Available online at www.ijpe-online.com

**International Journal of Performability Engineering**

vol. 14, no. 7, July 2018, pp. 1580-1589
DOI: 10.23940/ijpe.18.07.p22.15801589

# A Mongolian Language Model based on Recurrent Neural Networks

Zhiqiang Ma, Li Zhang, Rui Yang*, and Tuya Li

*College of Data Science and Application, Inner Mongolia University of Technology, Hohhot, 010080, China*

**Abstract**

In view of data sparsity and long-range dependence when training the N-Gram Mongolian language model, a Mongolian Language Model based on Recurrent Neural Networks (MLMRNN) is proposed. The Mongolian classified word vector is designed and used as the input word vector of MLMRNN in the pre-training phase, and the Skip-Gram word vector with context information is used at the input layer so that the input contains not only semantic information, but also rich context information. It effectively avoids the problem of data sparsity and long-range dependence. Finally, the training algorithm of MLMRNN is designed and the perplexity is used as the evaluation index of the language model to test the perplexity of N-Gram, RNNLM and MLMRNN on the training set and test set, respectively. The experimental results show that the perplexity of using MLMRNN is lower than that of other language models, and the performance of the language model is improved.

*Keywords*: language model; word vector; recurrent neural networks; N-Gram Mongolian Language Model

(Submitted on April 9, 2018; Revised on May 21, 2018; Accepted on June 23, 2018)



## 1. Introduction

Language model is widely used in natural language processing, which played an important role in speech recognition, machine translation, question answering systems and other applications. The language model is modeled by identifying the priori probability in word sequences allowed in the language, so it can provide grammatical and syntactic constraints for the word sequences [12,14,17,21]. In 1980, Jelinek proposed a statistical language model based on N-Gram [2,20] that could calculate the probability of a word sequence. Unfortunately, it suffers from data sparsity, long-range dependence and lack of semantic information injection. Researches [3,6,8,19] added smoothing techniques to the N-Gram language model in order to address the issue of data sparsity. Bengio continued N-Gram's assumptions in 2003 and proposed NNLM [4], which avoided data sparsity but still failed to capture the long historical information. In response to this problem, Mikolov proposed RNNLM in 2010 [13]. It can obtain the long historical information. After that, researchers applied RNN modeling methods to make the Chinese language model [18], and the fusion modeling method was proposed [11]. The experimental results showed that the perplexity of the RNN model is lower than the traditional N-Gram language model in Chinese corpus. Then, another method based on word vector features was proposed, and demonstrated through experiments a further reduction of perplexity [10,22], but it still lacked the injection of semantic information.

In the study of the Mongolian language model, researchers [7] used the N-Gram language model and focused on the improvement of the N-Gram model. For example, the use of the Skip-N Mongolian statistical language model based on long-range dependence effectively improved the performance of the Mongolian-Chinese machine translation; the study of Mongolian text recognition algorithm [15] used the N-Gram model to identify the Mongolian language text in different language texts and word classes. The Mongolian N-Gram model [1] used the word clustering algorithm to determine the appropriate number of word classes. When the training corpus is insufficient, the language model based on word classes can effectively solve the problem of data sparsity.

This paper presents the Mongolian Language Model Based on Recurrent Neural Networks (MLMRNN), introducing context vector in the input layer and the Mongolian classified word vector of semantic information category. MLMRNN can

* Corresponding author.
*E-mail address*: 1045685828@qq.com

not only learn the historical information of longer distance, but also inject relevant semantic category information at the same time. In order to verify the validity of MLMRNN, the experimental results showed that the perplexity of MLMRNN is lower than that of other language models, and the performance of it is further improved.

## 2. Research Foundation

### *2.1. N -gram Language Model*

In 1980, Jelinek proposed a statistical language model based on N-Gram that used the Markov hypothesis. Suppose each word in a sentence is only related to the previous N-1 words. The larger N is, the stronger the distinguishability of the model; the smaller N is, the higher the reliability of the model [9]. The Conditional Probability is defined as Equation (1):

$$P\left(w_t|w_1,\cdots,w_{t-1}\right)\approx P\left(w_t|w_{t-(n-1)},\cdots,w_{t-1}\right)=\frac{\text{count}\left(w_{t-(n-1)},w_{t-(n-2)},\cdots,w_t\right)}{\text{count}\left(w_{t-(n-1)},w_{t-(n-2)},\cdots,w_{t-1}\right)} \tag{1}$$

The advantages of the N-Gram language model are simple and effective, but it only considers the positional relationship of words without word similarity; it also suffers from data sparsity, long-range dependence and lack of semantic information injection.

### *2.2. RNN Language Model*

In 2010, Mikolov and other researchers [13,16] in Brno University of Technology proposed the Recurrent Neural Network Language Model (RNNLM). This model can remember longer historical information through a continuous circulation in the hidden layer. That means the RNNLM model performance has been further improved. The structure of RNN is shown in Figure 1.

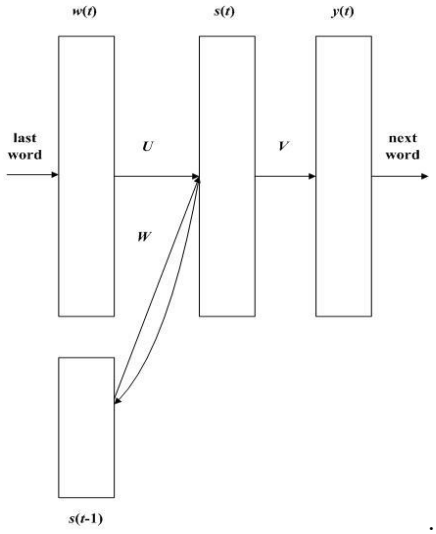



Figure 1. Recurrent Neural Network structure diagram

The RNN consists of three network layers, where $x(t)$、$s(t)$ and $y(t)$ denote the input layer, hidden layer and output layer, respectively. The input layer includes the current word vector $w(t)$ and the $s(t\text{-}1)$ state of the hidden layer at the previous moment. After the training text corpus isd trained by this RNN structure, the definition of the probability that the word vector $W_i$ in the current word appears is given as Equation (2):

$$P\left(w_i|s_i\right)=P\left(w_i|w_{i-1},s_{i-1}\right) \tag{2}$$

Assuming that the sample of the input words at time t is called the current word vector $w(t)$, the dimension is determined by the number of word samples |V| in the corpus. The $s(t)$ is determined by the input of the current word vector $w(t)$ and also

determined by the $s(t\text{-}1)$, which is the hidden layer state of historical information at the previous moment. We make the hidden layer state of the t-1 moment as a part of the time t input through the connection of the hidden layer to the input layer. The $y(t)$ represents the probability distribution information of the next word at the current moment, and the output layer nodes have the same number of input layer nodes as |V|. The relationship between layers is calculated by using the following Equations (3), (4), and (5):

$$x(\mathrm{t}) = w(\mathrm{t}) + s(\mathrm{t}-1) \tag{3}$$

$$s(\mathrm{t}) = f(\mathrm{U} \cdot w(\mathrm{t}) + \mathrm{W} \cdot s(\mathrm{t}-1) \tag{4}$$

$$\mathrm{y}(\mathrm{t}) = g(\mathrm{V} \cdot \mathrm{s}(\mathrm{t})) \tag{5}$$

Where $x(t)$, $s(t)$, $y(t)$, $f$ and $g$ denote the network input at time t, state of the network hidden layer, output of the network, sigmoid activation function and softmax activation function, respectively.

## 3. Mongolian Language Model based on Recurrent Neural Networks

### *3.1. Language Model Structure*

$x(t)$, $s(t)$ and $y(t)$ are included in the structure of the MLMRNN model. They denote the input layer, hidden layer and output layer, respectively. The structure of the MLMRNN model is shown in Figure 2.

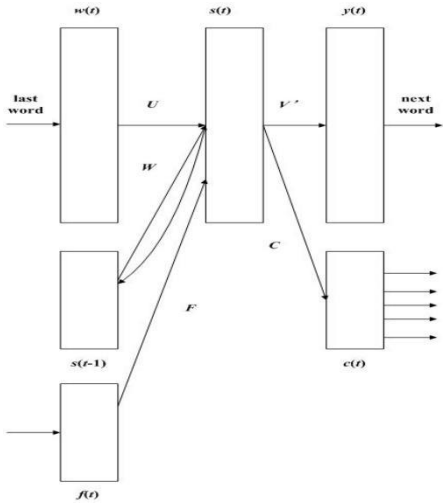



Figure 2. MLMRNN model structure diagram

At the time of t, $w(t)$, $s(t\text{-}1)$ and $f(t)$ are included in the input vector $x(t)$. The $w(t)$ represents the Mongolian classified word vector, which is input at time t; the $s(t\text{-}1)$ is the output of the hidden layer at time t-1; the $f(t)$ is the context-word vector trained by Skip-Gram at time t [23]; the dimension is far less than |V|, and the $s(t)$ is the hidden layer. The output vector is represented by $y(t)$, which includes two parts: one is the neurons of the category level and the other is the Mongolian words neurons. The $c(t)$ is the category layer of the clustering for word vectors in the vocabulary [5]. The output vector $y(t)$ represents the probability of the next word $w(t+1)$.

In the network, $U$, $W$, $F$ are the weight matrix between the input layer and hidden layer. $C$ is the weight matrix of the hidden layer and category layer. When the output layer is calculated, the probability distribution of the word class is calculated, and then the probability of the specific word is calculated from the required word class. The result of the computed output layer requires the output of the category. The output value of each layer in the network is expressed as following Equations (6), (7), (8), and (9):

$$x(t) = w(t) + s(t-1) + f(t) \tag{6}$$

$$s(t) = f\left(U \cdot w(t) + W \cdot s(t-1) + F \cdot f(t)\right) \tag{7}$$

$$y(t) = g\left(V \cdot s(t)\right) \tag{8}$$

$$y(t) = g\left(V' \cdot s(t)\right) \tag{9}$$

Where $V' \in R\left(c\left(w(t)\right)\right)$, $R(c(w(t))$ denotes the word set of the class to which $W_t$ belongs. $f$ and $g$ denote the sigmoid activation function.

### *3.2. Mongolian Classified Word Vector*

The classified word vector of the Mongolian word $w$ can be formalized as $W=I_{m1},I_{m2},...,I_{mi}I_{mi+1},...,I_{mk}(1 \le i \le k, 1 \le m_i \le V)$, where $k$ indicates the number of categories of the Mongolian classified word vector, $m_i$ represents the word vector category, $V$ represents the number of words in $m_i$ when W belongs to the $m_i$, $I_{mi}$ is 1, otherwise it is 0. $I_{mi}$ denotes one-hot word vector whose $w$ belongs to the $m_i$. The Mongolian classified word vector is shown in Figure 3.

ᠤᠰᠤ (water)、ᠠᠷᠢᠭ (wine) and ᠤᠮᠡᠳᠭᠡᠬᠦ (drinks) all belong to semantic category 1. There are 3 words in semantic category 1. Each word corresponds to the same $I_{m1}$ encoding format as the one-hot encoding format: $I_{m1}$ of ᠤᠰᠤ (water)、ᠠᠷᠢᠭ (wine) and ᠤᠮᠡᠳᠭᠡᠬᠦ (drinks) are [1,0,0,…,0], [0,1,0,…,0] and [0,0,1,…,0], respectively. The rest of the word vector encoding format from $I_{m2}$ to $I_{mk}$ are also the same as the one-hot encoding format, and $I_{m2}$ to $I_{mk}$ are [0, … ,0]. When all Mongolian words under semantic category 1 are encoded, word vectors of all Mongolian words under semantic category 2 are encoded until all semantic categories are completed. That means the corresponding Mongolian classified word vector of ᠤᠰᠤ (water)、ᠠᠷᠢᠭ (wine) and ᠤᠮᠡᠳᠭᠡᠬᠦ (drinks) are encoded as: [1,0,0,0,0,0,0,0,0, ⋯ ,0,0,0,0,0,0,0,0]、[0,1,0,0,0,0,0,0,0, ⋯ ,0,0,0,0,0,0,0,0] and [0,0,1,0,0,0,0,0,0, ⋯ ,0,0,0,0,0,0,0,0].

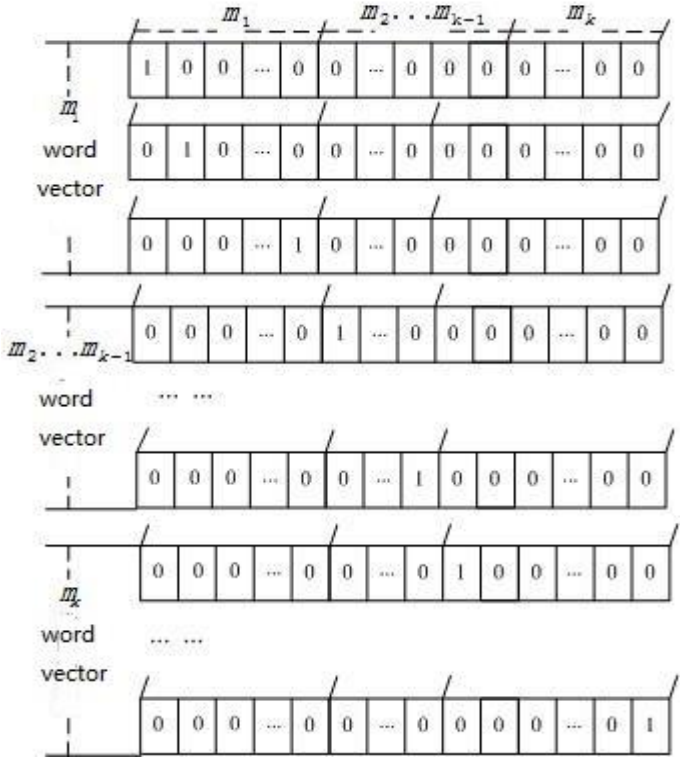



Figure 3. Representation of Mongolian classified word vector

In the pre-training phase of MLMRNN, Mongolian words and their semantic categories are obtained by using the Word2vec+k-means algorithm. It is shown in Algorithm 1.

**Algorithm 1** Training Algorithm of Mongolian Words and Their Classification

**Input**：$V$ represents a data set containing |V| Mongolian word vectors $\{w_1,\ldots,w_V\}$, and $k$ represents the number of clusters(k≤V).

**Output**：The set $S$ of $k$ clusters

1. Initialize the cluster centers $\mu_1,\mu_2,\ldots,\mu_k$
2. Repeat:
3. **For** $i\leftarrow 1;\ i\le V;i\leftarrow i+1$ **do**
4. **For** $j\leftarrow 1;\ j\le k;\ j\leftarrow j+1$ **do**
5. $S_i\leftarrow \arg min_j w_i-\mu_j^2$
6. $\mu_j\leftarrow \dfrac{\sum_{i=1}^{V}1\{S_i=j\}w_i}{\sum_{i=1}^{V}1\{S_i=j\}}$
7. **End For**
8. **End For**
9. until convergence

When training and calculating MLMRNN, a Mongolian classified word vector is created for each input Mongolian word according to the classification information, which is used as an input layer for calculation. According to the semantic, the Mongolian classified word vector is divided into $k$ categories. The Mongolian classified word vector algorithm is shown in Algorithm 2.

**Algorithm 2** Mongolian classified word vector algorithm

**Input**：The set $S$ of $k$ clusters，where $Si$ represents the ith set and $S$ has $k$ categories//$k$ represents the number of clusters and *index* represents the position of Mongolian word vectors in $k$ clusters.

**Output**：kc-one-hot// Mongolian classified word vectors in $k$ cluster.

1. $n=\sum_{i}^{k}size(S_i)$ //Read the size of the vocabulary as the dimension of the word vector
2. Initialize the length of *kc-one-hot* is $n$
3. **For** $i\leftarrow 0;\ i\le k;i++$ **do**
4. **For** $j\leftarrow 0;\ j\le\sum_{i}^{k}size(S_{i+1});j++$ **do**
5. index= $i*\sum_{i}^{k}size(S_{i+1})+j$;
6. kc-one-hot[index] = 1;
7. **End For**
8. **End For**

### *3.3. LMRNN Training Algorithm*

The MLMRNN training consists of two parts: one is forward calculation and the other is back propagation. The pre-training part before forward propagation provides the initial values needed for network training. In the pre-training part, Mongolian word vectors and Skip -Gram word vector are introduced, and backpropagation is mainly used to update the MLMRNN network weights so that the loss function is minimized. The entire network training activates functions to describe the relationship between layers. The input layer to the hidden layer use the sigmoid activation function. The use of the hidden layer to the output layer is the softmax activation function, so as to simulate the interaction between layers of neurons. The purpose of training is to be able to present the desired output for input. The network structure of MLMRNN is shown in Figure 4.

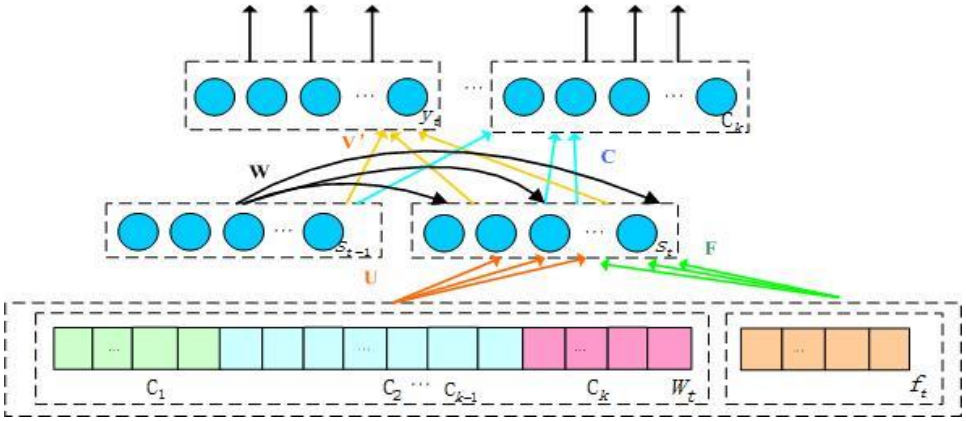



Figure 4. The network structure of MLMRNN

As shown in Figure 4 $W_t$ denotes the input Mongolian classified word vector at time t, and the dimension is the size of dictionary $V$. $f_t$ denotes the Skip-Gram word vector of the Mongolian pre-trained at time t. The dimension is M(M＜V), and the training algorithm of MLMRNN is shown in Algorithm 3:

**Algorithm 3** MLMRNN training algorithm

**Input:** $W_t$, $f_t$, α, epoch and $\tau$ denote the Mongolian classified word vector pre-trained at time t, the Mongolian Skip-Gram word vector pre-trained at time t, the learning rate, the number of cycles and the bptt size, respectively.

**Output:** Model parameters $W_{xs}, W_{ss}, W_{fs}, W_{sc}, W_{sy},$, $y_t$ denotes the probability of the next Mongolian word occurring under the first few words.

// $l_t$ is the expected output probability of the word in the class, $y_t$ is the class probability of the real output, $c'_t$ is the class probability of the expected output, and $c_t$ is the class probability of the real output.

1. Random initialization $W_{xs}, W_{fs}, W_{ss}, W_{sy}, W_{sc}$
2. **For** $j$ in 0 **to** epoch **do**
3. $x_t \leftarrow W_{xs} \cdot w_t + W_{ss} \cdot s_{t-1} + W_{sf} \cdot f_t$
4. $s_t \leftarrow sigmoid(x_t)$
5. $y_t \leftarrow \mathrm{softmax}(W_{sy} \cdot s_t)$
6. $c_t \leftarrow \mathrm{softmax}(W_{sc} \cdot s_t)$
7. $E_t(y_t, l_t) \leftarrow -\log y_t$
8. $E_t(c_t, c'_t) \leftarrow -c_t \log c'_t$
9. **For** $\tau \leftarrow 1;\ \tau \le t;\ \tau$ ++ **do**
10. $\delta_y^t = \partial E_t(y_t, l_t) / \partial W_{sy}$
11. $\delta_c^t = \partial E_t(c_t, c'_t) / \partial W_{sc}$
12. $\delta_s^{t-\tau-1} = \left[\partial E_{t-\tau}(y_{t-\tau}, l_{t-\tau}) + \partial E_{t-\tau}(c_{t-\tau}?, c'_{t-\tau})\right] / \partial W_{ss}$
13. $W_{xs}^{\tau} \leftarrow W_{xs}^{\tau} - \cdot \delta_s^{t-\tau} w_{t-\tau}^t$
14. $W_{ss}^{\tau} \leftarrow W_{ss}^{\tau} - \cdot \delta_s^{t-\tau-1} s_{t-\tau-1}^t$
15. $W_{fs}^{\tau} \leftarrow W_{fs}^{\tau} - \cdot \delta_s^{t-\tau} f_{t-\tau}^t$
16. $W_{sc}^{\tau} \leftarrow W_{sc}^{\tau} - \cdot \delta_c^{\tau} c_{\tau}^t$
17. $W_{sy}^{\tau} \leftarrow W_{sy}^{\tau} - \cdot \delta_y^{\tau} s_{\tau}^t$
18. **End For**
19. **End For**

## 4. Language Model Evaluation Method and Experimental Analysis

### *4.1. Experimental Design*

In order to verify the validity of the proposed MLMRNN Mongolian language model, the following experiments were designed:

(1) By testing the language model perplexity of different hidden layer nodes and bptt values, the experimental study on the influence of the number of hidden layer nodes and the bptt value on the language model is carried out. We take the hidden layer node number of the language model with the lowest perplexity as the parameter for the subsequent experiment.

(2) After obtaining the optimal parameters of the language model by (1), the same parameters are used to conduct comparative experiments, namely □ introducing Skip-Gram word vectors into the Mongolian language, that is, RNNLM + SK; □ using Mongolian classified word vectors, that is, KC + RNNLM; □ In this paper, we introduce the Skip-Gram word vector in Mongolian, using the Mongolian classified word vector, namely MLMRNN, and compare it with the Mongolian language model based on N-Gram in terms of perplexity.

### *4.2. Experimental Setup*

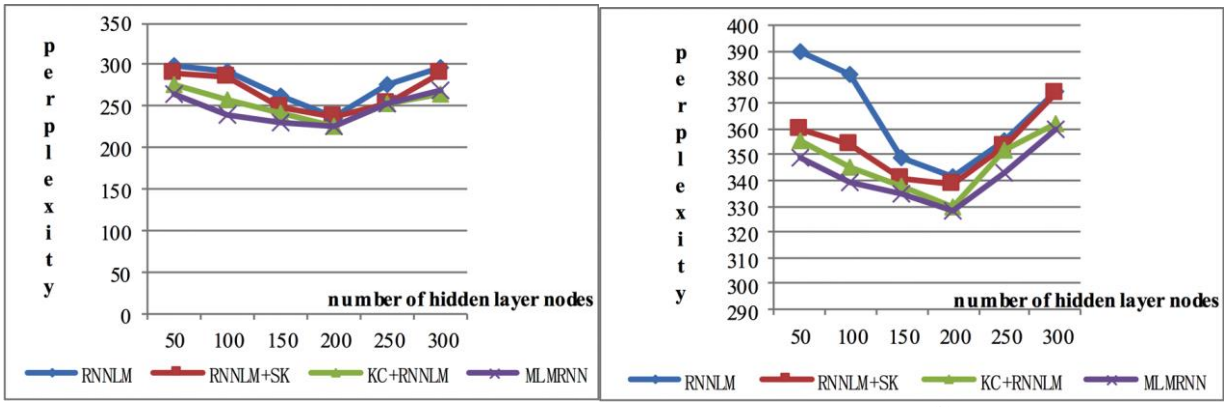
Experimental training data comes from the Mongolian corpus of the previous work. There are 8000 Mongolian sentences. The corpus is divided into a training set and test set according to the ratio of 3: 1; 6000 sentences are used for model training, and the other 2000 sentences are for the model perplexity test. The number of nodes in the hidden layer are set for training comparison in six groups, which is based on the recurrent neural network Mongolian language model. The same training and test dataset were used in the comparative experiment, with 11,940 words. In the experiment, the Skip-Gram model of Google's Word2vec tool was used to get the Mongolian word vectors with a dimension of 50 and a window length of 2 on training set.

The evaluation index adopted in this experiment is the perplexity of the language model [3], which is the most commonly used index to evaluate the performance of a language model. The meaning of the perplexity is the geometric mean of the candidate words after each word when the language model predicts a certain linguistic phenomenon. The lower perplexity, the stronger the language model's ability to constrain the context, and therefore, the better model performance. The experiment divides the Mongolian words into 10 categories based on semantic similarity, so the number of categories in the experiment is set as 10. The number of hidden layer nodes is determined by comparative experiments. Other parameters are set as bptt3, bptt-block3, and class10.

### *4.3. Experimental Results and Analysis*

(1) The influence of the number of hidden layer nodes on the performance for the language model

In the training of the language model, the number of hidden layer nodes plays an important role. By adjusting the number of hidden layer nodes, the influence of the number of hidden layer nodes on the perplexity of different language models is tested. There are 6 groups of different numbers of hidden layer nodes: 50, 100, 150, 200, 250, and 300. Comparing the perplexity of the different language models on the training set and the test set, the influence of the number of hidden layer nodes in the training set on the language model's perplexity is shown in Figure 5(a). The result of the influence of the hidden layer nodes in the test set on the perplexity is shown in Figure 5(b).




(a) (b)

Figure 5. The perplexity of language model in (a) training set and (b) test set with different number of hidden layer nodes

The experimental results showed that with an increase in number of hidden layer nodes, the perplexity of each language model decreased. But when the number of hidden layer nodes increased to a certain extent, the perplexity of the language model rose instead. So, it needed to adjust and select the relatively good performance parameters. Among them, when the number of hidden layer nodes is 200, the language model has a lower perplexity and better performance.

(2) The influence of bptt value on the performance of language model

When the optimal number of hidden layer nodes is determined, it is vital to set five different bptt values, and the different bptt values of the language model on the training set and test set are compared to determine the optimal bptt value. The experimental results are shown in Figure 6.

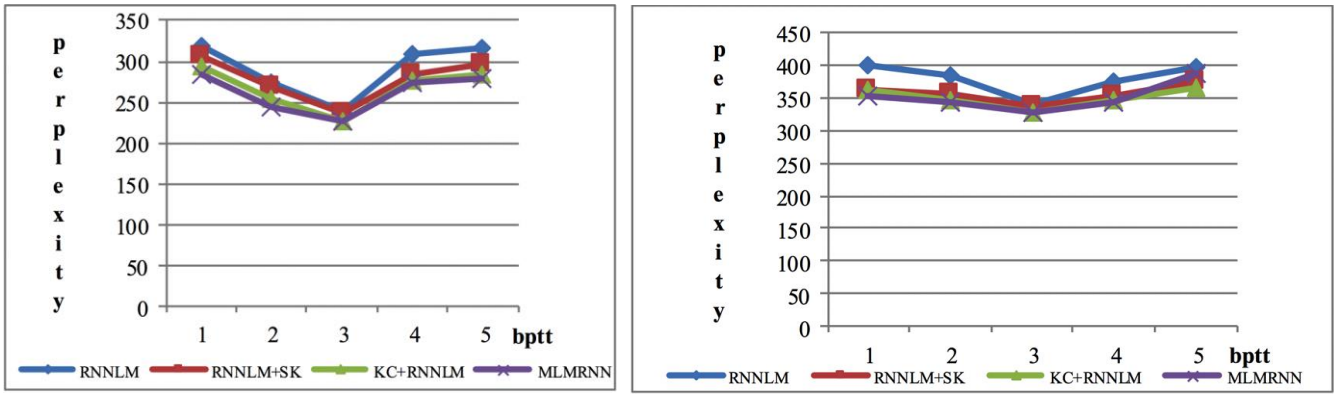



Figure 6. The perplexity of performance of language model with different bptt values in (a) training set and (b) test set

The experimental results showed that with an increase of bptt value, the perplexity of each language model decreased. However, when the bptt value increased to some extent, the perplexity of the language model rose instead. Therefore, an adjustment will be needed to choose a relatively good bptt value. Among them, when the bptt value is in the range of three, the language model has a lower perplexity and better performance.

(3) Comparison of the reduction rate of perplexity for different language models

It can be seen from the experimental results that when the number of hidden layer nodes is 200 and the bptt value is 3, the language model has lower perplexity and better performance. Therefore, the above parameters are applied to the comparison of experimental results in other models, where RNNLM200 indicates that the number of neurons in the hidden layer is 200, which is denoted as RNNLM200. RNNLM200 + SK represents a Mongolian Recurrent Neural Network Language Model using a pre-trained Mongolian 50-dimensional Skip-Gram word vector; KC + RNNLM200 represents a Mongolian Recurrent Neural Network Language Model using a Mongolian classified word vector. MLMRNN represents a Mongolian Recurrent Neural Network Language Model using a Mongolian classification word vector with 200 hidden layer neurons and using a pre-trained Mongolian 50- dimensional Skip-Gram word vector.

Table 1. Perplexity comparison of different language models

| Model | PPL | |
|---|---|---|
| | Train set | Test set |
| 3-gram(KN3) | 575.43 | 693.49 |
| RNNLM200 | 237.99 | 341.72 |
| RNNLM200+SK | 237.01 | 338.79 |
| KC+RNNLM200 | 226.20 | 329.67 |
| MLMRNN | 225.86 | 328.04 |

Table 1 lists the experimental results of five different language models. The experimental results showed that MLMRNN has a lower perplexity in the training set and the test set compared to other language models. It showed that the performance of the Mongolian Recurrent Neural Network Language Model is further improved by introducing the multidimensional Skip-Gram word vectors and using Mongolian classified word vectors. It further illustrated the effectiveness of the proposed method.

The reduction rate of the language model perplexity refers to the difference between the current language model's perplexity and the perplexity of the original language model with the percentage of the perplexity of the original language model. The reduction rate is calculated by using Equation (10):

$$\text{The rate of decrase in language model perplexity} = -\frac{\text{perplexity of the current language model} - \text{perplexity of the original language model}}{\text{perplexity of the original language model}} \times 100\% \quad (10)$$

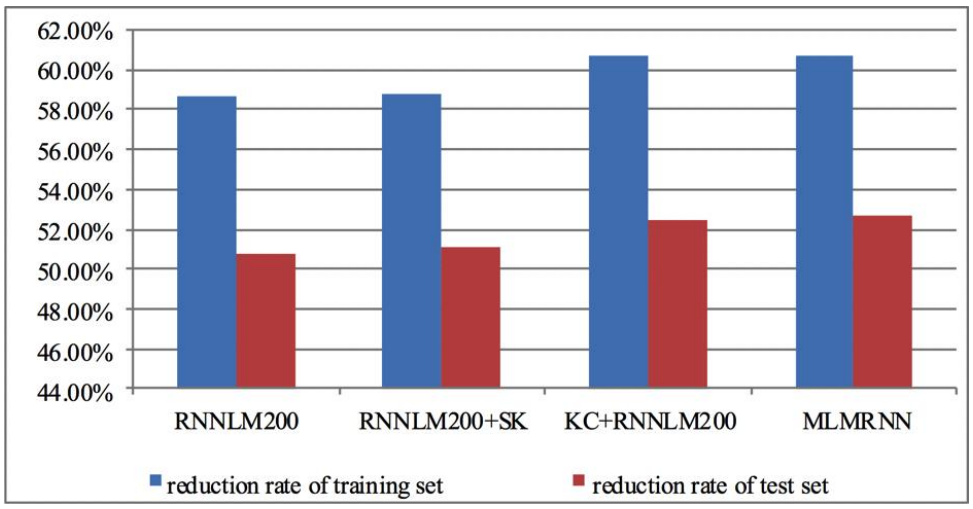



Figure 7. Reduction rate of perplexity in language model

Figure 7 further compared the reduction rate of perplexity between the training set and the test set by using different language models and 3-Gram language models, respectively. The calculation formula of the reduction rate is shown in Formula 10, and it can be seen from Figure 7 that when the reduction rate of MLMRNN is larger, the performance is better.

## 5. Conclusions

This paper presents a Mongolian Recurrent Neural Network Language Model while also introducing Mongolian classified word vectors and Skip-Gram word vectors with context information at the input layer. The experiments showed that this method can reduce the perplexity of the Mongolian Language Model.

(1) The influence of the number of hidden layer nodes on the perplexity for the language model was tested, and six groups of different hidden layer nodes were set up for experimental comparison. The experimental results showed that when the number of hidden layer nodes is 200, the language model has lower perplexity.

(2) By setting five groups of different bptt values, the influence of different bptt values on the perplexity of the language model is compared. It can be seen from the results that when the bptt value is three, the language model has lower perplexity.

(3) We compared the perplexity of different language models with optimal parameters. Finally, we can see from the experimental results that MLMRNN has lower perplexity and better performance compared to other language models.

## Acknowledgments

Funding project: National Natural Science Foundation of China (61762070, 61650205).

## References

1. X. Ai, “Researching of Mongolian Language Model based on Speech Recognition,” Hohhot: Inner Mongolia University, 2007
2. H. Bourlard and N. Morgan, “Continuous Speech Recognition by Connectionist Statistical Methods” IEEE Transactions on Neural Networks, vol. 4, no. 6, pp. 893-909, 1993
3. P. F. Brown, P. V. Desouza, and R. L. Mercer, “Class-based N-gram Models of Natural Language,” Computational Linguistics, vol. 18, no. 4, pp. 467-479, 1997

4. Y. Bengio, Ducharme and Jean, "A Neural Probabilistic Language Model," Journal of Machine Learning Research, vol. 3, no. 6, pp. 1137-1155, 2003
5. A. Coates and A. Y. Ng, "Learning Feature Representations with K-Means," Lecture Notes in Computer Science, 2012, vol. 7700, pp.561-580
6. S. F. Chen and J. Goodman, "An Empirical Study of Smoothing Techniques for Language Modeling," Meeting of the Association for Computational Linguistics, pp. 24-27, June 1996, University of California, Santa Cruz, California, Usa, Proceedings. pp. 359-393, 1996
7. H. X. Hou, Q. Liu and Z. W. Liu, "Skip-N Mongolian Statistical Language Model," Journal of Inner Mongolia University (natural edition), vol. 39, no. 2, pp. 220-224, 2008
8. R. Kuhn and R. M. De "A Cache-Based Natural Language Model for Speech Recognition," IEEE Transactions on Pattern Analysis & Machine Intelligence, vol. 12, no. 6, pp. 219-228, 1990
9. R. Kneser and H. Ney, "Improved Backing-off for N-gram Language Modeling," In Proceedings of the international conference on acoustics, speech and signal processing (ICASSP), pp. 181-184, 1995
10. H. Li, D. Qu and W. L. Zhang, "Recurrent Neural Network Language Model with Global Word Vector Features," signal peocessing, vol. 32, no. 6, pp. 715-723, 2016
11. Y. X. Li, J. Q. Zhang and D. Pan, "A Study of Speech," Journal of Integrative Plant Biology, vol. 51, no. 9, pp. 1936-1944, 2014
12. T. A. Mikolov, "Statistical Language Models based on Neural Networks," 2012
13. T. Mikolov, M. Karaat and L. Burget, "Recurrent Neural Network based Language Mode," Proceedings of Interspeech, pp. 1045-1048, 2010
14. W. D. Mulder S. Bethard and M. F. Moens, "A Survey on the Application of Recurrent Neural Networks to Statistical Language Modeling," Computer Speech & Language, vol. 30, no. 1, pp. 61-98, 2015
15. Z. Q. Ma, Z. G. Zhang and R. Yan, "N-Gram Based Language Identification for Mongolian Text," Journal of Chinese Information Processing, vol. 30, no. 1, pp. 133-139, 2016
16. I. Sutskever. "Training Recurrent Neural Networks," Toronto: University of Toronto, 2013
17. A. Vinciarelli, S. Bengio and H. Bunke, "Offline Recognition of Unconstrained Handwritten Texts Using HMMs and Statistical Language Models," IEEE Transactions on Pattern Analysis & Machine Intelligence, vol. 26, no. 6, pp. 709-20, 2004
18. L. Wang, J. A. Yang and L. Chen, "Recurrent Neural Network based Chinese Language Modeling Method," Acoustic technology, vol. 34, no. 5, pp. 431-436, 2015
19. P. Xu and F. Jelinek, "Random Forests in Language Modelin," Conference on Empirical Methods in Natural Language Processing, EMNLP 2004, A Meeting of Sigdat,A Special Interest Group of the Acl,Held in Conjunction with ACL 2004, pp. 25-26 July 2004, Barcelona, Spain. pp. 325-332, 2004
20. Y. K. Xing and S. P. Ma, "A Survey on Statistical Language Models," Computer Science, vol. 30, no. 9, pp. 22-26, 2003
21. C. X. Zhai, "Statistical Language Models for Information Retrieval," Now Publishers Inc, 2008
22. J. Zhang, D. Qu and Z. Li, "Recurrent Neural Network Language Model based on Word Vector Features," Pattern Recognition and Artificial Intelligence, vol. 28, no. 4, pp. 299-305, 2015
23. L. Zhou, "Exploration of the Working Principle and Application of Word2vec," Library and Information Guide, no. 2, pp. 145-148, 2015



**Zhiqiang Ma** is a professor from the College of Data Science and Application, Inner Mongolia University of Technology. He is a CCF member and his research interests include machine learning, speech recognition, and natural language processing.

**Li Zhang** is a Master's student from the College of Data Science and Application, Inner Mongolia University of Technology. She is a CCF member and her research interests include machine learning, speech recognition, and natural language processing.

**Rui Yang** is a Master's student from the College of Data Science and Application, Inner Mongolia University of Technology. She is a CCF member and her research interests include machine learning, speech recognition, and natural language processing.

**Tuya Li** is a Master's student from the College of Data Science and Application, Inner Mongolia University of Technology. She is a CCF member and her research interests include machine learning, speech recognition, and natural language processing.