IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received March 20, 2020, accepted March 29, 2020, date of publication April 3, 2020, date of current version April 17, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2985418

NEWLSTM: An Optimized Long Short-Term Memory Language Model for Sequence Prediction

QING WANG[1], RONG-QUN PENG[1], JIA-QIANG WANG[2], ZHI LI[3], AND HAN-BING QU[2]

[1]School of Computer Science and Technology, Shandong University of Technology, Zibo 255049, China
[2]Key Laboratory of Artificial Intelligence and Data Analysis, Beijing Academy of Science and Technology, Beijing 100094, China
[3]School of Economics and Management, University of Chinese Academy of Sciences, Beijing 100049, China

Corresponding author: Rong-Qun Peng (pengrq2006@126.com)

This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFF0301000, Grant 2018YFC0809700, and Grant 2018YFC0704800, in part by the National Natural Science Foundation of China under Grant NSF91746207, in part by the Beijing Postdoctoral Research Foundation under Grant ZZ-2019-76, and in part by the China Postdoctoral Science Foundation under Grant 2019M660540.

ABSTRACT The long short-term memory (LSTM) model trained on the universal language modeling task overcomes the bottleneck of vanishing gradients in the traditional recurrent neural network (RNN) and shows excellent performance in processing multiple tasks generated by natural language processing. Although LSTM effectively alleviates the vanishing gradient problem in the RNN, the information will be greatly lost in the long distance transmission, and there are still some limitations in its practical use. In this paper, we propose a new model called NEWLSTM, which improves the LSTM model, and alleviates the defects of too many parameters in LSTM and the vanishing gradient. The NEWLSTM model directly correlates the cell state information with current information. The traditional LSTM's input gate and forget gate are integrated, some components are deleted, the problems of too many LSTM parameters and complicated calculations are solved, and the iteration time is effectively reduced. In this paper, a neural network model is used to identify the relationship between input information sequences to predict the language sequence. The experimental results show that the improved new model is simpler than traditional LSTM models and LSTM variants on multiple test sets. NEWLSTM has better overall stability and can better solve the sparse words problem.

INDEX TERMS Gate fusion, exploding gradient, long short-term memory, recurrent neural network.

I. INTRODUCTION

Along with the transition from the traditional n-gram language model to the neural language model, research has revealed the potential of the statistical language model on the basic task of natural language processing (NLP) [1]. A language model based on a neural network performs the implicit clustering of words in a low-dimensional space, which can be used to predict many types of signals including language [2] and has attracted widespread research attention. Sequence prediction and classification in natural language processing is a ubiquitous and challenging problem that usually requires identifying complex dependencies between long-term inputs. The recurrent neural network (RNN), which can model sequence data for sequence recognition and prediction, shows strong performance in various NLP tasks [3]. The RNN performs the implicit clustering of words in a low-dimensional space, accepts input vectors at each time step, and updates the hidden state using a non-linear activation function. The RNN responds to time dependence through short-term memory implemented using feedback [4], which can effectively predict sequences at the next time step, and its hidden state can store information as a high-dimensional distributed representation, forming a model with rich information. In addition, the RNN can realize effective and powerful calculations by using nonlinear dynamics, which can be applied to sequences with highly complex structures to perform modeling and prediction tasks.

There are still many problems with using the RNN. For example, forward and back propagation in the RNN is performed sequentially, which is time-consuming during training, and vanishing and exploding gradients may occur. When faced with long-term memory, it is difficult for the RNN to train them effectively. Therefore, there are still many challenges in using the RNN for long sequence learning.

In recent research, many attempts have been made to overcome these difficulties. In research in the word embedding field, Takase *et al.* [5] constructed word embeddings

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

from character n-gram embeddings and proposed an RNN language model that uses character information combined with ordinary embeddings. The method achieved the best confusion on each dataset. Wen *et al.* [6] proposed a statistical language generator based on joint recursive and convolutional neural network structures, which can be trained on dialog actions without any semantic alignment or predefined syntax trees. Pappas and Henderson [7] studied the usability of powerful shared mappings for output labels and proposed a deep residual output mapping. The method performed well at capturing the structure of the output space and avoiding overfitting and solved the shortcoming of the lack of shared parameters across output labels in neural language models.

The long short-term memory (LSTM) proposed by Smagulova and James [8] is a variant of the RNN. It overcomes the vanishing gradients in the traditional RNN by loading information at each time step for transmission. It improves the structure of the traditional RNN hidden layer and solves the problem that the long-term dependence of the encoding in the input sequence cannot be achieved due to the vanishing gradient in the RNN [9]. In previous studies, for general sequence modeling, LSTM has proven to be stable and powerful for modeling remote dependencies [10]–[13]. However, although the text generated by LSTM shows long-term correlation characteristics on reproducible scales, it does not solve the common problem of constructing simple recursive networks. Although the vanishing gradient problem of the RNN is partially avoided, the information loss in the long distance transmission is very serious [14]. LSTM cannot explore the information of prediction sequences that have significant changes over a short period of time. To improve the network convergence speed, people add gate structures to the standard LSTM structure to enhance the long-term dependent learning ability of LSTM variants [15], [16]. By using a periodic function to parameterize each hidden unit to affect the gradient information flow, the initialization parameters need to be fine-grain adjusted. In addition, when dealing with complex long-term and short-term dependence problems when capturing multi-dimensional time series data including future time steps, increasingly more attention-based multi-time series models have been applied [17].

According to the problems existing in LSTM, this article proposes that the NEWLSTM model is improved in the following three aspects. (1) The forget gate and input gate of the LSTM model are integrated and processed together to reduce the number of parameters and simplify the calculation of the model, which reduces the complexity of the model and effectively reduces the iteration time. (2) The vanishing gradient problem in some regions is solved. (3) It pays attention to the hidden information of the current information, emphasizes the subject of the input information, and makes the model pay more attention to the context.

II. RELATED WORKS

LSTM alleviates the vanishing gradient problem caused by hindering the back propagation in the RNN. It can solve many tasks that cannot be solved by recurrent neural network learning algorithms. It has been widely used in speech recognition, picture description, natural language processing and other fields [18]. Yao *et al.* [19] proposed a convolutional LSTM (ConvLSTM) and used it to build a trainable model of end-to-end precipitation nowcasting. Aiming at the internal covariate shift between time steps, Cooijmans *et al.* [20] proposed the re-parameterization of LSTM, which proved the effectiveness of the batch standardized conversion of hidden layer to hidden layer and used batch normalization to perform recursive network optimization. Zhang *et al.* [21] developed a deep adaptive long short-term memory (DA-LSTM) architecture, which can dynamically adjust the structure based on the information distribution without prior knowledge. When faced with very little information, shallow structures can be used to achieve faster convergence and consume less computing resources. Ali *et al.* [22] proposed a new type of semantic knowledge based on the Word2vec model, which used the bidirectional long short-term memory (Bi-LSTM) method to improve the traffic feature extraction and text classification tasks.

Aiming at the problems of LSTM, in recent decades, people have done much work to optimize LSTM models [9], [14], [23], [24], [25]. Kent and Salem [23] proposed a slim model to simplify the LSTM model by removing certain components, thereby effectively speeding up the training and running time in the case of performance changes. Jie and Lu [24] proposed an LSTM-CRF model guided by dependencies. By encoding the complete dependency tree and capturing the syntactic relationships for named entity recognition, the study showed that there is a strong relevance. de Lhoneux *et al.* [25] used the features extracted by BiLSTM to recursively combine subtrees in a transition-based parser to study the effect of adding tree layers to the sequential model. The results indicate that Bi-LSTM can effectively capture related subtrees' information. By associating forward and backward LSTM with functions and results with language attributes, it is proved that the improved backward LSTM is particularly important for the formation of the final language.

III. MODELS

A. LONG SHORT-TERM MEMORY

The RNN supports variable-length inputs. By mapping the target vector from all previous records of previous inputs, it overcomes the bottleneck that traditional neural network structures can only map from the current input to the target vector. It can capture the content of time data and make inferences; it is also very effective at the dynamic modeling of continuous sequence data [18]. However, when the gradient is calculated by RNN backpropagation, it is easy for the gradient to diverge and disappear due to too many layers, it is impossible to learn long-term information, and the method is very sensitive to the length of the article.

LSTM uses multiple gates in a similar structure to the RNN to adjust the amount of information of each node state,

which effectively solves the problem that the previous information disappears in the RNN because the hidden layer continuously superimposes the input sequence in the new time state. The disadvantage of forward propagation can effectively overcome the vanishing gradient problem [26]. Its hidden layer update is replaced by a dedicated memory unit C_t, which performs better in the face of longer sequences and is suitable for context-sensitive language learning [27]. C_t is essentially an accumulator of state information, which is capable of remembering information that needs to be stored for a long time.

In a two-layer feedforward neural network, connections exist between adjacent layers and hidden layers, and the nodes in the hidden layers are unconnected. The simple recurrent network adds a feedback connection from the hidden layer to the hidden layer, allowing information about the state of the hidden layer to be propagated over time. If the state of each moment is regarded as a layer of a feedforward neural network, the recurrent neural network can be considered as a neural network with weights shared in the time dimension.

The LSTM model controls the transmission state through the gated state, which can be accessed through several self-parameterized control gates, namely, the forget gate f_t, the input gate i_t, and the output gate o_t. However, due to the introduction of much content, the number of parameters is increased, which also makes the training more difficult. The basic unit in the hidden layer of the LSTM network is a memory block, which contains one or more memory units and a pair of multiply gated units to gate all the units contained in the hidden layer. The specific implementation of the LSTM internal structure is as follows:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \quad (1)$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \quad (2)$$

$$c_t = f_t c_{t-1} + i_t tanh(W_{xc}x_t + W_{hc}h_{t-1} + W_{hc}h_{t-1} + b_c) \quad (3)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \quad (4)$$

$$h_t = o_t \tanh(c_t) \quad (5)$$

Each time a new message x_t is input, the forget gate f_t is open, and the past cell state may be discarded during this process. Otherwise, the input gate i_t is activated, the information is accumulated in the unit, and the output gate o_t judges whether it will be propagated to the final state h_t. At a certain time point in the time series, the output information is uploaded to the memory unit to prevent the problem of information disappearing after too long.

In recurrent neural networks, gradients can explode or vanish exponentially over time. Therefore, the gradient either dominates the next weight adaptation step or effectively disappears.

The LSTM propagates backwards from the top, not from the last error signal, and the error signal decreases or explodes with the multiple layers of non-linear transformation. However, traditional LSTM performs best on data sequences where information is evenly distributed between steps. For a high information flow, the transfer function from one hidden state to the next hidden state in the LSTM is too shallow and there is a lack of depth between the design steps, which can be considered a single linear transformation with activation; and the underlying structure in the sequence cannot be captured for sequences with uneven information flows between steps [28].

B. NEWLSTM

Aiming at the problems in LSTM, this paper proposes a new variant called NEWLSTM. NEWLSTM loads the information at each time step for transmission. The forget gate and input gate are fused to combine the current input information with the cell state; the information is accumulated in the cell; and, finally, the output gate o_t determines whether the information is propagated to the final state h_t.

NEWLSTM modified the network architecture and used the tanh activation function to compress new information to improve the vanishing gradient problem.

$$\tanh x = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (6)$$

The tanh function compresses the input value to the range of $-1\sim1$, which solves the non-zero-centered problem of the sigmoid function. NEWLSTM saves the selective discarding of some input information in the forget gate, which is beneficial to long sequence memory information.

NEWLSTM retains the use of the tanh activation function in the input gate, focuses on the hidden information in the sentence, and makes the model focus on the topic of the sentence. The specific implementation of the internal cell structure of the NEWLSTM is as follows:

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \quad (7)$$

$$c_t = f_t(c_{t-1} + tanh(W_{xc}x_t + W_{hc}h_{t-1} + W_{hc}h_{t-1} + b_c) \quad (8)$$

x_t is an input vector at time t, which is used to store all useful information at time t. W is the weight matrix of the hidden state h_t and is the weight of each gate. b_f, b_c and b_o represent the deviation vectors. f_t and o_t are the gated scalars of the network, h_t is the output cell state, and c_t is the storage unit state. NEWLSTM loads the information at each time step for transmission. The input at time t depends on the output at $t-1$. At time t, when new information x_t is input, if the gate f_t forgets to open, the past cell state may be discarded during this process. Otherwise, the input gate i_t is activated, the information is accumulated in the unit, and the output gate o_t judges whether it will be propagated to the final state h_t.

The internal structure of NEWLSTM is as follows.

Here, x is the input element and t is the time step. x represents the elements of time t, which can represent a set of codes for word features, dense vector features, or sparse features. The output layer represents the probability distribution of the label at time t, and its size is the same as the label size. The hidden layer at time t combines the output of the hidden layer at time $t-1$ to realize the transfer of information. After the memory unit is updated, the hidden layer calculates h_t according to the result obtained by the current output gate.

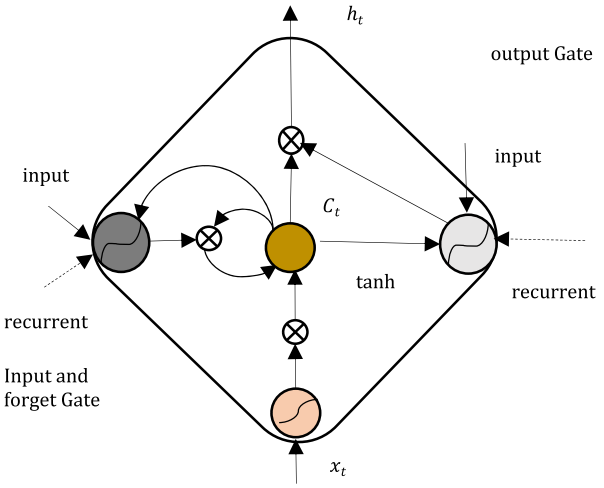


FIGURE 1. New model block.

NEWLSTM controls the transmission through the gated state, and combines the input gate and the forget gate to simplify the processing of the input layer. When new information is input each time, it is first combined with the previous cell state to simplify the calculation of the input gate. It forgets what it has learned and new information, and selectively deletes or adds information to the next unit by adjusting the information flow of the forget gate and output gate. The current information is combined with the past cell state, the amount of retained information is determined by the sigmoid function, and the information to be retained in the final state is determined according to the output gate o_t. NEWLSTM retains the dedicated memory storage unit of LSTM and retains the dependency relationship, which facilitates the discovery and establishment of long-term dependencies between input values.

IV. EXPERIMENTS

A. TRAINING PROCEDURE

In many applications, neural network models have shown excellent performance and the potential to surpass other computing technologies [18]. The neural network model receives the input sequence and attempts to accurately predict the output sequence based on the input sequence. A given neural network model can be trained using a large number of training examples and iteratively repeated until the neural network model can consistently draw similar inferences from the training examples that humans may make. It is difficult to train a recurrent neural network via back-propagation using time [29], which is mainly because the gradient propagated back through the network will vanish or grow exponentially.

This experiment builds a model on the basis of a neural network commonly used in sequence tasks to predict the probability of the next word appearing. The validity of our model is verified by comparing the two-layer original LSTM with two-layer LSTM variants.

By creating a dictionary using the original text, the articles, questions, and answers are mapped to a vocabulary and further mapped in the form of a vector. The results are imported into the two-layer LSTM through the input layer, and then the internal features of the sentence are learned from the information output by the LSTM using the attention mechanism. Attention overcomes the limitations in the codec architecture by making the network aware of the input attention position of each item in the output sequence. The final sequence label is predicted by preserving the intermediate outputs from each step of the input sequence of the encoder LSTM and training the model to learn to selectively focus on these inputs and associate them with items in the output sequence.

1) DATA

This experiment uses 10034964 pieces of data from the Penn Treebank Corpus. Adjacent sentences can be derived from paragraphs or adjacent paragraphs. Due to the particularity of the data, the output and unit state of the last hidden layer of each batch of data are used as the input of the next hidden layer and unit state.

In the experiment, we used perplexity (ppl) to measure the language model, implemented a two-layer LSTM network, and then, the LSTM results were used to predict the probability of the next word appearing. We calculate the cross entropy of each probability and the actual next word, then we sum them up and calculate the power of e to obtain the confusion ppl. By calculating the ppl, the probability of a sentence appearing is calculated based on each word, and the sentence length is used for normalization. It is expressed as follows:

$$\mathrm{PP}\,(\mathrm{S}) = \mathrm{P}(w_1, w_2 \ldots w_n)^{-\frac{1}{N}} \tag{9}$$

Here, S represents the sentence, N is the sentence length, and P(w_i) represents the probability of the i-th word. If the ppl decreases, P (w_i) becomes larger and the model becomes more accurate.

The variation curves of the training ppl of NEWLSTM are shown in Fig. 2, respectively. It can be seen that the initial prediction accuracy rate is very low. As the number of iterations increases, the validation ppl and training ppl first decrease significantly, the prediction accuracy rate greatly increases, and it finally stabilizes after the number of iterations is greater than 5.

2) RESULTS

The evaluation index of the experiment is the perplexity, which is calculated simultaneously for the training set and the test set. Some of the hyperparameters used in this paper are divided into small, medium, and large specifications, which are mainly derived from the experience of previous paper studies, such as the learning rate, the maximum gradient value used to control the exploding gradient, and the dropout ratio. In addition, some parameters are configured according to the depth of the LSTM and hardware conditions, such as the batch size and the number of GPUs. To verify the validity of our model, we use the three configurations of Small, Medium,

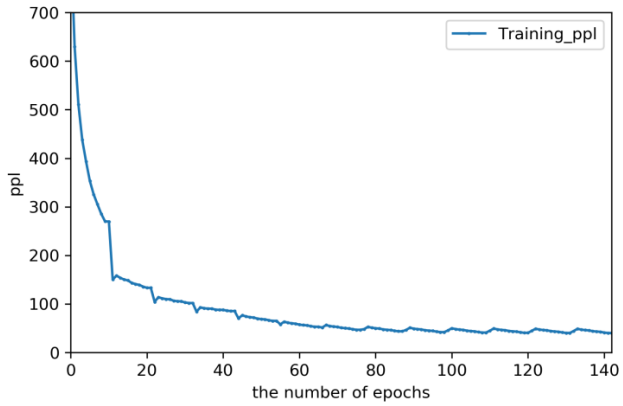


FIGURE 2. Training ppl of NEWLSTM.

and Large for comparison. The experimental configurations are as follows.

In our experiment, we chose the traditional LSTM and the classic LSTM variant for the experimental comparison. Salem and Fathi [30] proved that by simplifying the LSTM layer, three different LSTM variants, SLIM1, SLIM2, and SLIM3, can be obtained. It is possible to speed up the training and running time by changing the configuration of the parameters with limited performance changes. PhasedLSTM converges faster than traditional LSTM, improves the performance of LSTM, and reduces the amount of calculations by orders of magnitude at runtime [31]. HyperLSTM generates non-shared weights for LSTM, requires fewer parameters to learn, and obtains nearly new results on various sequence modeling tasks [32]. DA-LSTM is beneficial for processing sequential data with a non-uniform information distribution, learning the potential structure of the sequential input, and dynamically adjusting the number of operations performed, and it can greatly reduce the amount of calculations [21].

In this experiment, the CPU of the machine operates at 2.8 GHz, and its L3 cache is 6 MB. The size of the model and the number of training rounds are sequentially increased according to the three different specifications of Small, Medium, and Large. Through the prediction analysis of the language sequence, different experimental results are obtained.

As seen from the comparison of the results of the three different configurations, compared to LSTM, SLIM1, SLIM2, and SLIM3, the NEWLSTM model can better solve the problem of sparse words, the prediction of language sequences is good, and the overall performance exceeds that of the traditional LSTM. Experiments have shown that although the standard LSTM imposes a certain structure, the actual performance is not as good as that of NEWLSTM. Selecting the best hyperparameters in SLIM1, SLIM2, and SLIM3 may achieve powerful performance in each variable. However, it can be seen that the performance of NEWLSTM in the three specifications is relatively stable.

For PhasedLSTM, due to the incompatibility between the underlying structure involved in the sequence and the time assumptions of the phased LSTM, the effect cannot exceed that of NEWLSTM. In all experimental comparisons, the performance of DA-LSTM is not very good. HyperLSTM shows good performance under the small experimental configuration, but as the experimental specifications increase,

TABLE 1. Three configurations of the experiment.

Symbol	Quantity	Explanation
Small	num_layers=2	The number of neural network layers, representing the depth of the model
	batch_size=20	Batch size, which represents the number of samples for word training
	hidden_size=200	The number of hidden nodes in the hidden layer
	num_steps=20	Step size, which represents the length of the context
	init_scale=0.1	The initial values of the related parameters are randomly and uniformly distributed
	max_grad_norm=5.0	Maximum gradient value
	epoch_start_decay=4	The exponential decay modifies the learning rate of the vanishing gradient
	max_epoch=13	When epoch <max_epoch, lr_decay value = 1; when epoch> max_epoch, lr_decay gradually decreases
	dropout=0.0	The most effective regularization method in neural networks
	lr_decay=0.5	Learning rate decay
Medium	num_layers=2 batch_size=20 hidden_size=650 num_steps=35 init_scale=0.05 max_grad_norm=5.0 epoch_start_decay=6 max_epoch=39 dropout=0.5 lr_decay=0.8	
Large	num_layers=2 batch_size=20 hidden_size=1500 num_steps=35 init_scale=0.04 max_grad_norm=10.0 epoch_start_decay=14 max_epoch=55 dropout=0.65 lr_decay=1.0/1.15	

TABLE 2. ppl in Small configuration.

Models	Training ppl	Validation ppl	Test ppl	Time
SLSTM1	40.98	119.24	114.71	583.18
SLSTM2	40.68	118.48	113.94	590.27
SLSTM3	40.68	118.69	114.84	601.03
DA-LSTM	41.10	119.75	115.06	600.94
PhasedLSTM	40.42	118.94	113.44	609.59
HyperLSTM	40.38	119.36	113.57	619.45
LSTM	41.14	120.80	115.44	588.93
NEWLSTM	40.37	118.85	113.85	583.38

TABLE 3. ppl in Medium configuration.

Models	Training ppl	Validation ppl	Test ppl	Time
SLSTM1	40.73	117.90	114.97	604.90
SLSTM2	40.61	119.66	114.80	593.78
SLSTM3	40.46	119.38	114.27	595.78
DA-LSTM	40.73	118.86	115.29	597.71
PhasedLSTM	40.44	118.86	113.41	607.51
HyperLSTM	40.28	118.64	113.62	598.38
LSTM	40.57	119.87	115.05	584.40
NEWLSTM	40.32	118.14	114.55	580.60

TABLE 4. ppl in Large configuration.

Models	Training ppl	Validation ppl	Test ppl	Time
SLSTM1	40.03	118.71	113.67	607.62
SLSTM2	40.81	119.83	115.11	604.26
SLSTM3	40.72	120.47	115.44	598.29
DA-LSTM	40.20	117.79	114.21	599.89
PhasedLSTM	40.44	119.79	114.36	600.25
HyperLSTM	40.13	118.23	114.58	597.10
LSTM	40.81	119.69	115.13	581.42
NEWLSTM	40.29	118.87	113.63	578.91

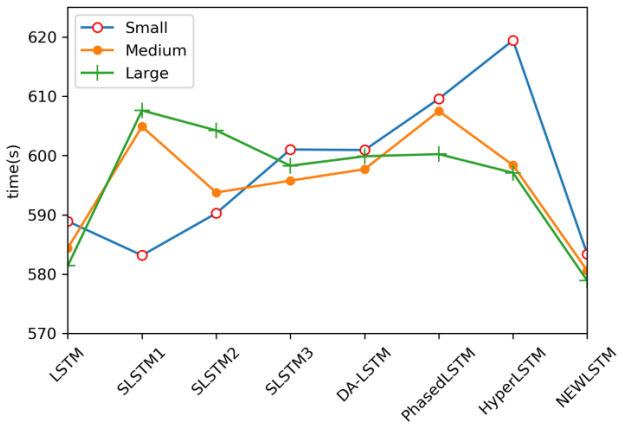


FIGURE 3. Average time taken for each training iteration.

its perplexity also increases, and the performance of the structure decreases.

Under the three different configuration specifications of Small, Medium, and Large, the average time taken by each LSTM iteration and variant during training is shown in Fig. 3.

The NEWLSTM simplifies the processing of the input gate in the LSTM and combines it with the forget gate to simplify the calculation complexity. As shown in Fig. 3, for the three different specifications of the NEWLSTM configuration, the time consumed during the iteration has been effectively reduced In addition, the performance is more stable than those of SLIM1, SLIM2, and SLIM3, and the time consumed does not fluctuate greatly. The time consumed by the traditional LSTM has not changed significantly with the change of the configuration specifications.

From the experiment, we can see that the convergence time of the single-cell structure is significantly shorter than that of the layered structure. Compared with the traditional LSTM and multiple LSTM variants, NEWLSTM maintains a simple structure, reduces the amount of calculations, and greatly reduces the convergence time.

V. CONCLUSION

Aiming at the shortcomings of the complex parameters in LSTM, the large amount of calculations, and the large loss of information transmitted too far away, this paper proposes an optimization model called NEWLSTM. It combines the current input information with the cell state after processing, and the information is loaded at each time. The steps are transmitted, then the information is accumulated in the cell unit, and finally the output gate judges whether the information is propagated to the final state. NEWLSTM improves the vanishing gradient problem of LSTM. By receiving the input information and predicting the language sequence, experiments prove that compared with the traditional LSTM model, the language sequence prediction accuracy of the NEWLSTM model is higher, which can effectively reduce the iteration time. Compared with SLIM1, SLIM2, and SLIM3, NEWLSTM does not experience large fluctuations in the training time due to configuration parameter changes, and its performance is relatively stable. The predictions of the model in natural language processing show a more optimized effect than those of LSTM and can better solve the problem of sparse words.

REFERENCES

[1] Y. Bengio and D. R. V. Pascal, "A neural probabilistic language model," *J. Mach. Learn. Res.*, vol. 3, pp. 932–938, Jan. 2000.
[2] T. Mikolov, S. Kombrink, L. Burget, J. Cernocky, and S. Khudanpur, "Extensions of recurrent neural network language model," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Prague, Czech Republic, May 2011, pp. 5528–5531, doi: 10.1109/ICASSP.2011.5947611.
[3] Y. Zhao, Y. Shen, and J. Yao, "Recurrent neural network for text classification with hierarchical multiscale dense connections," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 5450–5456.
[4] J. Koutník, K. Greff, F. Gomez, and J. Schmidhuber, "A clockwork RNN," in *Proc. 31st Int. Conf. Mach. Learn.*, 2014, pp. 1–4. [Online]. Available: http://home.process.com/Intranets/wp2.htp
[5] S. Takase, J. Suzuki, and M. Nagata, "Character n-gram embeddings to improve RNN language models," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 5074–5082. [Online]. Available: https://arxiv.org/abs/1906.05506
[6] T.-H. Wen, M. Gasic, D. Kim, N. Mrksic, P.-H. Su, D. Vandyke, and S. Young, "Stochastic language generation in dialogue using recurrent neural networks with convolutional sentence reranking," 2015, *arXiv:1508.01755*. [Online]. Available: http://arxiv.org/abs/1508.01755
[7] N. Pappas and J. Henderson, "Deep residual output layers for neural language generation," May 2019, *arXiv:1905.05513*. [Online]. Available: http://arxiv.org/abs/1905.05513
[8] K. Smagulova and A. James, "Overview of long short-term memory neural networks," in *Deep Learning Classifiers With Memristive Networks*. Cham, Switzerland: Springer, 2020, pp. 139–153.
[9] F. A. Gers, "Learning to forget: Continual prediction with LSTM," in *Proc. 9th Int. Conf. Artif. Neural Netw. (ICANN)*, Edinburgh, U.K., vol. 2, 1999, pp. 850–855, doi: 10.1049/cp:19991218.

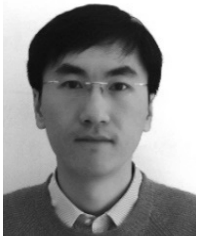
[10] F. M. Bianchi, E. Maiorino, M. C. Kampffmeyer, A. Rizzi, and R. Jenssen, "An overview and comparative analysis of recurrent neural networks for short term load forecasting," May 2017, *arXiv:1705.04378*. [Online]. Available: http://arxiv.org/abs/1705.04378
[11] A. T. Mohan and D. V. Gaitonde, "A deep learning based approach to reduced order modeling for turbulent flow control using LSTM neural networks," Apr. 2018, *arXiv:1804.09269*. [Online]. Available: http://arxiv.org/abs/1804.09269
[12] R. G. Hefron, B. J. Borghetti, J. C. Christensen, and C. M. S. Kabban, "Deep long short-term memory structures model temporal dependencies improving cognitive workload estimation," *Pattern Recognit. Lett.*, vol. 94, pp. 96–104, Jul. 2017.
[13] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," 2014, *arXiv:1409.3215*. [Online]. Available: http://arxiv.org/abs/1409.3215
[14] S. Chang, Y. Zhang, W. Han, M. Yu, X. Guo, W. Tan, X. Cui, M. Witbrock, M. Hasegawa-Johnson, and T. S. Huang, "Dilated recurrent neural networks," 2017, *arXiv:1710.02224*. [Online]. Available: http://arxiv.org/abs/1710.02224
[15] Y. Yu, X. Si, C. Hu, and J. Zhang, "A review of recurrent neural networks: LSTM cells and network architectures," *Neural Comput.*, vol. 31, no. 7, pp. 1235–1270, Jul. 2019.
[16] J. Hu and W. Zheng, "Transformation-gated LSTM: Efficient capture of short-term mutation dependencies for multivariate time series prediction tasks," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Budapest, Hungary, Jul. 2019, pp. 1–8.
[17] T. Guo, T. Lin, and N. Antulov-Fantulin, "Exploring interpretable LSTM neural networks over multi-variable data," in *Proc. ICML*, 2019, pp. 2494–2504.
[18] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," *Comput. Sci.*, pp. 338–342, Sep. 2014.
[19] K. Yao, T. Cohn, K. Vylomova, K. Duh, and C. Dyer, "Depth-gated recurrent neural networks," 2015, *arXiv:1508.03790v1*. [Online]. Available: https://arxiv.org/abs/1508.03790v1
[20] T. Cooijmans, N. Ballas, C. Laurent, Ç. Gülşehre, and A. Courville, "Recurrent batch normalization," Mar. 2016, *arXiv:1603.09025*. [Online]. Available: http://arxiv.org/abs/1603.09025
[21] Y. Zhang, K.-H. Chow, and S.-H. G. Chan, "DA-LSTM: A long short-term memory with depth adaptive to non-uniform information flow in sequential data," Jan. 2019, *arXiv:1903.02082*. [Online]. Available: http://arxiv.org/abs/1903.02082
[22] F. Ali, S. El-Sappagh, and D. Kwak, "Fuzzy ontology and LSTM-based text mining: A transportation network monitoring system for assisting travel," *Sensors*, vol. 19, no. 2, p. 234, 2019, doi: 10.3390/s19020234.
[23] D. Kent and F. Salem, "Performance of three slim variants of the long short-term memory (LSTM) layer," in *Proc. IEEE 62nd Int. Midwest Symp. Circuits Syst. (MWSCAS)*, Dallas, TX, USA, Aug. 2019, pp. 307–310.
[24] Z. Jie and W. Lu, "Dependency-guided LSTM-CRF for named entity recognition," in *Proc. Conf. Empirical Methods Natural Lang. Process., 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, 2019, pp. 1–3, doi: 10.18653/v1/D19-1399.
[25] M. de Lhoneux, M. Ballesteros, and J. Nivre, "Recursive subtree composition in LSTM-based dependency parsing," 2019, *arXiv:1902.09781*. [Online]. Available: http://arxiv.org/abs/1902.09781
[26] F. A. Gers and E. Schmidhuber, "LSTM recurrent networks learn simple context-free and context-sensitive languages," *IEEE Trans. Neural Netw.*, vol. 12, no. 6, pp. 1333–1340, Nov. 2001.
[27] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proc. IEEE*, vol. 105, no. 12, pp. 2295–2329, Dec. 2017.
[28] R. Pascanu and Y. Bengio, "Revisiting natural gradient for deep networks," *Comput. Sci.*, vol. 37, nos. 10–11, pp. 1655–1658, 2013.
[29] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, Oct. 1986.
[30] F. M. Salem, "SLIM LSTMs," 2018, *arXiv:1812.11391*. [Online]. Available: http://arxiv.org/abs/1812.11391
[31] D. Neil, M. Pfeiffer, and S.-C. Liu, "Phased LSTM: Accelerating recurrent network training for long or event-based sequences," in *Proc. NIPS*, 2016, pp. 3882–3890. [Online]. Available: https://arxiv.org/abs/1610.09513
[32] D. Ha, A. Dai, and Q. V. Le, "HyperNetworks," 2016, *arXiv:1609.09106*. [Online]. Available: http://arxiv.org/abs/1609.09106

QING WANG was born in Linyi, Shandong, China, in 1998. She is currently pursuing the degree with the School of Computer Science and Technology, Shandong University of Technology. Her research interests include machine learning, data mining, and natural language processing.

RONG-QUN PENG was born in Zibo, Shandong, China, in 1971. He received the B.S. degree in radio techniques from Shandong University, Jinan, in 1993, the M.S. degree in electrical engineering and automation from China Agricultural University, Beijing, in 2002, and the Ph.D. degree in information and communication engineering from the Nanjing University of Posts and Telecommunications, Nanjing, in 2012. Since 1993, he has been with the School of Computer Science and Technology, Shandong University of Technology, Zibo, as a Teacher and a Research Staff. His research interests include future network architectures and key technologies, 5G, and the IoT.

JIA-QIANG WANG received the M.S. degree from the Shandong University of Technology (SDUT), in 2013, and the Ph.D. degree from the Hebei University of Technology (Hebut), in 2018. He is currently a Postdoctoral Scholar with the Beijing Academy of Science and Technology (BJAST) and the Director of the Research Center for Big Data. His research interests include biometrics, machine learning, and data mining.

ZHI LI received the B.S. degree from the Southwest University of Political Science and Law (SWUPL), in 2001, and the M.S. degree from Xinjiang University (XJU), in 2010. He is currently pursuing the Ph.D. degree with the University of Chinese Academy of Sciences (UCAS). His research interests include machine learning and data mining.



HAN-BING QU received the M.S. degree from the Harbin Institute of Technology (HIT), in 2003, and the Ph.D. degree with the Institute of Automation, Chinese Academy of Sciences (CASIA), in 2007. He is currently a Professor with the Beijing Academy of Science and Technology (BJAST). He is also a member of the Intelligent Automation Committee of the Chinese Association of Automation (IACAA). His research interests include biometrics, machine learning, pattern recognition, and computer vision.