SANDSTONE POROSITY PREDICTION BASED ON GATED RECURRENT UNITS

HUI SONG[1], WEI CHEN*[2,3], HUA ZHANG[4], YANG WANG[5,6] and YAJUAN XUE [7]

[1] *College of Geophysics and Petroleum Resources, Yangtze University, Daxue Road 111, Caidian District, Wuhan 430100, P.R. China. 201400567@yangtzeu.edu.cn*
[2] *Key Laboratory of Exploration Technology for Oil and Gas Resources of Ministry of Education, Yangtze University, Daxue Road 111, Wuhan 430100, P.R. China.*
[3] *Hubei Cooperative Innovation Center of Unconventional Oil and Gas, Daxue Road 111, Caidian District, Wuhan 430100, P.R. China. *chenwei_yangtze@126.com*
[4] *School of Geophysics and Measurement-control Technology, East China University of Technology, Nanchang 330013, P.R. China.*
[5] *Faculty of Materials Science & Engineering, Hubei University, Wuhan 430062, China.*
[6] *Tianshu New Energy Material Industry Research and Design Institute, Hubei University, Wuhan 430062, P.R. China.*
[7] *School of Communication Engineering, Chengdu University of Information Technology, Chengdu 610225, P.R. China.*

(Received March 1, 2019; revised version accepted February 28, 2020)

ABSTRACT

Song, H., Chen, W., Zhang, H., Wang, Y. and Xue, Y.J., 2020. Sandstone porosity prediction based on gated recurrent units. *Journal of Seismic Exploration*, 29: 371-388.

Sandstone porosity prediction is a difficult task because of the heterogeneity of reservoir rock. Deep learning has been widely used in various fields, but it is rarely used for sandstone porosity prediction. Recurrent neural networks (RNNs) are currently very popular algorithms for deep learning and have achieved good performance in processing sequence data, such as speech recognition and machine translation. Since sandstone porosity prediction belongs to sequence data prediction, this paper proposes to use the gated recurrent units (GRUs), which is a variant of RNNs, to predict sandstone porosity using the well logs data. Six well logs data are divided into training set, validation set and testing set. We apply three deep learning architectures including convolutional neural networks (CNNs), recurrent neural networks (RNNs) and gated recurrent units (GRUs) to predict sandstone porosity. Results show that GRUs can extract the nonlinear characteristics of data more effectively and are more suitable for sandstone porosity prediction.

KEY WORDS: convolutional neural networks (CNNs), recurrent neural networks (RNNs), gated recurrent units (GRUs), sandstone porosity prediction.

INTRODUCTION

Porosity refers to the volume of pores contained in reservoir rocks, reflecting the ability of rocks to store fluids. It is an essential petrophysical parameter for evaluating reservoir properties and its estimation is susceptible to noise (Chen et al., 2016a,b; Siahsar et al., 2017; Chen et al., 2017a; Liu and Chen, 2019). Therefore, it is necessary to improve the prediction accuracy of reservoir porosity. Porosity of small cores or cuttings is measured in the laboratory, which is a time-consuming process. In addition to time-consuming, coring is also expensive. In many cases, coring is limited to certain intervals of the reservoir, which makes it difficult to obtain sufficient physical analysis data for reservoir evaluation. Using well logs data to estimate porosity is a relatively inexpensive method compared to expensive core data. Different well logs data can reflect the change of reservoir porosity to some extent. According to the correlation between well logs data and porosity, some regression equations are proposed, but the prediction accuracy of this method is not high, because there is no absolute linear correlation between well logs data and reservoir porosity. Machine learning has many excellent algorithms for dealing with nonlinear problems, such as support vector machines (SVM), K-means clustering, and artificial neural networks (ANNs) (Pal, 2006; Cracknell and Reading, 2013; Zhao et al., 2014; Chen et al., 2017b; Reynen and Audet, 2017; Paitz et al., 2017), and has shown great advantages in reservoir prediction, such as porosity, permeability, and saturation (Rogers et al., 1995; Helle et al., 2001; Ahmadi et al., 2013; Iturrar´an-Viveros and Parra, 2014; Fattahi and Karimpouli, 2016; Zhang et al., 2018b).

Deep learning is a new field in machine learning, and its concept stems from the research of ANNs (Hinton et al., 2006). Compared with the traditional shallow learning, deep learning can realize complex function approximation and layer-by-layer feature transformation by constructing a machine model structure with many hidden layers. At present, deep learning is widely used in computer vision, natural language processing, and speech recognition (Kavukcuoglu et al., 2010; Mikolov et al., 2013; Hinton et al., 2012). Deep learning includes many popular network structures, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs). Since CNNs can effectively extract features of an image through multiple convolution kernels, it is often applied to image processing, such as image denoising and image classification (Zhang et al., 2017, 2018a; Yuan et al., 2018). Compared with CNNs, RNNs specialize in solving time series problems. RNNs have the characteristics of memory and parameter sharing, which makes it possible to learn the nonlinear characteristics of the sequence with high efficiency. When it comes to certain sequential machine learning tasks, such as speech recognition, language modeling, and machine translation, other algorithms are hard to exceed the effects of RNN processing. RNNs have the function of remembering previous information. However, due to the problem of gradient disappearance, memory is short-term memory and cannot store long-term information (Bengio et al., 1994). Gating algorithm is an important method to deal with long-distance dependence and long-short memory networks (LSTMs) are the earliest proposed RNNs gating algorithms (Hochreiter and Schmidhuber, 1997). Hidden layer of LSTMs contains three gates: input gate, forgetting gate, and output gate. Three gates control the transmission of information in the hidden layer: the input gate determines what new information is stored in the cell state; the forgetting gate determines what information is discarded from the cell state; the output gate determines what information is output from cell state. Gated recurrent units (GRUs) are very popular variants of LSTMs, and its corresponding hidden layer contains only 2 gates: update gate and reset gate (Chung et al., 2014). The GRUs are simpler than LSTMs, which improves learning efficiency, but can achieve almost the same functions as LSTMs. The function of the reset gate is similar to that of the input gate of the LSTMs, and the update gate realizes the functions of the forgetting gate and the output gate of the LSTMs.

In this paper, we propose three different deep learning algorithms for sandstone porosity prediction, such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and gated recurrent units (GRUs). We focus our research on GRUs and conduct experiments to determine the optimal network hyper-parameters for GRUs. Moreover, we compare the performance the three models on different blind testing wells. The final results show that GRUs have the best performance in sandstone porosity prediction.

THEORY

Convolutional neural networks

Convolutional neural networks (CNNs) are feedforward neural networks, which is widely used in the field of computer vision. The difference between a convolutional neural network and a simple neural network is that it contains a special extraction structure consisting of a convolutional layer and a pooling layer. The convolutional layer is the core of CNNs, and most of the computational load is at this layer. In a convolutional layer, the maps of the previous layer are convolved by the learnable convolution kernels, and then the results of the convolution are input to the activation function to obtain the output feature maps. Each output feature map may be results of the combined convolution of multiple input feature maps, and the features learned by each convolution kernel are different. The *j*-th output feature map of the *l*-th layer can be expressed as:

$$x_j^l = f(\sum_{i \in M_j} x_i^{l-1} * w_{ij}^l + b_j^l) \quad , \tag{1}$$

where, x_i^{l-1} represents the *i*-th output feature map at the *l*-1 layer, w_{ij}^l represents the weights learned by the convolution kernel used to extract features. Since a convolution kernel cannot extract all features, multiple

RNN is known as recurrent because it performs the same operation on every element of a sequence, and its subsequent output depends on previous computations. Another way to understand RNNs is that RNNs have some "memory" that captures some of the information that were previously calculated. A typical RNNs structure include input layer, hidden layer and output layer, as shown in Fig. 1a, which shows that RNNs can be viewed as a process in which the same neural structure is repeated multiple times. Loop diagrams are concise, but it cannot not describe the computational flow. The expansion diagram can clearly describe the calculation process, as shown in Fig. 1b, which helps us understand how RNNs work. Fig. 1b shows that the parameters (W_{hx}, W_{hh}, W_y) of RNNs are shared at all times, while the traditional neural networks use different parameters at each layer, which reflects the fact that we are performing the same operation at every time step with different inputs. This structure greatly reduces the number of parameters to learn. For input vector sequence, x_t, the calculation of the hidden sequence vector h_t and the output sequence vector y_t can be expressed as:

$$\begin{aligned} h_t &= tanh(W_{hh}h_{t-1} + W_{hx}x_t + b_h) \\ &= tanh([W_{hh}, W_{hx}][{}^{h_{t-1}}_{x_t}] + b_h) \\ &= tanh(W_h[h_{t-1}, x_t] + b_h) \end{aligned} \tag{4}$$

$$y_t = \sigma(W_y h_t + b_y) \tag{5}$$

where W and b denote weights and bias(for example, W_h and b_h represent weights and bias of hidden layer, respectively), σ denotes the sigmoid function.

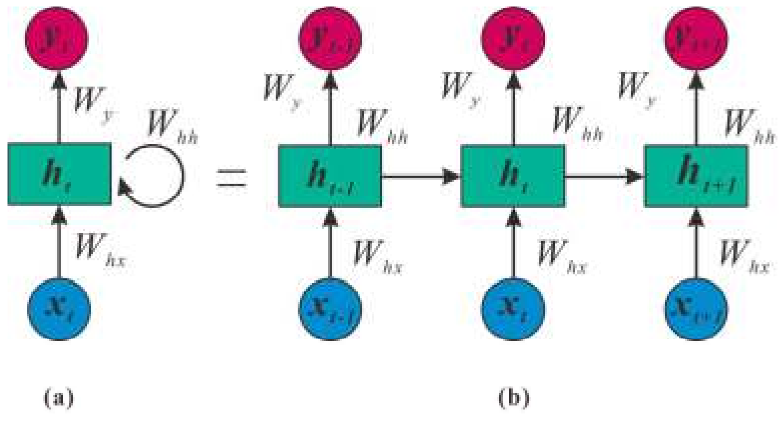


Fig 1. (a) Loop diagrams of RNNs. (b) Expansion diagram of RNNs.

convolution kernels are often used in the same convolutional layer to learn different features. M_j represents a selection set of input feature maps, and all or a specific combination of feature maps can be selected from M_j as input feature maps, $b_j{}^l$ represents the j-th bias of the l-th layer, symbol * represents convolution operation, f represents the nonlinear activation function rectified linear unit (Relu), which is widely used in CNNs. Relu solves the problem of gradient disappearance and speeds up the convergence of backpropagation. Pooling layer is the fix function in CNNs, which can be regarded as a special convolution kernel without any parameters to be learned. The pooling layer is generally used after the convolutional layer for feature dimensionality reduction and reduction of the number of parameters.

We hope that the difference between the predicted value and the desired value is as small as possible. A loss function is defined as:

$$L(\theta) = \frac{1}{N}\sum_{k=1}^{N}(\hat{y}_\theta(x_k) - y_k)^2 \quad , \tag{2}$$

where $\theta = \{w,b\}$ represents weights and bias, N is the number of data used for model training, $\hat{y}_\theta$represents the network output value when the input is x_k, y_k represents the desired result, which is the label of the data, k represents the serial number of the sample, L represents the degree of difference between the predicted value and the desired value. The purpose of the neural network is to learn specific parameters θ to minimize L. The backpropagation algorithm and the gradient descent method are used together to update parameters θ. The network parameters are updated as follows:

$$\theta \leftarrow \theta - \eta\frac{\partial L}{\partial \theta} \quad , \tag{3}$$

where η denotes learning rate.

Recurrent neural networks

Now that we have CNNs, why do we have recurrent neural networks (RNNs)? And the reason for that is very simple, whether it's a convolutional neural network or a deep neural network, their premise is that the previous inputs are completely unrelated to the latter inputs. However, there are cases where the previous information are related to the later information, or the latter information are related to the previous information. For example, if you want to predict the porosity of a well at a certain depth, you need to use the porosity information at the previous depth, because porosity does not exist independently for a particular interval.

In order to increase the expressive ability of the models, multiple hidden layers are stacked in the hidden layer of the RNNs. Such RNNs become deep recurrent neural networks (DRNNs), as shown in Fig. 2, which shows that the DRNNs repeat the hidden layer of each time step multiple times. The parameters of same hidden layer are shared at different time step, but the parameters of different hidden layers can be different at the same time step.

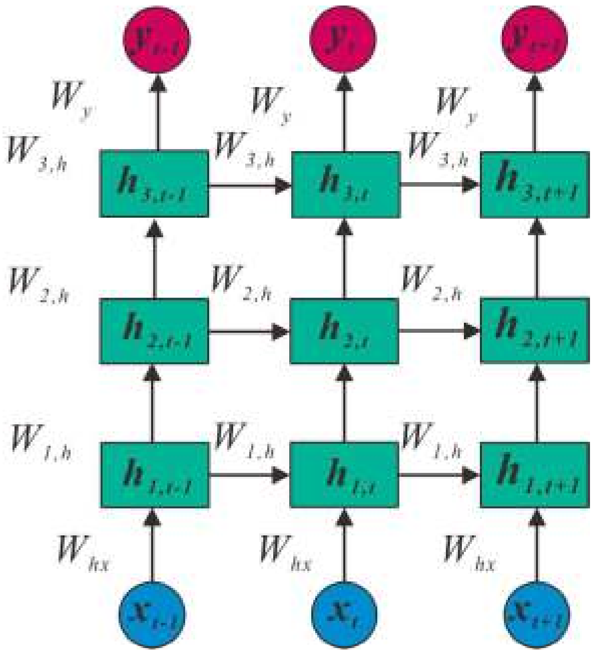


Fig. 2. Multi-layer RNNs structure.

Gated recurrent units

The hidden layer has no ability to measure the value of information in RNNs. Therefore, the hidden layer treats the status information of each time step as the same, which results in that the hidden layer often stores some useless information, while the really useful information is squeezed out by the useless information. The most important idea of GRUs is to introduce the gated mechanism to control the transmission of information, as shown in Fig. 3. The process of information transmission in the hidden layer of GRUs is as follows:

$$z_t = \sigma(W_z\left[h_{t-1}, x_t\right] + b_z) \tag{6}$$

$$r_t = \sigma(W_r\left[h_{t-1}, x_t\right] + b_r) \tag{7}$$

$$\tilde{h}_t = tanh(W_h[h_{t-1} \otimes r_t, x_t] + b_h) \tag{8}$$

$$h_t = (1 - z_t) \otimes h_{t-1} + z_t \otimes \tilde{h}_t \quad , \tag{9}$$

where z_t and r_t denote update gate and reset gate, respectively. $\tilde{h}_t$ denotes the candidate sequence vector, ⊗ denotes elementwise product. When the values of z_t and r_t are both 1, the update gate and reset gate in the GRUs will no longer 'work'. In this case, the transmission of information in the hidden layer becomes as follows:

$$\tilde{h}_t = tanh(W_h[h_{t-1}, x_t] + b_h) \tag{10}$$

$$h_t = \tilde{h}_t \quad . \tag{11}$$

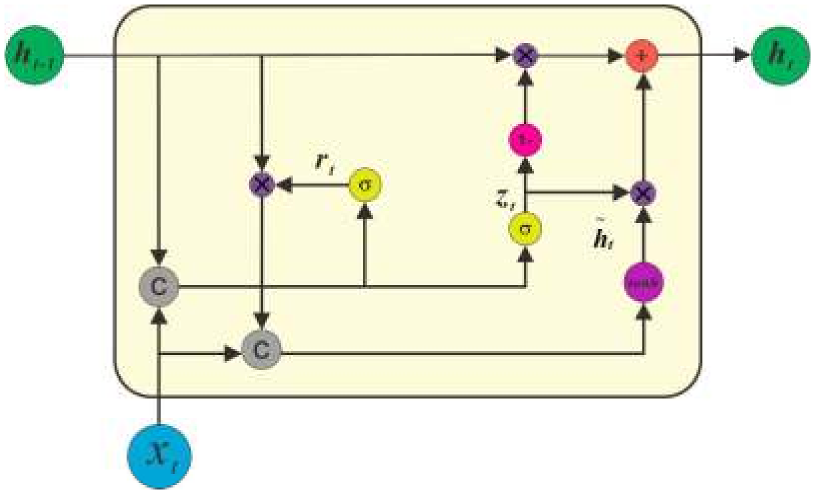


Fig. 3. The hidden layer structure of the GRUs.

DETAILS OF MODEL TRAINING

Porosity prediction is a regression problem, which is a kind of supervised learning. Supervised learning uses existing data (including input and output) to train the model, and then new data is fed into the trained model to get the output. If the output is continuous, it is called regression; if the output is discrete, it is called classification. In our study, we use natural gamma ray (GR), acoustic wave (AC), density (DEN), and shale content (SH) as inputs and porosity as output.

In our study, well logs data are obtained from an oil field. The reservoir is a sedimentary environment of the salt lake. The lithology is dominated by fine sandstone with medium separation. The sandstone composition is mainly quartz, followed by feldspar, which is feldspar quartz sandstone. The reservoir type is pore type with less cement content. 5920 samples are selected from 6 wells as data sets for the model, and 6 wells are named W1 to W6.

In supervised deep learning, in order to prevent over-fitting, the data set is generally divided into three independent parts: training set, validation set, and testing set. The training set is used to determine the common parameters of the model, such as weights and bias, which are obtained by automatic learning. The validation set is used to determine the model hyper-parameters, such as the number of network layers and the number of network units, which are manually set. The testing set does not participate in model training and is only used to test the generalization ability of the model. In our study, the testing set consisted of all the samples in the well W3. 20% of the remaining data are randomly selected as the validation set, and the rest of the data are used as a training set for training the model.

The dimensionality difference of the input data is relatively large, which leads to a slow or even non-convergence in the process of model training. Therefore, we need to use the following formula to normalize the data so that our data are as close to the same dimension as possible.

$$Y = \frac{X - X_{min}}{X_{max} - X_{min}} \quad . \tag{12}$$

The formula applies the linearization method to scale the original data equally to the range of [0,1]. Where, Y is the normalized value, X is the original value, X_{max} is the maximum value in the original data, X_{min} is the minimum value in the original data.

Models use the RMSProp optimizer and set the batch size as 360 for training. Let $360 = T \times n$, where T represents time step, n represents how many groups of T have been calculated, and then the network parameter is updated once. The training will be terminated if the validation error does not decrease after 3 iterations. Mean absolute error (MAE) and goodness of fit (R^2) are used to evaluate the predictive performance of the models.

$$\mathrm{MAE} = \frac{1}{N}\sum_{k=1}^{N}\left| p_k - \hat{p}_k \right| \tag{13}$$

$$R^2 = 1 - \frac{\sum_{k=1}^{N} (p_k - \hat{p}_k)^2}{\sum_{k=1}^{N} (p_k - \bar{p})^2} , \tag{14}$$

where p_k, $\hat{p}_k$ and $\bar{p}$ represent the actual value, predicted value and average value, respectively. The closer MAE is to 0 and R^2 is to 1, the better the models predict.

EXPERIMENTS

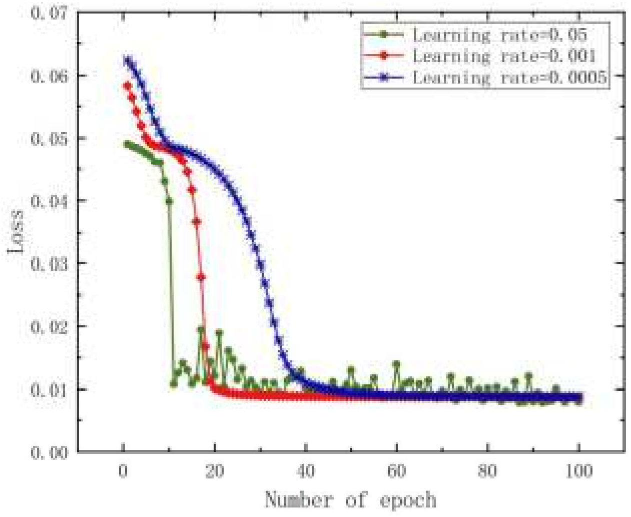
Hyper-parameters in deep learning play an important role in model performance. Finding the optimal hyper-parameters of the model is called tuning. The GRU model is the most complex model in this study. Thus, in the following research, we will focus on GRUs for model tuning, such as learning rate, number of hidden layers, number of network units and time step. We choose the TensorFlow framework of deep learning to implement the models. In order to reduce the random effects on experiment results, the average of 5 runs is taken as the final evaluation result.

Learning rate

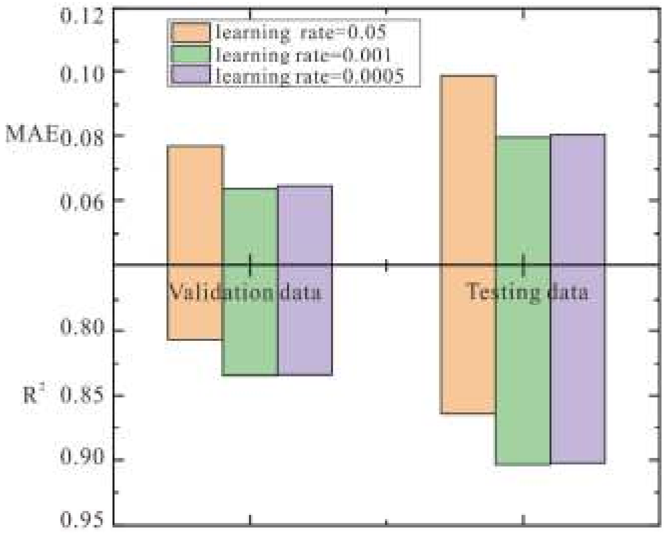
The learning rate is one of the important hyper-parameters that affect the performance of the model. If the learning rate is set too high, the model cannot converge to the local minimum. If the learning rate is set too small, it can ensure that no local minimum is missed, but it will take longer to make the model converge. Therefore, it is very critical to set the appropriate learning rate for model training. Fig. 4a shows that when the learning rate is 0.001, GRUs can not only ensure the stability of training, but also reduce the training time. Fig. 4b shows the experiment results of learning rate of 0.05, 0.001, and 0.0005 for GRUs. Results show that the GRUs have best performance on the validation and testing data with the learning rate of 0.001.

Hidden layers

For simple problems or small data volumes, one or two hidden layers can achieve good results. For more complex problems or large data volumes, you can increase the appropriate number of hidden layers to achieve better performance. Fig. 5 shows the experiment results of the number of hidden layers for GRUs. In this study, it is evident that the one hidden layer is sufficient for completing the network task, and the GRUs are more difficult to train and the performance is worse as the number of hidden layers increases.

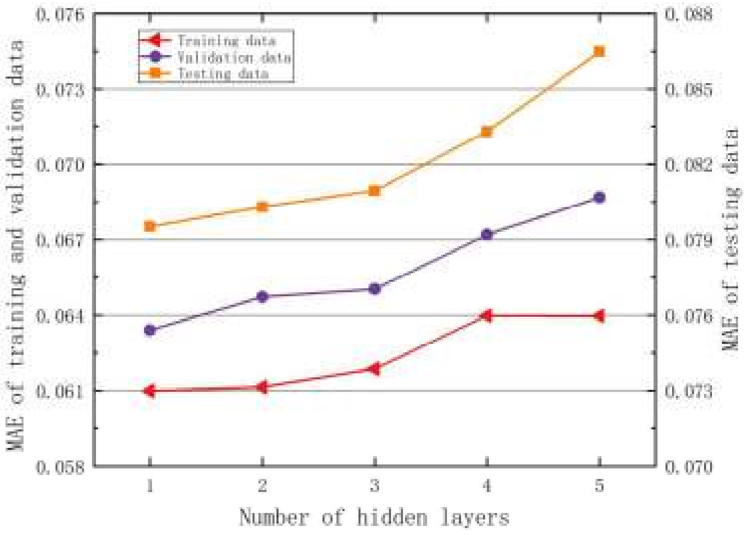


(a)

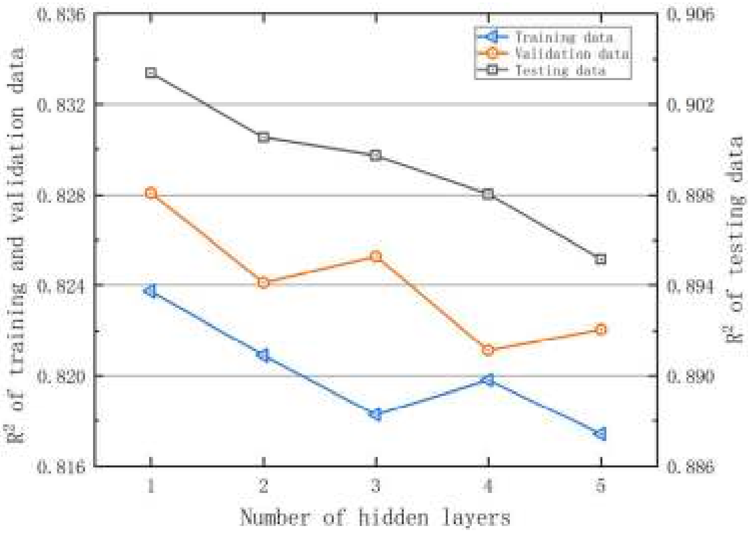


(b)

Fig. 4. (a) Effects of different learning rate on the loss function. (b) Effects of different learning rate on training, validation, and testing data for GRUs when using W3 as the blind testing well. The model uses one hidden layer with 20 network units and is trained with a time step of 6.



(a)

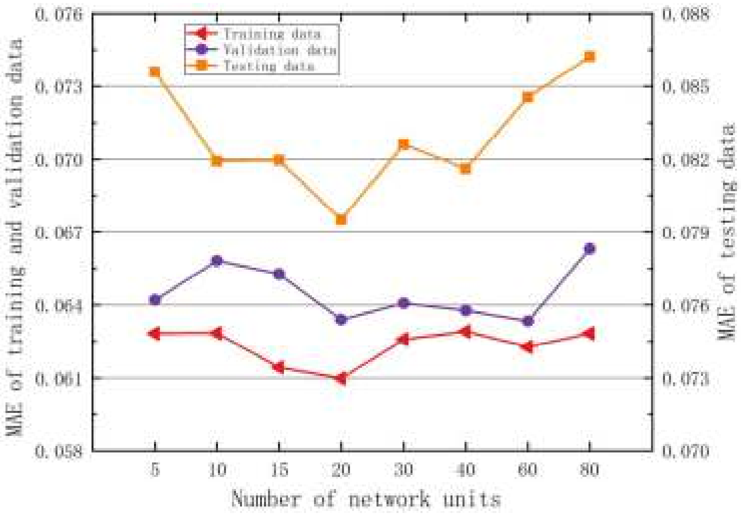


(b)

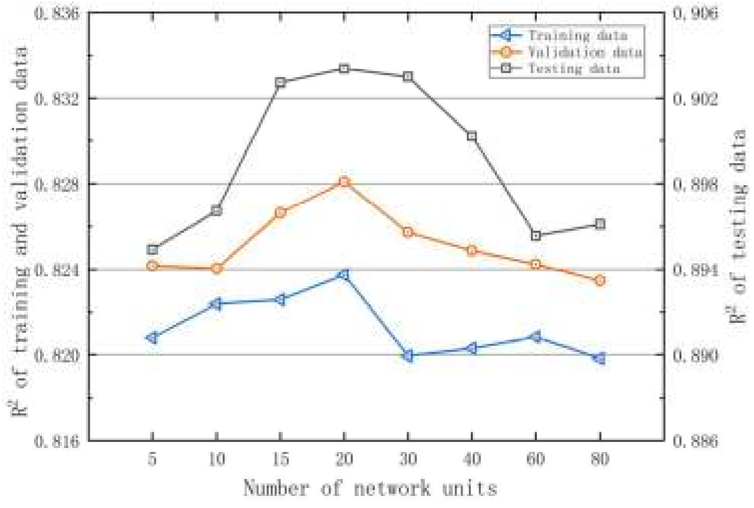
Fig. 5. Effects of number of hidden layers on training, validation, and testing data for GRUs when using W3 as the blind testing well. The model uses 20 network units and is trained with a learning rate of 0.001 and time step of 6.

Network units

According to the network task, the number of network units in the input layer and output layer is easily determined. For sandstone porosity prediction, the number of network units in the input layer is set as 4, and the number of network units in the output layer is set as 1. The number of network units in the hidden layer cannot be determined directly. Thus, the optimal number of network units must be selected experimentally. Fig. 6 shows the experiment results of the number of network units for GRUs. Results show that the GRUs have best performance on all the training set, validation set and testing set when the number of network units is 20.



(a)

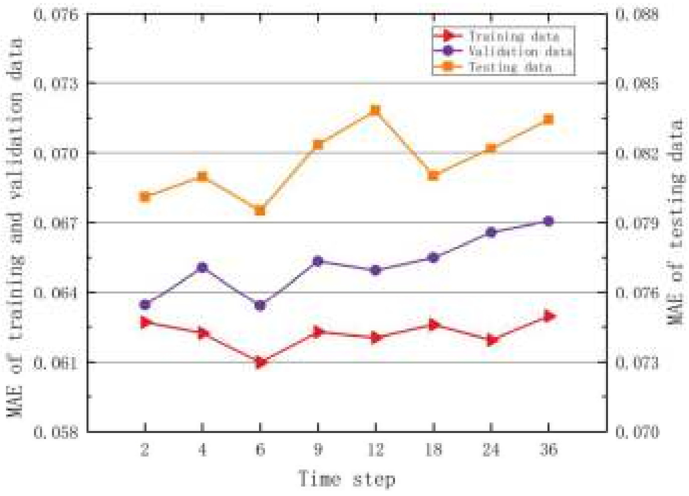


(b)

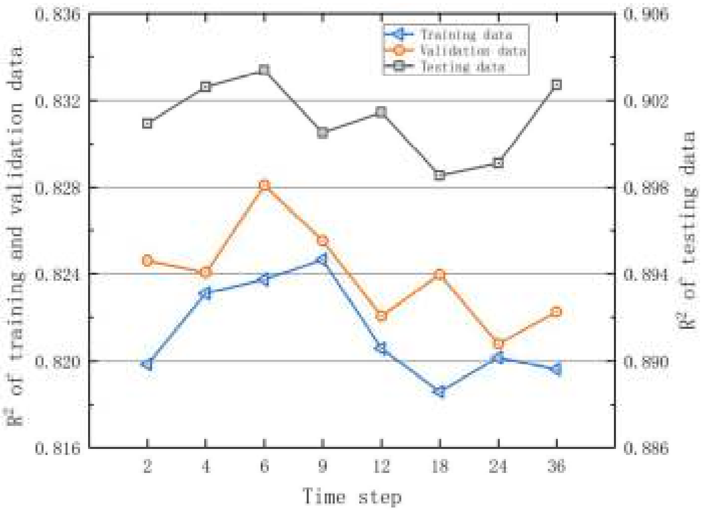
Fig. 6. Effects of number of network units on training, validation, and testing data for GRUs when using W3 as the blind testing well. The model uses one hidden layer and is trained with a learning rate of 0.001 and time step of 6.

Time step

Fig. 7 shows the experiment results of the time step for GRUs. The time step is not sensitive to the MAE indicator in training data set and validation data set. The GRUs have the best performance on the validation, and testing data set when the time step is 6.



(a)



(b)

Fig. 7. Effects of time step on training, validation, and testing data for GRUs when using W3 as the blind testing well. The model uses one hidden layer with 20 network units and is trained with a learning rate of 0.001.

Generalization ability of the model

Through the above experiments, the optimal hyper-parameters of GRUs are determined. Fig. 8 shows the prediction results of GRUs and Fig. 9 shows the degree of fit between the predicted sandstone porosity and the actual sandstone porosity on W3. It can be seen that the overall trend of sandstone porosity change is basically predicted correctly, although in some places it does not predict well. To test the generalization ability of the model, we take turns to use other wells as the blind testing well and compared with the results of CNNs and RNNs test. CNNs use two convolutional layers, each containing 20 convolution kernels of 1×3. Pooling layers and fully connected layers are not used. The hyper-parameter settings of the RNNs are the same as GRUs. Fig. 10 shows the performance of three models test on other blind testing wells. In terms of R^2 indicators, GRUs perform best on all blind testing wells. In terms of MAE indicators, GRUs perform better than RNNs on all blind testing wells, while it performs worse than CNNs on W2 and W6. Overall, the GRUs perform best on all blind testing wells (Table 1).

Table 1. The average performance of different models on all blind testing wells.

Model	The average of MAE			The average of R^2		
	Training data	Validation data	Testing data	Training data	Validation data	Testing data
CNNs	0.0572	0.0602	0.0848	0.8540	0.8441	0.8639
RNNs	0.0582	0.0606	0.0837	0.8477	0.8427	0.8671
GRUs	0.0571	0.0606	0.0818	0.8497	0.8473	0.8695

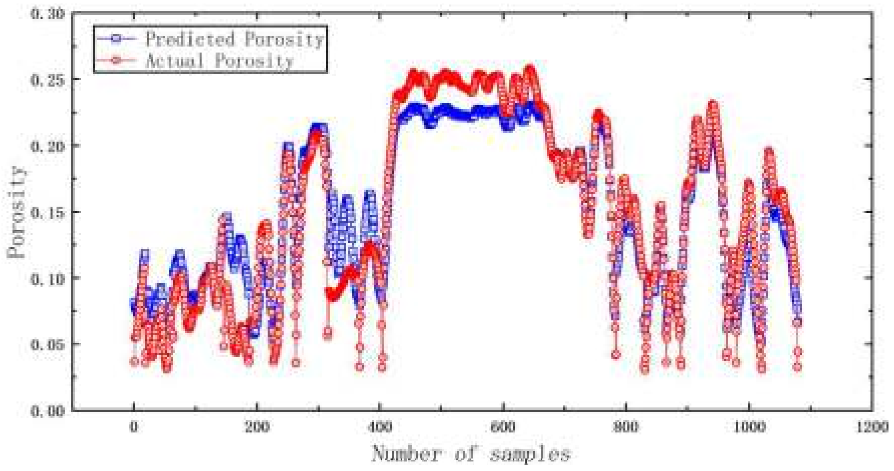


Fig. 8. Results of sandstone porosity prediction on W3 by GRUs.

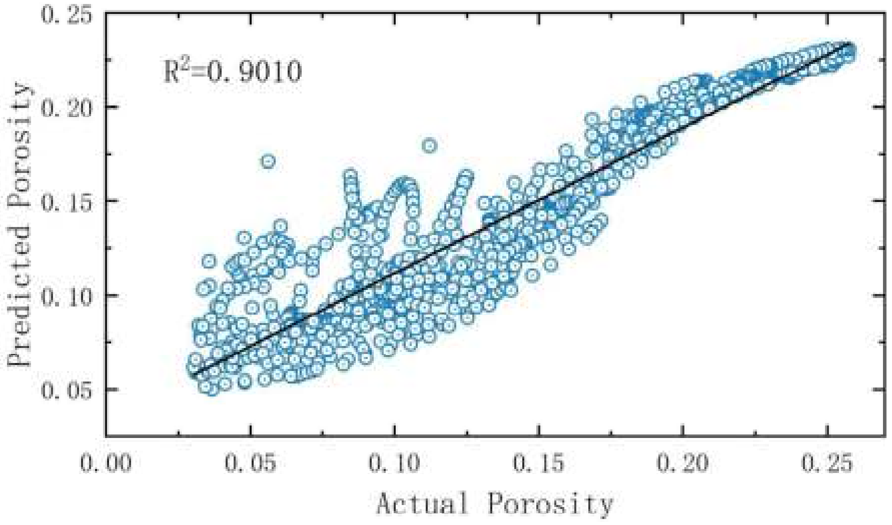
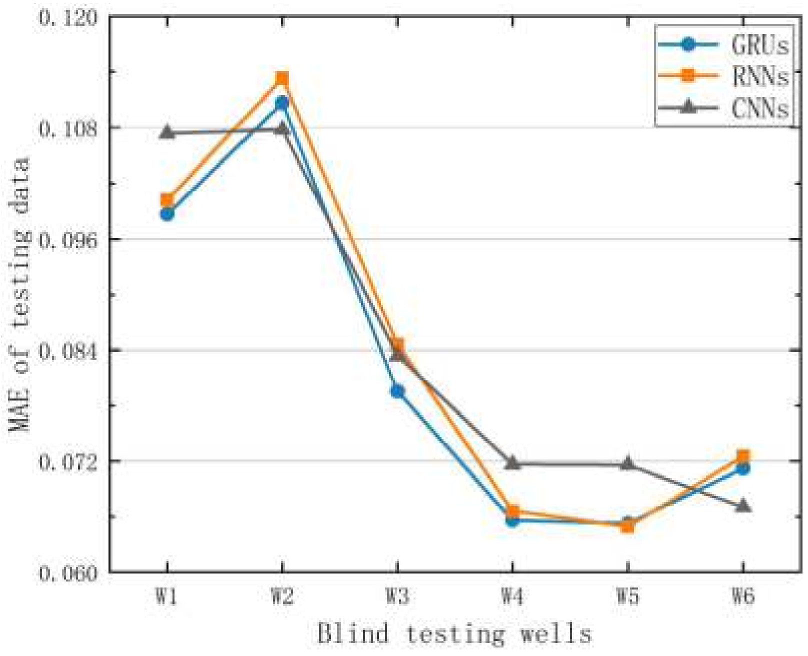


Fig. 9. Cross plot of predicted porosity versus actual porosity on W3 by GRUs.

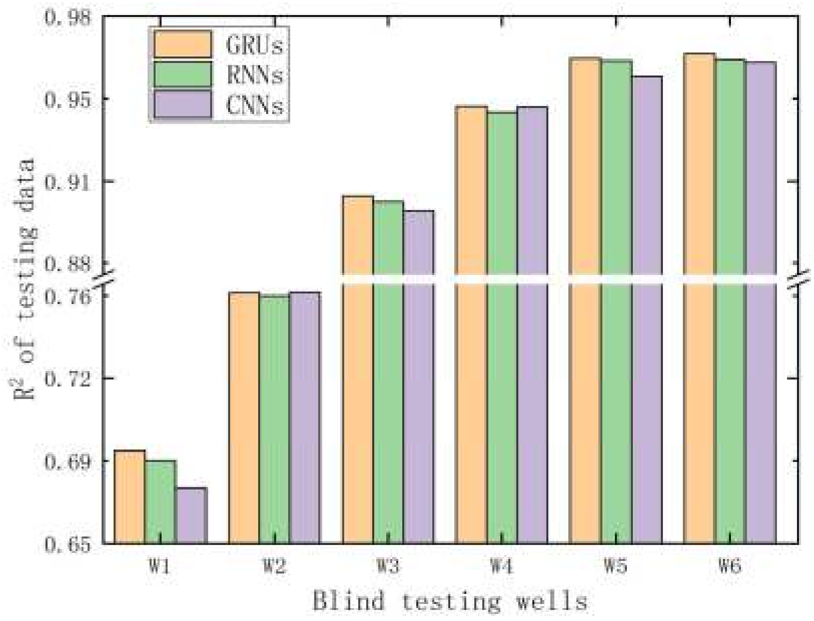
DISCUSSION

Compared with the other two models, GRUs have best performance in sandstone porosity prediction. However, it takes a lot of time to determine the optimal hyper-parameters of the GRUs, such as learning rate, number of hidden layers, number of network units, and time step. However, there are still a few tricks for model tuning. For simple problems such as porosity prediction, the number of hidden layers can be directly selected as one or two, and then gradually determine other hyper-parameters. Three models have different performances on different Wells. However, the performance gap on W1 and W5 is large. The reason for this phenomenon may be that the selected data set cannot represent the sandstone porosity information of the whole region. Therefore, how to select representative samples is a challenging problem.

Compared with RNNs, GRUs are able to remember more useful information about previous moments by using gates. Therefore, GRUs are able to take full advantage of changes in the time dimension of data. CNNs have the characteristics of local perception and can make full use of the spatial connection between different characteristic parameters. Fig. 10a shows that in terms of MAE indicators, CNNs perform better than GRUs on some blind testing wells. CNNs and GRUs have their own advantages to some extent. For sandstone porosity prediction, it is necessary to consider not only the spatial relationship between different characteristic parameters, but also the change of data in time dimension. Therefore, how to fully combine the advantages of the two models to build a model capable of expressing spatial and temporal characteristics of data will be studied in the future.



(a)



(b)

Fig. 10. (a) MAE of GRUs, RNNs, and CNNs on different blind testing wells. (b) R^2 of GRUs, RNNs, and CNNs on different blind testing wells.

CONCLUSION

GRUs are good at processing sequence data and are used for sandstone porosity prediction for the first time with good performance. Compared with RNNs and CNNs, GRUs have best performance in sandstone porosity prediction. Compared with standard RNNs, GRUs can better extract data features by introducing gated unit to control information transmission. We also find that one hidden layer is sufficient for sandstone porosity prediction. Sandstone porosity prediction is essentially a regression problem, and GRUs can also be used for other regression problems and classification problems, such as permeability prediction, saturation prediction, and lithology prediction.

ACKNOWLEDGEMENT

The research in this paper is partially supported by the National Natural Science Foundation of China (Grant Nos. 41804140, 41664006, 41874126, 41404102), the Open Fund of Key Laboratory of Exploration Technologies for Oil and Gas Resources (Yangtze University), Ministry of Education (Grant No. PI2018-02), the Natural Science Foundation of Jiangxi Province (Grant No. 20181BAB203028) and the Sichuan Youth Science and Technology Foundation (Grant No. 2016JQ0012). Special thanks also go to the SINOPEC Key Laboratory of Geophysics for their support.

REFERENCES

Ahmadi, M.A., Zendehboudi, S., Lohi, A., Elkamel, A. and Chatzis, I., 2013. Reservoir permeability prediction by neural networks combined with hybrid genetic algorithm and particle swarm optimization. Geophys. Prosp., 61, 582-598.

Bengio, Y., Simard, P. and Frasconi, P., 1994. Learning long-term dependencies with gradient descent is difficult. IEEE Transact. Neural Netw., 5: 157-166.

Chen, W., Xie, J., Zu, S., Gan, S. and Chen, Y., 2016a. Multiple-reflection noise attenuation using adaptive randomized-order empirical mode decomposition. IEEE Geosci. Remote Sens. Lett., 14: 18-22.

Chen, W., Yuan, J., Chen, Y. and Gan, S., 2017a. Preparing the initial model for iterative deblending by median filtering. J. Seismic Explor, 26: 25-47.

Chen, Y., Hill, J., Lei, W., Lefebvre, M., Tromp, J., Bozdag, E. and Komatitsch, D., 2017b. Automated time-window selection based on machine learning for full-waveform inversion. Expanded Abstr., 87th Ann. Internat. SEG Mtg., Houston: 1604-1609.

Chen, Y., Huang, W., Zhang, D. and Chen, W., 2016b. An open-source matlab code package for improved rank-reduction 3D seismic data denoising and reconstruction. Comput. Geosci., 95: 59-66.

Chung, J., Gulcehre, C., Cho, K. and Bengio, Y., 2014. Empirical evaluation of gated recurrent neural network on sequence modeling. arXiv preprint arXiv:1412.3555.

Cracknell, M.J. and Reading, A.M. 2013. The upside of uncertainty: Identification of lithology contact zones from airborne geophysics and satellite data using random forests and support vector machines. Geophysics, 78(3): WB113-WB126.

Fattahi, H. and Karimpouli, S., 2016. Prediction of porosity and water saturation using pre-stack seismic attributes: a comparison of Bayesian inversion and computational intelligence methods: Comput. Geosci., 20: 1075-1094.

Helle, H.B., Bhatt, A. and Ursin, B., 2001. Porosity and permeability prediction from wire-line logs using artificial neural networks: a north sea case study. Geophys. Prosp., 49: 431-444.

Hinton, G., Deng, L., Yu, D., Dahl, G.E., Mohamed, A.-R., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P. and Sainath, T.N., 2012. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. IEEE Sign. Process. Mag., 29: 82-97.

Hinton, G.E., Osindero, S. and Teh, Y.-W., 2006. A fast learning algorithm for deep belief nets. Neural Comput., 18: 1527-1554.

Hochreiter, S. and Schmidhuber, J., 1997. Long short-term memory. Neural Comput., 9: 1735-1780.

Iturrar´an-Viveros, U. and Parra, J.O., 2014. Artificial neural networks applied to estimate permeability, porosity and intrinsic attenuation using seismic attributes and well-log data. J. Appl. Geophys., 107: 45-54.

Kavukcuoglu, K., Sermanet, P., Boureau, Y.-L., Gregor, K., Mathieu, M. and Cun, Y.L., 2010. Learning convolutional feature hierarchies for visual recognition. Adv. Neur. Inform. Process. Syst., 1090-1098.

Liu, W. and Chen, W., 2019. Recent advancements in empirical wavelet transform and its applications. IEEE Access, 7: 103770-103780.

Mikolov, T., Chen, K., Corrado, G. and Dean, J., 2013. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

Paitz, P., Gokhberg, A. and Fichtner, A., 2017. A neural network for noise correlation classification. Geophys. J. Internat., 212: 1468-1474.

Pal, M., 2006. Support vector machines-based modelling of seismic liquefaction potential. Internat. J. Numer. Analyt. Meth. Geomechan., 30: 983-996.

Reynen, A. and Audet, P., 2017. Supervised machine learning on a network scale: Application to seismic event classification and detection. Geophys. J. Internat., 210: 1394-1409.

Rogers, S., Chen, H., Kopaska-Merkel, D. and Fang, J., 1995. Predicting permeability from porosity using artificial neural networks. AAPG Bull., 79: 1786-1796.

Siahsar, M.A.N., Gholtashi, S., Torshizi, E.O., Chen, W. and Chen, Y., 2017. Simultaneous denoising and interpolation of 3-D seismic data via damped data-driven optimal singular value shrinkage. IEEE Geosci. Remote Sens. Lett., 14: 1086-1090.

Yuan, S., Liu, J., Wang, S., Wang, T. and Shi, P., 2018. Seismic waveform classification and first-break picking using convolution neural networks. IEEE Geosci. Remote Sens. Lett., 15: 272-276.

Zhang, G., Wang, Z. and Chen, Y., 2018a. Deep learning for seismic lithology prediction. Geophys. J. Internat., 215: 1368-1387.

Zhang, G., Wang, Z., Li, H., Sun, Y., Zhang, Q. and Chen, W., 2018b. Permeability prediction of isolated channel sands using machine learning. J. Appl. Geophys., 159: 605-615.

Zhang, K., Zuo, W., Chen, Y., Meng, D. and Zhang, L., 2017. Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising. IEEE Transact. Image Process., 26: 3142-3155.

Zhao, T., Jayaram, V., Marfurt, K.J. and Zhou, H., 2014. Lithofacies classification in barnett shale using proximal support vector machines. Expanded Abstr., 84th Ann. Internat. SEG Mtg., Denver: 1491-1495.