IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received August 24, 2020, accepted October 7, 2020, date of publication October 14, 2020, date of current version October 27, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3031200

Natural Language Word Prediction Model Based on Multi-Window Convolution and Residual Network

JINGYUN YANG, HENGJUN WANG, AND KEXIANG GUO
Zhengzhou Information Science and Technology Institute, Zhengzhou 450001, China
Corresponding author: Jingyun Yang (yangjy_0611@163.com)

ABSTRACT In this paper, we propose MCNN-ReMGU model based on multi-window convolution and residual-connected minimal gated unit (MGU) network for the natural language word prediction. First, the convolution kernels with different sizes are used to extract the local feature information of different graininess between the word sequences. Then, the extracted features are fed to the residual-connected MGU network. Finally, the prediction results are output by the SoftMax layer. Through the residual-connection processing of MGU network in the model, not only the problems of vanishing gradient and network degradation are effectively solved, but also the long-term dependence between word sequences is effectively extracted to predict the next word accurately. Meanwhile, the introduction of the convolution kernel in a convolutional neural network (CNN) enables the feature information between word sequences to be extracted more fully. The experimental results on the Penn Treebank and WikiText-2 datasets show that the proposed method has certain advantages in the word prediction task.

INDEX TERMS Deep learning, convolution operation, residual connection, word prediction.

I. INTRODUCTION

Word prediction is one of the essential tasks of language models in the field of natural language processing (NLP), which uses a language model to capture the joint distribution of natural language word sequences. Some characters or words are typically given to predict the probability of the following characters or words. The traditional probability-based language models, such as the Hidden Markov, Conditional Random Field, and Decision Tree, are widely used to solve the word prediction problem [1]. With the advance of deep learning technology, language models based on neural networks have been widely adopted in the field of NLP. Recurrent neural network (RNN) [2] can capture the sequence connection between words within a sentence, and apply contextual semantic information to the current situation, thereby enhancing the semantic relevance within the sentence. However, useful historical information is weakened in processing the long-time information due to gradient dispersion [3]. To address this problem, Hochreiter and Schmidhuber [4] adopted a long short-term memory (LSTM) network, by adding order dependence between word sequences, to make full use of historical information. Also, the gated recurrent unit (GRU) network [5], [6] effectively solved the gradient dispersion problems in RNN, and GRU is a more simple structure and trained faster than LSTM [7]. The model based on minimal gated unit (MGU) [8], [9] is efficiently trained with fewer parameters, and a single forget gate, achieving comparable prediction accuracy comparable to other gate units [10].

As discussed above, the advance of the internal structure unit can improve the performance of RNN in natural language word prediction. However, deeper network or increased number of hidden nodes often induces the network degradation and over-fitting issues. Li *et al.* [11] proposed a robust independent recurrent neural network (IndRNN) where neurons in the same layer are set independently of each other, and neurons in different layers are cross-connected. This method enabled the model to be robust at a relatively large network depth and effectively overcame the problem of network degradation. Zilly *et al.* [12] proposed Recurrent Highway Network (RHN), which is an extension of the LSTM, to allow multiple hidden state updates at each time step. In RHN, the degradation of the weight matrix was alleviated so that the vanishing gradient was resolved. The residual network [13] addressed the gradient dissipation problem in the back-propagation process by utilizing the nature of the identity connection.

In order to reduce the over-fitting problem, two approaches: the dropout [14]–[16] operation and batch

The associate editor coordinating the review of this manuscript and approving it for publication was Gang Li.

normalization [17]–[19] are widely used. In the training process, the dropout randomly prevented some neurons from participating in the training and weakened the joint adaptability between the neuron nodes to avoid over-fitting. The batch normalization normalized the input data to ensure that the data distribution remains unchanged and improved the generalization ability of the model.

The convolutional neural network (CNN) [20] consists of deep-stacked convolutional layers, which has a strong processing ability for local information of data. CNN has been successfully adopted in many computer vision applications. In this paper, utilizing the ability of local information processing of CNN and the ability of the word sequences feature extracting of residual-connected MGU network, multi-window convolution and residual-connected MGU network (MCNN-ReMGU) is proposed. First, the convolution kernels extract the local feature relationships between the word sequences. Then the residual-connected MGU network fully learns the long dependency relationship between the word sequences. Thus, both global and local feature information of the word sequence is thoroughly used in word prediction. Also, L2-norm [21], [22] and batch normalization are employed to avoid the over-fitting of the network.

In this paper, we employed the high-dimensional feature extraction capability of convolution kernels to fully obtain the depth feature information between word sequences, At the same time, we carried on the residual connection processing to the MGU network to make the network more sensitive to gradient changes, and solved the network degradation problem caused by deep network depth and finally proposed the MCNN-ReMGU model.

The main contributions of our work and the innovation of this paper are as follows:

1. In order to fully mine the local feature information between word sequences and improve the prediction accuracy of the model, we introduced the convolution kernel of CNN into the language model of sequence prediction task, and checked the convolution operation of adjacent word sequences by using convolution of different window sizes, thus enhancing the influence of adjacent word sequences on the prediction target.
2. To solve the problems of vanishing gradient and network degradation caused by the increase in the number of network layers, we conducted residual connection based on MGU unit, so that the model could carry out more in-depth training when the network depth was relatively deep, and proved this theoretically.
3. In this paper, the MCNN-ReMGU model can effectively obtain the global and local feature information between the word sequences, and deeply excavate the word sequence connection in sentences, thus improving the word prediction ability of the model.
4. The research results of this paper can provide technical support for automatic writing, text content input and other tasks.

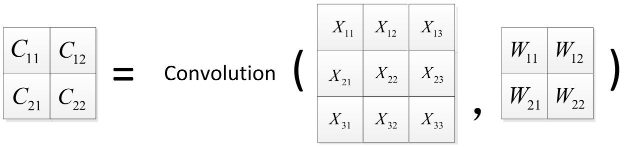


FIGURE 1. Convolution operation.

II. PRELIMINARIES

In conventional methods, the hand-crafted features were usually used to extract the characteristics between word sequences. Recently, neural network models (such as CNN and RNN) have been increasingly adopted with the rapid advance of deep learning, which can automatically extract features from data, and have successfully extracted features regardless of that is local information or temporal information.

A. CNN

A deep CNN can effectively extract the feature information from locally-adjacent words by its inherent ability in the local feature learning [23], [24]. A typical CNN is composed of an input layer, convolution layer, pooling layer, and fully connected layer. In NLP, the input layer of CNN is a vector representation of words defined as:

$$E \in R^{n\times m}, \tag{1}$$

where E is the vector representation of words of the input layer of CNN, m is the dimension of the word vector, and n is the length of the given sentence.

The nodes of the convolution layer are only connected to a local input, rather than fully connected to each input point. Convolution layer convolves the input matrix with a different size of kernels, extracting local features of input data:

$$\mathbf{c} = f(W \otimes x + b), \tag{2}$$

where $\mathbf{c}$ is the text feature vector, x is the word embedding matrix. $\mathbf{W}$ is the weight matrix, and b is the offset. f is the activation function and $\otimes$ indicates the convolution operation. Taking Fig. 1 as an example, the specific convolution operation process is:

$$\mathbf{C_{11}} = \mathbf{W_{11}X_{11}}+\mathbf{W_{12}X_{12}}+\mathbf{W_{21}X_{21}}+\mathbf{W_{22}X_{22}}, \tag{3}$$

$$\mathbf{C}_{12} = \mathbf{W_{11}X}_{12}+\mathbf{W_{12}X}_{13}+\mathbf{W_{21}X_{22}}+\mathbf{W_{22}X}_{23}, \tag{4}$$

$$\mathbf{C_{21}} = \mathbf{W_{11}X_{21}}+\mathbf{W_{12}X_{22}}+\mathbf{W_{21}X}_{31}+\mathbf{W_{22}X}_{32}, \tag{5}$$

$$\mathbf{C}_{22} = \mathbf{W_{11}X}_{22}+\mathbf{W_{12}X}_{23}+\mathbf{W_{21}X}_{32}+\mathbf{W_{22}X}_{33}, \tag{6}$$

The function of the pooling layer is to compress the information from the previous layer. The pooling layer takes the feature vector obtained by the convolution layer, then extracts locally more important feature information through the down-sampling process. The fully connected layer produces the output of the entire network, taking the output of the pooling layer as an input.

B. RESIDUAL NETWORK

The residual network [25], [26] can effectively solve the problem of network degradation. It principally transforms the fitted identity mapping function $H(x) = x$ into the optimized

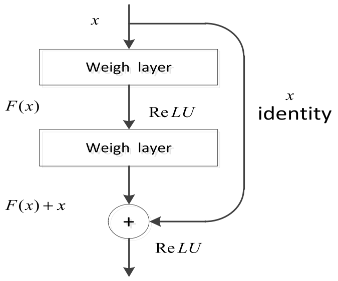


FIGURE 2. A diagram of the residual network.

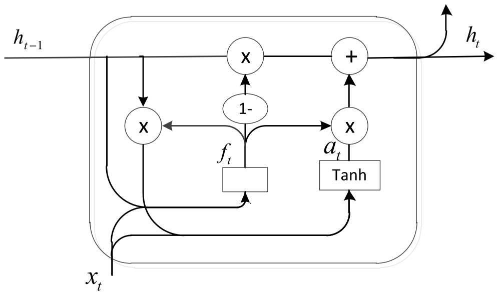


FIGURE 3. MGU structure diagram.

residual function $F(x) = H(x) - x$. When $F(x) = 0$ in the residual function constitutes the identity mapping of $H(x) = x$. That is, the network identity of this layer maps the input of the previous layer, thereby avoiding the redundancy generated by the redundant network layer and not increasing the number of network parameters. The diagram of the residual network is shown in Fig. 2.

C. MINIMAL GATED UNIT

MGU is an effective simplified variant of the gated unit, which is inspired by the GRU structure. Because the forget gate is the most essential and indispensable [5], [27], [28], only the forget gate structure is employed in MGU, unlike the other recurrent neural networks such as LSTM and GRU, as shown in Fig. 3.

Similarly to GRU, MGU merges reset gate and update gate, thus modifying the calculation method of hidden state in the recurrent neural network as follows:

$$f_t = \sigma(\mathbf{W}_f \cdot x_t + \mathbf{U}_f \cdot h_{t-1} + b_f), \tag{7}$$

$$a_t = \tanh(\mathbf{W}_a \cdot x_t + \mathbf{U}_a(h_{t-1} \cdot f_t) + b_a), \tag{8}$$

$$h_t = (1 - f_t) \cdot h_{t-1} + f_t \cdot a_t, \tag{9}$$

where x_t represents the input value of the current layer at time t. h_t and h_{t-1} are the state vector at the time t and $t-1$, respectively. σ is the sigmoid function. a_t is the candidate hidden state at time t, that is the output value of the forgetting gate. tanh is the hyperbolic tangent activation function of the candidate hidden state. $\mathbf{W}_f$, $\mathbf{W}_a$, $\mathbf{U}_f$ and $\mathbf{U}_a$ are the weight parameter matrices; b_f and b_a are the offset vectors.

III. MCNN-REMGU MODEL

The proposed MCNN-ReMGU model mainly consists of four parts: word embedding layer, CNN local perception layer, residual connection MGU network layer, and softmax output layer. The overall structure framework is shown in Fig. 4.

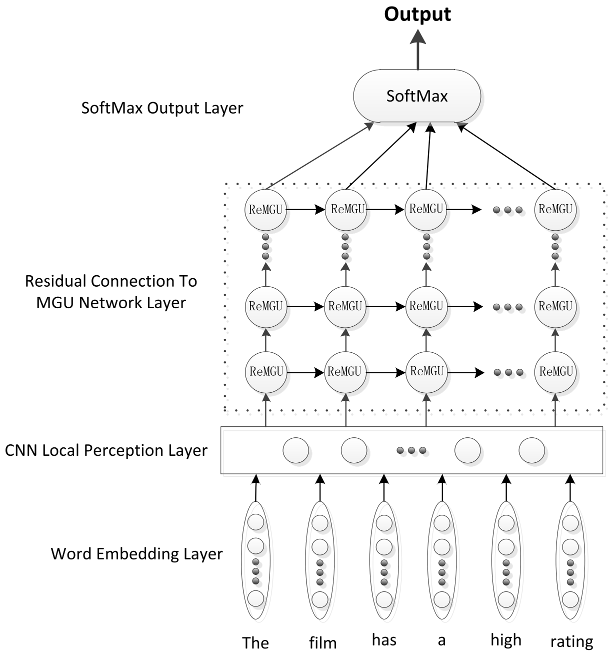


FIGURE 4. Model framework.

A. WORD EMBEDDING LAYER

Word embedding [29] is a distributed expression of words, which maps words from a high-dimensional sparse space to a relatively low-dimensional real vector space. In the input layer, each word is first represented by a real vector through the word embedding process. With the word embedding, a given sentence $\mathbf{S} = [x_1, \ldots, x_i, \ldots, x_n]$ where x_i is the corresponding word in the sentence is expressed as follows:

$$S_\omega = [\omega_1, \ldots, \omega_i, \ldots \omega_n] \in \mathrm{R}^{n \times d}, \tag{10}$$

where n is the number of words in the sentence, d is the dimension of the word vector, and ω_i is the vector representation of x_i in the sentence.

B. CNN LOCAL PERCEPTION LAYER

CNN extracts sequence feature information with different granularity by using different window sizes of convolution kernels [30]. In the CNN local perception layer of this paper, multiple $h \times d$ convolution kernels (h is the window size of the convolution kernel and$d$ is the dimension of the embedded vector) are used to conduct convolution operation on the sentence $\mathbf{S} = [x_1, \ldots, x_n] \in R^{n \times d}$, and the local features of the data are mined to achieve the purpose of feature enhancement and reduction of model calculation parameters, as shown in Fig. 5. The batch normalization is applied after each convolution operation to prevent the transfer of covariates within the data. The batch normalization is defined as follows:

$$\mu_B = \frac{1}{n} \sum\nolimits_{i=1}^{n} x_i, \tag{11}$$

$$\sigma_B^2 = \frac{1}{n} \sum\nolimits_{i=1}^{n} (x_i - \mu_B)^2, \tag{12}$$

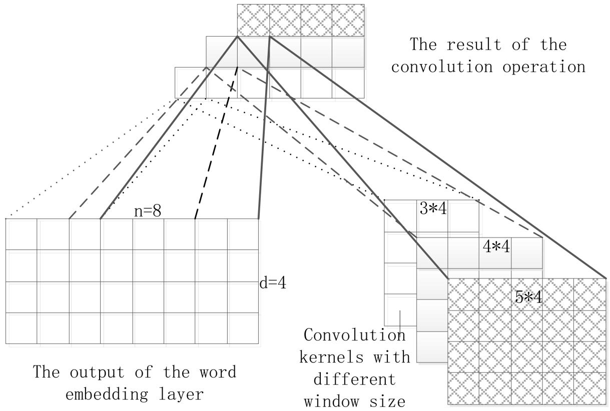

The result of the convolution operation
n=8
d=4
3*4
4*4
5*4
Convolution kernels with different window size
The output of the word embedding layer

FIGURE 5. CNN local perception layer structure diagram.

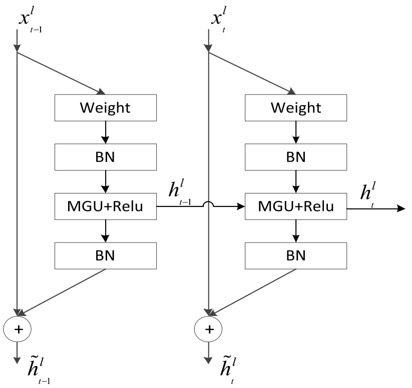


FIGURE 6. ReMGU structure.

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}}, \tag{13}$$

$$BN_{\gamma,\beta}(x_i) = \gamma \hat{x}_i + \beta, \tag{14}$$

where B is a small batch of data of size n as $B = \{x_1, x_2, \ldots, x_n\}$.$\mu$ and σ_B^2 are mean and variance respectively. γ and β are the parameters to be learned. Since the batch normalization process can eliminate the deviation, the offset vector in (2) is removed, that is as:

$$m_i = f(BN(\mathbf{W}_c \otimes x_{i:i+h-1})), \tag{15}$$

where m_i represents the i^{th} feature obtained by the convolution operation. BN and $\otimes$ indicate the batch normalization and convolution operations, respectively. f represents the non-linear function ReLU. $\mathbf{W}_c$ represents the weight matrix of the convolution, which is a feature set obtained by a convolution operation on the input feature x with a convolution kernel of window size h. Note that the beginning of the word sequences are filled with the zero vector to avoid introducing feature information of predicted words.

Then, connect the result obtained by sentence convolution to obtain the output of the convolution layer, that is as:

$$\mathbf{M} = [m_1, m_2, \ldots, m_n]. \tag{16}$$

The output size of this layer is $n \times r$ after r convolution operations due to the zero-padding in the word sequences.

TABLE 1. Parameters setting.

parameter	number
hidden units	{150, 300, 600, 900}
filter sizes	{3, 4, 5}
filter numbers	{150(50, 50, 50). 300(100, 100, 100). 600(200, 200, 200). 900(300, 300, 300)}
Layers	{1, 3, 5, 7}
batch sizes	20
number steps	35
dropout	0.5
learning rates	0.25
regularization coefficient λ	0.001

C. RESIDUAL CONNECTION TO MGU NETWORK LAYER

As mentioned earlier, when the depth of the network is increased without special restrictions, redundant network layers will be generated, which eventually leads to poor network performance. In order to solve the problem of network degradation, this paper adopts the residual connection to the original MGU. Also, the activation function of the candidate hidden state in (8) is modified to the ReLU activation function, which can avoid the vanishing gradient caused by the saturation function so that the deeper network can be trained [31]. The ReLU function is defined as:

$$ReLU(x) = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases} \tag{17}$$

The derivative of ReLU is 0 when $x < 0$, whereas it is a constant: 1 when $x > 0$ so that the back-propagated gradient does not vanish.

Batch normalization processes the weighted input x_i and the output h_t of MGU for each layer. So the improved ReMGU structure is shown in Fig. 6:

For a given input $x_1, x_2, \ldots, x_n$, at time, the ReMGU is computed as follows:

$$f_t^l = \sigma(BN(\mathbf{W}_f^l \cdot x_t^l) + \mathbf{U}_f^l \cdot h_{t-1}^l), \tag{18}$$

$$a_t^l = ReLU(BN(\mathbf{W}_a^l \cdot x_t^l) + \mathbf{U}_a^l(h_{t-1}^l \cdot f_t^l)), \tag{19}$$

$$h_t^l = (1 - f_t^l) \cdot h_{t-1}^l + f_t^l \cdot a_t^l, \tag{20}$$

$$\tilde{h}_t^l = BN(h_t^l) + x_t^l, \tag{21}$$

where a_t^l represents the candidate hidden state at time t of the l^{th} layer. $\tilde{h}_t^l$ is the state vector passed to the next layer after the residual connection processing of the l^{th} layer, that is, the input of the $l + 1$ layer at time t. h_t^l is only used as the state vector transmitted to the l layer at time $t + 1$.

TABLE 2. Test results of different models on PTB dataset. The gray background indicates the best among the methods with the same hidden unit and the same number of network layers. The bold indicates the best among RNN, LSTM, GRU, and MGU for the same parameters, comparing the performance between MGU-based and other gate-units-based neural networks.

Model	Layers	1		3		5		7	
	Hidden Units	Valid	Test	Valid	Test	Valid	Test	Valid	Test
RNN	150	164.3	163.2	155.4	154.1	175.7	174.1	199.1	198.6
LSTM		**111.9**	**114.0**	**105.8**	**103.9**	114.0	113.3	132.4	131.2
GRU		112.1	114.4	105.9	104.3	**113.5**	**112.8**	**131.8**	**131.0**
MGU		112.6	114.9	106.2	104.6	113.7	113.0	132.2	131.4
ReMGU		110.1	109.3	104.3	103.1	103.0	101.7	101.3	100.4
MCNN-MGU		101.5	100.4	96.3	94.9	103.3	102.6	109.0	107.5
MCNN-ReMGU		95.2	94.7	89.6	88.7	89.3	88.8	86.8	86.2
RNN	300	144.4	143.7	137.1	136.5	158.7	158.0	172.0	171.3
LSTM		100.9	100.3	**94.4**	**93.6**	105.8	105.2	123.3	122.0
GRU		100.6	99.9	95.1	94.2	104.5	103.9	**122.6**	**121.2**
MGU		**100.5**	**99.7**	94.8	94.3	**104.4**	**103.7**	122.8	121.6
ReMGU		98.1	97.4	93.4	92.5	91.5	90.7	89.7	89.1
MCNN-MGU		84.5	83.7	80.7	80.2	89.4	88.9	96.9	96.2
MCNN-ReMGU		82.7	81.5	79.5	78.3	77.8	76.7	76.2	75.2
RNN	600	137.2	136.3	133.7	133.1	151.3	150.6	180.5	179.2
LSTM		88.6	87.7	**80.6**	**80.4**	93.4	92.6	109.0	108.3
GRU		88.9	88.1	81.5	81.0	92.7	91.8	**108.6**	**107.7**
MGU		**88.3**	**87.6**	81.4	80.8	**92.3**	**91.5**	109.2	108.0
ReMGU		86.4	85.5	80.0	79.2	78.9	78.0	77.7	76.7
MCNN-MGU		71.8	70.4	67.2	66.2	73.9	73.1	78.4	77.7
MCNN-ReMGU		70.6	68.7	65.2	63.7	60.8	59.3	58.1	56.6
RNN	900	134.9	134.0	133.1	131.9	148.3	147.5	177.4	176.6
LSTM		**80.6**	**79.7**	**77.0**	**76.2**	88.2	87.5	102.8	102.1
GRU		81.4	80.2	77.4	76.5	**88.1**	**87.4**	102.2	101.4
MGU		81.2	80.8	77.1	76.4	88.4	87.7	**101.7**	**101.0**
ReMGU		79.7	79.1	75.7	75.5	74.4	73.5	72.8	71.9
MCNN-MGU		77.3	76.7	75.1	74.3	80.3	79.5	85.2	84.4
MCNN-ReMGU		75.9	74.4	73.7	72.6	72.0	71.2	70.4	69.3

relationship between the word sequences in the sentence

The offset vector b in (7) and (8) are removed due to the batch normalization.

Here, only the back-propagation (BP) between ReMGU layers are discussed because the ReMGU is a residual network based on the upper and lower layers of the network. Assuming that l is the current number of network layers, L be a layer deeper than the l, and several network layers exist between the L layer and the layer, (22) is obtained from (21) as follows:

$$\begin{aligned} x_t^L &= \tilde{h}_t^{L-1} \\ &= BN(h_t^{L-1}) + x_t^{L-1} \\ &= x_t^l + \sum\nolimits_{i=l}^{L-1} BN(h_t^l). \end{aligned} \tag{22}$$

Therefore, the partial derivative of x_t^L with respect to x_t^l can be expanded as follows:

$$\begin{aligned} \frac{\partial x_t^L}{\partial x_t^l} &= \frac{\partial (x_t^l + \sum_{i=l}^{L-1} BN(h_t^l))}{\partial x_t^l} \\ &= 1 + \frac{\partial}{\partial x_t^l} \sum\nolimits_{i=l}^{L-1} BN(h_t^l). \end{aligned} \tag{23}$$

According to the chain rule, the gradient of the loss function ε with respect to can be expressed as:

$$\begin{aligned} \frac{\partial \varepsilon}{\partial x_t^l} &= \frac{\partial \varepsilon}{\partial x_t^L} \frac{\partial x_t^L}{\partial x_t^l} \\ &= \frac{\partial \varepsilon}{\partial x_t^L} (1 + \frac{\partial}{\partial x_t^l} \sum\nolimits_{i=l}^{L-1} BN(h_t^l)). \end{aligned} \tag{24}$$

During the entire training process, $\frac{\partial}{\partial x_t^l} \sum_{i=l}^{L-1} BN(h_t^l)$ cannot always be -1 [32], and thus the vanishing gradient caused by the continuous multiplication of multiple layers between the L layer and the l layer can be avoided. The ReMGU designed in this paper is more sensitive to the gradient change of deep network, which is conducive to effective information transmission between network layers and thus solving network degradation.

D. OUTPUT LAYER

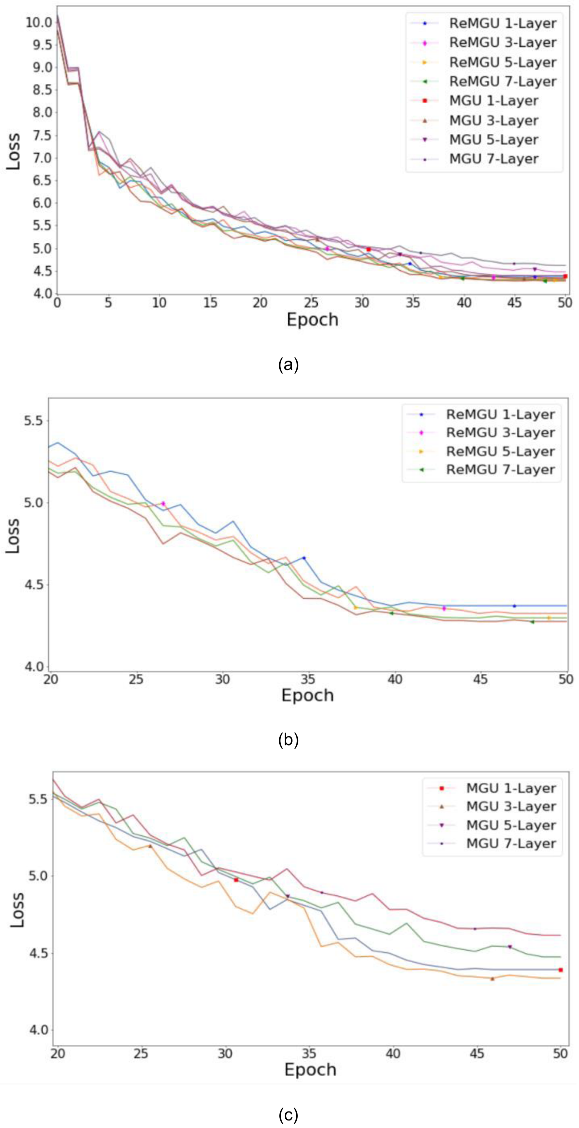
The softmax function provides the normalized values for the final word prediction, where the output $\tilde{h}$ from the ReMGU is taken as an input:

$$\hat{y} = \mathrm{softmax}(\mathbf{W}_s \tilde{h} + b_s). \tag{25}$$

where $\mathbf{W}_s$ and b_s are weight matrix and bias vector, respectively. In this paper, cross-entropy with L2 norm is adopted as the loss function, defined as:

$$Loss = -\sum\nolimits_{i=1}^{D} \sum\nolimits_{k=1}^{C} y_i^k \log \hat{y}_i^k + \lambda ||\theta||^2. \tag{26}$$

where D and C are the size of the training datasets and the number of categories of data, respectively. y_i and $\hat{y}_i$ are the actual and predicted categories, respectively. $\lambda||\theta||^2$ is the regularization-term to avoid the over-fitting and negative migration, where λ and θ are the regularization coefficient and model parameter.



(a)

(b)

(c)

FIGURE 7. (a) the complete variation curve, (b) partially enlarged diagram of ReMGU and (c) partially enlarged diagram of MGU.

IV. EXPERIMENTAL RESULTS

This section validates the predictive performance of the proposed method. First, we analyze the impacts of multi-window convolutions and residual connection MGU network on the prediction. Then, the proposed method is comprehensively compared with compared methods.

A. EXPERIMENTS DETAILS

The proposed MCNN-REMGU model is implemented in the Tensorflow 1.14 framework. Specifically, several neural networks based on different gate units: RNN, LSTM, GRU, MGU, and the proposed ReMGU are implemented. Also, the proposed MCNN-MGU is implemented. They were analyzed and compared on the Penn Treebank (PTB) dataset [33]. Further, the performance of the proposed method is compared with the state-of-the-art methods [11]–[16], [34] on the PTB and WikiText-2 (WT2) datasets. Lastly, further ablation studies on the L2-norm and batch normalization are conducted to investigate the effect of corresponding regularization measures on model over-fitting.

TABLE 3. The compared results on the PTB dataset.

Model	Parameters	Valid	Test
Zaremba et al. [14] (2014)–LSTM (medium)	20M	86.2	82.7
Zaremba et al. [14] (2014)–LSTM (large)	66M	82.2	78.4
Zilly et al. [12] (2016)–Variational RHN (tied)	23M	67.9	65.4
Li et al. [11] (2017)–res-IndRNN (11 layers)	22M	66.5	65.3
Melis et al. [15] (2017)–4-layer skip connection LSTM (tied)	24M	60.9	58.9
Merity et al. [16] (2018)–AWD-LSTM	24M	60.7	58.8
Dai et al. [34] (2019) Transformer-XL	24M	58.6	56.9
Ours–MCNN-ReMGU (600-hidden units,7-layer)	23M	58.1	56.6

TABLE 4. The compared results on the WT2 dataset.

Model	Parameters	Valid	Test
Zaremba et al. [14] (2014)–LSTM (medium)	20M	94.7	92.1
Zaremba et al. [14] (2014)–LSTM (large)	66M	88.9	86.2
Li et al. [12] (2017)–res-IndRNN (11 layers)	22M	73.2	71.4
Melis et al. [15] (2017)–2-layer skip connection LSTM (tied)	24M	69.1	65.9
Merity et al. [16] (2018)–AWD-LSTM	24M	67.4	65.3
Dai et al. [34] (2019) Transformer-XL	24M	66.7	64.8
Ours MCNN-ReMGU (1200-hidden units,3-layer)	26M	64.4	63.2

B. PARAMETERS SETTING

After a lot of manual attempts and the parameter setting standards recommended in reference [14], we set the experimental parameters in this paper as follows. The sizes of convolutional kernels are 3, 4, and 5, and their filters number are evenly distributed according to hidden units, whose sum is equal to the word vector dimension. To evaluate the performance of the language model with different network depths and different hidden units, we selected them from sets {1, 3, 5, 7} and {150, 300, 600, 900}, respectively. Also, random sampling and batch processing are conducted on experimental data. The comparison model also adopts the dropout mechanism, and all non-hyper parameters are randomly initialized and adjusted as the network is trained. The detailed parameter settings are summarized in Table 1.

C. EXPERIMENTAL RESULTS AND ANALYSIS

In this paper, perplexity (PPL) is used as the evaluation index for the language models. Generally, a lower perplexity in the probability distribution model indicates better sample prediction of the language model [35]. The experimental results of different models on PTB dataset are shown in Table 2.

TABLE 5. Ablation study for different regularization methods: L2-norm and batch normalization.

Model	Valid	Test
MCNN-ReMGU (W/O L2 norm)	70.2	67.8
MCNN-ReMGU (W/O batch normalization)	85.6	84.1
MCNN-ReMGU	65.2	63.7

Table 2 shows that the proposed MCNN-ReMGU gives the lowest PPL value compared to the models having the same number of hidden units and the same network depth. Moreover, for the same hidden unit condition, the PPL value of the model proposed in this paper decreases as the number of network layers increases.

The following conclusions can be drawn from Table 2:

1. All the neural network models based on RNN, LSTM, GRU, and MGU units can obtain the long-term dependence between word sequences, but the networks outperform based on LSTM, GRU, and MGU are better than the traditional RNN. Also, the prediction accuracy of the network based on MGU is competitive to that of LSTM and GRU networks, which verifies the conclusion of reference [10].

2. ReMGU and MCNN-ReMGU provide better prediction accuracy than MGU and MCNN-MGU, respectively, showing that the residual connection can improve the prediction performance.

3. MCNN-MGU and MCNN-ReMGU provide better prediction accuracy than MGU and ReMGU, respectively. The results show that the convolution operations of multiple sizes can fully extract and learn the characteristic relationship between the word sequences in the sentence (especially the word sequences that are close to the predicted word position), thereby improving the predictive ability of the model.

4. As the number of layers in the network increases up to 3, the PPL values are decreased, indicating that properly increasing the network depth for deep learning can improve the learning ability of the model. However, when the network depth is further deepened, the PPL values of the neural network models (such as RNN, LSTM, GRU, and MGU) that have not undergone residual processing increase. But the PPL value of the neural network models connected through the residual (such as ReMGU and MCNN-ReMGU) further decreases, indicating that the residual connection can effectively solve the problem of the vanishing gradient and network degradation of a multi-layer neural network, thus enabling deeper learning of the model.

5. As the number of hidden units increases, the PPL value also decreases, indicating that adding hidden units at the network layer can enrich the network structure to enhance the learning ability of the network. However, when the number of hidden units reaches 900, the PPL values of MCMM-MGU and MCNN-ReMGU increase. This is because the increase in model complexity causes the over-fitting problem, thereby deteriorating prediction accuracy.

Fig. 7 shows the training losses in the whole epochs and part of the epochs period for ReMGU and MGU. As shown in Fig. 7, when the network depth reaches three layers, the loss value of MGU increases as the network depth deepens, whereas the loss value of ReMGU continues to decrease. At the same time, ReMGU has lower loss values than MGU, and the convergence of ReMGU is faster, indicating that the GRU network after the residual processing effectively solves the problem of vanishing gradient and network degradation so that the network is adequately trained.

Tables 3 and 4 compare the performance of the proposed method with the state-of-the-art methods on PTB and WT2 datasets. As shown in Tables 3 and 4, the proposed MCNN-ReMGU provides better or competitive performance (lower PPL values), indicating that the inroduction of residual-connected MGU network and multi-window convolution kernels can fully extract the features between word sequences, so that the model can predict the results more accurately. Note that the test results of the proposed MCNN-ReMGU on the WT2 and PTB datasets are better than those of Transformer-XL [34], and the advantage on the WT2 dataset is more obvious. This is because the size of the WT2 dataset is twice that of PTB, so a large number of 1,200-hidden units and 3-layers are used in the MCNN-ReMGU model based on a larger dataset, which improves the prediction performance without over-fitting. At the same time, it can be seen from Table 4 that a large number of hidden units can enhance the learning ability of the neural network model, so as to more fully extract the feature information between word sequences.

The ablation study for L2-norm and batch normalization is conducted. The network with 600 hidden units and three layers are analyzed on the PTB dataset. As shown in Table 5, significant performance improvements are achieved by both employing L2-norm and batch normalization.

V. CONCLUSION

For the prediction task of natural language words, this paper proposes the MCNN-ReMGU model based on multi-window convolution and residual-connected MGU network combined with data regularization technology. Based on the PTB dataset, we verify the effectiveness of multi-window convolution and residual-connected MGU network in extracting high-dimensional features between locally adjacent words and feature information between word sequences, respectively. At the same time, it is found through experiments that the residual connection to the MGU network not only addresses the vanishing gradient problem and network degradation but also fully learns the long dependence relationship between word sequences. Also, L2-norm and batch normalization are employed to alleviate the over-fitting effectively. The overall experimental results show that the proposed MCNN-ReMGU significantly improves the performance of the word prediction task over the traditional methods. Also, the proposed method provides competitive performance to state-of-the-art methods.

REFERENCES

[1] Y. K. Xing and S. P. Ma, "A survey on statistical language models," *Comput. Sci.*, vol. 30, no. 9, pp. 22–26, 2003.

[2] O. Irsoy and C. Claire, "Opinion mining with deep recurrent neural networks," in *Proc. Empirical Methods Natural Lang. Process.*, 2014, pp. 720–728.

[3] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 157–166, Mar. 1994.

[4] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[5] J. Chung, C. Gulcehre, K. H. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," Dec. 2014, *arXiv:1412.3555*. [Online]. Available: http://arxiv.org/abs/1412.3555

[6] L. Wang, L. Liu, L. Niu, F. Y. Hu, and T. Peng, "Relation extraction method based on relation trigger words and single-layer GRU model," *J. Jilin Univ., Sci. Ed.*, vol. 58, no. 1, pp. 95–103, 2020.

[7] W. Wang, Y. Sun, Q. Qi, and X. Meng, "Text sentiment classification model based on BiGRU-attention neural network," *Appl. Res. Comput.*, pp. 3558–3564, 2019, doi: 10.19734/j.issn.1001-3695.2018.07.0413.

[8] G.-B. Zhou, J. Wu, C.-L. Zhang, and Z.-H. Zhou, "Minimal gated unit for recurrent neural networks," *Int. J. Autom. Comput.*, vol. 13, no. 3, pp. 226–234, Jun. 2016.

[9] A. Dong, Z. Du, and Z. Yan, "Round trip time prediction using recurrent neural networks with minimal gated unit," *IEEE Commun. Lett.*, vol. 23, no. 4, pp. 584–587, Apr. 2019.

[10] J. X. Liu and S. C. Chen, "Non-stationary multivariate time series prediction with MIX gated unit," *J. Comput. Res. Develop.*, vol. 56, no. 8, pp. 1642–1651, 2019.

[11] S. Li, W. Li, C. Cook, C. Zhu, and Y. Gao, "Independently recurrent neural network (IndRNN): Building a longer and deeper RNN," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5457–5466.

[12] J. G. Zilly, R. K. Srivastava, J. Koutník, and J. Schmidhuber, "Recurrent highway networks," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2016, pp. 4189–4198.

[13] S. Jastrzębski, D. Arpit, N. Ballas, V. Verma, T. Che, and Y. Bengio, "Residual connections encourage iterative inference," in *Proc. Int. Conf. Learn. Represent.*, 2017, pp. 1–14.

[14] Z. Wojciech, I. Sutskever, and O. Vinyals, "Recurrent neural network regularization," 2014, *arXiv:1409.2329*. [Online]. Available: https://arxiv.org/abs/1409.2329

[15] G. Melis, C. Dyer, and P. Blunsom, "On the state of the art of evaluation in neural language models," in *Proc. ICLR Conf. Blind Submission*, 2018, pp. 1–10.

[16] M. Stephen, N. S. Keskar, and R. Socher, "Regularizing and optimizing LSTM language models," in *Proc. 5th Int. Conf. Learn. Represent. (ICLR)*, 2017, pp. 1–10.

[17] T. Cooijmans, N. Ballas, C. Laurent, Ç. Gülçehre, and A. Courville, "Recurrent batch normalization," in *Proc. 5th Int. Conf. Learn. Represent. (ICLR)*, 2016, pp. 1–13.

[18] J. Bjorck, C. P. Gomes, B. Selman, and K. Q. Weinberger, "Understanding batch normalization," in *Proc. Neural Inf. Process. Syst.*, Montreal, QC, Canada, 2018, pp. 7694–7705.

[19] G. Yang, J. Pennington, V. Rao, J. Sohl-Dickstein, and S. S. Schoenholz, "A mean field theory of batch normalization," in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–95.

[20] Y. Kim, "Convolutional neural networks for sentence classification," in *Proc. Empirical Methods Natural Lang. Process.*, 2014, pp. 1746–1751.

[21] Z. Dai, W. Chen, X. Huang, B. Li, L. Zhu, L. He, Y. Guan, and H. Zhang, "CNN descriptor improvement based on L2-normalization and feature pooling for patch classification," in *Proc. IEEE Int. Conf. Robot. Biomimetics (ROBIO)*, Dec. 2018, pp. 144–149.

[22] J. Wang, Y. Li, Z. Miao, X. Zhao, and Z. Rui, "Multi-level metric learning network for fine-grained classification," *IEEE Access*, vol. 7, pp. 166390–166397, 2019.

[23] R. Song, X. Chen, Y. Hong, and M. Zhang, "Combination of convolutional recurrent neural network," *J. Chin. Inf. Process.*, vol. 33, no. 10, pp. 64–72, 2019.

[24] D. Cheng, Y. Gong, S. Zhou, J. Wang, and N. Zheng, "Person re-identification by multi-channel parts-based CNN with improved triplet loss function," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1335–1344.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[26] X. T. Chen and T. Lu, "Efficient face recognition algorithm using global deep separable convolutional and residual network," *J. Wuhan Inst. Technol.*, vol. 41, no. 3, pp. 276–282, 2019.

[27] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2222–2232, Oct. 2017.

[28] J. Rafal, Z. Wojciech, and S. Ilya, "An empirical exploration of recurrent network architectures," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 2342–2350.

[29] C. Ronan and W. Jason, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 160–167.

[30] H. J. Yuan, X. Zhang, W. H. Niu, and K. B. Cui, "Sentiment analysis based on multi-channel convolution and bi-directional GRU with attention mechanism," *J. Chin. Inf. Process.*, vol. 33, no. 10, pp. 109–118, 2019.

[31] G. Xavier and B. Yoshua, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.

[32] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 630–645.

[33] M. P. Marcus and M. A. Marcinkiewicz, and B. Santorini, "Building a large annotated corpus of English: The penn treebank," *Comput. Linguistics*, vol. 19, no. 2, pp. 313–330, 1993.

[34] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. Le, and R. Salakhutdinov, "Transformer-XL: Attentive language models beyond a fixed-length context," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 2978–2988.

[35] T. Mikolov and G. Zweig, "Context dependent recurrent neural network language model," in *Proc. IEEE Spoken Lang. Technol. Workshop (SLT)*, Miami, FL, USA, Dec. 2012, pp. 234–239.

JINGYUN YANG was born in Liaoning, China, in 1992. He received the B.S. degree in electronic engineering from the Zhengzhou Information Science and Technology Institute, China, in 2015, where he is currently pursuing the M.D. degree in computer science and technology. His research interests include artificial intelligence and natural language processing.

HENGJUN WANG was born in Hunan, China, in 1973. He received the Ph.D. degree from the Zhengzhou Information Science and Technology Institute, in 2007. He is currently an Associate Professor with the Zhengzhou Information Science and Technology Institute. His research interests include intelligent information processing, natural language processing, and machine learning.

KEXIANG GUO was born in Fujian, China, in 1992. He received the B.S. degree in computer science and technology from the Xi'an University of Technology, China, in 2015. He is currently pursuing the M.D. degree in computer technology with the Zhengzhou Information Science and Technology Institute, China. His research interests include natural language processing and deep learning.