Received September 6, 2021, accepted October 8, 2021, date of publication October 15, 2021, date of current version October 29, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3120482

Combining a Fully Connected Neural Network With an Ensemble Kalman Filter to Emulate a Dynamic Model in Data Assimilation

MANHONG FAN, YULONG BAI, LILI WANG, AND LIN DING
College of Physics and Electrical Engineering, Northwest Normal University, Lanzhou 730070, China

Corresponding author: Yulong Bai (baiyulong@nwnu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Project 41861047, Project 41461078, and Project 62066041; in part by the Foundation of Northwest Normal University of China under Grant NWNU-LKQN2020-21 and Grant NWNU-LKQN2019-18; and in part by the Scientific Research Program of the Higher Education Institutions of Gansu Province under Grant 2020A-016.

ABSTRACT Using neural network technology, dynamic characteristics can be learned from model output or assimilation results to train the model, which has greatly progressed recently. A data-driven data assimilation method is proposed by combining fully connected neural network with ensemble Kalman filter to emulate dynamic models from sparse and noisy observations. First, the hybrid model couples the original dynamic model with the surrogate model. The surrogate model is learned from model forecast values and assimilation results, and its performance is verified using the training accuracy/loss and the validation accuracy/loss at different training times. Second, the assimilation process includes a ''two-stage'' procedure. Stage 0 generates the training sets and trains the surrogate model. Then, the hybrid model is used for the next assimilation period in Stage 1. Finally, several numerical experiments are conducted using the Lorenz-63 and Lorenz-96 models to demonstrate that the proposed approach is better than the ensemble Kalman filter in different model error covariances, observation error covariances, and observation time steps. The proposed approach has also been applied to sparse observations to improve assimilation performance. This hybrid model is restricted to the form of the ensemble Kalman filter. However, the basic strategy is not restricted to any particular version of the Kalman filter.

INDEX TERMS Data assimilation, fully connected neural network, machine learning, ensemble Kalman filter, Lorenz model.

I. INTRODUCTION

Data assimilation (DA) is an important method to fuse observation information and nonlinear physical models. It is aided by estimation theory, cybernetics, optimization methods and error estimation theory in mathematics [1]. Early DA methods were mainly polynomial interpolation, successive corrections, and optimal interpolation in the 1990s. After this period, modern DA methods have been widely studied and rapidly applied [2], [3]. DA methods are mainly divided into two categories: continuous DA and sequential DA. The former mainly includes three-dimensional and four-dimensional variational assimilation (3DVar and 4DVar, respectively) [4]. The latter mainly includes the Kalman filter (KF), ensemble Kalman filter (EnKF) and particle filter (PF). These approaches refer to the updating of the model state based on the weighting of the observation error and the model error to obtain an a posteriori optimal estimation of the model state when the system is running [5]–[8]. After the status update, the model is reinitialized with the new state and continues to integrate forward until new observation information is obtained. Among the various methods, applying the EnKF to a set represented by the Monte Carlo sequential DA method has been a huge success [9], [10] and has been widely used in the fields of ocean, land surface and atmospheric DA [11].

Since objective reality is unknown, people usually replace the model operator with an expanded and refined mathematical model, but such models cannot be equal to objective reality. Due to the diversity of mathematical model representation

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

methods, environmental changes and other factors, model errors and observation errors always exist [12], [13]. At present, many practical DA schemes rely on Gaussian or linear assumptions. In particular, most practical data assimilation schemes are approximations of the famous Kalman filter (KF), all of which have been introduced to reduce computational costs and improve statistical predictions. Although current DA methods have made great improvements in error processing and correction, errors are inevitable due to the influence of various uncertain factors, such as the initial value, boundary condition and model structure [14], [15]. Therefore, an important challenge of DA is how to intelligently utilize existing methods in the presence of various errors to solve practical problems, such as the incomplete understanding of physical processes, lack of large computational resources, and understanding of interactions across scales.

Machine learning (ML) methods are widely used to generate alternative models of low-dimensional chaotic systems [16] or sea surface temperatures [17]. Then, these new models, such as recurrent neural networks (RNNs) [18], convolutional neural networks (CNNs) [19], and adversarial neural networks (ANNs) [20], are used to make predictions. With these advancements in ML, the problem of using surrogate models to identify unknown processes based on observations has also been solved through sparse regression and DA approaches. Tang *et al.* [21] suggested using artificial neural networks as a possible DA technology. However, Campos Velho *et al.* [22] used neural networks in all DA spatial domains. Later, this approach was improved by Harter and Campos Velho [23].

The output of DA is clearly dependent on the uncertainty in the numerical model, which has led to the development of techniques for considering model errors in the DA process [24]. One can accomplish this by using parameterizing model errors in model equations or by adding random noise to deterministic models [13]. In any case, there must be a dynamic model, and its existence is the key to DA. It is worth noting that for the ML problem, the optimization problem solved by DA is equivalent to the ML problem when model errors exist. For instance, the DA approach was used to infer ordinary differential equation representations of dynamical models [25]. The use of analog techniques to replace the numerical forecast model has been described by Lguensat [26]. A model-free filter based on the EnKF and Takens theory was introduced by Hamilton *et al.* [27], [28]. Brajard *et al.* [19] combined DA and ML to emulate a dynamical model from sparse and noisy observations using the Lorenz-96 model. Laloyaux *et al.* [29] explored the potential and limitations of weak-constraint 4DVar. However, none of these studies include ensemble approaches or only use a simple linear regression to realize forecasts. In this study, a hybrid approach is proposed that couples the fully connected neural network (FCNN) with the EnKF to correct errors. It is applied in Lorenz models to improve assimilation performance.

The structure of this study is as follows. In Section 2, the principles of EnKF and FCNN are briefly introduced, and this section describes the generation of training sets, the training of the surrogate model and the application of the hybrid model. The experimental setup using the Lorenz-63 and Lorenz-96 models is chosen to illustrate the hybrid approach in Section 3. Section 4 summarizes the advantages and disadvantages of this approach and proposes a new perspective and challenges for future work.

II. METHODOLOGY

A. ENSEMBLE KALMAN FILTER

The KF is the closed-form solution to Bayesian filtering and is obtained in the linear, Gaussian case [30]–[34]. Similar to the KF, the EnKF consists of the recursive application of a forecast step and an analysis step. In this study, EnKF is used to verify the assimilation performance. However, the basic strategy is not restricted to any particular version of the Kalman filter. In the forecast step, a group of random variables $X_{i,t}^a$ conform to the Gaussian distribution ($i = 1, \ldots, N$) at time t. N is the size of the ensemble. The forecasting values $X_{i,t+1}^f$ and the observation values $y_{i,t+1}$ of each random variable at time $t+1$ are expressed in equations (1–2).

$$X_{i,t+1}^f = M_t\left(X_{i,t}^a\right) + w_{i,t}, w_{i,t} \sim \mathbb{N}(0, Q_t) \tag{1}$$

$$y_{i,t+1} = H_t\left(X_{i,t}^f\right) + \varepsilon_{i,t}, \varepsilon_{i,t} \sim \mathbb{N}(0, R_t), \tag{2}$$

where $X_{i,t}^a$ represents the analysis values of the random variable at time t. $M_t(\cdot)$ and $P_t(\cdot)$ are nonlinear model operators. Q_t and R_t denote the model error covariance and the observation error covariance, respectively. $w_{i,t}$ is Gaussian white noise with an expectation value and a variance equal to 0 and Q_t, respectively. $\varepsilon_{i,t}$ is Gaussian white noise with an expectation value and a variance equal to 0 and R_t, respectively. The Kalman gain matrix K_{t+1} at time $t+1$ is calculated using equations (3–7):

$$K_{t+1} = P_{t+1}^f H^T \left(HP_{t+1}^f H^T + R_{t+1}\right)^{-1} \tag{3}$$

$$P_{t+1}^f = \frac{1}{N-1}\sum_{i=1}^{N}\left(X_{i,t+1}^f - \overline{X_{t+1}^f}\right)\cdot\left(X_{i,t+1}^f - \overline{X_{t+1}^f}\right)^T \tag{4}$$

$$P_{t+1}^f H^T = \frac{1}{N-1}\sum_{i=1}^{N}\left(X_{i,t+1}^f - \overline{X_{t+1}^f}\right)\cdot\left(H\left(X_{i,t+1}^f\right) - H\left(\overline{X_{t+1}^f}\right)\right)^T \tag{5}$$

$$HP_{t+1}^f H^T = \frac{1}{N-1}\sum_{i=1}^{N}\left(H\left(X_{i,t+1}^f\right) - H\left(\overline{X_{t+1}^f}\right)\right)\cdot\left(H\left(X_{i,t+1}^f\right) - H\left(\overline{X_{t+1}^f}\right)\right)^T \tag{6}$$

$$\overline{X_{t+1}^f} = \frac{1}{N}\sum_{i=1}^{N} X_{i,t+1}^f, \tag{7}$$

where $\overline{X^f_{t+1}}$ is the mean of the forecast values at time $t+1$ and $H(\cdot)$ is the observation operator.

In the analysis step, the observation and forecast values are weighted to obtain the best estimation values at time $t+1$. The mean $\overline{X^a_{t+1}}$ of the analysis values and the background field error covariance matrix P^a_{t+1} at time $t+1$ are calculated using equations (8–10):

$$X^a_{i,t+1} = X^f_{i,t+1} + K_{t+1}\left[y_{t+1} - H(X^f_{i,t+1}) + v_{i,t+1}\right], \quad v_{i,t+1} \sim \mathbb{N}(0, R_t) \tag{8}$$

$$\overline{X^a_{t+1}} = \frac{1}{N}\sum_{i=1}^{N} X^a_{i,t+1} \tag{9}$$

$$P^a_{t+1} = \frac{1}{N-1}\sum_{i=1}^{N}\left(X^a_{i,t+1} - \overline{X^a_{t+1}}\right)\cdot\left(X^a_{i,t+1} - \overline{X^a_{t+1}}\right)^T, \tag{10}$$

where X^a_{t+1} is the analysis value at time $t+1$ and $v_{i,t+1}$ is Gaussian white noise with an expectation value and variance equal to 0 and R_t, respectively. This is the principle of the EnKF in one cycle.

B. FULLY CONNECTED NEURAL NETWORK

An FCNN is a network in which adjacent network layers are fully connected to each other, and the most basic principles are derived from back propagation [35], [36]. Neurons are grouped in layers. Figure 1 is the schematic diagram of an FCNN. The leftmost layer is called the input layer, and it is responsible for receiving input data. The rightmost layer is called the output layer, and we can obtain the neural network output data from this layer. The layers between the input and output layers are called hidden layers because they are not visible to the outside [37], [38]. Each connection has a weight. There are no connections between neurons in the same layer. Each neuron in layer L is connected to all the neurons in layer L-1 (this is the meaning of a fully connected layer), and the output of the neurons in layer L-1 is the input of the neurons in layer L.

In neural networks, the activation functions of different layers can be different. Common functions are the following: the *sigmoid* function, the *tanh* function, and the *rectified linear unit* (*ReLU*) function [39]. The disadvantages of the *tanh* and *sigmoid* functions are obvious. When the value of the dependent variable is too large, the speed of the parameter update will become very slow, which is very unfavorable to the implementation of the gradient descent algorithm. At present, the most popular activation function in ML is the *ReLU*. Since this function generally makes learning much faster, it has become the default choice for hidden layer activation functions. Therefore, the *ReLU* is chosen as the activation function in this study [40].

C. EnKF COUPLED WITH FCNN

The general idea of this algorithm is that the FCNN provides a surrogate forward model to the EnKF; in contrast, the EnKF

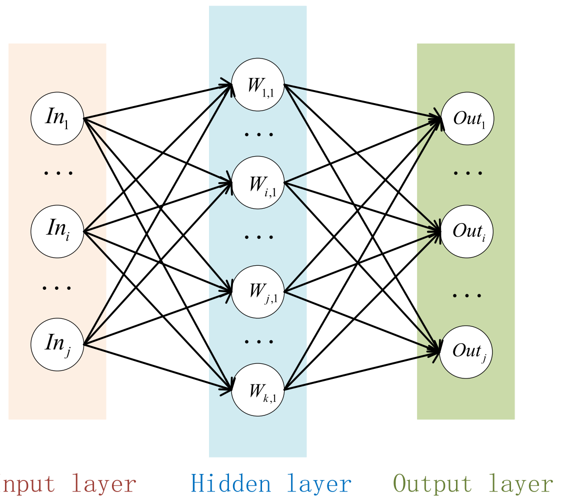


FIGURE 1. Schematic diagram of the FCNN. The leftmost layer is the input layer, the rightmost layer is the output layer, and the middle is the hidden layers. In_j is the training set, Out_j is the training model, and $W_{k,1}$ is the neural network weight in the hidden layer.

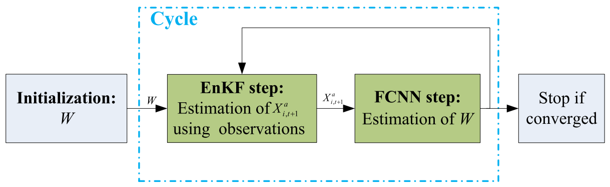

Cycle
Initialization: W
W
EnKF step: Estimation of $X^a_{i,t+1}$ using observations
$X^a_{i,t+1}$
FCNN step: Estimation of W
Stop if converged

FIGURE 2. Scheme of the two-step algorithm. EnKF step: Using the observations to estimate the analysis values based on W. FCNN step: Using neural network model and analysis values to estimate W.

provides a complete time series to train the neural network. The scheme of the algorithm is displayed in Figure 2, and the schematic diagram of an EnKF coupled with an FCNN is shown in Figure 3. The assimilation process is separated into two stages. In stage 0, the nonlinear physical model is denoted as $M_t(\cdot)$. The training sets consist of the EnKF results over a period of time ($t = 1, 2, \ldots,$ m). The forecast values $X^f_{i,1}, X^f_{i,2}, \cdots, X^f_{i,\mathrm{m}}$ and the analysis values $X^a_{i,1}, X^a_{i,2}, \cdots, X^a_{i,\mathrm{m}}$ are used as the input and output of the training sets in the FCNN. Using the FCNN, this process generates a surrogate model $\xi(\cdot)$ through training. Equation (11) represents a hybrid model using an already existing model:

$$\xi(M_t(\cdot)) \mapsto M_t(\cdot) + \xi(\cdot) \tag{11}$$

where $M_t(\cdot)$ is the original model and $\xi(\cdot)$ is the trainable model.

In the training process, the optimal weight W (the weight of an artificial neural network) is determined using an iterative minimization process of the cost function. In this case, the cost function to minimize is

$$L\left(W, X^a_{i,t}\right) = \frac{1}{2}\sum_{t=0}^{\mathrm{m}-1}\left\|y_t - H_t\left(X^a_{i,t}\right)\right\|^2_{R_t^{-1}} + \frac{1}{2}\sum_{t=0}^{\mathrm{m}-1}\left\|X^a_{i,t+1} - \xi(M_t(W, X^a_{i,t})\right\|^2_{Q_t^{-1}}, \tag{12}$$

where m is the length of the assimilation or training window and W is also the set of parameters of the surrogate model. This equation also estimates the model error using sparse and

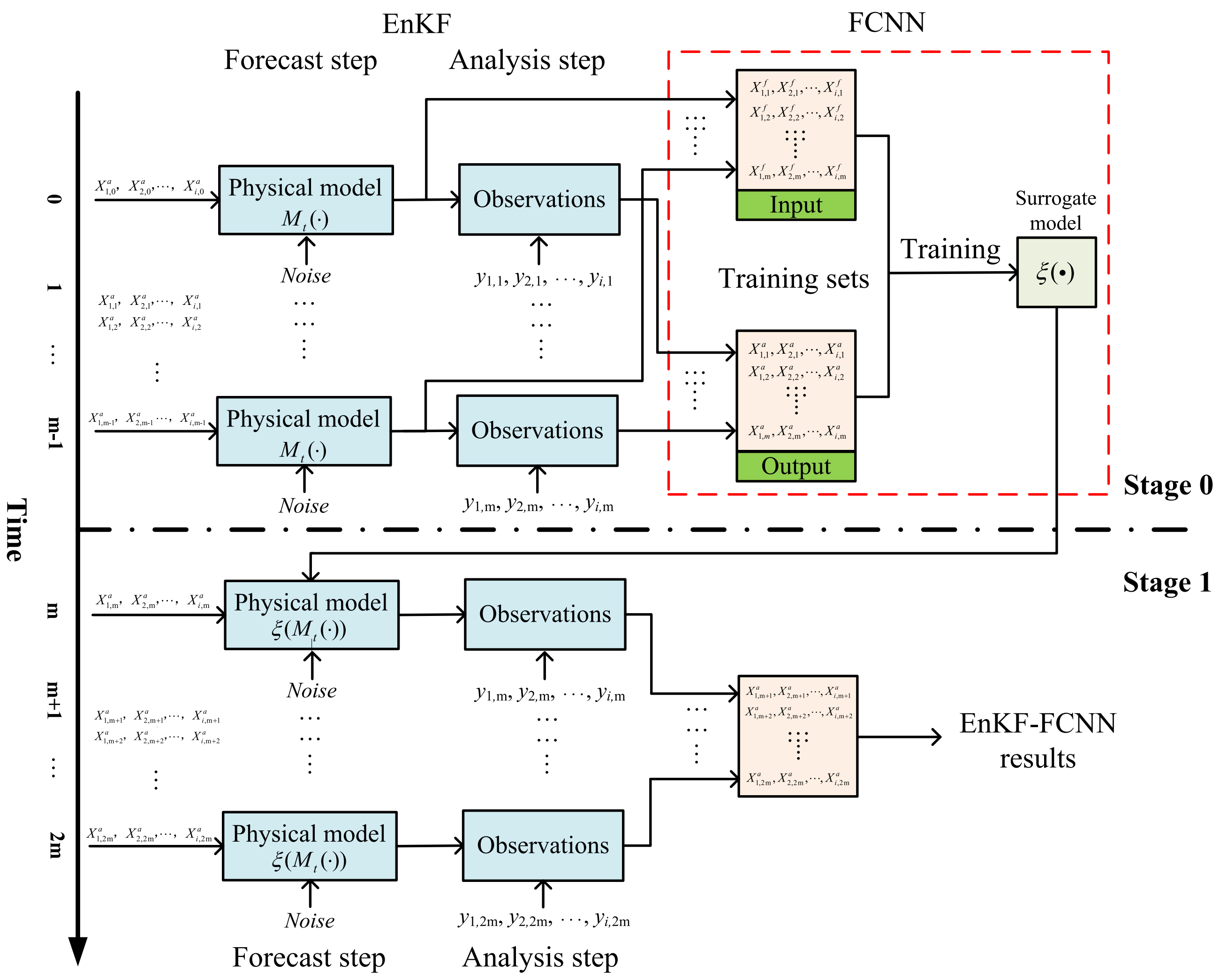


FIGURE 3. Schematic diagram of an EnKF coupled with an FCNN. Stage 0: Using the EnKF results over a period of time (t=1, 2, …, m) to train the surrogate model. Stage 1: Coupling this surrogate model with the EnKF to realize assimilation in the next period of time (t=m, m+1, …, 2m).

noisy observations. If y_t is the full observation and $R_t = 0$ (no observation noise), then the standard ML cost function is shown in equation (13).

$$L\left(W, X_{i,t}^a\right) = \frac{1}{2}\sum_{t=0}^{m-1}\left\|X_{i,t+1}^a - \xi(M_t(W, X_{i,t}^a)\right\|_{Q_t^{-1}}^2 \quad (13)$$

In stage 1, the surrogate model can be iteratively formed $\xi(M_t(\cdot))$. This hybrid model is coupled with the EnKF to realize assimilation in the next period of time (t = m, m+1, …, 2m). The results of the EnKF and EnKF-FCNN are defined as X^{EnKF} and $X^{EnKF-FCNN}$, respectively. The forecast value $X_{i,t+1}^f$ changes as

$$X_{i,t+1}^f = \xi\left(M_t\left(X_{i,t}^a\right)\right) + w_{i,t}, w_{i,t} \sim \mathbb{N}\left(0, Q_t\right) \quad (14)$$

From equations (11–13), Q_t also plays the role of the hybrid model error covariance matrix. In other words, the surrogate model error is associated with $M_t(\cdot)$, and its error covariance matrix is also estimated by equation (4). Q_t is a symmetric positive semidefinite matrix and is itself estimated using the EnKF. In the optimization process, different weights are given to different states according to their uncertainties.

TABLE 1. Neural network architecture in EnKF-FCNN.

Neural network architecture	
Input size	m
Output size	m
Number of layers	3
Number of weights	4000
Activation function	*ReLU*
Optimizer	Adam
Loss	mean_squared_error
Metrics	Accuracy

There are many parameter configurations in addition to the input and output of the training sets. The *epochs* are defined as a single training iteration of all batches in forward and backward propagation. This means that one cycle is a single forward and backward transfer of the entire input data. The *batch_ size* is defined as the number of samples in each batch when performing gradient descent. The validation sets *val_split* is a certain proportion of the data from the training sets. The neural network architecture is shown in Table 1 An FCNN framework based on TensorFlow and an EnKF-FCNN algorithm framework are built in Appendix A and Appendix B, respectively.

D. ASSIMILATION EVALUATION METRICS

(1) The root mean square error (*RMSE*) is used to quantify the assimilation performance, as shown in equation (15).

$$RMSE = \sqrt{\frac{1}{m}\sum_{t=0}^{m}(X_{i,t}^{a} - x(t))^2}, \tag{15}$$

where m is the integral number of times and $x(t)$ is one of the variables of the true states (Lorenz-63 or Lorenz-96) at time t. $X_{i,t}^{a}$ represents the analysis values of the random variables (assimilation results) at time t.

(2) The mean absolute error (*MAE*) is chosen as the cost function to determine the influence of the surrogate model on data assimilation performance. The formula for the *MAE* is shown in equation (16).

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|X_{i,t}^{a} - x(t)\right| \tag{16}$$

III. RESULTS

A. NUMERICAL EXPERIMENTS USING THE LORENZ-63 MODEL

The Lorenz-63 model is a nonlinear spectral model used in the study of the finite amplitude convection of a fluid and proposed by Lorenz and Saltzman in 1963 [41], [42]. Because of its nonlinear chaotic characteristics and low dimensionality, the Lorenz-63 model is often used to verify the performance of data assimilation systems. The Lorenz-63 model is defined as shown in equation (17):

$$\begin{aligned}\frac{dx(t)}{dt} &= \sigma(y(t) - x(t))\\ \frac{dy(t)}{dt} &= rx(t) - y(t) - x(t)z(t)\\ \frac{dz(t)}{dt} &= x(t)y(t) - bz(t)\end{aligned} \tag{17}$$

In this configuration, $\sigma = 10, r = 28, b = 8/3$, the integration of the system is based on fourth-order Runge-Kutta explicit iterative methods [43]. The integral time step is $t = 0.01$ units, the dimension is $n = 3$, and the integral iterations (the length of the assimilation or training window) is m = 1000. The observation time step is $Obs = 0.08$ units (every eight integration time steps). The size of the ensemble is $N = 50$. These configurations have been used in previous studies [44]. This study chooses the variable $x(t)$ to observe the assimilation results.

1) TRAINING THE SURROGATE MODEL

For the dynamic system described by equation (14), we run the system and record the true value $x(t)$ of the system. Then, the observations add Gaussian noise to the true value $x(t)$, and the EnKF algorithm is applied to generate a sequence of forecast values, $X_{i,1}^{f}, X_{i,2}^{f}, \cdots, X_{i,\mathrm{m}}^{f}$, and a sequence of analysis values, $X_{i,1}^{a}, X_{i,2}^{a}, \cdots, X_{i,\mathrm{m}}^{a}$. The length of the assimilation or training window is 1000, $Q_{\mathrm{t}} = 0.1$, and $R_{\mathrm{t}} = 0.1$. The size of the ensemble is $N = 50$. The training sets are built by the time series $\{X_{i,m}^{f}, X_{i,m}^{a}\}$. In the FCNN, the other parameter configurations are the number of training steps, $epochs = \{1, 2, \ldots, 200\}$, $batch\_size = 8$, and validation sets, $val\_split = 0.1$. Regarding the entire training dataset, 90% of the data are used for training, and 10% are used for validation. Then, the surrogate model $\xi(\cdot)$ is trained.

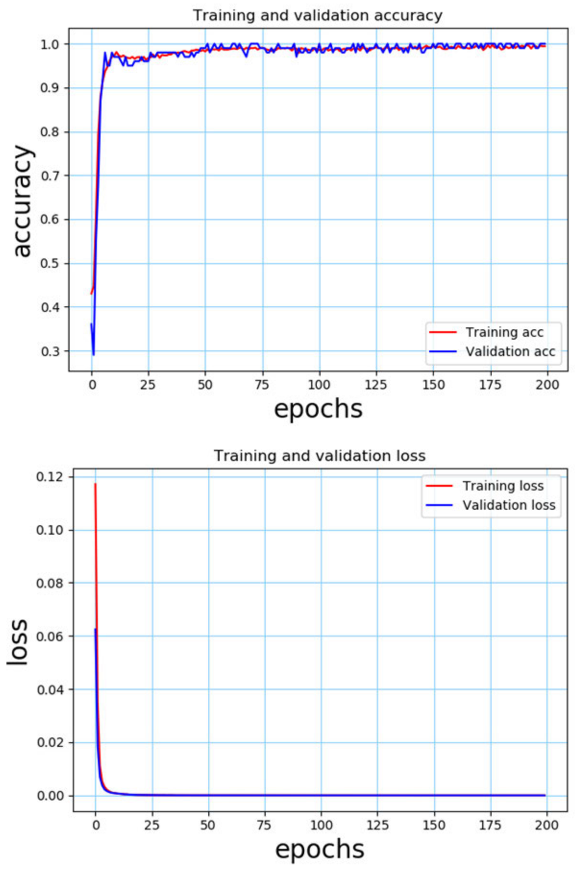


FIGURE 4. Process of training the surrogate model: *epochs*={1, 2, …, 200}, the size of the training sets is 1000 (training: 90%, validation: 10%), and the *batch_size* = 8.

To verify the performance of the surrogate model, the training accuracy (loss) and the validation accuracy (loss) are compared using different *epochs*. Figure 4 shows the following: (1) the training and validation accuracies perform better when the number of *epochs* increases, and they are over 95% when *epochs*=200. (2) The training and validation losses are low when *epochs* ≥ 5.

2) PERFORMANCE OF THE EnKF-FCNN WITH DIFFERENT MODEL ERROR COVARIANCES

In the EnKF-FCNN, the surrogate model $\xi(\cdot)$ is added to the forecast step. The forecast values are calculated by equation (11). In these experiments, the model error covariance

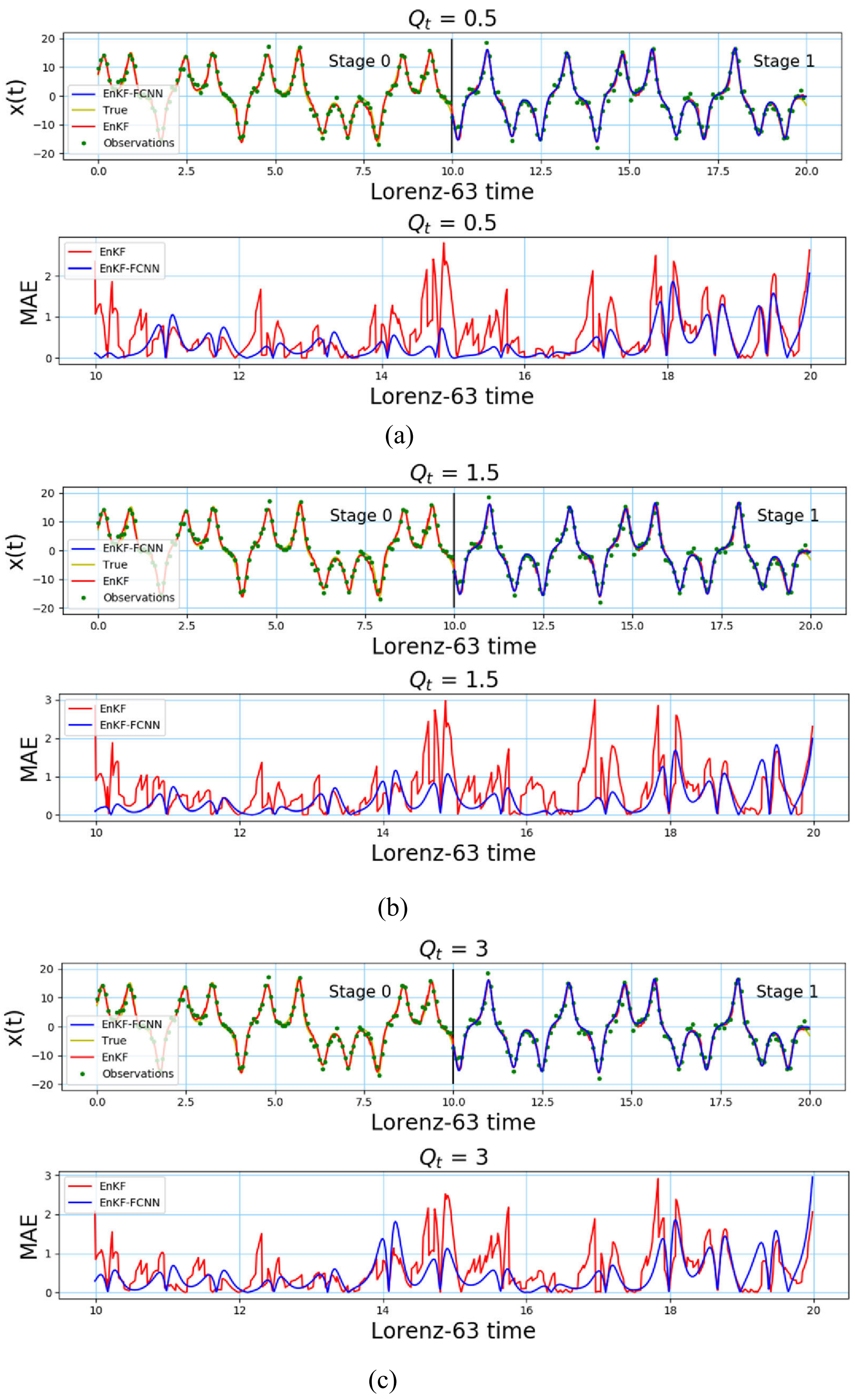


FIGURE 5. ***MAE* performance comparison of two approaches (EnKF-FCNN and EnKF, $Q_t = \{0.5, 1.5, 3\}$). The horizontal axis represents the Lorenz-63 time, and the vertical axis represents the *MAE*. The red solid line represents MAE_{EnKF}, and the blue solid line represents $MAE_{\text{EnKF-FCNN}}$. Panel (a) shows $Q_t = 0.5$, Panel (b) shows $Q_t = 1.5$, and Panel (c) shows $Q_t = 3$.**

TABLE 2. ***RMSEs* of two approaches (EnKF and EnKF-FCNN) with different Q_ts.**

Q_t	0.5	1.5	3
$RMSE_{\text{EnKF}}$	1.053	1.215	1.356
$RMSE_{\text{EnKF-FCNN}}$	0.699	0.741	0.802

$Q_t = \{0.5, 1.5, 3\}$ and the observation error covariance $R_t = 2$. The other configurations are the same as in the training of the surrogate model. Figure 5 compares the *MAE* of the EnKF-FCNN approach with that of the EnKF. The *RMSEs* of the different approaches are compared in Table 2. The conclusion of this experiment is as follows: (1) the *MAE* and *RMSE* of the EnKF-FCNN approach are significantly lower than those of the EnKF. (2) As Q_t increased, although the *RMSE* of the new method also increased, the rate of the increase was not as fast as that of the EnKF. These results show that the EnKF-FCNN approach performs better than the EnKF, especially when the model error increases.

3) PERFORMANCE OF THE EnKF-FCNN WITH DIFFERENT OBSERVATION ERROR COVARIANCES

Another group of experiments with the observation error covariance $R_t = \{1, 2, 3\}$ is conducted using the model error covariance $Q_t = 0.1$, which is different from that used in the training process. The other configurations are the same as in the previous experiments. Figure 6 also compares the *MAE* of the EnKF-FCNN approach with that of the EnKF. The same result is obtained, which is that the *MAE* of the EnKF-FCNN approach is significantly lower than that of the EnKF. The *RMSEs* of the different approaches are compared in Table 3. Through experiments, $RMSE_{\text{EnKF−FCNN}} = \{0.383, 0.471, 0.664\}$ and $RMSE_{\text{EnKF}} = \{0.598, 0.732, 0.913\}$ when $R_t = \{1, 2, 3\}$, respectively. The *RMSE* of the EnKF becomes increasingly severe when the observation error increases. However, the *RMSE* of the EnKF-FCNN change less. Therefore, the EnKF-FCNN has a good effect on reducing the influence of the observation error.

4) PERFORMANCE OF THE EnKF-FCNN WITH DIFFERENT OBSERVATION TIME STEPS

The last set of experiments compares the performance of the EnKF-FCNN approach with that of the EnKF built using different observation time steps $Obs = \{0.02, 0.1, 0.16\}$ to demonstrate the effectiveness of using sparse observations. The model error covariance $Q_t = 0.1$, and the observation error covariance $R_t = 2$. Figure 7 also compares the *MAE* of the EnKF-FCNN approach with that of the EnKF with different *Obs*s. The *RMSEs* of the different approaches are compared in Table 4. The results demonstrated the following: (1) the performance of both methods is good when *Obs* is smaller (= 0.02) and the EnKF-FCNN is slightly better. (2) $RMSE_{\text{EnKF−FCNN}}$ is clearly lower than $RMSE_{\text{EnKF}}$ when *Obs* increases ($\geq$ 0.1), especially when *Obs*=0.16, $RMSE_{\text{EnKF−FCNN}} = 1.174$, and $RMSE_{\text{EnKF}} = 1.762$. The choices of Q_t and R_t do not seem to be critical to the EnKF-FCNN performance as long as the *Obs* is short. However, when the *Obs* is longer, the advantage of the EnKF-FCNN over the EnKF becomes quite obvious.

TABLE 3. ***RMSEs* of two approaches (EnKF and EnKF-FCNN) with different R_ts.**

R_t	1	2	3
$RMSE_{\text{EnKF}}$	0.598	0.732	0.913
$RMSE_{\text{EnKF-FCNN}}$	0.383	0.471	0.664

TABLE 4. ***RMSEs* of two approaches (EnKF and EnKF-FCNN) with different *Obss*.**

Obs	0.02	0.1	0.16
$RMSE_{\text{EnKF}}$	0.402	1.273	1.762
$RMSE_{\text{EnKF-FCNN}}$	0.293	0.732	1.174

B. NUMERICAL EXPERIMENTS USING THE LORENZ-96 MODEL

The Lorenz-96 system is also widely used to verify various assimilation algorithms [45]. The expression is shown as follows:

$$\frac{dx_j(t)}{dt} = -x_{j-2}(t)x_{j-1}(t) + x_{j-1}(t)x_{j+1}(t) - x_j(t) + F, \quad j = 0, 1, 2 \cdots \tag{18}$$

These variables represent several basic features of the atmosphere, such as nonlinear advection-like terms, a damping term, and an external forcing. In this experiment, we set $j = \{0, 1, \ldots, 39\}$ and $F = 8$. For computational stability, the time step is 0.05 units. A fourth-order Runge-Kutta scheme is used for temporal integration in this study. These configurations have also been used in previous studies.

In the training model process, the model error covariance $Q_t = 0.1$ and the observation error covariance $R_t = 0.1$. The number of training *epochs* = 200, *batch_size* = 8, and validation set *val_split* = 0.1. The size of the ensemble $N = 50$. Using the FCNN algorithm, the surrogate model $\xi(\cdot)$ is trained.

In the assimilation process, the surrogate model $\xi(\cdot)$ is used to calculate the forecast values. The observation error covariance $R_t = 2$, and the other configurations are the same as in the training process. The *MAE* is used to reflect the error of the analysis values with the true values. Figure 8 compares the performance of the EnKF-FCNN with that of the EnKF. In these configurations, $RMSE_{\text{EnKF}} = 1.272$ and $RMSE_{\text{EnKF−FCNN}} = 0.864$. These results show that the EnKF-FCNN performs better than the EnKF. Another group of experiments, with $Q_t = 2$ and $R_t = 2$, is also conducted in the numerical simulation. All the other parameters remain the same. Figure 9 shows the performance of the two approaches in these configurations. The $RMSE_{\text{EnKF}}$ and $RMSE_{\text{EnKF−FCNN}}$ are equal to 2.121 and 1.067, respectively. In conclusion, the *MAE*s and *RMSE*s demonstrate that the EnKF-FCNN approach significantly improves the performance.

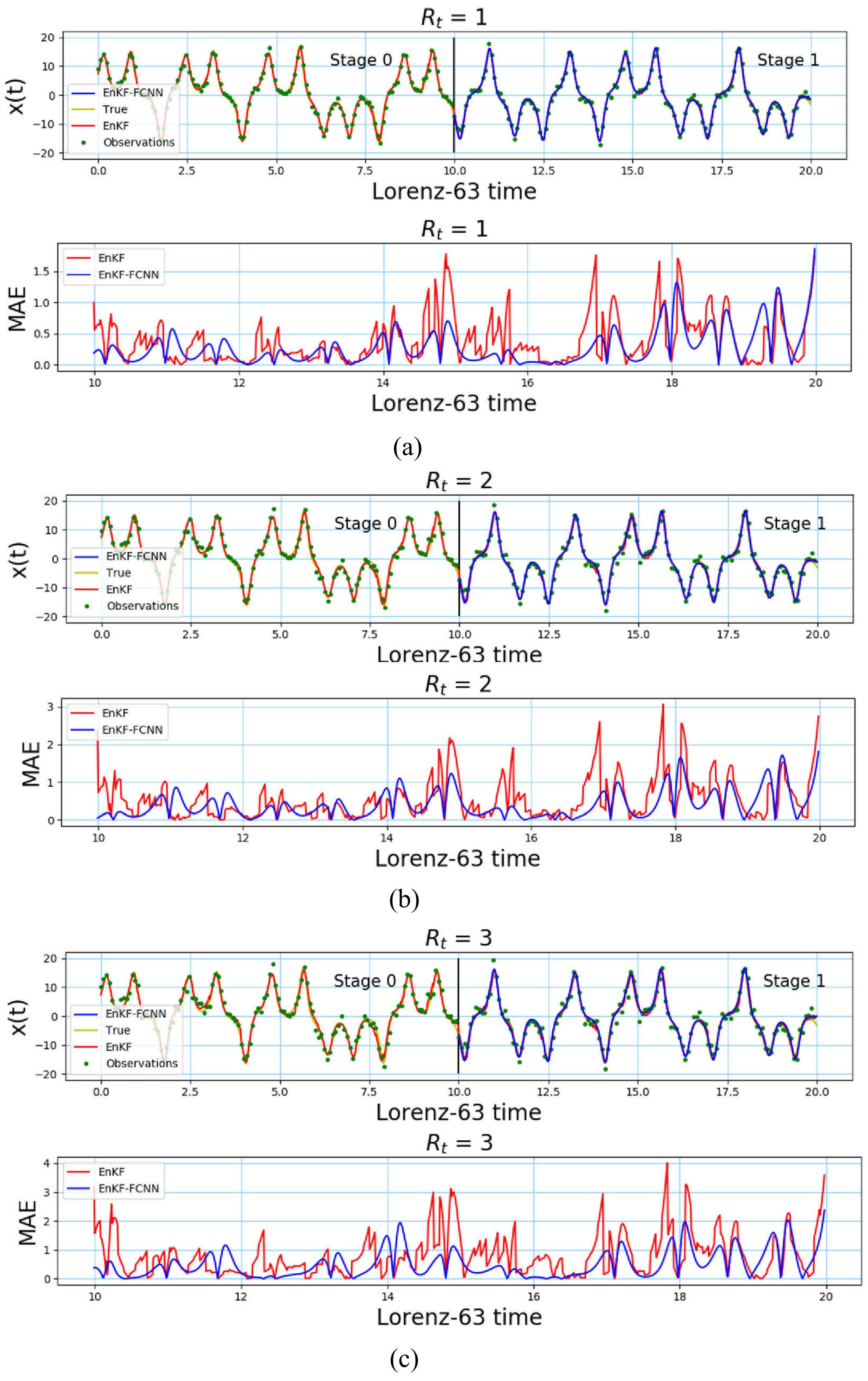
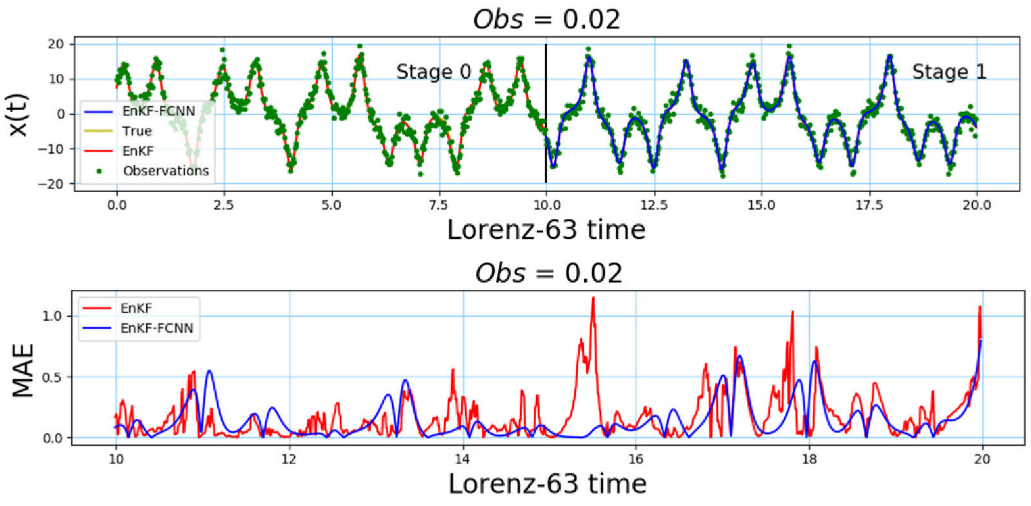
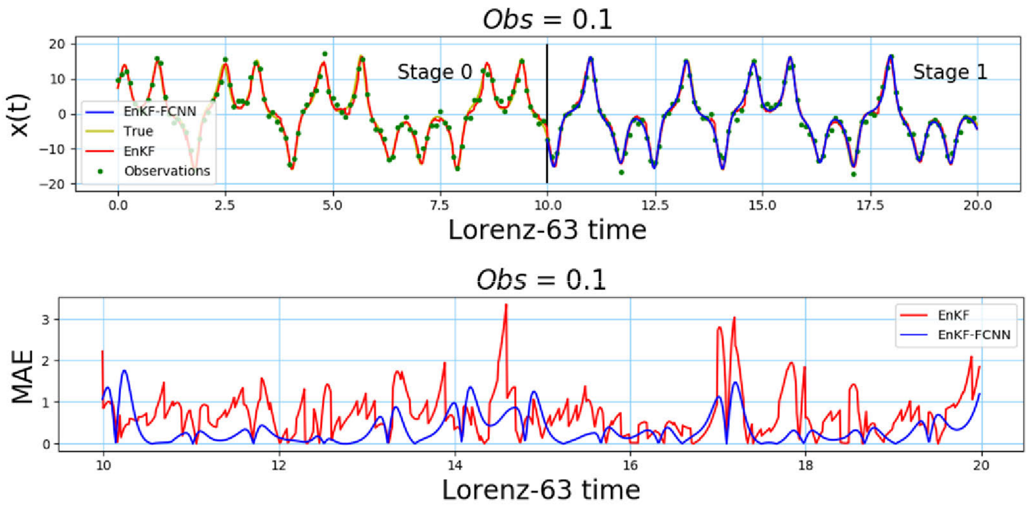


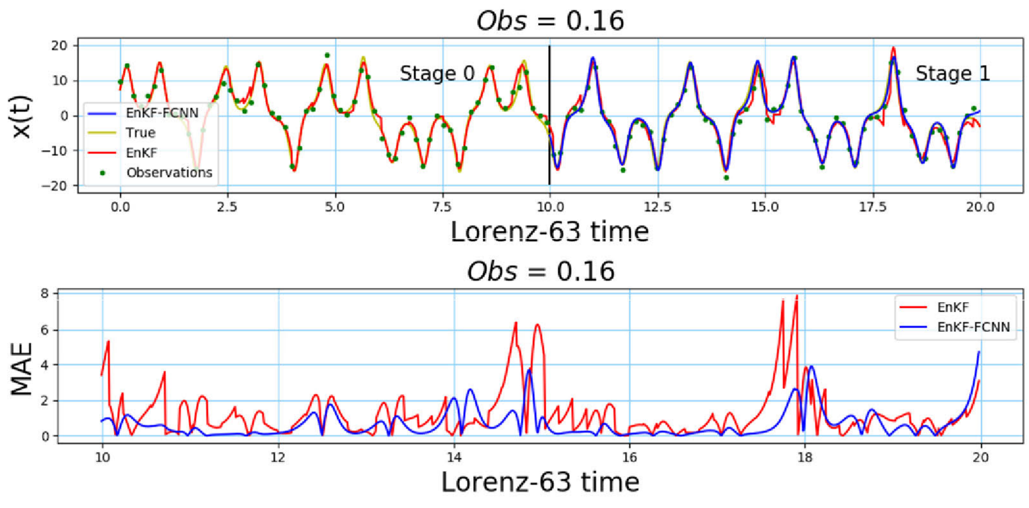
FIGURE 6. ***MAE*** **performance comparison of two approaches (EnKF-FCNN and EnKF, $R_t = \{1, 2, 3\}$). Panel (a) shows $R_t = 1$, Panel (b) shows $R_t = 2$, and Panel (c) shows $R_t = 3$.**



(a)



(b)



(c)

FIGURE 7. ***MAE*** **performance comparison of two approaches (EnKF-FCNN and EnKF, *Obs*={0.02, 0.1, 0.16}). Panel (a) shows *Obs* = 0.02, Panel (b) shows *Obs* = 0.1, and Panel (c) shows *Obs* = 0.16.**

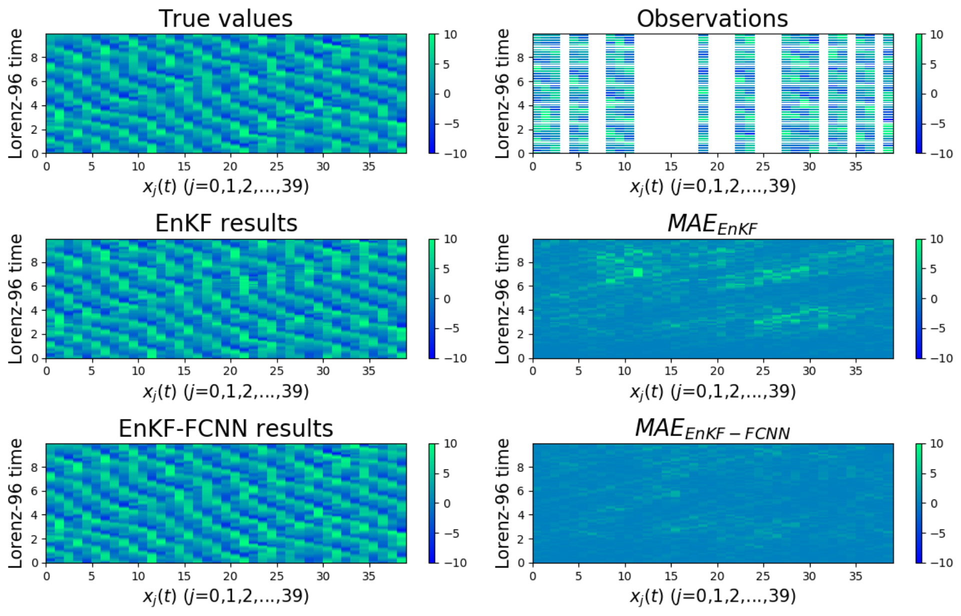


FIGURE 8. **Performance comparison of the two approaches ($R_t = 2$ and $Q_t = 0.1$). The horizontal axis represents $x_j(t)$ $j = \{0, 1, \cdots, 39\}$, and the vertical axis represents the time of the Lorenz-96 model.**

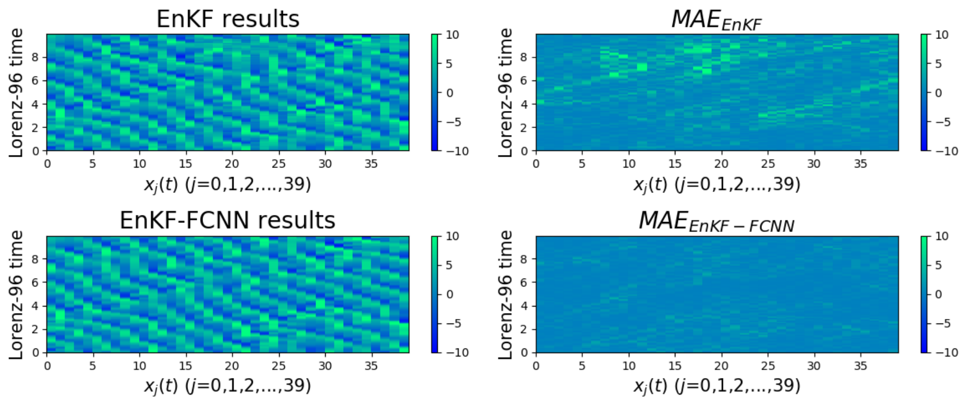


FIGURE 9. **Performance comparison of the two approaches ($R_t = 2$ and $Q_t = 2$).**

IV. DISCUSSION

A. TRAINING AND LOSS

With fewer training epochs, the learning efficiency is low, and the accuracy of the training model will be poor. Therefore, a reasonable number of training step *epochs* is an important indicator of the quality of the model. In this study, from the accuracy and loss based on different *epochs*, the accuracy reaches 95%, training stops, and then, the training model has high accuracy. Loss is the punishment for bad forecasting. That is, loss indicates how accurate the model's predictions are for a single sample. If the model's predictions are completely accurate, then the loss is 0; otherwise, the loss is large. The goal of the training model is to find a set of weights and deviations with a smaller average loss from all the samples.

B. UNCERTAINTIES

The EnKF-FCNN approach performs fairly well in the Lorenz-63 and Lorenz-96 models. However, there are still some uncertainties and limitations to the current study. For instance, (1) although the EnKF-FCNN approach works well in Lorenz models, choosing an appropriate size for the training time step and hidden layers still needs to be addressed. (2) This new approach has not been verified in high-dimensional models (the quasi-geostrophic model (QG), the Noah-MP land surface model (NFLS), and the weather research and forecasting model (WRF)). (3) The application of the EnKF-FCNN approach to multiscale and larger systems is another important challenge for the future. Merging multiscale ML and sparse coding greatly improves saliency detection [46]. A multiscale convolutional neural network effectively solves remote sensing images with complex backgrounds [47]. A multiscale target detection algorithm based on a region proposal convolutional neural network can detect differences in the target scales of images and improve the detection speed [48]. Therefore, coupling the EnKF-FCNN strategies with multiscale space theory is a very promising research direction.

C. THE COMPUTATIONAL COST

In the EnKF-FCNN approach, the computational cost of Stage 0 is related to the cost of the FCNN step. The neural network computing capacity largely depends on the amount of data and the depth and complexity of the network. This study focuses on two factors to reduce the computing cost: (1) vectorization programming, which first appears in the MATLAB programming language. Because MATLAB and Python are analytical executions, the efficiency is low when using the basic cycle. However, when choosing the vectorization method for calculation, the computer can run multiple data simultaneously in parallel, and the computing performance will be greatly improved. (2) For ML, the cost of data acquisition can be very expensive, so selecting reasonable training datasets for the model is critical. The main tools to improve the performance of learning algorithms are learning curves and verification curves. Taking Lorenz-63 as an example, when the size of the training sets is $m = 10^2$, the highest training accuracy of the model is only approximately 70%; when $m = 10^4$, the training accuracy and verification loss rate of the model are good, which was similar to $m=10^3$. Therefore, training sets that are too large will also cause resource waste and high calculation costs. An effective approach is to end the training when the training accuracy or validation loss has not improved over several cycles. This will save considerable computing costs.

V. CONCLUSION

In this study, a new method for an FCNN coupled with an EnKF is discussed, including the generation of training sets, the training of the surrogate model and how to apply it. To verify the performance of the surrogate model, the training accuracy (loss) and validation accuracy (loss) were compared using different *epochs*. The algorithm is verified by a numerical example of model uncertainty. The performance of the EnKF-FCNN is proven with different model error covariances, different observation error covariances, and different observation time steps. The new method is applied to noisier observed systems to realize observation error correction, and it is also applied to sparse observations to improve assimilation performance. The hybrid approach was designed for a threefold scope: (1) coupling the original dynamic model with the FCNN-trained model is built; (2) the assimilation process includes a "two-stage" procedure. Stage 0 generates the training sets and trains the surrogate model. Then, the hybrid model is used for the next period assimilation in Stage 1. (3) The FCNN is coupled with the DA algorithm.

Nevertheless, the effectiveness of the proposed algorithm has yet to be verified in specific fields with large state spaces and more complex internal and external mechanisms. Future work should include the implementation of a DA+NN under more general conditions [49], [50]. For instance, the combination of parametric model inaccuracy and structural model uncertainty will be studied in the future [51]. In addition, modern deep learning tools (such as CNNs and RNNs) can also be introduced into data assimilation to improve the adaptability and performance to data under different conditions [20].

CONFLICTS OF INTEREST

The authors declare no conflicts of interest.

APPENDIX A

An FCNN Framework Based on TensorFlow: This study builds an FCNN framework using Python and TensorFlow. TensorFlow is a symbolic mathematical system based on data

EnKF-FCNN Algorithm

```
Stage 0
    for $t$ from 0 to m do
        -initialization: generate true state, observations, the length range of the data is 0~m, the physical model is, the model error
          covariance $Q_t = 0.1$ and the observation error covariance $R_t = 0.1$
        -Forecast step: generate forecast values $(X^f_{i,1}, X^f_{i,2}, \cdots, X^f_{i,m})$
        -Analysis step: generate analysis values $(X^a_{i,1}, X^a_{i,2}, \cdots, X^a_{i,m})$
    end
    for *epochs* from 1 to 200 do
        -Training sets: $\{X^f_{i,1}, X^f_{i,2}, \cdots, X^f_{i,m}, X^a_{i,1}, X^a_{i,2}, \cdots, X^a_{i,m}\}$
        -Normalization: the training sets are normalized from $[-1, 1]$
        -Neural network configuration: *ReLU* activation function, *Adam* optimizer, *Mean_squared_error* loss function, *Accuracy* evaluation
          standard
        -Training: *batch_size* = 8, *val_split* = 0.1. Generate the surrogate model $\xi(\cdot)$
    end
    - Calculate the accuracy and loss
Stage 1
    for $t$ from m to 2m do
        -initialization: generate true state, observations, the length range of the data is m~2m, the physical model is $\xi(M_t(\cdot))$, chose
          different $Q_t$ and $R_t$
        -Forecast step: generate forecast values $(X^f_{i,m}, X^f_{i,m+1}, \cdots, X^f_{i,2m})$
        -Analysis step: generate analysis values $(X^a_{i,m}, X^a_{i,m+1}, \cdots, X^a_{i,2m})$
    end
End
```

flow programming. TensorFlow is widely used in the programming and implementation of various machine learning algorithms. The form of TensorFlow is the Dist-Belief neural network algorithm library of Google [52]. *Keras* is a third-party high-level neural network API that supports TensorFlow, TheNaO, and MicroSOFT-CNTK [53]. The main steps include the following:

(1) Corresponding libraries are imported, mainly including the following: *keras.model*s, *keras.layers*, *keras.callbacks*, *keras.optimizers*, *sklearn.preprocessing*, *etc*.

(2) The training sets are normalized to [−1, 1] and imported into the input layer. In this study, the training sets are generated from the forecast values and the analysis values of the EnKF process in stage 0.

(3) A neural layer with 32 nodes is defined using the "*ReLU*" activation function, and the dimension of the data of the input layer matched that of Lorenz-63 or Lorenz-96.

(4) It is also important to select the appropriate optimizer, loss function and accuracy evaluation criteria. In this study, the optimizer is "Adam", the loss function is "*Mean_squared_error*", and the accuracy evaluation standard is "*Acc*".

(5) The training process is performed. *epochs* is the number of training epochs. *batch_size* is the number of samples in each batch when performing gradient descent. The share of the training data belonging to the validation set is *val_split.*

(6) The training accuracy (loss) and the validation accuracy (loss) are calculated.

APPENDIX B

See EnKF-FCNN Algorithm

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions.

REFERENCES

[1] Y. Bai and X. Li, "Evolutionary algorithm-based error parameterization methods for data assimilation," *Monthly Weather Rev.*, vol. 139, no. 8, pp. 2668–2685, Aug. 2011, doi: 10.1175/2011MWR3641.1.

[2] R. H. Reichle, "Data assimilation methods in the earth sciences," *Adv. Water Resour.*, vol. 31, no. 11, pp. 1411–1418, Nov. 2008, doi: 10.1016/j.advwatres.2008.01.001.

[3] C. L. Huang and X. Li, "Experiments of soil moisture data assimilation system based on ensemble Kalman filter," *Plateau Meteorol.*, vol. 112, no. 3, pp. 665–671, 2008, doi: 10.1111/j.1745-4557.2006.00081.x.

[4] A. Carrassi and S. Vannitsem, "Accounting for model error in variational data assimilation: A deterministic formulation," *Monthly Weather Rev.*, vol. 138, no. 9, pp. 3369–3386, Sep. 2010, doi: 10.1175/2010MWR3192.1.

[5] R. N. Bannister, "A review of operational methods of variational and ensemble-variational data assimilation," *Quart. J. Roy. Meteorolog. Soc.*, vol. 143, no. 703, pp. 607–633, Jan. 2017, doi: 10.1002/qj.2982.

[6] Y. L. Bai, X. Li, and X. J. Han, "A review of error problems for land data assimilation systems," *Adv. Earth Sci.*, vol. 26, no. 8, pp. 795–804, 2011. [Online]. Available: http://www.adearth.ac.cn/EN/10.11867/j.issn.1001-8166.2011.08.0795

[7] F. Bouttier and P. Courtier, "Data assimilation concept and methods training course," in *Meteorological Training Course Lecture Series*. London, U.K.: ECMWF, 2002, p. 85.

[8] J. J. Ruiz, M. Pulido, and T. Miyoshi, "Estimating model parameters with ensemble-based data assimilation: A review," *J. Meteorolog. Soc. Jpn. Ser. II*, vol. 91, no. 2, pp. 79–99, 2013, doi: 10.2151/jmsj.2013-201.

[9] A. Sanpei, T. Okamoto, S. Masamune, and Y. Kuroe, "A data-assimilation based method for equilibrium reconstruction of magnetic fusion plasma and its application to reversed field pinch," *IEEE Access*, vol. 9, pp. 74739–74751, 2021, doi: 10.1109/ACCESS.2021.3081146.

[10] P. N. Raanes, A. Carrassi, and L. Bertino, "Extending the square root method to account for additive forecast noise in ensemble methods," *Monthly Weather Rev.*, vol. 143, no. 10, pp. 3857–3873, Oct. 2015, doi: 10.1175/MWR-D-14-00375.1.

[11] R. Jin and X. Li, "Improving the estimation of hydrothermal state variables in the active layer of frozen ground by assimilating *in situ* observations and SSM/I data," *Sci. China Ser. D, Earth Sci.*, vol. 52, no. 11, pp. 1732–1745, Nov. 2009, doi: 10.1007/s11430-009-0174-0.

[12] R. C. Aster, C. H. Thuber, and B. Borchers, *Parameter Estimation and Inverse Problems*, 2nd ed. Amsterdam, The Netherlands: Elsevier, 2013.

[13] M. Bocquet, "Parameter-field estimation for atmospheric dispersion: Application to the Chernobyl accident using 4D-Var: Atmospheric dispersion parameter-field estimation," *Quart. J. Roy. Meteorolog. Soc.*, vol. 138, no. 664, pp. 664–681, Apr. 2012, doi: 10.1002/qj.961.

[14] M. Eltahan and S. Alahmadi, "Numerical dust storm simulation using modified geographical domain and data assimilation: 3DVAR and 4DVAR (WRF-Chem/WRFDA)," *IEEE Access*, vol. 7, pp. 128980–128989, 2019, doi: 10.1109/ACCESS.2019.2930812.

[15] P. N. Raanes, M. Bocquet, and A. Carrassi, "Adaptive covariance inflation in the ensemble Kalman filter by Gaussian scale mixtures," *Quart. J. Roy. Meteorolog. Soc.*, vol. 145, no. 718, pp. 53–75, Jan. 2019, doi: 10.1002/qj.3386.

[16] J. Pathak, B. Hunt, M. Girvan, Z. Lu, and E. Ott, "Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach," *Phys. Rev. Lett.*, vol. 120, no. 2, Jan. 2018, Art. no. 024102, doi: 10.1103/PhysRevLett.120.024102.

[17] X. Shi, Z. Gao, L. Lausen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-C. Woo, "Deep learning for precipitation nowcasting: A benchmark and a new model," 2017, *arXiv:1706.03458*. [Online]. Available: http://arxiv.org/abs/1706.03458

[18] D.-C. Park, "A time series data prediction scheme using bilinear recurrent neural network," in *Proc. Int. Conf. Inf. Sci. Appl.*, 2010, pp. 1–7.

[19] J. Brajard, A. Carrassi, M. Bocquet, and L. Bertino, "Combining data assimilation and machine learning to emulate a dynamical model from sparse and noisy observations: A case study with the Lorenz 96 model," *J. Comput. Sci.*, vol. 44, Jul. 2020, Art. no. 101171, doi: 10.1016/j.jocs.2020.101171.

[20] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[21] Y. Tang, W. W. Hsieh, B. Tang, and K. Haines, "A neural network atmospheric model for hybrid coupled modelling," *Climate Dyn.*, vol. 17, nos. 5–6, pp. 445–455, Mar. 2001.

[22] H. F. Campos Velho, N. L. Vijaykumar, S. Stephany, A. J. Preto, and A. G. Nowosad, *A Neural Network Implementation for Data Assimilation Using MPI. Application of High Performance Computing in Engineering*, C. A. Brebia, P. Melli, A. Zanasi, Eds. Southampton, U.K.: WIT Press, Southampton, 2002, Sec. 5, pp. 211–220.

[23] F. P. Härter and H. F. de Campos Velho, "New approach to applying neural network in nonlinear dynamic model," *Appl. Math. Model.*, vol. 32, no. 12, pp. 2621–2633, Dec. 2008, doi: 10.1016/j.apm.2007.09.006.

[24] J. Harlim, "Model error in data assimilation," in *Nonlinear and Stochastic Climate Dynamics*, C. Franzke and T. O'Kane, Eds. Cambridge: Cambridge Univ. Press, 2017, pp. 276–317.

[25] M. Bocquet, J. Brajard, A. Carrassi, and L. Bertino, "Data assimilation as a deep learning tool to infer ODE representations of dynamical models," *Nonlin. Processes Geophys.*, vol. 26, no. 1, pp. 143–162, 2019, doi: 10.5194/npg-2019-7.

[26] R. Lguensat, P. Tandeo, P. Ailliot, M. Pulido, and R. Fablet, "The analog data assimilation," *Monthly Weather Rev.*, vol. 145, no. 10, pp. 4093–4107, Oct. 2017.

[27] F. Hamilton, T. Berry, and T. Sauer, "Predicting chaotic time series with a partial model," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 92, no. 1, Jul. 2015, Art. no. 010902, doi: 10.1103/PhysRevE.92.010902.

[28] F. Hamilton, T. Berry, and T. Sauer, "Ensemble Kalman filtering without a model," *Phys. Rev. X*, vol. 6, no. 1, Mar. 2016, Art. no. 011021, doi: 10.1103/PhysRevX.6.011021.

[29] P. Laloyaux, M. Bonavita, M. Chrust, and S. Gürol, "Exploring the potential and limitations of weak-constraint 4D-Var," *Quart. J. Roy. Meteorolog. Soc.*, vol. 146, no. 733, pp. 4067–4082, Oct. 2020, doi: 10.1002/qj.3891.

[30] M. Bocquet, ''Ensemble Kalman filtering without the intrinsic need for inflation,'' *Nonlinear Processes Geophys.*, vol. 18, no. 5, pp. 735–750, Oct. 2011, doi: 10.5194/npg-18-735-2011.
[31] M. Bocquet, P. N. Raanes, and A. Hannart, ''Expanding the validity of the ensemble Kalman filter without the intrinsic need for inflation,'' *Nonlinear Processes Geophys.*, vol. 22, no. 6, pp. 645–662, Nov. 2015, doi: 10.5194/npg-22-645-2015.
[32] G. Evensen, ''The ensemble Kalman filter: Theoretical formulation and practical implementation,'' *Ocean Dyn.*, vol. 53, no. 4, pp. 343–367, Nov. 2003, doi: 10.1007/s10236-003-0036-9.
[33] G. Evensen, *Data Assimilation: The Ensemble Kalman Filter*, 2nd ed. Berlin, Germany: Springer, 2009.
[34] P. Sakov, D. S. Oliver, and L. Bertino, ''An iterative EnKF for strongly nonlinear systems,'' *Monthly Weather Rev.*, vol. 140, no. 6, pp. 1988–2004, Jun. 2012, doi: 10.1175/MWR-D-11-00176.1.
[35] T. M. Khoshgoftaar, E. B. Allen, J. P. Hudepohl, and S. J. Aud, ''Application of neural networks to software quality modeling of a very large telecommunications system,'' *IEEE Trans. Neural Netw.*, vol. 8, no. 4, pp. 902–909, Jul. 1997, doi: 10.1109/72.595888.
[36] Z. Liu, Q. Peng, Y. G. Xu, G. Ren, and H. T. Ma, ''Misalignment calculation on off-axis telescope system via fully connected neural network,'' *IEEE Photon. J.*, vol. 12, no. 4, Aug. 2020, Art. no. 0600112.
[37] D. Laredo, S. F. Ma, G. Leylaz, O. Schütze, and J. Q. Sun, ''Automatic model selection for fully connected neural networks,'' *Int. J. Dyn. Control*, vol. 8, no. 4, pp. 1063–1080, 2020, doi: 10.1007/s40435-020-00708-w.
[38] L. Zhao, Z. Shang, L. Zhao, T. Zhang, and Y. Y. Tang, ''Software defect prediction via cost-sensitive Siamese parallel fully-connected neural networks,'' *Neurocomputing*, vol. 352, pp. 64–74, Aug. 2019, doi: 10.1016/j.neucom.2019.03.076.
[39] D. P. Kingma and J. Ba, ''Adam: A method for stochastic optimization,'' *Comput. Sci.*, vol. 12, pp. 1–15, Dec. 2014.
[40] N. Ketkar, ''Stochastic gradient descent,'' in *Deep Learning With Python*. Berkeley, CA, USA: Apress, 2017, doi: 10.1007/978-1-4842-2766-4_8.
[41] J. L. Anderson and S. L. Anderson, ''A Monte Carlo implementation of the nonlinear filtering problem to produce ensemble assimilations and forecasts,'' *Monthly Weather Rev.*, vol. 127, no. 12, pp. 2741–2758, 1999, doi: 10.1175/1520-0493(1999)127<2741:AMCIOT>2.0.CO;2.
[42] I. Hoteit, D.-T. Pham, G. Triantafyllou, and G. Korres, ''A new approximate solution of the optimal nonlinear filter for data assimilation in meteorology and oceanography,'' *Monthly Weather Rev.*, vol. 136, no. 1, pp. 317–334, Jan. 2008, doi: 10.1175/2007MWR1927.1.
[43] J. R. Dormand and P. J. Prince, ''A family of embedded Runge–Kutta formulae,'' *J. Comput. Appl. Math.*, vol. 6, no. 1, pp. 19–26, 1980.
[44] P. J. van Leeuwen, ''Nonlinear data assimilation in geosciences: An extremely efficient particle filter,'' *Quart. J. Roy. Meteorolog. Soc.*, vol. 136, no. 653, pp. 1991–1999, Oct. 2010, doi: 10.1002/qj.699.
[45] E. Lorenz, ''Predictability: A problem partly solved,'' in *Proc. Seminar Predictability ECMWF United Kingdom*, vol. 1, 1996, pp. 1–19.
[46] Y. Tang, L. Zhu, and J. Wu, ''Salient object detection on multiscale learning and sparse coding,'' in *Proc. 36th Chin. Control Conf. (CCC)*, Jul. 2017, pp. 11047–11052, doi: 10.23919/ChiCC.2017.8029121.
[47] Z. Shao, Y. Pan, and C. Y. Diao, ''Cloud detection in remote sensing images based on multiscale features-convolutional neural network,'' *IEEE Trans. Geosci. Remote Sens.* vol. 57, no. 6, pp. 4062–4076, Jun. 2019, doi: 10.1109/TGRS.2018.2889677.
[48] T. Y. Yang, *Multi-Scale Target Detection Research Based on Convolution Neural Network*. Wuhan, China: Huazhong Univ. Science and Technology, 2017.
[49] R. Fablet, P. Viet, R. Lguensat, and B. Chapron, ''Data-driven assimilation of irregularly-sampled image time series,'' in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2017, pp. 4302–4306.
[50] G. Lu, Z. Lie, and L. Hang, ''Deep belief network software defect prediction model,'' *Comput. Sci.*, vol. 44, no. 1, pp. 229–233, 2017.
[51] M. Reichstein, G. Camps-Valls, B. Stevens, M. Jung, J. Denzler, N. Carvalhais, and M. Prabhat, ''Deep learning and process understanding for data-driven Earth system science,'' *Nature*, vol. 566, pp. 195–204, Feb. 2019, doi: 10.1038/s41586-019-0912-1.
[52] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, and J. Dean, ''Tensorflow: A system for large-scale machine learning,'' in *Proc. OSDI*, 2016, pp. 265–283.
[53] D. Baylor, E. Breck, H. T. Cheng, N. Fiedel, C. Y. Foo, and Z. Haque, ''TFX: A TensorFlow-based production-scale machine learning platform,'' in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2017, pp. 1387–1395, doi: 10.1145/3097983.3098021.

MANHONG FAN received the B.S. and M.S. degrees from Northwest Normal University, Lanzhou, China, in 2008 and 2012, respectively, where he is currently pursuing the Ph.D. degree with the College of Physical and Electrical Engineering. His research interests include data assimilation and measurement control.

YULONG BAI received the B.S. degree from Northwest Normal University, Lanzhou, China, in 1995, the M.S. degree from Lanzhou University, in 1998, and the Ph.D. degree from the University of Chinese Academy of Sciences, in 1998. He is currently a Professor with Northwest Normal University. His research interests include data assimilation and automatic control.

LILI WANG received the B.S. and M.S. degrees from Northwest A&F University, in 2006 and 2009, respectively, and the Ph.D. degree from the University of Chinese Academy of Sciences, in 2018. She is currently an Associate Professor with Northwest Normal University, Lanzhou, China. Her research interest includes wind speed forecasting.

LIN DING received the B.S. degree from Northwest Normal University, Lanzhou, China, in 2020, where she is currently pursuing the Ph.D. degree with the College of Physical and Electrical Engineering. Her research interest includes wind speed forecasting.