IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 24, 2021, accepted December 10, 2021, date of publication December 23, 2021,
date of current version January 6, 2022.

Digital Object Identifier 10.1109/ACCESS.2021.3138194

Logging Data Completion Based on an MC-GAN-BiLSTM Model

LIANG GUO[1], LUO RENZE[1], LI XINGYU[1], TUO JUANJUAN[1], CANRU LEI[1], AND ZHOU YANG[2]
[1]State Key Laboratory of Oil and Gas Reservoir Geology and Exploitation, School of Earth Sciences and Technology, Southwest Petroleum University, Chengdu 610500, China
[2]School of Earth Sciences and Technology, Southwest Petroleum University, Chengdu 610500, China

Corresponding author: Liang Guo (hihiwoo@163.com)

This work was supported in part by the Deep Earthquake Special Project of National Key Research and Development Program under Grant 2016YFC0601100, and in part by the Science and Technology Project of Sichuan Province under Grant 2019CXRC0027.

ABSTRACT Due to environmental interference and operational errors, problems such as incomplete and random missing logging data have occurred during the geophysical logging data collection process. Since it is difficult to establish a geophysical model based on logging data and geological information, the data complementation effect of conventional methods is not very satisfactory. In this paper, we propose an MC-GAN-BiLSTM model based on spatiotemporal sequence prediction. In the model, we adopt a generative adversarial network (GAN) as a network framework, and a long short-term memory (LSTM) neural network and a bi-directional long short-term memory (Bi-LSTM) as the basic modules. We use the LSTM instead of a fully-connected layer in the GAN to extract the potential information in the logging data depth domain. We complete the logging data missing values through an encoding-decoding structure that includes the Bi-LSTM. In addition, the generator module also uses multiscale convolution to fully extract the logging data features. We use logging data random missing values and consecutive missing values to simulate a field data acquisition environment and threshold control to simulate a laboratory processing environment for experiments. The experimental results show that the coefficient of determination (R^2) of the GAN-LSTM model reaches 0.906 when 30% of random logging data are missing and 0.851 when 30% of consecutive logging data are missing. The effect of the model proposed in this paper is significantly higher than the commonly used random forest (RF), sequence to sequence (seq2seq) and generative adversarial interpolation network (GAIN) models.

INDEX TERMS Logging data, missing value, GAN, LSTM, GAIN, GAN-LSTM.

I. INTRODUCTION

High-quality and high-integrity logging data are the prerequisite and basis for geological work such as lithology identification. In the process of geophysical logging data collection, due to the objective natural environment and subjective human operations, problems such as incomplete logging data collection, data omission, and random loss are often caused. To complement the missing logging data, conventional methods of analysis and comparison can be used. These methods use existing complete logging curve data to establish a geophysical model of logging data and stratum lithology based on geological and rock geophysical properties to directly fill in the missing data. These methods have strong theoretical basis and strong pertinence, and are widely used in actual production [1]–[3]. However, the theoretical basis of these methods is based on an extreme simplification of an underground geological environment, which is very hypothetical. When staff are faced with different geological environments, the effect is often unsatisfactory. Additionally, the choice of model parameters is highly subjective. This is manifested in the fact that when facing the same problem, different workers using the same model may have different understandings of the geological conditions and choose different parameters, leading to different complement results.

Another effective method is based on statistics. This type of method uses the principles of statistics, such as regression algorithms, to find the internal relationships and overall characteristics of logging data, and complete a prediction of

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

the missing logging data. For example, Fan *et al.*. used a ridge regression method (RRM) for logging acoustic curves and achieved high accuracy [4]. However, the relationship between logging data and formation lithology is not a simple linear mapping, but a complex nonlinear relationship. In recent years, many scholars have tried to use data-driven machine learning methods to characterize the relationship between logging data and formation lithology. Cheng *et al.*. used a combination of principal component analysis and support vectors to establish a nonlinear relationship between logging curves and reservoir permeability [5]. Shi *et al*. successfully used multiple regression analysis (MRA), a backpropagation neural network (BPNN), and a support vector machine (SVM) to model earth science data [6]. Ibrahim *et al.*. used an SVM and random forest (RF) to reconstruct a gamma logging curve. The experimental results show that both the SVM and RF-produced models were able to predict the gamma ray (GR) log with high accuracies, and the SVM predicted the natural GR log with R^2 and AAPE values of 0.98, and 1.42%, respectively [7]. Garcia A P et *al*. strengthened the rock attribute evaluation of the missing data wells, and reconstructed the missing logging data through machine learning and supervised neural networks. The reconstructed well logs agreed with the actual measurements with relative errors of less than 10% [8].

In recent years, deep learning has been widely used in various fields and has achieved better results than traditional machine learning methods [9]–[11]. Deep learning can learn extremely complex multi-layer neural networks and build multiple hidden layers between the input and output layers to extract high-dimensional features of the data for complex and nonlinear modeling [12]–[15]. Deep learning methods are very good at constructing complex nonlinear relationships, and they are used by many scholars in the completion of logging data. For example, Mo *etal.* used a genetic neural network (GNN), which is better than a traditional back propagation neural network (BPNN), to reconstruct a logging curve [16]. Zhang *et al.*. proposed a cascade system based on long short-term memory (LSTM) neural networks. Testing using real well log data shows that the results from the LSTM neural network are of higher accuracy than those of a traditional fully-connected neural network (FCNN) [17]. Rolon *et al.*. uses a generalized regression neural network (GRNN) to generate artificial logging curves, and, compared with the results of an MRA, the network has higher accuracy [18]. Alizedel *et al*. uses artificial neural network and cluster analysis technology (CA) to successfully establish a model between logging data and organic carbon (TOC) [19].

Logging data not only has horizontal spatial characteristics but also has vertical time series characteristics [20]. Although most deep learning models can learn the distribution of the original data and characterize the spatial characteristics of the data when they are used for logging data completion, it is easy to ignore the longitudinal correlation and change trend of the logging data [21]–[23]. The result is simple, isolated, and lacks the geological significance of the continuity of the underground horizon. A recurrent neural network (RNN) uses previous data and information to comprehensively process current tasks [24]. As a special RNN, an LSTM network can not only process the prior and subsequent data information but also avoid the problem of gradient explosion or disappearance as the sequence increases by using a gated recurrent unit [25], [26]. This feature is very suitable for processing long-sequence logging data that needs to be compared with prior and subsequent information [27]–[29]. A bi-directional long short-term memory (Bi-LSTM) combines a forward LSTM network with a backward LSTM network, making it more suitable for long-sequence data. Khan et *al*. use multilayer bi-directional long short-term memory (MBD-LSTM) to extract features of Mitochondrial proteins of Plasmodium falciparum. The identification rate of Plasmodium mitochondrial proteins using this model is as high as 99.5% [30]. A generative adversarial network (GAN) provides us with a tolerant framework. Theoretically, all the differentiable functions can construct the generating and discriminating modules of a GAN framework. This paper proposes a GAN-LSTM model for the completion of logging data from the perspective of spatial and temporal characteristics. In the generator, we use a codec structure to obtain the low-dimensional representative features of the logging data [31]. In the encoder of the generator, we abandon the fully-connected (FC) layer and replace it with Bi-LSTM to strengthen the potential connection between the missing data and the logging data of the upper and lower horizon. We trained and tested the model using the data of 170 wells. The performance of the model proposed in this article is better than a RF, sequence to sequence (Seq2seq) model, and generative adversarial interpolation network (GAIN) in random missing data and consecutive missing data completion experiments.

II. PRINCIPLE OF MODELS

A. MULTISCALE CONVOLUTION MODULE

When we use the deep learning method to complete the logging data, the missing rate of a certain logging curve data is too large, and it can be directly discarded. If we forcibly complete, the obtained complete data features may introduce noise data, which will affect the final result and subsequent work.

In the GAN generator, we use a multiscale convolution module to extract the spatial features of the logging data. In the convolution operation, different convolution kernels have different receptive fields, and the features of the extracted data are also different. The large-scale convolution kernel is suitable for extracting global information, and the small-scale convolution kernel is suitable for extracting local information. The features of the logging data extracted in this way are more comprehensive. The multiscale convolution structure is shown in Fig. 1.

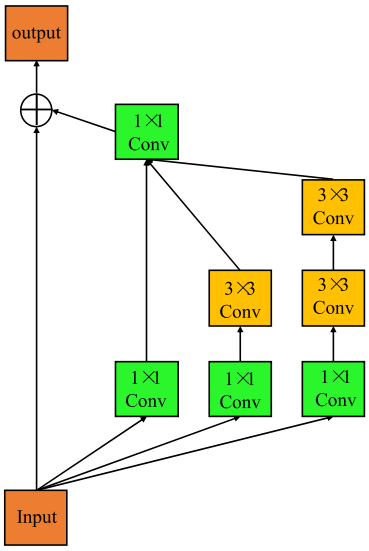

FIGURE 1. Two types of cyclic neural network calculation diagrams. The multiscale convolution adopted by MC-GAN-BiLSTM has three channels. The first convolution of each channel uses a 1 × 1 convolution kernel. The second channel adds a 3 × 3 convolution kernel, and the third channel adds two 3 × 3 convolution kernels.

B. BiLSTM MODULE

An RNN mainly processes sequence-based data, traversing all recurrent units in a recursive manner along the direction of the sequence. An RNN is used to process data with temporal characteristics, such as natural speech sequence processing, speech image processing, and machine translation. As a neural network with short-term memory, an RNN has two major characteristics. First, an RNN can not only receive the current self-information but also the information from the previous time point. Second, an RNN uses the same parameters at all times to realize parameter sharing in the time dimension, as presented in Fig. 2(a). At time t, assuming that the network input is x_t, the state of current hidden layer h_t is not only related to input x_t at the current time but also related to the hidden state h_{t-1} at the previous time. They are calculated using Eq. (1) and Eq. (2).

$$h_t = f(\boldsymbol{U}x_t + \boldsymbol{W}h_{t-1} + \boldsymbol{b}) \tag{1}$$
$$o_t = \boldsymbol{V}h_t + \boldsymbol{c} \tag{2}$$

where f is an activation function, o_t is the output value of the current layer, $\boldsymbol{U}$, $\boldsymbol{W}$, and $\boldsymbol{V}$ are weight coefficients, and $\boldsymbol{b}$ and $\boldsymbol{c}$ are bias terms.

An RNN can only predict the output of the next time point based on the timing information of the previous time point. However, the predicted value to be output may depend on the entire input sequence in the application of logging data completion. It requires that the current output is not only related to the previous state but also has some relationship with the future state. A bi-directional recurrent neural network (BRNN) can satisfy such needs. In the design of a BRNN, at time t, the input will pass the data in two directions to a hidden layer neuron. The neuron will calculate the deep-level feature information from the forward and reverse directions. The calculated result represents the "past" and "future" information, and the BRNN combines them to yield an output result, as presented in Fig. 2(b). The values are calculated using Eq. (3), Eq. (4) and Eq. (5).

$$h_t = f(\boldsymbol{U}_1x_t + \boldsymbol{W}_1h_{t-1} + \boldsymbol{b}_1) \tag{3}$$
$$g_t = f(\boldsymbol{U}_2x_t + \boldsymbol{W}_2g_{t-1} + \boldsymbol{b}_2) \tag{4}$$
$$o_t = \boldsymbol{V}_1h_t + \boldsymbol{V}_2g_t + \boldsymbol{c} \tag{5}$$

where f is an activation function, h_t is the current forward hidden layer state, g_t is the current reverse hidden layer state, o_t is the current layer output value, $\boldsymbol{U}_1$, $\boldsymbol{W}_1$, and $\boldsymbol{V}_1$ are the weight coefficients of the forward cyclic network, $\boldsymbol{U}_2$, $\boldsymbol{W}_2$, and $\boldsymbol{V}_2$ are the weight coefficients of the reverse cyclic network, and $\boldsymbol{b}_1$, $\boldsymbol{b}_2$, and $\boldsymbol{c}$ are the corresponding bias terms. From the perspective of "time" order, a BRNN has two states: one is passed from the beginning of the sequence back with time; the other is passed from the end of the sequence forward with time. As a result, the output can benefit from past data-related features as well as future data-related features. In general, a BRNN's structure allows the output unit to have the data feature constraints of the past and future sequences at the same time, and it can also maintain the sensitivity of the current data-related features. Additionally, when a BRNN evaluates the state of the current input data, it is not necessary to expand the scope of input in order to increase the overall control. However, whether it is an RNN or BRNN, when long-term tasks are involved, the gradient decreases exponentially with the corresponding calculation, or even disappears. This ultimately causes the weight to update slowly, and the model performance decreases. In practical applications, a gating system is usually used to solve the problem of gradient disappearance in an RNN. LSTM is the most typical gating system, and its exploded view is presented in Fig. 2.

LSTM controls the flow of the history and current information through a forget gate, input gate, output gate and cell state. State C is like dealing with a "fast car" on a high-speed channel, which only runs in the channel and rarely interacts with the hidden unit of the recurrent neural network. Then it is relatively easy for state C to maintain smooth fluctuations in a long-time scale, as shown in Fig. 3(a). State C passes through the discriminative control of the door, and transmits the original information state h_t to itself by adding or removing, as shown in Figure 4-6(b) The forget gate controls the degree of forgetting previous cell states, as shown in Fig. 3(a)-Fig. 3(f). The first gate is called the forget gate, which discriminates the information that the system should discard from the original cell state, see Fig. 3(c) and Eqn. (6).

$$f_t = \sigma\left(\boldsymbol{W}_f \cdot [h_{t-1}, x_t] + \boldsymbol{b}_f\right) \tag{6}$$

For the status update, see Figure 3(d) and Figure 3(e). The process of updating from C_{t-1} to C_t determines what new content is delivered to the state. For Figure 3(d), the input is still the accumulation state h_{t-1} of the hidden unit of the

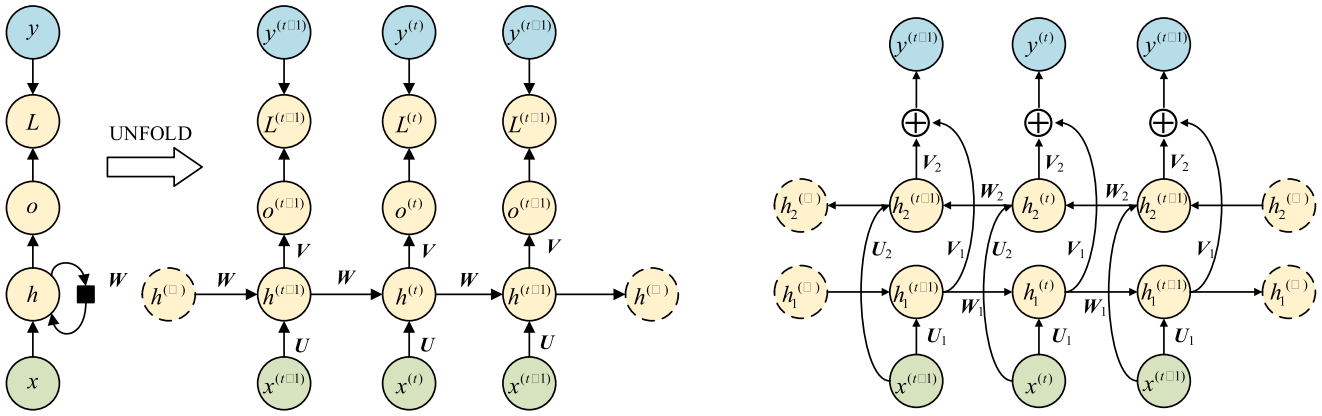

FIGURE 2. Two types of cyclic neural network calculation diagrams. The bi-directional neural network (BRNN) has two states from the perspective of "time". First is from the beginning of the sequence going backward with time. The second is from the end of the sequence going forward with time. Then, the output can benefit from past data-related features as well as data-related features from the future.

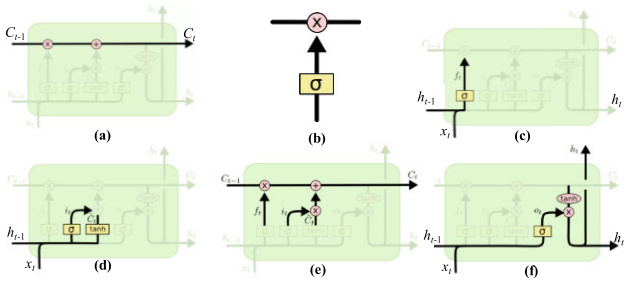

FIGURE 3. Exploded view of LSTM.

recurrent neural network and the current data sequence input x_t is then divided into two parts of calculation. The first part, similar to the previous forget gate, uses the sigmoid neural network layer to determine the updated content, and the output is i_t; the second part uses the tanh function layer to create a new candidate value vector $\tilde{C}_t$. The operating calculations of i_t and $\tilde{C}_t$ are as follows:

$$i_t = \sigma\left(\boldsymbol{W}_i \cdot [h_{t-1}, x_t] + \boldsymbol{b}_i\right) \tag{7}$$

$$\tilde{C}_t = \tanh\left(\boldsymbol{W}_c x_t + \boldsymbol{U}_c h_{t-1} + \boldsymbol{b}_c\right) \tag{8}$$

Finally, the LSTM merges the product of the updated content i_t of the current input information and the candidate value $\tilde{C}_t$. The specific calculation diagram is shown in Figure 3(e), and the calculation is as follows:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{9}$$

where $\tilde{C}_t$ is the candidate information of the current depth. It is multiplied by the input gate state i_t to determine which new information is updated to the cell state. The output gate determines the information that is output from the cell state at the current depth, see Fig. 3(f) and Eqn. (10), Eqn. (11).

$$o_t = \sigma\left(\boldsymbol{W}_o x_t + \boldsymbol{U}_o h_{t-1} + \boldsymbol{b}_o\right) \tag{10}$$

$$h_t = o_t \odot \tanh\left(C_t\right) \tag{11}$$

where x_t is the input to the current LSTM unit, h_t is the hidden state of the output of the current LSTM unit, $\boldsymbol{W}_f$, $\boldsymbol{U}_f$, $\boldsymbol{W}_i$, $\boldsymbol{U}_i$, $\boldsymbol{W}_o$, $\boldsymbol{U}_o$, $\boldsymbol{W}_c$, and $\boldsymbol{U}_c$ are the weight matrices of each structure, $\boldsymbol{b}_f$, $\boldsymbol{b}_i$, $\boldsymbol{b}_o$, $\boldsymbol{b}_c$ are corresponding bias terms, σ is a sigmoid function, and $\odot$ is the corresponding element multiplication.

In general, ordinary neurons can only extract data features from the past and current inputs to evaluate the current state. In the process of logging data completion, many current predictions need to rely on the input sequence of the overall logging signal data. For example, when logging data are consecutively missing due to the different logging signal characteristics of different formations, the logging data characteristics of the formation boundary not only depend on the logging data of the previous layer but also the analysis and comparison of the logging data from the next layer or even the overall logging data. If the current data changes are not obvious or abnormal (including unreasonable data processing in the previous period, such as data distortion and other reasons), it may be necessary to expand the data area forward (into the future) or backward (into the past) for identification. An ordinary RNN cannot do this, which brings us to a BRNN. For the problem of vanishing or exploding gradients, LSTM can solve this problem. Therefore, we use Bi-LSTM based on LSTM as the basic module to be introduced into the entire logging data completion framework.

C. NETWORK FRAMEWORK GAN AND GAIN

A GAN is a deep learning architecture based on game theory. Since end-to-end training can be performed in the neural network, it can learn the potential distribution of training samples. In the case of fewer artificial priors, it can use random noise to generate "real" samples, and finally generate usable data. Compared with other generation algorithms, a GAN has obvious advantages. It can generate

samples in parallel and can be flexibly combined with other networks, such as a CNN or RNN.

The structure of a GAN is composed of two parts: a generator and discriminator. The generator is used to synthesize simulated samples that are almost the same as real samples. The discriminator is used to determine whether a sample comes from the real world or was generated by simulation. The function of the generator is to generate samples that ''mix the spurious with the genuine'', which makes it difficult for the discriminator to distinguish. The function of the discriminator is to distinguish between synthetic samples and real samples. The purpose of the generator and the discriminator is opposite and there is a contradictory relationship. By putting two mutually-independent models together for synchronous training, the simulated samples generated by the generator will be more realistic, and the discriminator will make more accurate judgments of the samples.

Specifically, the generator and discriminator are both functions in nature. The generator is responsible for capturing the distribution of sample data and mapping it to a new data space with a noise vector z that obeys a uniform or Gaussian distribution. Finally, it tries to generate simulated samples $G(z)$ that obey the distribution of real data $G(z)$. The discriminator takes simulated sample $G(z)$ or real sample $\boldsymbol{x}$ as input, and the output is a scalar, representing the probability that the input sample is a real sample. The optimization goals of a GAN are as follows:

$$\min_G \max_D V(D, G) = E_{x\sim p_{data}(x)}[logD(x)] + E_{z\sim p_z(z)}[log(1 - D(G(z)))] \quad (12)$$

where $\boldsymbol{x}$ is the real sample, z is random noise, p_{data} is the real data distribution, and p_z is the noise data distribution. The GAN optimization problem is a maximum and minimum problem. First, we need to define a priori input noise distribution $p_z(z)$. During the model learning process, the discriminator should try to improve the prediction accuracy of the discriminant generated samples, and the generator should try to reduce the optimal performance of the discriminator. When the optimal performance of the discriminator reaches the lowest, the generator distribution p_g is closest to the real data distribution p_{data} at this moment. The process and structure of a GAN is presented in Fig. 4:

The GAIN network was proposed by Yoon *etal*. [32] in 2018. A GAIN is similar to the GAN model, and it is also composed of a generator and a discriminator. The difference from a GAN is that a GAIN has no special requirements for input data. A GAIN does not need complete data as the input to the model, and directly uses missing data that needs to be completed, as shown in Fig. 5.

Generally, the input data to a GAN is random noise, while the input to a GAIN is data that needs to be completed. The discriminator of a GAN model mainly discriminates the authenticity of a sample, while the discriminator of a GAIN model discriminates the authenticity

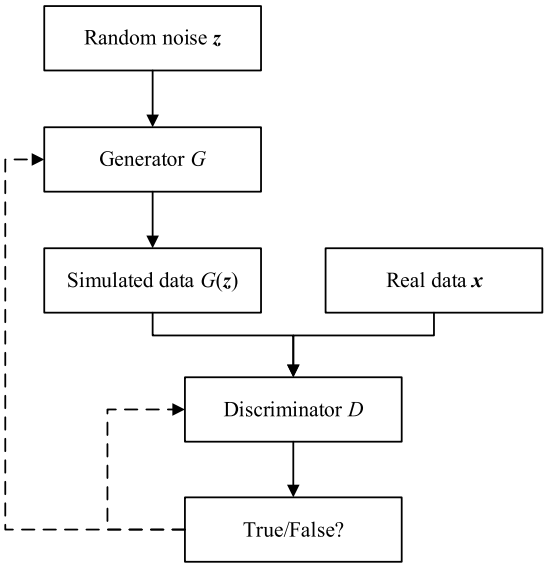

FIGURE 4. The process and structure of a GAN.

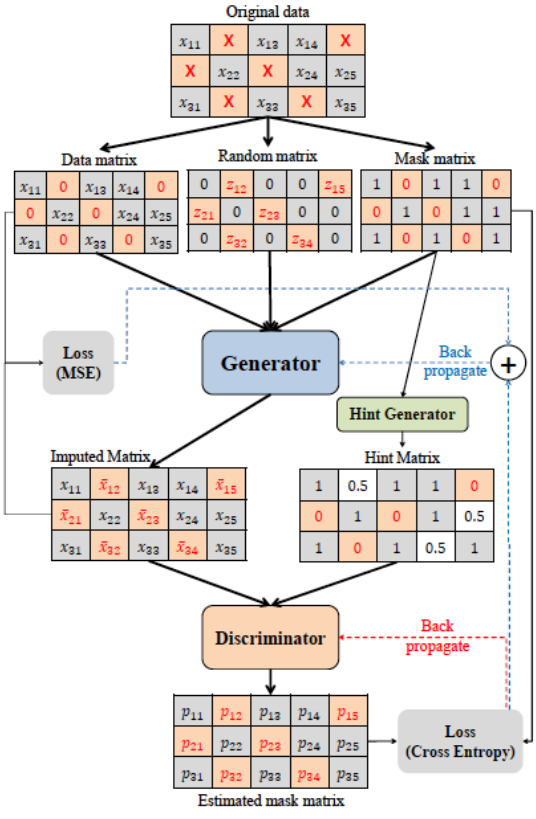

FIGURE 5. The structure of a GAIN. The structure diagram comes from reference [32].

of each element in the sample. Specifically, the missing data are converted into three different forms as input data. First, the GAIN fills in the missing data with 0, and combines the original data to form a Data matrix. Second, the GAIN fills in the original data with 0, and fills in the missing data with

random data to generate a Random matrix. Finally, the GAIN replaces the original data with 1 and the missing data with 0 to generate a Mask matrix that records the location of the missing data. The above three matrices are used as inputs to the generator. The output of the generator is the Imputed Matrix of the GAIN model. Additionally, the Mask matrix is transformed into a Hint matrix through a Hint generator. The Hint matrix combines the Imputed Matrix output from the generator as the input to the discriminator. The output of the discriminator is also a matrix, and the value of each element of the matrix represents the probability of missing data. The model calculates the error between the input of the generator and the initial completion matrix, which is also called reconstruction error. The reconstruction error, the input to the discriminator and the Mask matrix are used to calculate the cross entropy, and this cross entropy is used as the loss function of the model. Finally, the generator and the discriminator are updated iteratively by backpropagation until the network converges. At this time, the performance of the generator and the discriminator has reached a relatively optimal level. The generator can complete the missing data, and the result of the completion is close to the real data.

D. NETWORK FRAMEWORK MC-GAN-BiLSTM

In a traditional GAN and GAIN, the generator and discriminator use an FCNN. The generator of the MC-GAN-BiLSTM is realized by a self-encoder which consists of an encoder and a decoder. Before input, we use multiscale convolution to fully extract the spatial features of the logging data. In addition, logging data are the response of underground lithology. The formation and lithology of an oil field are related to the depositional cycle. The depositional cycle is caused by periodic changes in the global sea level. Specifically, a depositional cycle affects the deposition and the deposition conditions that allows them to be repeated in the same order and deposited into a sequence. Therefore, a depositional cycle has the characteristics of alternating lithology. As a response to lithology, logging data also has a similar trend. This trend leads to the data that needs to be completed not only being related to the overall characteristics of the input sequence but also related to the top (past information) and bottom (future information) logging data of the missing data. From a geological point of view, missing lithology data are not only related to the entire sedimentary environment but also related to the upper and lower contact lithology of the strata. Therefore, we used a Bi-LSTM network that can extract past information, future information, and overall information as the basic module of the model. The encoder of the MC-GAN-BiLSTM adopts a Bi-LSTM model, which can establish the potential connection between a missing value and the logging value of the upper and lower formations. It compresses the input missing logging data into a low-dimensional intermediate vector z. The decoder uses the LSTM model to obtain the generated complete logging data by decoding z. The discriminator is composed of an LSTM cyclic layer and a fully-connected layer, and its input

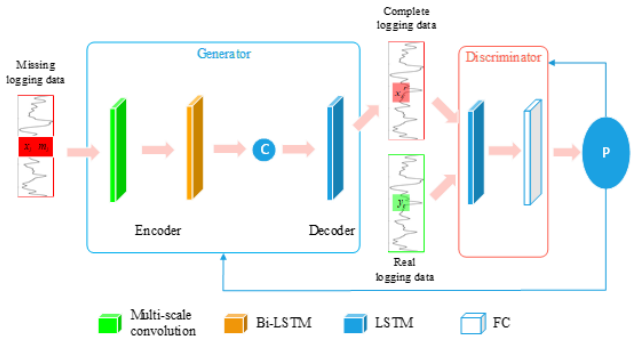

FIGURE 6. Network structure diagram of the MC-GAN-BiLSTM logging curve completion model.

includes two types: generated complete logging data and real complete logging data. Through the loop layer and the fully-connected layer, the discriminator maps the input to a one-dimensional vector, and obtains the probability value that the input data are "true". When the generator and discriminator reach a balanced state, the model training is completed. The network structure of the MC-GAN-BiLSTM logging curve completion model is presented in Fig. 6.

The methodology implemented in this research is shown in Fig. 7. First, due to the different measurement methods, the dimensions of each type of logging data are also different. Unifying the dimensions of the logging data facilitates the joint input of data. Second, according to the content of the experiment, we generated two sets of 5 different test sets, which represented different missing rates. Finally, we use an RF, sequence to sequence Seq2seq, GAIN and the MC-GAN-BiLSTM to analyze and compare the results of each experiment. Through 10 different missing rate experiments, we determined the maximum missing rate of each model when dealing with different types of missing data.

III. MC-GAN-BiLSTM LOGGING DATA COMPLETION PROCESS

The problem solved by a logging data complement method is to complement the logging data of each well in the data set, that is, to replace all missing values with reasonable logging values, as presented in Fig. 5 of the estimated mask matrix.

When the MC-GAN-BiLSTM method complements missing logging data, the first step is to determine the input and output variables. First, we define the missing log data and missing identification matrix.

$$\boldsymbol{X} = (x_1, x_2, \cdots, x_t, \cdots, x_T) \in \mathbb{R}^{D\times T} \tag{13}$$

$$\boldsymbol{M} = (m_1, m_2, \cdots, m_t, \cdots, m_T) \in \mathbb{R}^{D\times T} \tag{14}$$

$$\mathrm{x}_\mathrm{t}^\mathrm{d} = \begin{cases} \text{if } \mathrm{x}_\mathrm{t}^\mathrm{d} \text{ exist} & m_t^d = 1 \\ \text{if } \mathrm{x}_\mathrm{t}^\mathrm{d} \text{ does not exist} & m_t^d = none \end{cases} \mathrm{x}_\mathrm{t}^\mathrm{d} \in \mathbb{R}^{D\times T} \tag{15}$$

where x_t is the logging value at sampling point t, x_t^d is the d_{th} logging value at sampling point t, m_t is the missing identifier at sampling point t, D is the dimension of the data, and T is the length of the data. Therefore, the corresponding missing

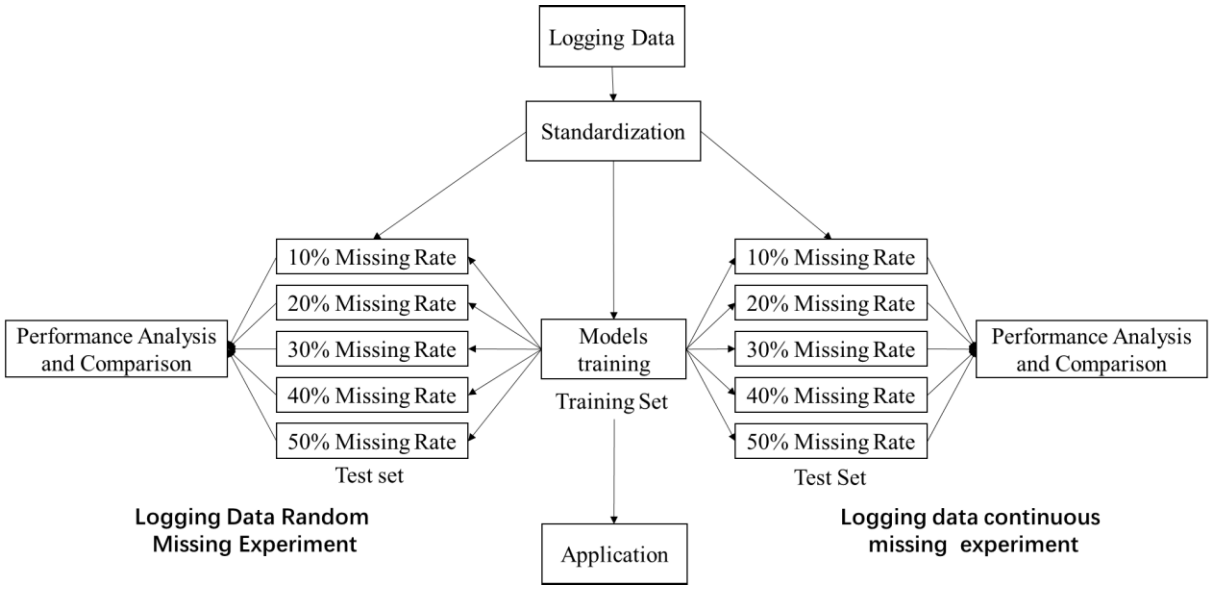

FIGURE 7. The framework of this research.

identification is:

$$\boldsymbol{M} = \begin{bmatrix} 1 & none & 1 & 1 \\ 1 & 1 & 1 & none \\ none & 1 & none & 1 \\ 1 & 1 & none & none \end{bmatrix} \tag{16}$$

where 1 means data exists and *none* means data are missing. So, the generated complete logging data and real and complete logging data can be expressed as:

$$\boldsymbol{X}' = (x_1', x_2', \cdots, x_T') \in \mathbb{R}^{D \times T} \tag{17}$$

$$\boldsymbol{Y} = (y_1, y_2, \cdots, y_T) \in \mathbb{R}^{D \times T} \tag{18}$$

After determining the input and output variables, we use the generator to generate simulation data from them. The sum of the missing data and the noise vector z is used as the generator input. Adding noise makes the input data and the original data have a certain degree of difference to reduce the over-fitting phenomenon in the process of generating data. The input to the generator is mapped to a low-dimensional semantic encoding vector $\boldsymbol{c}$ through the encoder. The task of the decoder is to reconstruct vector $\boldsymbol{c}$ into complete logging data $\boldsymbol{X}'$ and obtain a complete logging sequence.

The loss function of the generator includes two parts: the discrimination loss that makes the discriminator misjudgment and the reconstruction loss that reconstructs the original data. The generator loss is presented in Eq. (19):

$$G_{loss} = -(1-\lambda)D(G(\boldsymbol{X}+\boldsymbol{z})) + \lambda \left\| M \odot ((G(\boldsymbol{X}+\boldsymbol{z}) - \boldsymbol{X}) \right\|_2 \tag{19}$$

where λ is a coefficient of reconstruction loss and $\|\cdot\|_2$ is the $L2$ norm.

We use the discriminator to distinguish the authenticity of the data. The real samples $\boldsymbol{Y}$ and the generated samples $\boldsymbol{X}'$ are input into the discriminator. The discriminator is composed of an LSTM cyclic layer and a fully-connected layer. The cyclic layer is responsible for processing the real or generated logging data and obtaining the historical memory vector from the sample data. The fully-connected layer maps the historical memory vector into a one-dimensional output and uses a Sigmoid function to calculate the probability that the input data are "true". The training goal of the discriminator is to recognize the real samples as "true" and the generated samples as "false" as much as possible. The discriminator loss function is:

$$D_{loss} = -D(\boldsymbol{Y}) + D(G(\boldsymbol{X}+\boldsymbol{z})) \tag{20}$$

Finally, we optimize the objective function to obtain complementary data through iteration. The generator and the discriminator have opposite goals and oppose each other. In the optimization process of iterative training, the overall performance of the model continues to improve until it reaches a balance. The distribution of the simulated sample $\boldsymbol{X}'$ generated at this time is sufficiently close to the distribution of the real sample $\boldsymbol{Y}$. We replace missing values with generated values to obtain the completed data $\boldsymbol{Y}$.

$$\hat{\boldsymbol{Y}} = \boldsymbol{Y} \odot \boldsymbol{M} + (1 - \boldsymbol{M}) \odot \boldsymbol{X}' \tag{21}$$

IV. DATA PREPARATION, EVALUATION INDEX, AND PARAMETER ADJUSTMENT

A. DATA PREPARATION

The study area is the Jing'an Oilfield Dalugou II, located in the north central Shanbei Slope Basin, and the overall monoclinic structure is west-inclined at an angle less than 1°. The structure in the area is relatively simple, with only a few faults and folds developed, and the distribution of wells is shown in Fig. 8.

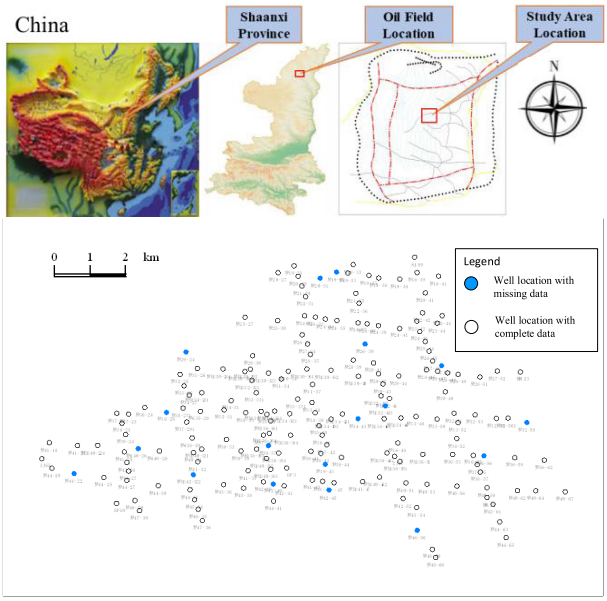

FIGURE 8. Drilling distribution in the study area.

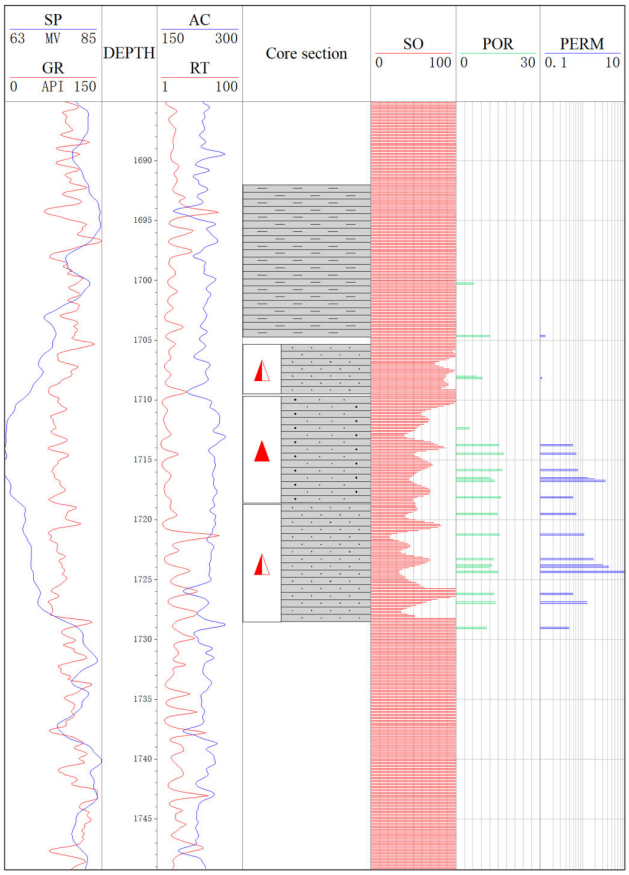

FIGURE 9. Seven types of logging data.

During the logging phase in the study area, four types of logging curves of spontaneous potential (SP), natural gamma ray (GR), acoustic time difference (AC) and true resistivity (RT) were recorded. There is a complex correlation between these four curves. Physical parameters such as porosity (POR), permeability (PERM) and water saturation (SW) can also be expressed linearly by these logging data, as in Fig. 9.

In the experiment, we build a model for these 7 types of data to complement the 18 wells in the area that had missing logging curves. We used the missing rate to represent the proportion of missing values in the input data, which is expressed by Eq. (24).

$$Missing\ Rate = \frac{\sum_{t=1}^{T}\sum_{d=1}^{D} 1 - m_t^d}{T \times D} \tag{22}$$

where m_t^d is the missing identifier of the d_{th} type logging value at sampling point t, D is the dimension of the data, that is the type of curve, and T is the length of the data.

According to our statistics, 18 wells in the study area had missing data, and the missing rate was approximately 5-40%. We used the remaining 177 wells with complete logging curves as standard data to establish a complementary model. Among them, 150 wells were used as the training set, and 30% of the data was randomly discarded. Under the guidance of the real data, the training model made the missing values approach the known values. Twenty-seven wells were used as the test set, and the discarding ratios were set to 10%, 20%, 30%, 40%, and 50%. According to the different discarding ratios, two types of random deletion and consecutive deletion in the actual logging data were simulated.

B. PERFORMANCE EVALUATION INDEX

The evaluation index of the model performance adopts the root mean square error (RMSE) and coefficient of determination (R^2). The RMSE reflects the difference between the complement value and the true value, which is calculated using Eq. (23). The R^2 represents the degree of fit between the complemented log curve and the original curve, which is calculated in Eq. (24). To prevent the mutation data from adversely affecting the statistics of the completion results, we also adopted the Mean Absolute Error (MAE), which is calculated in Eq. (25) [33]. The MAE is the deviation between the complement value and the true measurement value, and then the absolute value is taken and averaged. The deviation is the absolute value, and there will be no positive and negative offset, Therefore, the MAE can better reflect the actual situation of the predicted value error [34].

$$RMSE = \sqrt{\frac{\sum_{t=1}^{T}\left(y_t - y_t'\right)^2}{T}} \tag{23}$$

$$R^2 = 1 - \frac{\sum_{t=1}^{T}\left(y_t - y_t'\right)^2}{\sum_{t=1}^{T}\left(y_t - \bar{y}_{miss}\right)^2} \tag{24}$$

$$MAE = \frac{\sum_{t=1}^{T}\left|\left(y_t - y_t'\right)\right|}{T} \tag{25}$$

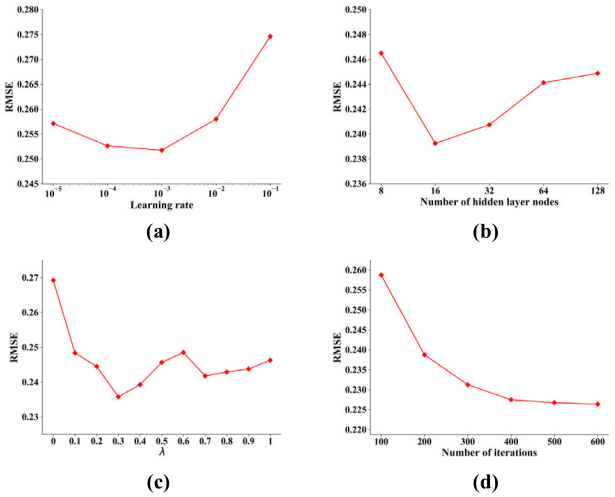

FIGURE 10. Comparison chart of the relationship between parameters and the RMSE. (a) is the learning rate, (b) is the number of hidden nodes, (c) is the λ, and (d) is the number of iterations.

where y_i is the real log value of the i_{th} sampling point in the missing section, y_i' is the generated log value of the i_{th} sampling point in the missing section, and N is the total number of sampling points in the missing section.

C. PARAMETER ADJUSTMENT

During the training process, the performance of the model was optimized by adjusting various parameters. In this process, the RMSE was used as the evaluation index. According to the comparison of the parameters listed in Fig. 10, when the learning rate is 0.001 and the number of hidden layer nodes is 16, the RMSE is the lowest and the model has the best performance. So, we set the number of hidden layer nodes of the LSTM layer and the learning rate to 16 and 0.001, respectively, and then adjusted the parameter λ of the model. The completion effect under different λ values is presented in Fig. 10(c). When the parameter is close to 1 or 0, the completion effect is poor. This implies that the two parts of the generator's loss function are not functioning as they should. When parameter λ is set to 0.3, the RMSE is the lowest and the completion effect is the best. When the number of iterations is 400, the RMSE basically stabilizes, as seen in Fig. 10(d). So, we set the maximum number of iterations to 400. In summary, the learning rate, the number of hidden nodes, the λ parameter, and the number of iterations are set to 0.001, 16, 0.3, and 400, respectively.

While optimizing the model parameters, we also adjusted the parameters of the GAIN to be optimized. The GAIN uses a traditional GAN framework, and both the generator and discriminator use a fully-connected neural network. In the following experiments, we use the GAIN model as a reference to compare the effects of the model proposed in this article.

TABLE 1. Model accuracy for random missing data completion.

Model	Missing Rate	Random Missing		
		MAE	RMSE	R2
RF	10%	0.28	0.34	0.87
	20%	0.32	0.37	0.84
	30%	0.35	0.39	0.81
	40%	0.40	0.46	0.78
	50%	0.46	0.53	0.77
Seq2seq	10%	0.24	0.26	0.94
	20%	0.28	0.31	0.87
	30%	0.29	0.33	0.85
	40%	0.35	0.40	0.83
	50%	0.40	0.46	0.82
GAIN	10%	0.16	0.24	0.94
	20%	0.18	0.26	0.92
	30%	0.24	0.33	0.88
	40%	0.26	0.35	0.83
	50%	0.36	0.47	0.80
MC-GAN-BiLSTM	10%	0.12	0.20	0.96
	20%	0.16	0.25	0.95
	30%	0.17	0.27	0.92
	40%	0.22	0.35	0.87
	50%	0.27	0.40	0.85

V. ANALYSIS AND DISCUSSION OF THE EXPERIMENTAL RESULTS

A. RANDOM MISSING DATA COMPLETION EXPERIMENT

In this section, we use conventional RF and Seq2seq models, a traditional GAIN, and the MC-GAN-BiLSTM to compare their accuracies. In the study area, the lack of logging data is mainly concentrated on the AC measurement. We combine the four types of logging data, GR, SP, RT and AC, and the three physical parameters, SO, POR and PERM, as the data input to the model. The advantage is that the joint multi-dimensional data have better constraints on the model, and can increase the completion accuracy of the model after training. For example, when the AC data in the study area is severely missing, it may affect the model to extract the characteristics of the AC data. However, when we combine other logging data as input, we can dilute the impact of the serious lack. In other words, when a certain logging curve data are seriously missing, we can rely on the other 6 logging data to complement the seriously missing data. Compared with the input of a single logging data type, the joint input increases the dimensionality of the input data from 1 to 7 dimensions, which increases the difficulty and cost of calculation. However, we believe that the processing of 7-D data is within an acceptable range compared with the dimensions of image processing.

We focus on the AC measurements with a large range of missing data in the study area. In this experiment, 27 wells were selected as the test set, and the amount of deleted data was controlled at 10%, 20%, 30%, 40%, and 50% by randomly deleting measurement points. This can simulate the real situation of field logging data loss. The performance results of the four models on the test set are shown in Table 1 and Fig. 11.

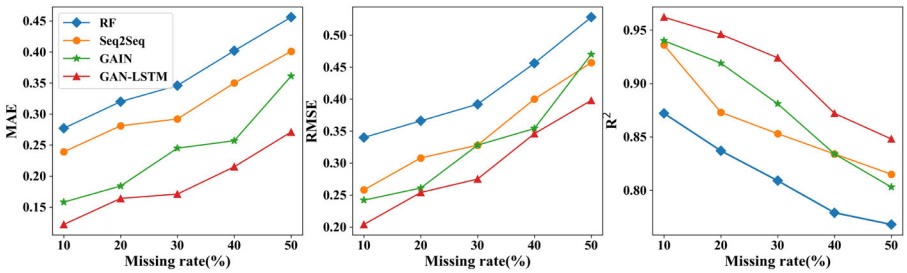

FIGURE 11. Completion of random missing data for different evaluation indicators.

We compare the performance of the four models through three evaluation indicators: the MAE, RMSE, and R^2. The smaller the value of the RMSE and MAE, the more accurate the completion value, and the better the performance of the model. The larger the R^2 value, the better the fit between the completed data and the real data. In general, the completion results of the MC-GAN-BiLSTM are slightly better than the other three models. The average values of the MAE and RMSE are 0.189 and 0.295, respectively, which is lower than the 0.36 and 0.417, 0.313 and 0.35, and 0.241 and 0.331 of the RF, Seq2seq and GAIN. The R^2 average value is 0.911, which is higher than the 0.813, 0.862, and 0.875 of the RF, Seq2seq and GAIN. Specifically, when the random missing rate is less than 40%, the R^2 completion accuracy is not much different. The three evaluation indicators are all within an acceptable range except for the RF. When the random missing rate exceeds 40%, the accuracy of the RF, Seq2seq and GAIN decreases. The R^2 of the RF is less than 0.85. A large proportion of random data missing has caused similar effects to consecutive missing data, which brings difficulty to the model. For example, the RF ignores the time series characteristics of the logging data, and the completed data are only related to other data values at that location. The RF lost the correlation of logging data from the upper and lower formations. When large areas of data are missing, the accuracy of completion decreases. The MC-GAN-BiLSTM proposed in this paper can effectively extract the spatial and temporal characteristics of the logging data. When the missing rate increases, the MC-GAN-BiLSTM can use the features of the intact data to ensure the accuracy of completion. The specific completion effects of each model are shown in Fig. 12.

Through the above experiments, we can count the missing data in the model applications. When there is less randomly missing data in the study area, we prefer a conventional completion model. This is because the model is simple, and the result will be more efficient within an acceptable range. When the missing rate is large, the MC-GAN-BiLSTM is used to ensure the accuracy of the completion

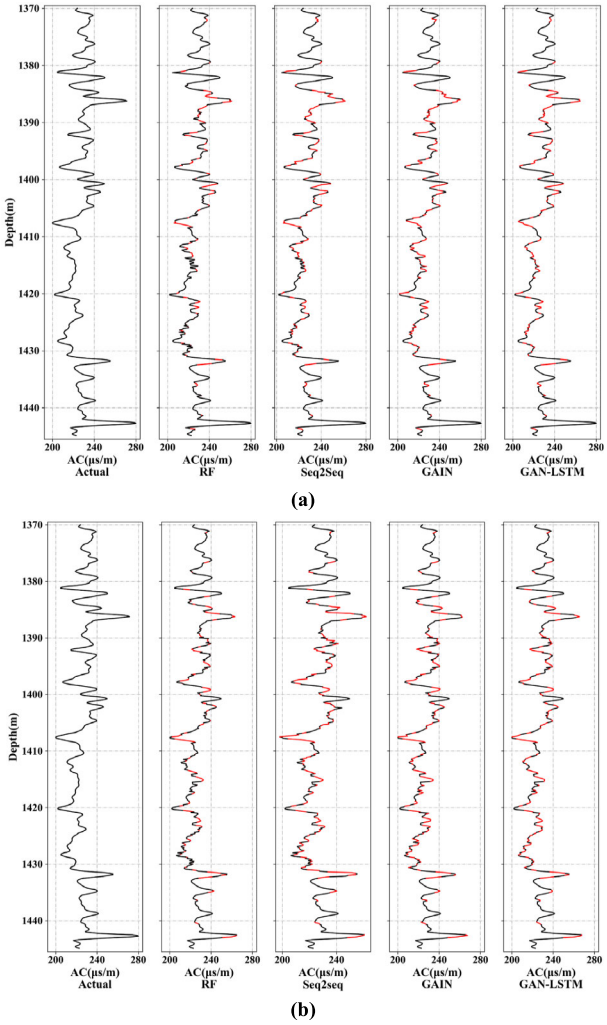

FIGURE 12. Practical application of the MC-GAN-BiLSTM for random data completion. (a) is a random missing data rate of 40% and (b) is a random missing data rate of 50%.

B. RANDOM MISSING DATA COMPLETION EXPERIMENT

We used the same four models to complete consecutive missing data. Due to the consecutive lack of data, the models lack the context of the data during the process of completion, which makes the experiment in this section more challenging than the random missing data completion experiment in the previous section. The training set also uses 150 wells with complete data, and each well also contains 7 types of measurement data. The test set selects 27 wells to control the

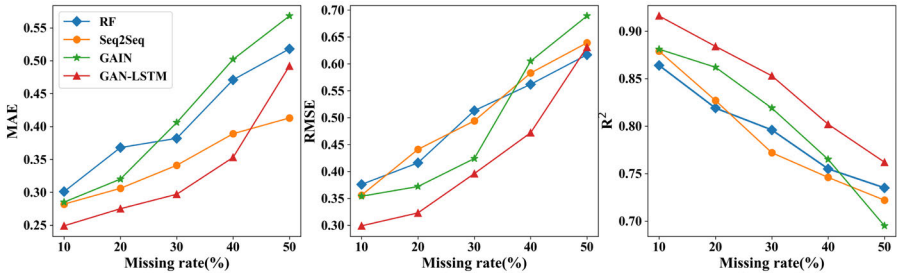

FIGURE 13. Completion of consecutive missing data for different evaluation indicators.

TABLE 2. Model accuracy of the consecutive missing data completion.

Model	Missing Rate	Consecutive Missing		
		MAE	RMSE	R2
RF	10%	0.30	0.38	0.86
	20%	0.37	0.42	0.82
	30%	0.38	0.51	0.80
	40%	0.47	0.56	0.76
	50%	0.52	0.62	0.74
Seq2seq	10%	0.28	0.36	0.88
	20%	0.31	0.44	0.83
	30%	0.34	0.49	0.77
	40%	0.39	0.58	0.75
	50%	0.41	0.64	0.72
GAIN	10%	0.29	0.35	0.88
	20%	0.32	0.37	0.86
	30%	0.41	0.42	0.82
	40%	0.50	0.61	0.77
	50%	0.57	0.69	0.70
MC-GAN-BiLSTM	10%	0.25	0.30	0.92
	20%	0.28	0.32	0.88
	30%	0.30	0.40	0.85
	40%	0.35	0.47	0.80
	50%	0.49	0.63	0.76

consecutive missing rate in the form of deletion at 10%, 20%, 30%, 40%, and 50%, which forms five test sets. We gradually increase the missing rate and analyze the accuracy of the model to determine the maximum missing rate for model completion.

We also train the model according to the joint input of 7 types of data. The output results of the 27 wells in the test set are shown in Table 2. Compared with the random data missing completion experiments, the accuracy of the consecutive data missing completion experiments decreased. For example, in the GAN-LSMT's random missing data completion experiment with a missing rate of 30%, the MAE is 0.3, the RMSE is 0.4, and the R^2 is 0.85. In the consecutive data missing and complete experiment, the MAE is 0.17, the RMSE is 0.27, and the R^2 is 0.92. When the missing rate is less than 30%, compared with the real data, the completion data of the four models have certain errors in their details, as shown with the blue marker boxes in Fig. 13(a). The trend of the curve shape has not changed much, and it is within an acceptable range. As the missing rate increases, the performance drops faster, as shown in Fig. 10. When the missing rate increases to more than 40%, the curve shape trend of the RF, Seq2seq and GAIN model data completion changes greatly (Fig. (b)). When the missing rate exceeds 50%, the curve shape trend of the GAIN-LSTM data completion has also changed (Fig. (c)). At this time, we should consider manual completion based on geological data.

C. DISCUSSION OF THE EXPERIMENT

The ultimate goal of this experiment is to use an objective and intelligent method to complement the missing logging data as much as possible. In this paper, we use the MC-GAN-BiLSTM model. The biggest advantage of the MC-GAN-BiLSTM model is that not only the spatial characteristics of logging data are used, but the RF and Seq2seq modules in the model can effectively extract the timing characteristics of the logging data. This advantage greatly increases the performance of the model. In addition, we adopt a method of the joint input of 7 types of data, which can effectively improve the accuracy of all models. For example, when the incomplete data are randomly missing, the accuracy of the four methods involved in this article are all within an acceptable range. Finally, we simulate the lack of real data collected in the field through two types of test sets with random deletion and consecutive deletion, so that the experiment is more in line with the actual application environment. Additionally, we gradually increase the missing rate of the data, determine the application scope of the model developed in this article according to the application effect, and indirectly improve the efficiency of logging data completion.

Although the MC-GAN-BiLSTM model proposed in this article has achieved results, it also has shortcomings. First, we use 7 types of data in a joint input methods. The method of joint input helps to improve the performance of the model, but it has strict data requirements. Although the logging data of most oilfields in China contains the same data used in this paper, a very small number of oilfields lack one or more types. Second, the model proposed in this paper requires a large amount of complete data to train the model, which forces the application of the model to be based on the completion of logging work. Finally, because the input of the LSTM module requires a certain amount of data space, the model is not useful for the completion of real-time data from logging while drilling.

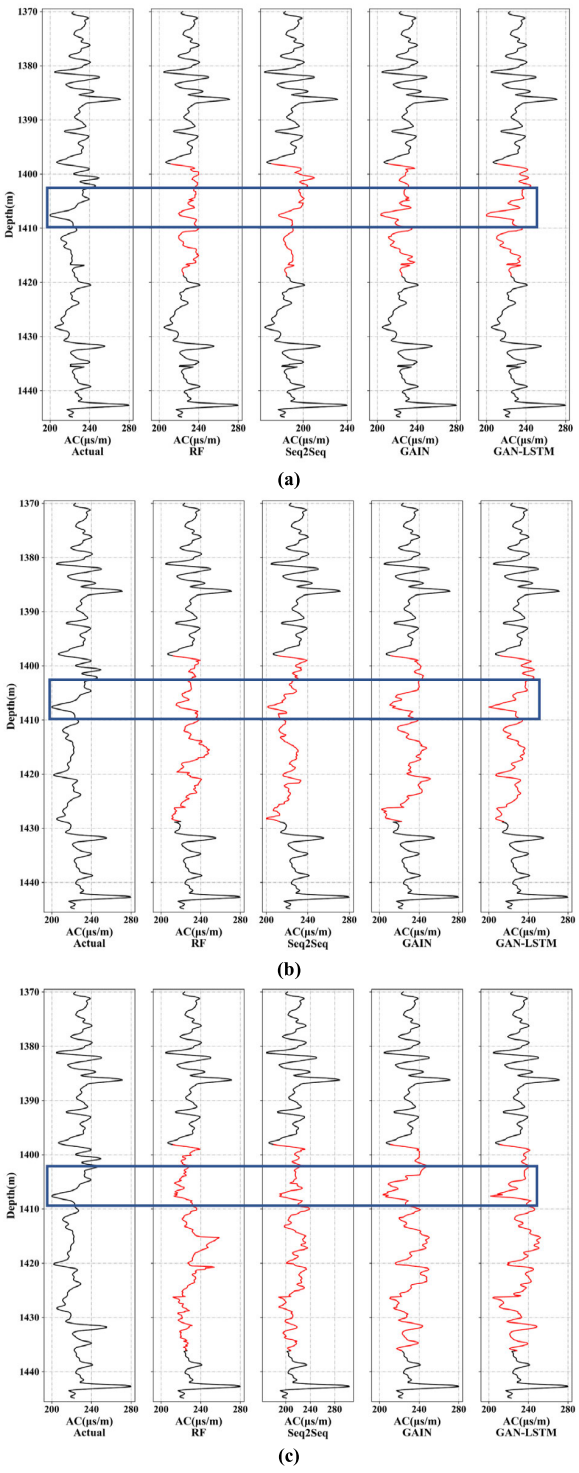

FIGURE 14. Practical application of the MC-GAN-BiLSTM consecutive random data completion. (a) is a consecutive missing data rate of 30%, (b) is a consecutive missing data rate of 40%, and (c) is a consecutive missing data rate of 50%.

VI. CONCLUSION

This paper proposes a new method of complementing logging data based on a MC-GAN-BiLSTM. This method combines the characteristics of a GAN module to learn the distribution of real data and the advantages of an BiLSTM module to process the time series. It can memorize the trend of logging curve changes with depth and the spatial correlation of logging data at different depths, which is more in line with actual geological analysis experience experiments. In addition, we use multiscale convolution to fully extract the spatial features of the logging data. Multiscale convolution combined with BiLSTM makes the extracted features have spatiotemporal characteristics. On this basis, we construct a log curve complement model, using known values to complement the missing values of the curve. The experimental results show that the completion effect of the MC-GAN-BiLSTM method proposed in this paper is better than the traditional GAIN method. It can complete the missing part of the logging curve with reasonable simulation values that are close to the real logging data. This method provides certain guidance for researchers and achieves the goal of reducing logging costs as much as possible. However, due to the complexity of the underground geological environment, when the missing rate of the curve is large ($>30\%$), it is difficult to accurately reflect the missing section based only on the limited known logging data, and there is a large deviation in the completion of this method. For the 18 wells in the study area with missing data, 15 of them were complemented by the MC-GAN-BiLSTM method, and logging data of 3 wells with a high missing rate were discarded.

REFERENCES

[1] R. K. Akhmadulin and A. I. Miraev, "Development of cross-platform software for well logging data visualization," in *Proc. 8th Int. Sci. Practical Conf. Innov. Technol. Eng.* Yurga, Russia: Yurga Inst. Technol., May 2017, pp. 2–8, doi: 10.1088/1757-899X/221/1/012005.

[2] A. Carrasquilla and R. Lima, "Basic and specialized geophysical well logs to characterize an offshore carbonate reservoir in the Campos Basin, southeast Brazil," *J. South Amer. Earth Sci.*, vol. 98, pp. 1–13, May 2020, doi: 10.1016/j.jsames.2019.102436.

[3] J. Lai, G. Wang, S. Wang, J. Cao, M. Li, X. Pang, C. Han, X. Fan, L. Yang, Z. He, and Z. Qin, "A review on the applications of image logs in structural analysis and sedimentary characterization," *Mar. Petroleum Geol.*, vol. 95, pp. 139–166, Aug. 2018, doi: 10.1016/j.marpetgeo.2018.04.020.

[4] P. P. Fan, R. Deng, J. Q. Qiu, Z. L. Zhao, and S. L. Wu, "Well logging curve reconstruction based on kernel ridge regression," *Arabian J. Geosci.*, vol. 14, no. 16, pp. 1–10, Aug. 2021, doi: 10.1007/s12517-021-07792-y.

[5] G. J. Cheng, H. X. Pan, and L. Cai, "Petroleum reservoir permeability prediction based on principal component analysis and support vector regression," in *Proc. 8th Int. Conf. Inf. Manage. Sci.*, Kunming, China, Jul. 2009, pp. 390–393.

[6] G. R. Shi, X. S. Yang, and Iccs, "Optimization and data mining for fracture prediction in geosciences," in *Proc. Int. Conf. Comput. Sci. (ICCS)*, Amsterdam, The Netherlands, May 2010, pp. 1353–1360, doi: 10.1016/j.procs.2010.04.151.

[7] A. F. Ibrahim and S. Elkatatny, "Real-time GR logs estimation while drilling using surface drilling data; AI application," *Arabian J. Sci. Eng.*, vol. 46, pp. 1–10, Jun. 2021, doi: 10.1007/s13369-021-05854-7.

[8] A. P. Garcia, A. Jagadisan, L. M. Hernandez, Z. Heidari, B. Casey, and R. Williams, "Enhanced formation evaluation in the Permian basin using a novel field-scale workflow including wells with missing data," *SPE Reservoir Eval. Eng.*, vol. 23, no. 3, pp. 865–878, Aug. 2020.

[9] S. U. Khan, I. U. Haq, S. Rho, S. W. Baik, and M. Y. Lee, "Cover the violence: A novel deep-learning-based approach towards violence-detection in movies," *Appl. Sci.*, vol. 9, no. 22, p. 4963, Nov. 2019, doi: 10.3390/app9224963.

[10] H. Zhang and X. J. Wu, "Research on the connotation and process of reflection-based deep learning," in *Proc. Int. Conf. Hum. Social Sci. (ICHSS)*, Guangzhou, China, Jun. 2014, pp. 351–355.

[11] M. Y. Kim, S. Lee, and J. H. Kim, "A wide & deep learning sharing input data for regression analysis," in *Proc. IEEE Int. Conf. Big Data Smart Comput. (BigComp)*, Busan, South Korea, Feb. 2020, pp. 8–12, doi: 10.1109/BigComp48618.2020.0-108.
[12] E. De la Rosa, W. Yu, and X. Li, "Nonlinear system modeling with deep neural networks and autoencoders algorithm," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Budapest, Hungary, Oct. 2016, pp. 2157–2162.
[13] E. De La Rosa and W. Yu, "Restricted Boltzmann machine for nonlinear system modeling," in *Proc. IEEE 14th Int. Conf. Mach. Learn. Appl. (ICMLA)*, Miami, FL, USA, Dec. 2015, p. 443, doi: 10.1109/ICMLA.2015.24.
[14] H. Zhang, H. Liu, R. Song, and F. Sun, "Nonlinear dictionary learning based deep neural networks," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Vancouver, BC, Canada, Jul. 2016, pp. 3771–3776.
[15] D. W. Chen, F. Hu, and G. K. Nian, "Deep residual learning for nonlinear regression," *Entropy*, vol. 22, no. 2, pp. 1–14, Feb. 2020, doi: 10.3390/e22020193.
[16] X. Mo, Q. Zhang, and X. Li, "Well logging curve reconstruction based on genetic neural networks," in *Proc. 12th Int. Conf. Fuzzy Syst. Knowl. Discovery (FSKD)*, Zhangjiajie, China, Aug. 2015, pp. 1015–1021.
[17] D. X. Zhang, Y. T. Chen, and J. Meng, "Synthetic well logs generation via recurrent neural networks," *Petroleum Explor. Development.*, vol. 45, no. 4, pp. 629–639, Aug. 2018, doi: 10.1016/S1876-3804(18)30068-5.
[18] L. Rolon, S. D. Mohaghegh, S. Ameri, R. Gaskari, and B. McDaniel, "Using artificial neural networks to generate synthetic well logs," *J. Natural Gas Sci. Eng.*, vol. 1, nos. 4–5, pp. 118–133, Nov. 2009, doi: 10.1016/j.jngse.2009.08.003.
[19] B. Alizadeh, S. Najjari, and A. Kadkhodaie-Ilkhchi, "Artificial neural network modeling and cluster analysis for organic facies and burial history estimation using well log data: A case study of the south pars gas field, Persian gulf, Iran," *Comput. Geosci.*, vol. 45, pp. 261–269, Aug. 2012, doi: 10.1016/j.cageo.2011.11.024.
[20] K. Zhou, S. Li, J. Liu, X. Zhou, and Z. Geng, "Sequential data-driven cross-domain lithology identification under logging data distribution discrepancy," *Meas. Sci. Technol.*, vol. 32, no. 12, Dec. 2021, Art. no. 125122, doi: 10.1088/1361-6501/ac29d2.
[21] J. Lin, H. Li, N. Liu, J. Gao, and Z. Li, "Automatic lithology identification by applying LSTM to logging data: A case study in X tight rock reservoirs," *IEEE Geosci. Remote Sens. Lett.*, vol. 18, no. 8, pp. 1361–1365, Aug. 2021, doi: 10.1109/LGRS.2020.3001282.
[22] A. Meghanani, C. S. Anoop, and A. G. Ramakrishnan, "An exploration of log-mel spectrogram and MFCC features for Alzheimer's dementia recognition from spontaneous speech," in *Proc. IEEE Spoken Lang. Technol. Workshop (SLT)*, Jan. 2021, pp. 670–677, doi: 10.1109/SLT48900.2021.9383491.
[23] E. Sharma, G. L. Ye, W. N. Wei, R. Zhao, Y. Tian, J. Wu, L. Lin, E. Lin, and Y. F. Gong, "Adaptation of rnn transducer with text-to-speech technology for keyword spotting," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Barcelona, Spain, May 2020, pp. 7484–7488.
[24] Z. Huang, J. Peng, H. Lian, J. Guo, and W. Qiu, "Deep recurrent model for server load and performance prediction in data center," *Complexity*, vol. 2017, pp. 1–10, Nov. 2017, doi: 10.1155/2017/8584252.
[25] W. J. Feng, N. Y. Guan, Y. Li, X, Zhang, and Z. G. Luo, "Audio visual speech recognition with multimodal recurrent neural networks," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Anchorage, AK, USA, May 2017, pp. 681–688.
[26] A. Hermanto, T. B. Adji, and N. A. Setiawan, "Recurrent neural network language model for English-Indonesian machine translation: Experimental study," in *Proc. Int. Conf. Sci. Inf. Technol. (ICSITech)*, Yogyakarta, Indonesia, Oct. 2015, pp. 132–136.
[27] S. Jha, D. Prashar, H. V. Long, and D. Taniar, "Recurrent neural network for detecting malware," *Comput. Secur.*, vol. 99, pp. 1–13, Dec. 2020, doi: 10.1016/j.cose.2020.102037.
[28] Y. G. Shi, B. Fernando, and R. Hartley, "Action anticipation with RBF Kernelized feature mapping RNN," in *Proc. 15th Eur. Conf. Comput. Vis. (ECCV)*, Munich, Germany, Sep. 2018, pp. 305–322, doi: 10.1007/978-3-030-01249-6_19.
[29] D. Jiang, W. Chen, F. Cai, and H. Chen, "Neural attentive personalization model for query auto-completion," in *Proc. IEEE 3rd Adv. Inf. Technol., Electron. Automat. Control Conf. (IAEAC)*, Chongqing, China, Oct. 2018, pp. 725–730.
[30] S. U. Khan and R. Baik, "MPPIF-Net: Identification of plasmodium falciparum parasite mitochondrial proteins using deep features with multilayer bi-directional LSTM," *Processes*, vol. 8, no. 6, p. 725, Jun. 2020, doi: 10.3390/pr8060725.
[31] S. Khan, T. Hussain, A. Ullah, and S. W. Baik, "Deep-ReID: Deep features and autoencoder assisted image patching strategy for person re-identification in smart cities surveillance," *Multimedia Tools Appl.*, vol. 80, Jan. 2021, doi: 10.1007/s11042-020-10145-8.
[32] J. Yoon, J. Jordon, and M. Schaar, "GAIN: Missing data imputation using generative adversarial nets," in *Proc. 35th Int. Conf. Mach. Learn. (ICML)*, Stockholm, Sweden, vol. 80, Jul. 2018, pp. 5689–5698.
[33] X. Zhang, C. Huang, Y. Xu, and L. Xia, "Spatial-temporal convolutional graph attention networks for citywide traffic flow forecasting," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2020, pp. 1853–1862.
[34] J. Hu and W. Zheng, "A deep learning model to effectively capture mutation information in multivariate time series prediction," *Knowl.-Based Syst.*, vol. 203, Sep. 2020, Art. no. 106139, doi: 10.1016/j.knosys.2020.106139.