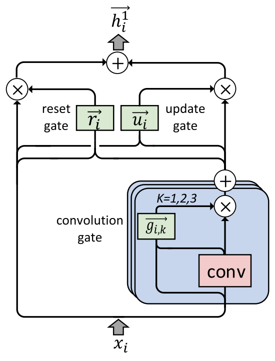
The embedding module focuses on mapping discrete characters into distributed semantic representations. As shown in Fig. 3a, for an input character c_i, the output of the embedding module x_i, is the concatenation of the character embedding x_i^c, the segmentation embedding x_i^w, and the glyph embedding x_i^g as:

$$x_i = [x_i^c; x_i^w; x_i^g], \tag{1}$$

where $x_i \in \mathbb{R}^{d_e}$ with d_e being the concatenated embedding size, and ; denotes the concatenation operation.



(a) Embedding Module.



(b) Gated Convolutional Unit.

Fig. 3. Architectural elements in BiGCNN. (a) The output of the embedding module is the concatenation of the character embedding, the segmentation embedding and the glyph embedding. (b) The gated convolutional unit includes the convolution gate, the update gate and the reset gate.

Character Embedding. Given a character c_i, the character embedding x_i^c is defined as:

$$x_i^c = E^c(c_i), \tag{2}$$

where E^c is the random-initialized dictionary and be fine-tuned during training.

Segmentation Embedding. Simply using the character embedding ignores the inherent word information in the sentence. Therefore, we integrate the word segmentation into the character-level feature. For a word w_i corresponding to the character c_i, its segmentation embedding x_i^w is extracted as:

$$x_i^w = E^w(w_i), \tag{3}$$

where E^w is initialized with pre-trained word embedding and be fine-tuned during training.

Glyph Embedding. Character and word embedding can model most of the semantic information. But the embeddings of rare and unknown characters or words are less reliable. Inspired by [29], we propose a simplified glyph representation to better encode Chinese structural property. We generate a 24×24 pixel-sized image[1] for every character according to its glyph morphology. Characters are rendered to binary images via API calls to *pygame* library[2]. Defining I_i as the glyph image of the character c_i, we use a 2D convolution with a 3×3

[1] Empirically, smaller sizes than 24×24 lead to blurry glyph images, while larger sizes are unnecessary.
[2] https://www.pygame.org/docs/.

kernel to extract the glyph-based feature $\hat{x}_i^g$, and then perform a max pooling operation to aggregate the convolution result into the glyph embedding x_i^g as:

$$\begin{aligned} \hat{x}_i^g &= \text{conv}(I_i), \\ x_i^g &= \text{max\_pooling}(\hat{x}_i^g). \end{aligned} \tag{4}$$

Compared with the radical embedding in [8], the glyph embedding can uniquely encode each character based on its logographic property. Our experiments further verify its effectiveness and superiority over the radical embedding.

2.2 Bidirectional CNN Module

The bidirectional CNN module takes concatenated character-level feature as input and separately model preceding and subsequent contexts to high-level features for entity type prediction. We describe each component in detail as follows.

Bidirectional Structure. We propose a bidirectional structure typically for the convolutional operation, which leverages a forward CNN and a backward CNN to exploit the preceding and subsequent contexts, respectively.

For a T-character sentence with embeddings $\boldsymbol{x} = x_1, x_2, \ldots, x_T$, which are derived by Eq. 1, the output of the forward CNN at position i only relies on the i-th character and its preceding ones, i.e., $[x_{i-(k-1)d}, x_{i-(k-2)d}, ..., x_i]$, as:

$$\overrightarrow{h_{i,k}} = \text{conv}_k([x_{i-(k-1)d}, x_{i-(k-2)d}, ..., x_i]). \tag{5}$$

Here, $\overrightarrow{h_{i,k}} \in \mathbb{R}^{d_h}$ with d_h being the hidden size, k is the kernel size and d is the dilation factor which means each character in the sliding window skips over d characters in the input sentence. Considering that the semantic dependency between characters is not limited by a fixed distance, as revealed in [3], we utilize three different convolutions, each with a kernel size of $k = 1$, 2, and 3, respectively, to extract multi-scaled features.

The backward CNN performs in a similar way, except that its inputs are in the reverse order, i.e., $[x_i, x_{i+d}, ..., x_{i+(k-1)d}]$. For simplicity, we take the forward CNN as an example in the following procedures to elaborate on the module.

Gated Convolutional Unit. As shown in Fig. 3b, the gated convolutional unit (GCU) contains three kinds of control schemes, including the convolution gate, the update gate and the reset gate.

- **The convolution gate**: Generally, the contexts with different length contribute differently to understanding semantics of a given entity. Hence, it is necessary to clarify the importance of the convolutional output with a certain kernel size. For the input embedding x_i, the convolution gate corresponding to kernel size k is calculated as:

$$\overrightarrow{g}_{i,k} = \sigma\left(\boldsymbol{W}_g^k \overrightarrow{h_{i,k}} + \boldsymbol{V}_g^k x_i + \boldsymbol{b}_g^k\right), \tag{6}$$

where $\boldsymbol{W}_g^k \in \mathbb{R}^{d_h \times d_h}$, $\boldsymbol{V}_g^k \in \mathbb{R}^{d_e \times d_h}$, and $\boldsymbol{b}_g^k \in \mathbb{R}^{d_h}$ are learned parameters. $\sigma(\cdot)$ is the sigmoid function, we define its output as the convolution gate. The filtered representation of the input x_i is the gated combination of multi-kernel convolutions as:

$$\overrightarrow{h_i}' = \sum\nolimits_{k=1,2,3} \overrightarrow{h_{i,k}} \otimes \overrightarrow{g_{i,k}}, \tag{7}$$

where $\otimes$ denotes the element-wise multiplication. The $\overrightarrow{h_i}'$ is the final combined convolutional hidden output, and is used to calculate the following update and reset gates.

- **The update gate**: The update gate focuses on filtering out the irrelevant information implied in the combined output above, which is defined as:

$$\overrightarrow{u_i} = \sigma\left(\boldsymbol{W}_u \overrightarrow{h_i}' + \boldsymbol{V}_u x_i + \boldsymbol{b}_u\right), \tag{8}$$

  where $\boldsymbol{W}_u \in \mathbb{R}^{d_h \times d_h}$, $\boldsymbol{V}_u \in \mathbb{R}^{d_e \times d_h}$ and $\boldsymbol{b}_u \in \mathbb{R}^{d_h}$ are trainable parameters.
- **The reset gate**: As indicted in [5], the input embedding x_i is essential for prevent information decay. Therefore, we defined the reset gate to regulate the flow of input semantic information as:

$$\overrightarrow{r_i} = \sigma\left(\boldsymbol{W}_r \overrightarrow{h_i}' + \boldsymbol{V}_r x_i + \boldsymbol{b}_r\right), \tag{9}$$

  which is parameterized by $\boldsymbol{W}_r \in \mathbb{R}^{d_h \times d_h}$, $\boldsymbol{V}_r \in \mathbb{R}^{e \times d_h}$ and $\boldsymbol{b}_r \in \mathbb{R}^{d_h}$.

The final hidden output of the gated convolutional unit is calculated by:

$$\overrightarrow{h_i} = \overrightarrow{h_i}' \otimes \overrightarrow{u_i} + x_i \otimes \overrightarrow{r_i}. \tag{10}$$

Similarly, we obtain $\overleftarrow{h_i}$ as the output of the backward CNN for the i-th input character x_i. Practically, we stack two convolutional layers of each direction to enlarge the receptive fields[3]. Denote $\overrightarrow{h_i^L}$ and $\overleftarrow{h_i^L}$ as the last hidden output of forward and backward CNN, the final result of the bidirectional CNN module is the concatenation as:

$$h_i = [\overrightarrow{h_i^L}; \overleftarrow{h_i^L}], \tag{11}$$

which will be mapped into entity type distributions and fed into the CRF module.

2.3 CRF Module

Conditional Random Field (CRF) is a probabilistic method that jointly models interactions between entity labels, which is incorporated in nearly all state-of-the-art NER models. Similarly, we utilize a CRF module over the bidirectional CNN module to calculate loss and perform label decoding.

[3] We also evaluated more convolution layers, but found the results comparable while the computational cost is higher.

Table 1. Statistics of Datasets.

Dataset	Type	Train	Dev	Test
OntoNotes4	Char	491.9 k	200.5 k	208.1 k
	Sentence	15.7 k	4.3 k	4.3 k
MSRA	Char	2169.9 k	–	172.6 k
	Sentence	46.4 k	–	4.4 k
Weibo	Char	73.8 k	14.5 k	14.8 k
	Sentence	1.4 k	0.27 k	0.27 k
Resume	Char	124.1 k	13.9 k	15.1 k
	Sentence	3.8 k	0.46 k	0.48 k

Loss Function. Suppose that the final output of the bidirectional CNN module forms a sequence $\boldsymbol{h} = h_1, h_2, \ldots, h_T$, where h_i corresponds to the hidden state of the i-th character derived from Eq. 11. Given a label sequence $\boldsymbol{y} = y_1, y_2, \ldots, y_T$, $p(\boldsymbol{y}|\boldsymbol{h})$ is defined as the probability of using $\boldsymbol{y}$ as the prediction sequence for the sentence as follows:

$$p(\boldsymbol{y}|\boldsymbol{h}) = \frac{\prod_{i=1}^{N} \phi_i(y_{i-1}, y_i, \boldsymbol{h})}{\sum_{y' \in \mathcal{Y}(\boldsymbol{h})} \prod_{i=1}^{N} \phi_i(y'_{i-1}, y'_i, \boldsymbol{h})}. \tag{12}$$

Here, $\mathcal{Y}(\boldsymbol{h})$ denotes the set of all possible label sequences. $\phi_i(y_{i-1}, y_i, \boldsymbol{h}) = \exp(\boldsymbol{W}_{\text{CRF}}^{y_i} h_i + \boldsymbol{b}_{\text{CRF}}^{y_{i-1} \rightarrow y_i})$, where $\boldsymbol{W}_{\text{CRF}} \in \mathbb{R}^{d_h \times d_l}$ and $\boldsymbol{b}_{\text{CRF}} \in \mathbb{R}^{d_l \times d_l}$ with d_l being the label vocabulary size. $\boldsymbol{W}_{\text{CRF}}^{y_i}$ is the column corresponding to label y_i, and $\boldsymbol{b}_{\text{CRF}}^{y_{i-1} \rightarrow y_i}$ is the transition probability from label y_{i-1} to y_i.

During training, the loss function $\mathcal{L}$ is defined as the negative log-likelihood:

$$\mathcal{L} = -\sum\nolimits_{\boldsymbol{h}} \log p(\boldsymbol{y}|\boldsymbol{h}). \tag{13}$$

Label Decoding. During inference, we predict the label sequence y^* with the maximal likelihood which can be efficiently settled by the Viterbi algorithm:

$$y^* = \arg\max\nolimits_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{h})} p(\boldsymbol{y}|\boldsymbol{h}). \tag{14}$$

3 Experiments

To demonstrate the effectiveness of the bidirectional gated CNN structure, we evaluate BiGCNN over four widely-used Chinese NER datasets, and compare it with existing state-of-the-art methods.

3.1 Datasets

We conduct extensive experiments on four datasets including OntoNotes4 [28], MSRA [16], Weibo [22] and Resume [32]. Table 1 shows the statistics of the four datasets. Gold-standard segmentation is provided for OntoNotes4 and MSRA training set. Since the golden segmentation is not available for Weibo and Resume datasets as well as the test set of MSRA, we adopt the automatic neural segmentor as [32] to construct the word segmentation for each sentence. The data splits and tagging scheme(i.e., BIOES) follow those in [32].

3.2 Settings

For evaluation, standard precision (P), recall (R) and F1-score (F1) are used as metrics in our experiments. Other settings of our model are described as follows:

Embeddings. The dimension of character embedding is 80 and the embedding matrix is randomly initialized with kaiming uniform [13]. We use the word segmentation embedding following paper [34], which is trained on Chinese Baidu encyclopedia [17]. For the out-of-vocabulary words, we initialize them with a uniform distribution as in [21]. The dimension of glyph embeddings is set to 20.

Weight Initialization. We initialize all weights of convolution operations (Eq. 4 and Eq. 5) in the same way as in [10], while other weights are initialized with kaiming uniform [13] and bias with zero.

Network Structure. The output channel of bidirectional convolution (Eq. 5) is set as 400. The first layer of the bidirectional CNN module uses a dilation factor $d = 1$ and the second layer uses $d = 2$.

Training. We use stochastic gradient descent (SGD) with momentum as the optimizer, where the batch size is 10 and the momentum is 0.9. The learning rate is initialized as $\eta_0 = 0.02$. At the end of each epoch, we update the learning rate with $\eta_t = \frac{\eta_0}{1+\rho t}$, where $\rho = 0.05$ is the decay rate and t refers to the epoch index. Dropout layers are added upon both the inputs and outputs of the bidirectional CNN module, with a dropout rate of 0.5. We train our model for 100 epochs and report the average P/R/F1 results of 5 runs for each experiment.

3.3 Experimental Results

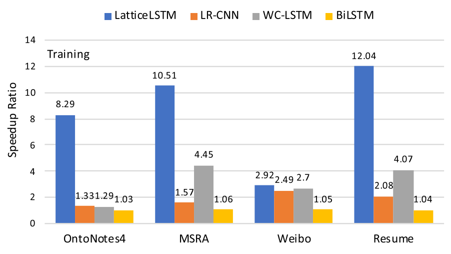
Table 2 shows the performance of BiGCNN. The first block of sub-tables lists the recent advances for Chinese NER. Among them, LR-CNN [11] is the latest Chinese NER model based on CNN. All others employ RNN structure as backbones. Note that WC-LSTM [19] adopts four strategies to encode word information, we listed their best results for better comparison. The second block lists three baselines with the same embedding module as BiGCNN (Eq. 1). Specifically, BiLSTM [14] is the base of most RNN-based varieties. GRN [3] is the recent CNN-based model for English NER. Transformer [27] constructs character representations through attention mechanism and enjoys the parallelism property similar to CNN. All of these models adopt CRF to perform label prediction.

Table 2. Performance comparisons on the four datasets. * denotes models based on RNN structure. † denotes models based on CNN structure. The "NE" and "NM" denote F1-scores for named entities and nominal entities, following [22].

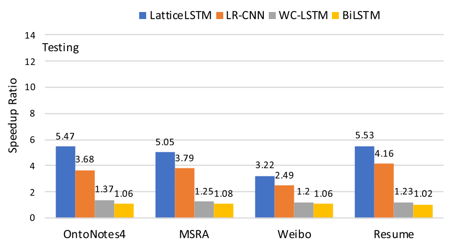
Model	**OntoNotes4**			**Resume**		
	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)
CAN-NER [34]*	73.63	70.82	72.20	95.05	94.82	94.94
LatticeLSTM [32]*	76.35	71.56	73.88	94.81	94.11	94.46
WC-LSTM [19]*	76.09	72.85	74.43	95.27	95.15	95.21
LR-CNN [11]†	76.40	72.60	74.45	95.37	94.84	95.11
BiLSTM*	75.94	70.42	73.03	93.69	94.76	94.23
GRN [3]†	75.81	70.97	73.33	93.92	94.89	94.40
Transformer [27]	73.25	71.39	72.31	93.61	92.95	93.27
BiGCNN w/o glyph	**76.32**	**73.66**	**74.97**	94.81	**95.50**	95.15
BiGCNN	**76.86**	**74.10**	**75.46**	94.84	**95.62**	**95.23**
Model	MSRA			Weibo		
	P(%)	R(%)	F1(%)	NE(%)	NM(%)	F1(%)
Cao et al. [2]*	91.73	89.58	90.64	54.34	57.35	58.70
CAN-NER [34]*	93.53	92.42	92.97	55.38	62.98	59.31
LatticeLSTM [32]*	93.57	92.79	93.18	53.04	62.25	58.79
WC-LSTM [19]*	94.58	92.91	93.74	52.55	67.41	59.84
LR-CNN [11]†	94.50	92.93	93.71	57.14	66.67	59.92
BiLSTM*	93.63	92.26	92.94	52.56	63.44	58.25
GRN [3]†	93.37	92.00	92.68	53.08	63.53	58.90
Transformer [27]	92.75	91.18	91.96	52.35	62.56	57.20
BiGCNN w/o glyph	**94.66**	**92.93**	**93.78**	**57.42**	66.82	**60.27**
BiGCNN	**94.63**	**93.14**	**93.88**	**57.60**	**67.56**	**61.54**

Comparisons with RNN-based Models. As shown in Table 2, without external labeled features, BiGCNN achieves the state-of-the-art performance on four datasets compared with all the RNN-based models. Even removing the glyph embedding, BiGCNN still outperforms the baselines on most of the datasets and is strongly competitive with the best result on Resume, despite the performance drop slightly. This well verifies the feasibility and effectiveness of replacing RNN with our well-designed CNN on Chinese NER task. Meanwhile, the extended glyph embedding is helpful for further improvement.

Comparisons with CNN and Transformer Based Models. BiGCNN outperforms the best CNN-based model LR-CNN [11] on four datasets. Especially on OntoNotes4 and Weibo, BiGCNN consistently increasing the F1 score by 1.01% and 1.62%. Without the glyph embedding, the improvement is also



(a) Training Speedups.



(b) Testing Speedups.

Fig. 4. Training and testing speedups gained by BiGCNN over baseline models.

considerable. Besides, compared with the latest CNN-based model for English NER, GRN [3], BiGCNN has an overall significant performance boost. Both LR-CNN and GRN are based on traditional convolutions. Hence, such improvements well demonstrate that the bidirectionality is beneficial for common CNN structures.

Additionally, the Transformer [27] also performs inferior to BiGCNN on the four datasets. We consider that the Transformer focuses on learning distant dependencies within a sequence. While for NER, the long-term dependency is not absolutely necessary [1]. Therefore, with limited data, extracting local features using CNNs is more helpful to improve NER performance than the Transformer.

3.4 Efficiency

In this section, we further evaluate the speedups gained by BiGCNN over latest state-of-the-art Chinese NER models and the basic BiLSTM model [21]. All models are trained for 10 epochs in total with the same batch size on the same physical machine and use CRF for label decoding. After each training epoch, we evaluate the learned model on the test set. We log the training and testing time costs for each epoch and calculate the average. Accordingly, the speedups are obtained for efficiency comparisons.

Speedups over State-of-the-art Chinese NER Models. LatticeLSTM [32], WC-LSTM [19] and LR-CNN [11] are the recent advanced models for Chinese NER. We run their publicly available implementations. As shown in Fig. 4, BiGCNN is much faster than the three baselines both in training and testing. Particularly on Resume, the training-time speedup is 12.04 times over LatticeLSTM. Despite that LR-CNN is also CNN-based, BiGCNN still gains noticeable speedups (an average of 1.86 and 3.53 times faster than LR-CNN for training and test). The reason may be that LR-CNN needs to stack multiple layers to extract multi-gram lexicon features, and readjust the weight of each layer after the convolutional operation, which inevitably limits efficiency. More importantly, BiGCNN does NOT sacrifice performance for speedup, meaning that it is more effective and efficient than the compared models.

Table 3. Experimental comparisons on the four datasets for enhancing BiGCNN with fine-tuned BERT. The results of BERT are listed following [29].

Model	OntoNotes4			Resume		
	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)
BERT	78.01	80.35	79.16	96.12	95.45	95.78
Glyce	82.06	68.74	74.81	95.72	95.63	95.67
BiGCNN+BERT	**79.63**	**80.41**	**80.02**	**95.42**	**96.76**	**96.08**
Model	MSRA			Weibo		
	P(%)	R(%)	F1(%)	NE(%)	NM(%)	F1(%)
BERT	94.97	94.62	94.80	67.12	66.88	67.33
Glyce	93.86	93.92	93.89	53.69	55.30	54.32
BiGCNN+BERT	**95.51**	**95.80**	**95.65**	**67.50**	**73.25**	**68.91**

Table 4. Ablation study results (F1 score) on the four datasets.

Model	OntoNotes4	MSRA	Weibo	Resume
BiGCNN	**75.46**	**93.88**	**61.54**	**95.23**
BiGCNN w/o direction	73.85	92.70	59.02	94.06
BiGCNN w/o GCU	74.21	93.06	59.41	94.39
BiGCNN replace GCN w/GLU	74.83	93.36	60.34	95.02
BiGCNN w/o glyph	74.97	93.78	60.27	95.15
BiGCNN replace glyph w/radical	74.40	93.23	60.12	94.90

Speedups over BiLSTM Model. As mentioned before, BiLSTM is the base of many RNN-based models with more complicated structures. So taking BiLSTM by itself into comparison, which reflects the lower-bound of speedups that BiGCNN can achieve. Specifically, we replace the bidirectional CNN module of BiGCNN with a BiLSTM module, and keep other modules and hyper-parameters the same. As shown in Fig. 4, BiGCNN gains speedups of 1.06 and 1.08 on the largest dataset MSRA during training and testing, respectively. Since the experiment is derived in an end-to-end manner, the time costs for calculating the embedding and CRF modules are also non-negligible. Therefore, such speedups are still noticeable. Moreover, the efficiency is more meaningful for large-scale NER applications, which can substantially improve overall system throughput.

3.5 Enhancing BiGCNN with Pre-trained Language Model

In this section, we verify the pre-trained language model (LM) can enhance the performance of BiGCNN. BERT [7] is taken as a representative as it is a powerful and most influential pre-trained LM currently. Glyce [29] is a pre-trained Chinese character representations and is used as a strong baseline.

We concatenate the Chinese BERT feature with our character embeddings and feed them into the bidirectional CNN module. The parameters of BERT are fine-tuned during training. As shown in Table 3, the BERT has already achieved a remarkable performance on the four datasets. When combining BiGCNN with BERT, the P/R/F1 scores on the four datasets improve significantly, and outperform Glyce. Especially, the performance increases obviously by 14.59% on Weibo dataset. All these results demonstrate that BERT can enhance the performance of BiGCNN and further verify the effectiveness of BiGCNN.

3.6 Detailed Analysis

Ablation Study. We consider the following variant models:

1. *BiGCNN w/o direction*, which adopts traditional convolution layers with unidirectional kernels. To guarantee the traditional CNN to have the same coverage as the bidirectional ones, we apply three different convolutions with kernel sizes as 1, 3, 5, respectively;
2. *BiGCNN w/o GCU*, which removes the convolution gate, the update gate, and the reset gate (Eq. 6-9);
3. *BiGCNN replace GCN w/ GLU*. The gated convolutional unit (GCU) is replaced with the gated linear unit (GLU) in [6]. For a fair comparison, we use multiple convolutions with kernel sizes the same as BiGCNN for GLU;
4. *BiGCNN replace glyph w/ radical*. The glyph embedding is replaced with the radical embedding [8] to verify its superiority. We split character c_i into a radical sequence[4] and feed it to an LSTM layer, where the last hidden state is referred as the radical embedding x_i^r. Then x_i^g is replaced with x_i^r in Eq. 1.

Except for the above changes in structure, other modules and experimental settings are kept the same as BiGCNN. The CRF module is also included in all these variants. Table 4 presents the comparison results, which suggests that:

- Bidirectionality plays a crucial role in modeling context information. The model without direction only uses a single CNN to extract both preceding and subsequent context simultaneously, and the degradation of performance is substantial on the four datasets (drops 1.61% on OntoNotes4). The result verifies that distinguishing contexts can be helpful for better performance.
- The proposed GCU is beneficial to the CNN-based structure. Regarding the model without the GCU, its performance drops significantly on the four datasets, which indicates that filtering out irrelevant information and retaining useful inputs is important to improve performance. Meanwhile, the result of *BiGCNN replace GCU w/ GLU* is lower than BiGCNN, indicating the superiority of GCU over GLU and the necessity of controlling the convolution input for the gating mechanism.
- Glyph embedding can further lead to a performance boost. As mentioned in Sect. 3.3, without the glyph embedding, the performance of BiGCNN drops

[4] https://github.com/kfcd/chaizi.

Table 5. An example in OntoNotes4 test set. Characters with blue and red text highlight the correct and incorrect labeled entities, respectively.

Sentence	这 一 计 划 得 到 了 英 国 威 康 公 司 的 大 力 支 持 The plan was strongly supported by Wellcome UK.
Segment	这/一/计划/得到/了/英国/威康/公司/的/大力/支持
BiGCNN	这 一 计 划 得 到 了 **[英 B-ORG]** **[国 I-ORG]** [威 I-ORG] [康 I-ORG] [公 I-ORG] [司 E-ORG] 的 大 力 支 持 (✓)
- direction	这 一 计 划 得 到 了 **[英 B-GPE]** **[国 E-GPE]** [威 B-ORG] [康 I-ORG] [公 I-ORG] [司 I-ORG] 的 大 力 支 持 (×)
- gate	这 一 计 划 得 到 了 **[英 B-GPE]** **[国 E-GPE]** [威 O] [康 O] [公 O] [司 O] 的 大 力 支 持 (×)

slightly. In addition, when replacing the glyph feature with the radical feature, there is still a performance gap with BiGCNN. We attribute it to that, splitting characters into radical sequences can not only lead to different characters sharing the same sequence (Sect. 1) but also the loss of structural information. Therefore, it may not be as effective as the glyph embedding.

Case Study. Table 5 shows an example in OntoNotes4 test set. In this case, the correct label of "英国 (*the UK*)", i.e., `ORG`, should be inferred from the subsequent context "威康公司 (*Wellcome*)". Otherwise, it could easily be mislabeled as Geo-Political Entities (`GPE`). Benefiting from the bidirectionality, BiGCNN can concentrate on the subsequent context, and predict the label correctly. The model without directional structures fuses the preceding/subsequent context and may somehow confuse the important information, and thus result in incorrect predictions. Additionally, the model without gating mechanism fails to distill the useful context clues, which also affects the recognition result.

4 Related Work

General NER Systems. Traditional NER systems are mostly based on statistical models with hand-crafted features [25]. To alleviate the heavy feature-engineering work, later studies applied recurrent neural networks to automatically extract features. For example, the BiLSTM is first introduced in [14] to capture word-level information. Paper [15,21] further integrated character-level features into the BiLSTM model. However, these methods only focused on learning context-independent representations. To enhance the generalization of learned features, some works combined the pre-trained language model with the RNN-based NER systems and gained considerable improvement [24]. More recently, BERT [7] is designed to pre-train bidirectional transformers and achieve state-of-the-art results on NER. Specifically, we also enhance BiGCNN with the pre-trained BERT and obtain further performance boost.

Chinese NER Systems. Compared with general NER, recognizing Chinese entities is more challenging as there are no word boundaries in Chinese text. Previous works proposed to preform segmentation first and then conduct word-level NER [12,23]. These methods are vulnerable to segmentation errors and thus later works are mostly based on character-level features. In particular, a position-sensitive model [20] is presented to train character representations. The character-level BiLSTM is used in [30] to extract context features. However, these models lack necessary word-level features. Recent studies introduced LatticeLSTM [32] and WC-LSTM [19] to integrated potential words information into character-level features, and gained greatly improvements. In addition, considering the typical structure of Chinese characters, radical information and historical glyph scripts/styles are further leveraged in [8,29]. Particularly, Glyce [29] collected extensive glyphs to pre-train character representations. We serve the glyph as a simple feature to better encode characters without any pre-training and manual work, which is totally different from Glyce.

CNN-based Networks for NER. As RNNs are limited in parallel processing, some works utilized CNNs to improve the computational efficiency for NER. ID-CNN [26] stacked layers of dilated convolutions to capture long-term context features. GRN [3] introduced a gated relation network to model the local contexts and the global relations in a sentence. LR-CNN [11] presented lexicon rethinking CNN to integrate lexicons and tackle conflicts between potential words. All of these methods are based on traditional CNNs, which use one convolution to model both preceding and subsequent context information simultaneously. Differently, we propose a novel bidirectional CNN structure using two independent convolutions to capture these two different contexts separately, which can better distinguish the entity-related information.

5 Conclusion

In this paper, we propose a bidirectional gated convolutional neural network (BiGCNN) for Chinese NER, which employs two independent CNNs to better differentiate the entity-related information between preceding and subsequent contexts. We also present an effective gated convolutional unit to control context information, and introduce additional glyph feature to further improve the representation of Chinese characters. Experimental results on four datasets demonstrate that BiGCNN outperforms both RNN-based and CNN-based models while enjoying a remarkable efficiency acceleration both in training and testing.

Acknowledgments. This work was supported in part by the National Natural Science Foundation of China (Grant Nos. U1636211, 61672081,61370126), the Beijing Advanced Innovation Center for Imaging Technology (Grant No. BAICIT-2016001), and the Fund of the State Key Laboratory of Software Development Environment (Grant No. SKLSDE-2019ZX-17).

References

1. Bai, S., Kolter, J.Z., Koltun, V.: An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. ArXiv (2018)
2. Cao, P., Chen, Y., Liu, K., Zhao, J., Liu, S.: Adversarial transfer learning for Chinese named entity recognition with self-attention mechanism. In: EMNLP, pp. 182–192 (2018)
3. Chen, H., Lin, Z., Ding, G., Lou, J., Zhang, Y., Karlsson, B.: Grn: gated relation network to enhance convolutional neural network for named entity recognition. In: AAAI, pp. 6236–6243 (2019)
4. Chen, Y., Xu, L., Liu, K., Zeng, D., Zhao, J.: Event extraction via dynamic multi-pooling convolutional neural networks. In: ACL, pp. 167–176 (2015)
5. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. In: NIPS Workshop (2014)
6. Dauphin, Y.N., Fan, A., Auli, M., Grangier, D.: Language modeling with gated convolutional networks. In: ICML , pp. 933–941 (2017)
7. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (2019)
8. Dong, C., Zhang, J., Zong, C., Hattori, M., Di, H.: Character-based LSTM-CRF with radical-level features for Chinese named entity recognition. In: Lin, C.-Y., Xue, N., Zhao, D., Huang, X., Feng, Y. (eds.) ICCPOL/NLPCC -2016. LNCS (LNAI), vol. 10102, pp. 239–250. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-50496-4_20
9. Fragkou, P.: Applying named entity recognition and co-reference resolution for segmenting English texts. Prog. Artif. Intell. **6**(4), 325–346 (2017)
10. Gehring, J., Auli, M., Grangier, D., Yarats, D., Dauphin, Y.N.: Convolutional sequence to sequence learning. In: ICML (2017)
11. Gui, T., Ma, R., Zhang, Q., Zhao, L., Jiang, Y.G., Huang, X.: CNN-based Chinese NER with lexicon rethinking. In: IJCAI (2019)
12. He, H., Sun, X.: F-score driven max margin neural network for named entity recognition in Chinese social media. In: EACL (2017)
13. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: ICCV (2015)
14. Huang, Z., Xu, W., Yu, K.: Bidirectional LSTM-CRF models for sequence tagging. ArXiv (2015)
15. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., Dyer, C.: Neural architectures for named entity recognition. In: NAACL-HLT (2016)
16. Levow, G.A.: The third international Chinese language processing bakeoff: word segmentation and named entity recognition. In: SIGHAN Workshop, pp. 108–117 (2006)
17. Li, S., Zhao, Z., Hu, R., Li, W., Liu, T., Du, X.: Analogical reasoning on Chinese morphological and semantic relations. In: ACL (2018)
18. Liu, L., et al.: Empower sequence labeling with task-aware neural language model. In: AAAI (2018)
19. Liu, W., Xu, T., Xu, Q., Song, J., Zu, Y.: An encoding strategy based word-character LSTM for Chinese NER. In: NAACL-HLT, pp. 2379–2389 (2019)
20. Lu, Y., Zhang, Y., Ji, D.H.: Multi-prototype Chinese character embedding. In: LREC, pp. 855–859 (2016)
21. Ma, X., Hovy, E.: End-to-end sequence labeling via bi-directional LSTM-CNNS-CRF. In: ACL (2016)

22. Peng, N., Dredze, M.: Named entity recognition for Chinese social media with jointly trained embeddings. In: EMNLP, pp. 548–554 (2015)
23. Peng, N., Dredze, M.: Improving named entity recognition for Chinese social media with word segmentation representation learning. In: ACL (2016)
24. Peters, M.E., et al.: Deep contextualized word representations. In: NAACL-HLT (2018)
25. Sekine, S., Nobata, C.: Definition, dictionaries and tagger for extended named entity hierarchy. In: LREC, Lisbon, Portugal, pp. 1977–1980 (2004)
26. Strubell, E., Verga, P., Belanger, D., McCallum, A.: Fast and accurate entity recognition with iterated dilated convolutions. In: EMNLP (2017)
27. Vaswani, A., et al.: Attention is all you need. In: NIPS (2017)
28. Weischedel, R., et al.: Ontonotes release 4.0. LDC2011T03 (2011)
29. Wu, W., et al.: Glyce: glyph-vectors for Chinese character representations. ArXiv (2019)
30. Yang, Y., Zhang, M., Chen, W., Zhang, W., Wang, H., Zhang, M.: Adversarial learning for Chinese NER from crowd annotations. In: AAAI (2018)
31. Zhang, L., Wang, H., Sun, X., Mansur, M.: Exploring representations from unlabeled data with co-training for Chinese word segmentation. In: EMNLP (2013)
32. Zhang, Y., Yang, J.: Chinese NER using lattice LSTM. In: ACL (2018)
33. Zheng, H., Li, Z., Wang, S., Yan, Z., Zhou, J.: Aggregating inter-sentence information to enhance relation extraction. In: AAAI (2016)
34. Zhu, Y., Wang, G.: Can-NER: convolutional attention network for Chinese named entity recognition. In: NAACL-HLT (2019)