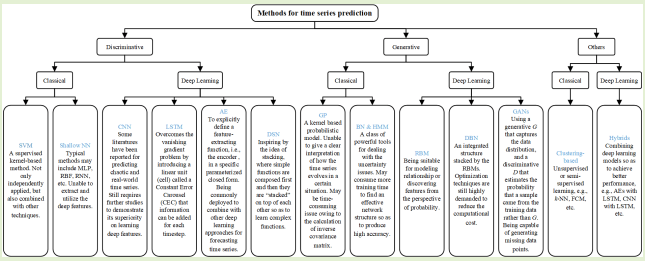
A Review of Deep Learning Models for Time Series Prediction

Zhongyang Han, *Member, IEEE*, Jun Zhao, *Member, IEEE*, Henry Leung, *Fellow, IEEE*, King Fai Ma, and Wei Wang, *Senior Member, IEEE*

***Abstract*—In order to approximate the underlying process of temporal data, time series prediction has been a hot research topic for decades. Developing predictive models plays an important role in interpreting complex real-world elements. With the sharp increase in the quantity and dimensionality of data, new challenges, such as extracting deep features and recognizing deep latent patterns, have emerged, demanding novel approaches and effective solutions. Deep learning, composed of multiple processing layers to learn with multiple levels of abstraction, is, now, commonly deployed for overcoming the newly arisen difficulties. This paper reviews the state-of-the-art developments in deep learning for time series prediction. Based on modeling for the perspective of conditional or joint probability, we categorize them into discriminative, generative, and hybrids models. Experiments are implemented on both benchmarks and real-world data to elaborate the performance of the representative deep learning-based prediction methods. Finally, we conclude with comments on possible future perspectives and ongoing challenges with time series prediction.**



***Index Terms*—Review, discriminative models, generative models, deep learning, time series prediction.**

I. Introduction

TIME series, as a collection of temporal observations, has attracted intensive attention initiating various studies and developments in the field of machine learning and artificial intelligence. Among the research aspects ranging from dimensionality reduction to data segmentation, time series prediction for acquiring future trends and tendency is one of the most important subjects. The results can provide a basis for various applications, e.g., production planning, control, optimization, etc. [1]–[3]. Therefore, numerous models have been proposed for solving this problem, e.g., Auto Regressive Integrated Moving Average (ARIMA) [4], [5], filtering-based methods [6], [7], support vector machines [8], etc.

Conventional techniques for time series prediction were limited in their ability to process big data with high dimensionality, as well as efficiently represent complex functions [9]. Also, designing an effective machine learning system requires considerable domain expertise of data. Recently, deep learning has emerged as the forefront of advanced artificial intelligence. Deep learning describes models that utilize multiple layers to represent latent features at a higher and more abstract level [10]. The representations are learned from data rather than constructed by human engineers. Regarding the above superiorities, deep learning-based models have been successfully applied in many fields pertinent to time series prediction, including remote sensing [11], multi-sensor fusion [12], [13], etc. They have also been explored as an effective approach to discovering complex relationships between multiple time series. Despite their effectiveness, we note that various deep learning models have their own individual advantages and drawbacks.

In this paper, we review a variety of deep learning models for time series prediction that have been developed to explicitly capture temporal relationships. The rest of this paper is organized as follows: Section II will give a brief description on time series prediction, including 3 groups of definitions and mathematical formulations. Section III elaborates a wealth of

Manuscript received May 13, 2019; revised June 5, 2019; accepted June 6, 2019. Date of publication June 20, 2019; date of current version February 17, 2021. This work was supported in part by the National Key R&D Program under Grant 2017YFA0700300, in part by the National Natural Sciences Foundation of China under Grant 61833003, Grant 61703071, Grant 61603069, and Grant 61533005, and in part by the Fundamental Research Funds for the Central Universities of China under Grant DUT18RC(3)074. The associate editor coordinating the review of this article and approving it for publication was Dr. You Li. *(Corresponding author: Zhongyang Han.)*

Z. Han, J. Zhao, and W. Wang are with the School of Control Sciences and Engineering, Dalian University of Technology, Dalian 116023, China (e-mail: hanzhongyang@dlut.edu.cn; zhaoj@dlut.edu.cn; wangwei@dlut.edu.cn).

H. Leung and K. F. Ma are with the Department of Electrical and Computer Engineering, University of Calgary, Calgary, AB T2N 1N4, Canada (e-mail: leungh@ucalgary.ca; kfma@ucalgary.ca).

Digital Object Identifier 10.1109/JSEN.2019.2923982

models for time series prediction, which are categorized as discriminative, generative and others. In Section IV, experimental studies are implemented on data involving commonly deployed benchmarks and real-world representative industrial data to report the performance of some state-of-the-art deep learning models. Finally, we conclude with a brief discussion of possible future topics in Section V.

II. Time Series Prediction

Generally, the aim of time series prediction is to forecast its value at some future time $t+h$ using available observations from a time series at time t. The detailed definition or mathematical formulation varies with respect to different cases. To provide background for the following sections, this study will introduce 3 typical groups of definitions formulating the problem of time series prediction.

A. Observations at Equal/Unequal Intervals of Time

For a large amount of existing studies, the observations are assumed to be available at equispaced intervals of time [14], [15]. In such a case, the problem of time series prediction can be formulated as follows

$$\hat{x}_{t+h} = f(x_t, x_{t-1}, \cdots\cdots, x_{t-N+1}) \tag{1}$$

where $x_t, x_{t-1}, \cdots\cdots, x_{t-N+1}$ refers to time series data points, $\hat{x}_{t+h}$ is the predicted results. N denotes the number of inputs, also named as embedded dimension in some studies [16], [17]. The timestamp h could be 1 [18], [19], or any positive integer, of which is named as multi-step-ahead prediction [20], [21].

Some studies also break the equispaced interval assumption, instead, processing time series data observed at unequal length of time [22], [23]. Under such circumstances, the problem of time series prediction should be expressed as follows

$$\hat{x}_{t+h} = f(x_{t-l_1}, x_{t-l_2}, \cdots\cdots, x_{t-l_N}, l_1, l_2, \cdots, l_N) \tag{2}$$

where timestamps $l_1, l_2, \cdots, l_N$ denote various time space among the observations.

B. Recursive/Direct Prediction Strategy

In order to predict the several-timesteps-ahead values of a time series, one of the most intuitive approaches is to deploy recursive prediction strategy [24]–[26]. This kind of iterative process can be expressed as follows

$$\begin{aligned}
\hat{x}_{t+1} &= f\left(x_t, x_{t-1}, \cdots\cdots, x_{t-N+1}\right) \\
\hat{x}_{t+2} &= f\left(\hat{x}_{t+1}, x_t, \cdots\cdots, x_{t-N+2}\right) \\
&\cdots\cdots \\
\hat{x}_{t+M} &= f\left(\hat{x}_{t+M-1}, \hat{x}_{t+M-2}, \cdots\cdots, x_{t+M-N}\right)
\end{aligned} \tag{3}$$

where M denotes the number of iterations.

The main drawback of recursive prediction is its accumulated error which gradually deteriorates the prediction accuracy. As such, some researchers are focusing to predict multiple data points in one-time iteration. For instance, Granular Computing [27]–[29], modeling on granules rather than single data points, embraces time-series prediction with data segments in both equal and unequal length. Such direct prediction strategy can be expressed as follows

$$\hat{\boldsymbol{X}} = f(x_t, x_{t-1}, \cdots\cdots, x_{t-N+1}) \tag{4}$$

where $\hat{\boldsymbol{X}} = \left\{\hat{x}_{t+1}, \hat{x}_{t+2}, \cdots\cdots, \hat{x}_{t+M}\right\}^T$ refers to the predicted vector.

C. Univariate/Multivariate Modeling

Time series prediction may be defined with regards to the dimensionality of the modeling variables. The problems are named as univariate [30] and multivariate [31], which are initially proposed for classifying ARIMA models [32], [33]. Detailed definitions for univariate and multivariate time series prediction can be described as following Eq. (5) and (6), respectively.

$$\hat{x}_{t+1} = f(x_t, x_{t-1}, \cdots\cdots, x_{t-N+1}) \tag{5}$$

$$\begin{aligned}
\hat{x}_{t+1} = f(&x_t^1, \cdots, x_{t-N+1}^1, x_t^2, \cdots, x_{t-N+1}^2, \cdots\cdots, \\
&x_t^L, \cdots, x_{t-N+1}^L)
\end{aligned} \tag{6}$$

where L denotes the number of related variables. The Eq. (5) can be regarded as a special case of Eq. (1), of which $h = 1$.

One of the most common conventional approaches when dealing with multivariate series is the Vector Autoregressive (VAR) model which considers linear relationship between variables, and has been widely applied in the field of economics [34]. Other approaches will be discussed in the following sections. The problem of embedded dimension selection also grows with multivariate data.

III. Classic and Deep Learning Models for Time Series Prediction

By modeling in view of conditional or joint probability, the deep learning models can be categorized as discriminative and generative approaches. This categorization is not only effective for classification, but also for time series prediction, for which the mechanism of the models remains the same. Besides, other unclassifiable but typical hybrid approaches are also described in this section.

In order to introduce models in a natural way as well as clarify the difference, we will review one or two classical methods at first for each category in this section. Then, several state-of-the-art deep learning models for time series prediction will be discussed in detail.

A. Discriminative

Discriminative models, or conditional models, are a class of methods depending on the observed data and learning to act from the given statistics. The orientation of these models is with respect to the conditional probability of the target Y given an observation X, which means they can be used to 'discriminate' the target given an observation [35]. As such, two classic and three deep learning-based models are deployed as representatives to be discussed in this section.

1) Representative Classical Model - Support Vector Machine: Support vector machine (SVM), proposed by Vapnik [36] in 1995, is a supervised kernel-based method and always applied to nonlinear time series prediction [37]–[41]. The classical SVM can be summarized as to solve a quadratic programming problem, which tends to cause curse of dimensionality with the increase of training dataset size. As a result, some algorithms, such as Sequential Minimal optimization (SMO), were proposed to overcome the shortcomings [42]. Also, Suykens and his fellowships have constructed a Least Square Support Vector Machine (LSSVM), in which the unequal constraints are replaced by equal ones so that the computing efficiency is remarkably enhanced [43].

SVM was not only independently applied regarding the superiority in modeling and predicting times series, but also combined with other techniques so as to obtain higher predictive accuracy considering the non-stationarity and complexity in time series. For example, in [44], the trend and fluctuation parts hidden in original time series were decomposed by the singular spectrum analysis before the establishment of an SVM. Multiple LSSVM models were built in a neuro-fuzzy framework to construct different local regimes of the input space for improving the accuracy [18]. In addition, SVM models have also been applied in multivariate time series consisting of multiple spatial observations. To overcome the instabilities of spatiotemporal forecasting, a multi-output SVM model with multi-task learning was reported in [45].

2) Representative Classical Model - Shallow Neural Networks: Another classical discriminative model for time series prediction is Neural Networks (NNs). In order to distinguish the machine learning and deep learning models, here we only investigate the shallow, or in other words, traditional or vanilla NNs. With almost 70 years development, NNs have successfully solved problems ranging from regression, classification to feature exaction, inference, etc. Regarding the subject of time series prediction, feedforward NNs were one of the most commonly deployed approach, represented by Multi-Layer Perceptron (MLP) and error Back Propagation (BP) [46]–[51]. These models consist of intuitive calculations among neurons so that they are easy to be interpreted and realized. They have been also used for nonlinear prediction in multivariate domain [52] by combining all inputs in the input layer. However, their shortcomings are also evident when facing complex tasks with high dimensionality and variance, which is low efficiency and inability to approximate complicated functions. By using Radial Basis Function (RBF) as activation functions, RBF network has been an alternative to forecast time series data, exhibiting superiority on convergence rate and being capable of approximating any nonlinear function [53]–[57].

Another wealth of classical networks is recurrent structures. Unlike feedforward NNs, Recurrent Neural Networks (RNNs) connects neurons' output to their inputs. Such a close-loop endows RNNs the ability of memorizing information involving trend and tendency. The training process for RNNs is a well-known strategy named as Backpropagation-Through-Time (BPTT) [58]–[62]. As well, Echo State Network (ESN), first proposed in 2001, has been also widely applied for regression and forecasting. Although currently it draws less attention comparing with the RNNs, its application on time series prediction still performs well on both computational cost and accuracy [63]–[66].

3) Representative Deep Learning Model - Convolutional Neural Network: In 1959, two neurobiologists, Hubel and Wiesel, found a unique neuron structure during the research on cat's receptive fields in striate cortex, which is the prototype of Convolutional Neural Network (CNN) [67]. The CNN was successfully applied for processing image, speech and time series in 1995 by LeCun and Bengio [68]. Using a variation of MLP designed to require minimal preprocessing, CNN is also known as Shift Invariant Artificial Neural Network (SIANN) owing to its shared-weights structure and translation invariance characteristics.

Similar with the conventional NNs, a CNN also consists of three parts, i.e., an input layer, multiple hidden layers and an output layer. Each layer may contain activation functions, such as Rectified Linear Unit (ReLU). The hidden layers typically include fully connected layers and operators involving convolutional layers along with pooling. The purpose of pooling is to achieve invariance to small local distortions and reduce the dimensionality of the feature space [69]. In order to mine the deep information, connected convolution and pooling operators often repeat many times in the network. The use of convolutional operations allows for the number of parameters to be far smaller than a fully connected network, thus resulting in efficient training and inference. The convolution technique has also been deployed on deep belief network, a generative deep learning model which we will review in the following section 3.2, to introduce probability into the pooling process [70].

The CNN model for a univariate sequence will operate using a set of filters or weights in the convolutional layer, and the output of the layer o_l is obtained by simply computing the dot product between the overlapping input x and the weights w similar to an autoregressive manner. $o_l = \sum_1^N w_i x_{t-i} + b_i$, where the receptive field of the filter is of size N. The receptive field determines the number of inputs that can influence the output similar to the autoregressive model. We note that a multivariate sequence may be thought of as a 2D image, for example several works have used the spectrogram as the input for acoustic event processing [71]. By considering time as one axis and the frequency (or in general, multivariate observations for each time) on the other, an "image patch" is formed. This structure allows for finding local patterns in the input series. This is followed by multiple layers of convolutional layers and pooling, and finally a fully connected layer. The basis of convolution uses the same weights for sliding window, and thus tries to learn the correlation and repeating patterns between the variables. However, the receptive field or window size is typically fixed in conventional CNN architectures. Recently, dilated convolutions were introduced where the filter is applied to every d inputs, such that $o_l = \sum_1^k w_i x_{t-d\times i} + b_i$. The use of dilated convolutions in CNN allows the filter to increase the receptive field and thus allowing for access to a longer history of time [74]. The dilated causal convolution architecture

without pooling operations was proposed in WaveNet for improved speech modeling [72].

Some literatures have been reported to use CNN for predicting chaotic as well as real-world time series. For instance, [76] reported a deep CNN model for dynamic occupancy grid prediction with data from multiple sensors. One recent work adapted the dilated CNN architecture to the stock market prediction problem [77]. A deep spatio-temporal residual network, as an extension of CNN, is presented in [78] for citywide crowd flows prediction. An ensemble model is established in [79] for wind power forecasting. Besides, remote sensing is also one of the hot topics for the real application of CNN [80]. To the best of our knowledge, we found that more works utilize the CNN architectures for classification rather than prediction of the next value with multivariate time-series, such as predicting heart failure from heartbeat data [73] or activity from biometric time series [75]. Compared with the extensive application on image recognition [81]–[83], using CNN for time series prediction still requires further studies to demonstrate its superiority on learning deep features so as to obtain better performance.

4) Representative Deep Learning Model - Long-Short Term Memory: Standard RNN's suffer from a vanishing and exploding gradient problem, as the BPTT procedure depends exponentially on weights for each timestep, thus failing to learn information over typically 5-10 timesteps [84]. The Long-Short Term Memory (LSTM) recurrent network overcomes the vanishing gradient problem by introducing a linear unit (cell) called a Constant Error Carousel (CEC) that information can be added for each timestep. Error flow control with a CEC is conducted using 'gates'. For example, the input gate controls the information added to the cell, the output gate regulates the flow of information out to the rest of the network, while the forget gate decays the activation of the previous timesteps. Thus, LSTM can maintain temporal information in the state for a long number of timesteps and is widely used in sequential data analysis, prediction and classification tasks [85]–[88] in univariate and multivariate [89], [90] domains.

We describe the standard LSTM use from the Auto Regressive (AR) model perspective in Eq. (7). The LSTM for a single timestep k where i_k, f_k, u_k, o_k, c_k represent input gate, forget gate, update for cell state, output gate, cell state, respectively [85]. The weights $W = \left[W_i, W_g, W_u, W_o\right], U = [U_i, U_g, U_u, U_o]$ and bias is denoted b. Internal LSTM states h_0, c_0 are initialized as zero.

$$x_{t+1} = f(x_t, x_{t-1}, x_{t-2}, \ldots x_{t-N+1}) \tag{7}$$

For $k = t - N + 1$ to t

$$\begin{bmatrix} i_k \\ f_k \\ u_k \\ o_k \end{bmatrix} = Wx_k + Uh_{k-1} + b \tag{8}$$

$$c_k = c_{k-1}\sigma\left(f_k\right) + \tanh\left(u_k\right)\sigma\left(i_k\right) \tag{9}$$

$$h_k = \sigma\left(o_k\right)\tanh(c_k) \tag{10}$$

where σ is sigmoid function. h_k at each timestep becomes the features learned from the history of the input. Finally, the features of the latest timestep obtained from Eq. (8)-(10) are used in a prediction algorithm, typically a feedforward neural network like MLP, in Eq. (11). The weights are learned from BPTT with the loss function such as mean squared error in Eq. (12).

$$\hat{x}_{t+1} = f(h_t) \tag{11}$$

$$L = \frac{1}{N}\sum_{t=1}^{N}\left(x_{t+1} - \hat{x}_{t+1}\right)^2 \tag{12}$$

The input gate and update i_k, u_k control the value of the current cell state update c_k, while the forget gate f_k controls forgetting factor of the previous state c_{k-1}.

There are several common variants proposed to LSTM. From Eq. (8), we see that the current state c_{k-1} has no effect on the gates. The output of the LSTM h_{k-1} has an effect, which is close to zero if the output gate is zero. The main usage is when extracting information from long timesteps are required. The peephole LSTM [94] modifies this to have an effect, such that the output of gates in (8) is dependent on c_{k-1} and another parameter $Wx_k + Uh_{k-1} + Vc_{k-1} + b$. The peephole LSTM was able to learn highly nonlinear spike trains with constant time delays and count delays between sharp spikes.

Gated Recurrent Unit (GRU) is a simplified version of the LSTM that combines forget and input gates to form an 'update' gate [96]. As a result, it has less parameters but reduced complexity than LSTM. The cell state and hidden state is also combined, and a 'reset' gate is used. While there have been several improvements and modifications to the standard (vanilla) LSTM, recent studies [87], [95] show that most variants do not significantly improve the performance for sequential tasks such as speed, music and handwriting recognition. In fact, it was shown that initializing a bias to a large value of 1 or 2 to the forget gate improved LSTM performance for long term dependencies [95].

The GRU authors also proposed an encoder-decoder framework [96], where instead of predicting at each timestep, the prediction is decomposed into two steps. An RNN encoder such as LSTM is used to obtain a hidden state from a variable length sequence $h_t = f(h_{t-1}, x_t)$. The representation at the end of sequence is denoted as c. A decoder RNN would decode the hidden state to predict multiple timesteps $h_t = f(h_{t-1}, y_{t-1}, c)$. Many conventional architectures utilize only the previous and current timestep, while the bidirectional LSTM utilizes the future timesteps, and was used in sequence-to-sequence speech recognition [100]. However, the encoder-decoder framework was shown to deteriorate in performance with longer sequences [99]. The use of attention mechanism in the encoder-decoder framework [96] addressed this, where a weighting of the previous hidden states is used, and allows the network to select relevant hidden states. Recently the attention mechanism was modified for use in time series prediction [97], [98]. Besides, the real-world examples of using LSTM and GRU for time series analysis are also reported in the application of diagnosis of neurodegenerative diseases [101], hydrologic prediction [102], multi-sensors fusion [98], [103], remote sensing [104]–[106], etc.

The LSTM can be extended to multivariate domain [88], [90], [91] by additional dimension on parameters W, U and b. We note the multivariate LSTM assumes equispaced intervals of time, thus many common approaches interpolate data [88], [92] to align and fit the recurrent model. Other LSTM variants have sought to incorporate timestep into the network architectures, such as adding a new time gate in the Phased LSTM [93].

5) Representative Deep Learning Model - Auto-Encoder: As a discriminative method for directly learning a parametric map from input to representation, Auto-Encoder (AE) is to explicitly define a feature-extracting function, i.e., the encoder f_θ, in a specific parameterized closed form [107]. Given each sample x_i from a dataset $x_1, x_2, \ldots, x_N$, the feature vector or code is computed as

$$h_i = f_\theta(x_i) \tag{13}$$

Another parameterized closed function g_θ, called the decoder, maps from feature space back into input space, producing a reconstruction as

$$r_i = g_\theta(h_i) \tag{14}$$

The form of f_θ and g_θ can be simply affine mappings as following

$$f_\theta(x_i) = \sigma_f(\mathbf{W}\mathbf{x} + \mathbf{b}) \tag{15}$$

$$g_\theta(h_i) = \sigma_g(\mathbf{W}^{'}\mathbf{h} + \mathbf{b}^{'}) \tag{16}$$

where $\mathbf{W}$ and $\mathbf{W}^{'}$ are the weight matrices for encoder and decoder, along with $\mathbf{b}$ and $\mathbf{b}^{'}$ as respective bias vectors. σ_f and σ_g denote the activation functions, which can be sigmoid, logistic, etc. [108], [109] Obviously, the parameters can be determined by minimizing reconstruction error $E(x, r)$, which is usually carried out by stochastic gradient descent technique [110].

After been initially deployed for dimensionality reduction by its original form, AE has been developed into many forms, such as sparse AE, regularized AEs involving Contractive AEs (CAEs) and Denoising AE (DAE), etc. [111]–[113] The application for time series prediction ranges from multimodal fusion for sensor data [114], traffic flow [115] to host load in cloud computing [116]. Reference [117] also reported an extreme deep learning approach using stacked AE to predict building energy consumption. Besides, AE is commonly deployed to combine with other deep learning approaches for forecasting time series. We will investigate them in the following sections.

6) Representative Deep Learning Model - Deep Stacking Network: Inspiring by the idea of stacking, a novel discriminative deep learning method emerged recently is the Deep Stacking Network (DSN), where simple functions are composed first and then they are "stacked" on top of each other so as to learn complex functions [118], [119]. The basic architecture of DSN is shown in Fig. 1 involving a number of layered modules, in which each module is a specialized neural network consisting of a single hidden layer and two sets of weights [120]. The figure only gives 4 such modules, while they could be up to hundreds in practice, especially for image and speech

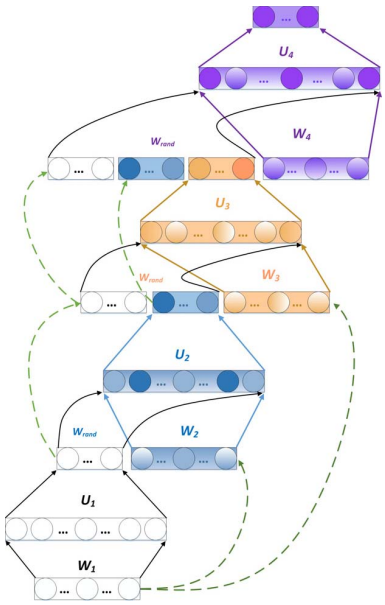


Fig. 1. A basic DSN architecture using input-output stacking. Dash lines denote copying layers.

classification [121], [122]. The weight matrix in lower-layer $\mathbf{W}$ connects the linear input layer and the hidden nonlinear layer, and the one in upper-layer $\mathbf{U}$ connects the hidden nonlinear layer with the linear output layer. Given the fact that $\mathbf{U} = \left(\mathbf{H}\mathbf{H}^T\right)^{-1}\mathbf{H}\mathbf{T}^T = F(\mathbf{W})$ ($\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_i, \ldots, \mathbf{h}_N]$, $\mathbf{h}_i = \sigma(\mathbf{W}^T\mathbf{x}_i)$, $\mathbf{x}_i$ denote the training vector), the weights in DSN can be learned by simple gradient computation with batch training and parallel strategy [123].

The above DSN has already been generalized to tensorized version, i.e., Tensor DSN (TDSN), which provides higher-order feature interactions comparing with conventional DSN [124]. Motivated by increase the size of the hidden units without increasing the number of parameters to learn, Kernel DSN (KDSN) has been proposed using kernel trick [125]. The DSN was also reported to be connected with a Conditional Random Field (CRF), which was successfully developed and applied for Natural Language Processing (NLP) [126].

As for time series prediction, [127] and [128] both presented a typical DSN-based approach, namely Deep-STEP, for spatiotemporal prediction of remote sensing data. Besides, the DSN is commonly combined with auto-encoders for forecasting time-series benchmarks [129] and practical data, such as crude oil price [130]. Reference [131] also presented a novel double deep ELMs ensemble system for time series prediction, which utilized DSN for generalization.

B. Generative

Different from the discriminative models, a generative model considers the joint probability distribution of both observation X and target Y. It can be used to 'generate' random instances regarding a set of observation and target, i.e., (X, Y). We will also review two conventional approaches along with three deep architectures in this section.

1) Representative Classical Model - Gaussian Process: Gaussian Process (GP) [132], as a kernel based probabilistic model, builds probabilistic relationships between the latent

functions of samples [133]–[136]. The advantage of a GP lies in its ability of modeling the uncertainty hidden in data, which is provided by predicting distributions. Considering the non-stationarity of a time series, a covariance matrix in a GP was designed in [137]. As for multiple-step ahead times series forecasting, in a noisy inputs-based GP model, the propagation of the uncertainty in each step prediction was realized by a Gaussian approximation [138]. For cases with missing points in time series, a GP framework based on semi-described and semi-supervised learning was reported in [139], in which the posterior distribution over the missing points was provided by using a variational inference. Moreover, the time series in some fields, such as financial, often exhibits heteroscedastic characteristic, i.e., the volatility or fluctuation of these data is time-varying rather than constant. To forecast such time series data, the GP-based volatility models were reported in recent years [140]–[142].

In a nutshell, the GP based models can give excellent predictive performance with prediction uncertainty. However, as a black box model, they cannot give a clear interpretation of how the time series evolves in a certain application situation. Besides, these methods may suffer from the time-consuming issue owing to the calculation of inverse for covariance matrix, especially with large datasets.

2) Representative Classical Model - Bayesian Networks and Hidden Markov Model: Bayesian Networks (BNs) [143], [144], treated as directed probabilistic graph models, are a class of powerful tools for dealing with the uncertainty issues. There already exist numerous studies on using BNs for time series prediction [145]–[148]. Compared with other generative models (e.g., a GP), BNs can be naturally applied to model the multivariate time series, where the relation between variables as well as the evolution over time will be both effectively captured [149]–[151].

Particularly, Hidden Markov Model (HMM), a special BN which has discrete latent nodes, is often employed to accomplish time series prediction tasks [152]–[154]. These HMM based prediction methods assume that the time series evolves under the control of 'events' or 'patterns' hidden in data, which will then transform over time with a certain probability [155]–[158].

It should be noted that although the BNs based prediction models can capture uncertain relations between the multivariate time series, they may consume more training time to find an effective network structure so as to produce high accuracy. And as for the HMM based models, how to determine the cardinality of the latent nodes is still an open topic deserving further discussion.

3) Representative Deep Learning Model - Restricted Boltzmann Machine: The Restricted Boltzmann Machine (RBM), proposed by Hinton and Sejnowski in 1986 [159], is a generative deep model concerning probabilistic relationship between input, i.e., visible units **v**and latent, i.e., hidden units **h** [160]. The visible and hidden units have their bias **a** and **b**, respectively, and they are connected with a weight matrix **W**. The energy function $E(\mathbf{v},\mathbf{h})$ of an RBM is defined as follows

$$E(\mathbf{v},\mathbf{h}) = -\sum_i a_i v_i - \sum_j b_j h_j - \sum_{ij} W_{ij} v_i h_j \quad (17)$$

where v_i, h_j, a_i, b_j and W_{ij} are the elements of **v**, **h**, **a**, **b** and **W** respectively. Based on this energy function, the joint probability distribution is yielded as

$$P(\mathbf{v},\mathbf{h}) = \frac{1}{Z} e^{-E(\mathbf{v},\mathbf{h})} \quad (18)$$

where Z is the partition function for guaranteeing that the distribution is normalized. Considering the special structure of RBM in which the connections are existed only between layers, the states of each units in hidden layer are conditionally independent given visible units, and vice versa. Therefore, the conditional probability $P(v_i|\mathbf{h})$ and $P(h_j|\mathbf{v})$ are given by

$$\begin{aligned} P(v_i|\mathbf{h}) &= \sigma\left(a_i + \sum_j W_{ij} h_j\right) \\ P(h_j|\mathbf{v}) &= \sigma\left(b_j + \sum_i v_i W_{ij}\right) \end{aligned} \quad (19)$$

where $\sigma(*)$ is the activation function, of which logistic function $\sigma(x) = \frac{1}{1+e^{-x}}$ is a common choice [161]. RBM has several extensions, e.g., conditional RBM [162], gated RBM [163], etc.

Some studies have already reported using RBM to forecast or predict time series data. For instance, [164] proposed a deep network structure consisting of two RBMs to realize prediction. One of the authors has also reviewed extended RBM for time series which often have recurrent structure so that BPTT is employed to learn the parameters [165]. Focusing on multiperiod wind speed prediction, [166] designed a deep Boltzmann machine regarding its competitive capability on approximating nonlinear and nonsmoothed functions. Besides, a dynamic RBM is presented in [167] which considers Gaussian properties of the data. In summary, this kind of deep learning approach is suitable for modeling relationship or discovering features from the perspective of probability.

4) Representative Deep Learning Model - Deep Belief Network: In order to further develop the ability of abstraction and the capacity of information, the RBMs are typically stacked as an integrated structure, to form a Deep Belief Network (DBN). A DBN with l layers models the joint distribution between observed variables v_i and hidden layers $\mathbf{h}^{(k)}, k = 1, 2, \cdots, l$ consisted of binary units $h_j^{(k)}$, which can be described as follows.

$$\begin{aligned} &p\left(\mathbf{v},\mathbf{h}^{(1)},\mathbf{h}^{(2)},\cdots,\mathbf{h}^{(l)}\right) \\ &\quad = P\left(\mathbf{v}|\mathbf{h}^{(1)}\right) P\left(\mathbf{h}^{(1)}|\mathbf{h}^{(2)}\right) \\ &\qquad \times \cdots P\left(\mathbf{h}^{(l-2)}|\mathbf{h}^{(l-1)}\right) P\left(\mathbf{h}^{(l-1)},\mathbf{h}^{(l)}\right) \end{aligned} \quad (20)$$

Assuming $\mathbf{v} = \mathbf{h}^{(0)}$, along with $\mathbf{a}^{(k)}$ the bias vector of layer k and $\mathbf{W}^{(k)}$ the weight matrix between layer k and $k+1$, the factorial conditional distribution in DBN can be formulated as follows.

$$P\left(\mathbf{h}^{(k)}|\mathbf{h}^{(k+1)}\right) = \prod_i P\left(h_i^{(k)}|\mathbf{h}^{(k+1)}\right) \quad (21)$$

where $P\left(h_i^{(k)} = 1|\mathbf{h}^{(k+1)}\right) = sig(a_i^{(k)} + \sum_j W_{ij}^{(k)} h_j^{(k+1)})$. As such, $p\left(\mathbf{h}^{(l-1)},\mathbf{h}^{(l)}\right)$ refers to an RBM [168].

Comparing with RBM, DBN have been more widely deployed as a learning model of predicting temporal data. The predictive objects are not only chaotic time series [169], [170], but also ranging from traffic flow, energy to drought index, etc. [171]–[176] As for the structure of this method, some scholars have constructed ensemble DBNs which aggregate the outputs so as to obtain better accuracy [177], [178]. DBN has also been used in combination with some classical models, such as ARIMA [179]. In the future, optimization techniques are still highly demanded to reduce the computational cost this deep model, especially for ones exhibiting ensemble structure.

5) Representative Deep Learning Model - Generative Adversarial Nets: At twenty-eighth conference on Neural Information Processing Systems (NIPS), Ian J. Goodfellow et al. proposed a new framework for estimating generative models via an adversarial process, i.e., Generative Adversarial Nets (GANs) [180]. Two neural networks are typically involved and simultaneously trained in GANs: a generative G that captures the data distribution, and a discriminative D that estimates the probability that a sample came from the training data rather than G. The objective of the D is to discriminate the training and generated data, whereas the G is trained to confuse the D as much as possible. As for the most straightforward to apply GANs, in which the G and D are both multilayer perceptrons, the entire system can be simply trained with backpropagation technique. Given input data $\mathbf{x}$ and the noise $\mathbf{z}$, the objective of training GANs can be described as a two-player minimax game with value function $V(G, D)$ between D and G:

$$V(G, D) = \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}\left[\log D(\mathbf{x})\right] + \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}(\mathbf{z})}\left[\log\left(1 - D(G(\mathbf{z}))\right)\right] \tag{22}$$

where $p_{data}(\mathbf{x})$ denotes the generator's distribution over data $\mathbf{x}$, and $p_{\mathbf{z}}(\mathbf{z})$ the prior distribution over noise. In practice, the training process is implemented by some iterative, numerical approach. One of the most commonly used technique is minibatch stochastic gradient descent, which updates the discriminators by ascending its stochastic gradient as well as the generator by descending its stochastic gradient:

$$\nabla_{\theta_d}\frac{1}{m}\sum_{i=1}^{m}\left[\log D(\mathbf{x}_i) + \log\left(1 - D(G(\mathbf{z}_i))\right)\right] \tag{23}$$

$$\nabla_{\theta_g}\frac{1}{m}\sum_{i=1}^{m}\log\left(1 - D(G(\mathbf{z}_i))\right) \tag{24}$$

where θ_d and θ_g denote the parameters for the D and G. $\mathbf{x}_i$ and $\mathbf{z}_i$ are the samples from data generating distribution $p_{data}(\mathbf{x})$ and noise prior $p_g(\mathbf{z})$ [181].

The variants of GANs include varying objective of the G, D and also the overall architecture, yield conditional GANs [182], InfoGAN [183], SeqGAN [184], Wasserstein GANs [185], least squares GANs [186], etc. The reported application cases are mostly for image processing [187]. Still, a number of studies applying GANs for time-series prediction has been also emerged recently. For instance, [188] presented a study of stock market prediction on high-frequency using GAN. And a GAN-based parallel prediction model is proposed in [189] to forecast building energy consumption. Reference [190] reported a LSTM-based GAN for predicting traffic flow. Besides, a deep generative adversarial architecture for network-wide spatial-temporal traffic-state estimation is proposed in [191]. Considering the capability of generating missing data points [192], the prediction for incomplete time series is a possible future topic deserving further study.

C. Others

1) Representative Classical Model - Clustering-Based Model: As the most commonly deployed semi-supervised technique, the clustering-based methods are reported in the literatures for predicting and forecasting time series data [193], [194]. The clusters, as a typical representative of the information granules, are used to construct a time series prediction model by the aid of techniques such as Fuzzy C-Means (FCM). Generally, this framework first clusters the data so as to produce the prototypes, which are then employed to extract fuzzy rules and finally obtains long-term prediction results [27], [28], [195]–[197]. These methods modeled the information granules rather than single point so that the accumulated errors caused by the iterative prediction are avoided. However, these granules are still intuitive which needs to be constructed in a deeper level.

Another typical clustering-based model is k-Nearest Neighbor (k-NN). This simple but powerful technique has been applied in many different fields, particularly for classification and prediction [198]–[201]. The points having shortest distance with the data in training set are selected as the nearest neighbors [202]–[204]. While k-NN has its shortcomings, such as high computational cost and space complexity, which limits its scope of application.

2) Representative Deep Learning Model - Hybrids: Besides the above-mentioned studies reporting individual deep learning models, there also exist some hybrids or combinational structures which are successfully applied on time series prediction [205], [206]. For instance, a deep learning approach using Self Organizing Maps (SOMs) and MLP were proposed in [207] for multi-sensor data prediction. Reference [115] reported a deep architecture model for traffic flow prediction, in which a stacked autoencoder is used to extract traffic flow features, and a logistic regression layer is applied for prediction. Combining CNN and LSTM, [208] proposed a hybrid deep learning framework, also to forecast future traffic flow. Reference [209] discussed prediction models using autoencoder and LSTM with various activation functions for solar power forecasting. An integrated approach is proposed in [210] which combines discriminatively trained predictive models with deep neural networks accounting for the joint statistics of a set of weather-related variables.

Such hybrid approaches take the advantage of both discriminative and generative models for applying deep learning on time series prediction. Still, resolving how to combine various models in a reasonable and effective way requires special consideration towards different application fields.

In order to give a brief summarization, we also organize a table as the Appendix to describe the advantage or disadvantage or both of each model.

IV. Experiments and Analysis

This section selects six methods, i.e., RNN with BPTT, CNN, LSTM, GANs, DBN and sparse AEs with LSTM as a hybrid, as the representative for demonstrating the performance of deep learning models on time series prediction. The data we use are two benchmarks including Mackey-Glass and Lorenz chaotic time series, along with Blast Furnace Gas (BFG) generation and Coke Oven Gas (COG) generation as the typical gaseous energy from steel industry. Mean Absolute Percentage Error (MAPE) [211] and Root Mean Square Error (RMSE) [212] are deployed as indices for error statistics, which are defined as follows.

$$\text{MAPE} = \frac{100}{N}\sum_{i=1}^{N}\frac{\left|y_i - \hat{y}_i\right|}{y_i} \tag{25}$$

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{26}$$

where N denotes the length of predicted results, i.e., $\hat{y}_i$. y_i refers to the real value. In this study, $N = 60$. Parameters include the number of units in input layer n_{input}, hidden layers $n_{hidden}^{(1)}, n_{hidden}^{(2)}, \cdots, n_{hidden}^{(k)}$ and output layer n_{output} for each model. As for CNN, this study deploys max pooling. The parameters further include the size of the convolutional kernel, the number of neurons n_{fc} in fully connected layer between the convolutional layer and the predictor. The predictor for CNN in this study deploys a simple backpropagation neural network. The step size for the convolutional nets is set to 1, and the size of the convolutional kernel as 1×5. The hybrid also includes sparsity ρ and the number of divided layers $n_{divided}$. All the parameters are determined by Differential Evolution (DE) algorithm [213]. The results are calculated on testing data.

A. Benchmarks

1) Mackey-Glass: The Mackey-Glass time series prediction is a typical nonlinear fitting problem, which has greatly attracted attentions. The series is generated by the following time delay differential equation:

$$\frac{dx(t)}{dt} = \frac{\alpha x(t-\tau)}{1 + x^{\beta}(t-\tau)} + \gamma\, x(t) \tag{27}$$

where $x(t)$ refers to the generated data. τ denotes the time delay parameter. α, β and γ are real numbers [214].

Predicted results of different models for Mackey-Glass time series are given in Fig.2. It can be depicted that almost all six models successfully forecasted the tendency of this data. In particular, LSTM performs best on accuracy, which can be also concluded from error statistics shown in Table I.

2) Lorenz: The Lorenz system can be described as the following set of equations

$$\begin{aligned}\frac{dx(t)}{dt} &= \sigma(x(t) - y(t))\\ \frac{dy(t)}{dt} &= rx(t) - x(t)z(t)\\ \frac{dz(t)}{dt} &= z(t)\,y(t) - bz(t)\end{aligned} \tag{28}$$

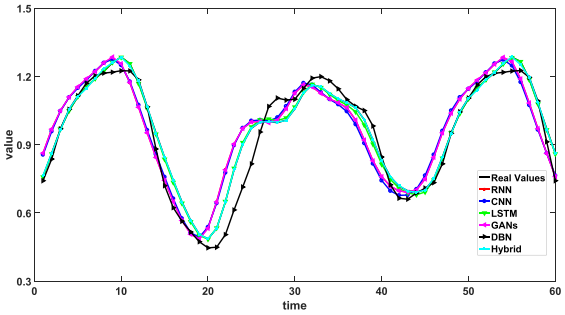


Fig. 2. Predicted results of different deep learning models for Mackey-Glass time series.

TABLE I
Error Statistics and Parameters of Different Deep Learning Models for Predicting Mackey-Glass Time Series

Method	MAPE	RMSE	Number of parameters	Parameters
RNN	4.65	0.0701	3	$n_{input} = 30, n_{hidden} = 10, n_{output} = 1$
CNN	4.81	0.0710	7	The number of convolutional layers: 2, the sizes of the kernels are both 1×5, $n_{fc} = 200$ Predictor: $n_{input} = 30, n_{output} = 1$, $n_{hidden}^{(1)} = n_{hidden}^{(2)} = 10$
LSTM	0.41	0.0728	4	$n_{input} = 50, n_{output} = 1$, $n_{hidden}^{(1)} = 50, n_{hidden}^{(2)} = 100$
GANs	4.67	0.0703	8	G: $n_{input} = 80, n_{output} = 1$, $n_{hidden}^{(1)} = n_{hidden}^{(2)} = 30$ D: $n_{input} = 100, n_{output} = 1$, $n_{hidden}^{(1)} = n_{hidden}^{(2)} = 50$
DBN	3.57	0.0643	4	$n_{input} = 5, n_{output} = 1$, $n_{hidden}^{(1)} = 30, n_{hidden}^{(2)} = 20$
Hybrid	0.21	0.0040	6	$n_{input} = 100, n_{output} = 1$, $n_{hidden}^{(1)} = n_{hidden}^{(2)} = 130$, $n_{divided} = 3, \rho = 0.38$

where σ, r and b refer to chaotical parameters. The series was generated from Eq. (28) by the Runge-Kutta algorithm [215]. Comparing with Mackey-Glass, the Lorenz data exhibits no obvious periodic variations so prediction on this data is somewhat more difficult. As a result, the numbers of hidden layers of the deep learning models are all higher than the ones for Mackey-Glass time series. Besides the RNN and CNN, the other four models still output satisfactory results, which demonstrate the applicability on time series prediction, as shown in Fig.3. Table II gives error statistics along with determined parameters.

B. Real-World Data

For the steel industry, the generating amount of gaseous energy plays a crucial role for supporting production and saving energy. The prediction on this time series data enables the operating staff to acknowledge the trend of gas flow in

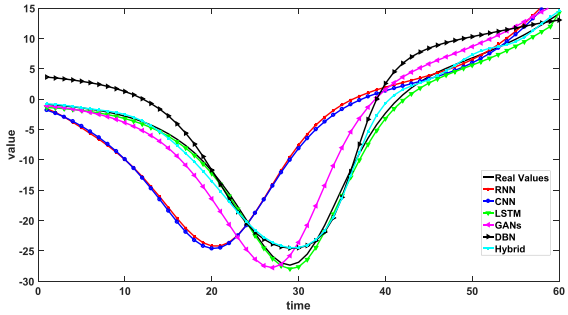


Fig. 3. Predicted results of different deep learning models for Lorenz time series.

TABLE II
ERROR STATISTICS AND PARAMETERS OF DIFFERENT DEEP LEARNING MODELS FOR PREDICTING LORENZ TIME SERIES

Method	MAPE	RMSE	Number of parameters	Parameters
RNN	24.67	8.9747	3	$n_{input} = 30, n_{hidden} = 20, n_{output} = 1$
CNN	24.26	8.8520	5	The number of convolutional layers: 3, the sizes of the kernels are 1×5, 1×3 and 1×3 $n_{fc} = 300$
LSTM	2.73	0.8214	4	$n_{input} = 50, n_{output} = 1,$ $n_{hidden}^{(1)} = 50, n_{hidden}^{(2)} = 100$
GANs	10.16	3.3104	10	G: $n_{input} = 100, n_{output} = 1,$ $n_{hidden}^{(1)} = n_{hidden}^{(2)} = n_{hidden}^{(3)} = 50$ D: $n_{input} = 120, n_{output} = 1,$ $n_{hidden}^{(1)} = n_{hidden}^{(2)} = n_{hidden}^{(3)} = 50$
DBN	9.94	3.1565	4	$n_{input} = 10, n_{output} = 1,$ $n_{hidden}^{(1)} = 30, n_{hidden}^{(2)} = 20$
Hybrid	2.80	1.0527	8	$n_{input} = 110, n_{output} = 1,$ $n_{hidden}^{(1)} = n_{hidden}^{(2)} = n_{hidden}^{(3)} = n_{hidden}^{(4)} = 150,$ $n_{divided} = 4, \rho = 0.5$

advance, which is then beneficial for scheduling and optimizing the utilization of gaseous energy. As a result, here we employed BFG generation and COG generation as the representative data application.

1) BFG Generation: By implementing LSTM and DBM on BFG generation data, we obtain the predicted results as shown in Fig.4. The RNN, CNN and GANs perform well for the first 40 points, then behave not as well as for predicting Mackey-Glass and Lorenz chaotic time series. While the three deep models give excellent results on this industrial data in all these 60 points. And as shown in Table III, the different models exhibit little difference on the accuracy involving MAPE and RMSE.

2) COG Generation: As shown in Fig.5, compared to time series of BFG generation, LSTM and hybrid behave substantially different with the other models for predicting COG generation amount. LSTM and hybrid that still perform well, whereas RNN, CNN GANs and DBN failed to accurately estimate the variation of this data, which can be also concluded from error statistics in Table IV.

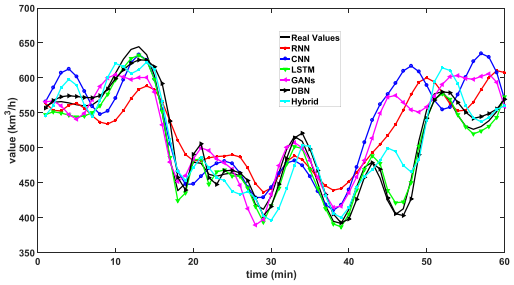


Fig. 4. Predicted results of different deep learning models for BFG generation.

TABLE III
ERROR STATISTICS AND PARAMETERS OF DIFFERENT DEEP LEARNING MODELS FOR PREDICTING BFG GENERATION

Method	MAPE	RMSE	Number of parameters	Parameters
RNN	5.55	47.6039	3	$n_{input} = 60, n_{hidden} = 30, n_{output} = 1$
CNN	5.96	59.2039	4	The number of convolutional layers: 3, the sizes of the kernels are 1×5, 1×5 and 1×3 $n_{fc} = 200$
LSTM	1.42	10.7508	4	$n_{input} = 50, n_{output} = 1,$ $n_{hidden}^{(1)} = 50, n_{hidden}^{(2)} = 100$
GANs	4.79	47.7879	10	G: $n_{input} = 120, n_{output} = 1,$ $n_{hidden}^{(1)} = n_{hidden}^{(2)} = n_{hidden}^{(3)} = 50$ D: $n_{input} = 100, n_{output} = 1,$ $n_{hidden}^{(1)} = n_{hidden}^{(2)} = n_{hidden}^{(3)} = 50$
DBN	1.87	15.1920	4	$n_{input} = 3, n_{output} = 1,$ $n_{hidden}^{(1)} = 30, n_{hidden}^{(2)} = 10$
Hybrid	3.22	27.6538	8	$n_{input} = 130, n_{output} = 1,$ $n_{hidden}^{(1)} = n_{hidden}^{(2)} = n_{hidden}^{(3)} = n_{hidden}^{(4)} = 145,$ $n_{divided} = 5, \rho = 0.6$

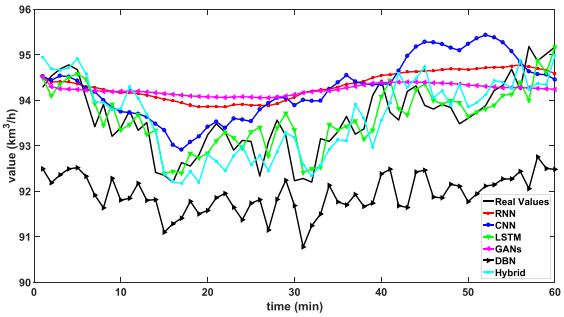


Fig. 5. Predicted results of different deep learning models for COG generation.

In summary, these two representative models perform well for selected time series including both benchmarks and real-world data. Analysis shows LSTM seems to be more stable than DBN in the implemented experiments.

TABLE IV
ERROR STATISTICS AND PARAMETERS OF DIFFERENT DEEP LEARNING MODELS FOR PREDICTING COG GENERATION

Method	MAPE	RMSE	Number of parameters	Parameters
RNN	0.84	0.9301	3	$n_{input} = 40, n_{hidden} = 20, n_{output} = 1$
CNN	0.79	0.9091	3	The number of convolutional layers: 2, the sizes of the kernels are both 1×5, $n_{fc} = 200$
LSTM	0.35	0.4286	4	$n_{input} = 50, n_{output} = 1, n_{hidden}^{(1)} = 50, n_{hidden}^{(2)} = 100$
GANs	0.84	0.9506	10	G: $n_{input} = 80, n_{output} = 1, n_{hidden}^{(1)} = n_{hidden}^{(2)} = n_{hidden}^{(3)} = 50$ D: $n_{input} = 100, n_{output} = 1, n_{hidden}^{(1)} = n_{hidden}^{(2)} = n_{hidden}^{(3)} = 60$
DBN	1.76	1.7634	4	$n_{input} = 5, n_{output} = 1, n_{hidden}^{(1)} = 30, n_{hidden}^{(2)} = 10$
Hybrid	0.45	0.5204	7	$n_{input} = 95, n_{output} = 1, n_{hidden}^{(1)} = n_{hidden}^{(2)} = n_{hidden}^{(3)} = 120, n_{divided} = 3, \rho = 0.55$

Considering the time for modeling deep features with multiple hidden layers, the training time for the deep learning models is comparatively longer than the one for conventional machine learning models. While the time for prediction consumes far less, which exhibits little difference with the traditional models. Bearing this in mind, the real-world application always run these two procedures in separate, i.e., train the model offline, possibly with parallel strategy, and predict the data online, so that the practical requirements on real-time performance can be satisfied.

In summary, these representative models perform well for selected time series including both benchmarks and real-world data. Analysis shows LSTM and hybrid seems to be more stable than other ones in the implemented experiments.

V. FUTURE PERSPECTIVES

Deep learning-based models are good at discovering intricate structure in large data sets. In addition, they have been shown to learn to discriminate patterns from multiple time series information. This superiority in the feature learning process also results in some mystery in interpretation with regards to the output. As for time series prediction, providing explainable result is beneficial for measuring its reliability. Therefore, enhancing interpretability becomes a future topic attracting intensive attention [216], [217]. For instance, IJCAI specially organized a workshop discussing Explainable Artificial Intelligence (XAI) from 2017 [218], [219]. As well, the best paper of ICML 2017 was awarded to a study on understanding black-box predictions [220].

TABLE V
BRIEF SUMMARY OF THE REVIEWED MODELS

Category	Family	Model	Brief Description
Discriminative	Classical	SVM	A supervised kernel-based method. Not only independently applied, but also combined with other techniques.
		Shallow NN	Typical methods may include MLP, RBF, RNN, etc. Unable to extract and utilize the deep features.
	Deep Learning	CNN	Some literatures have been reported for predicting chaotic and real-world time series. Still requires further studies to demonstrate its superiority on learning deep features.
		LSTM	Overcomes the vanishing gradient problem by introducing a linear unit (cell) called a Constant Error Carousel (CEC) that information can be added for each timestep. Gated recurrent unit (GRU) is a simplified version of the LSTM that combines forget and input gates to form an 'update' gate.
		AE	To explicitly define a feature-extracting function, i.e., the encoder f_θ, in a specific parameterized closed form. Being commonly deployed to combine with other deep learning approaches for forecasting time series.
		DSN	Inspiring by the idea of stacking, where simple functions are composed first and then they are "stacked" on top of each other so as to learn complex functions.
Generative	Classical	GP	A kernel based probabilistic model. Unable to give a clear interpretation of how the time series evolves in a certain situation. May be time-consuming issue owing to the calculation of inverse covariance matrix.
		BN & HMM	A class of powerful tools for dealing with the uncertainty issues. May consume more training time to find an effective network structure so as to produce high accuracy.
	Deep Learning	RBM	Being suitable for modeling relationship or discovering features from the perspective of probability.
		DBN	An integrated structure stacked by the RBMs. Optimization techniques are still highly demanded to reduce the computational cost.
		GANs	Using a generative G that captures the data distribution, and a discriminative D that estimates the probability that a sample came from the training data rather than G. Being capable of generating missing data points
Others	Classical	Clustering-based	Unsupervised or semi-supervised learning, e.g., k-NN, FCM, etc.
	Deep Learning	Hybrids	Combining deep learning models so as to achieve better performance, e.g., AEs with LSTM, CNN with LSTM, etc.

Acceleration for the learning process is another area in which deep learning needs to be improved over the next few years. In order to determine the weights of multiple layers in the network, deep learning-based approaches usually requires hours or even days for training, even using the latest GPU processors, which is unacceptable for time series prediction in some real-world application cases. For improving the computational efficiency, some studies have been conducted to propose algorithms and strategies for accelerating deep learning [221]–[223].

Besides combining with classic models, deep learning-based methods are also typically merged with meta-learning and reinforcement learning so as to utilize the ability of interaction with environment of these two methods [224]–[227]. There already exist some literatures which successfully make predictions by deep meta-learning or deep reinforcement learning [228], [229]. We believe that more studies will emerge in the near future as improvements on accuracy, efficiency and architectures are proposed for time series analysis.

APPENDIX

See Table V.

REFERENCES

[1] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time Series Analysis: Forecasting and Control*. Hoboken, NJ, USA: Wiley, 2015.

[2] P. J. Brockwell, R. A. Davis, and M. V. Calder, *Introduction to Time Series and Forecasting*. New York, NY, USA: Springer, 2002.

[3] C. Chatfield, *Time-Series Forecasting*. London, U.K.: Chapman & Hall, 2000.

[4] C. Liu, S. C. H. Hoi, and P. Zhao, "Online ARIMA algorithms for time series prediction," in *Proc. 13th AAAI Conf. Artif. Intell.*, 2016, pp. 1–7.

[5] A. A. Adebiyi, A. O. Adewumi, and C. K. Ayo, "Comparison of ARIMA and artificial neural networks models for stock price prediction," *J. Appl. Math.*, vol. 2014, Mar. 2014, Art. no. 614342.

[6] X. Wu and Y. Wang, "Extended and unscented Kalman filtering based feedforward neural networks for time series prediction," *Appl. Math. Model.*, vol. 36, no. 3, pp. 1123–1131, 2012.

[7] T. W. Joo and S. B. Kim, "Time series forecasting based on wavelet filtering," *Expert Syst. Appl.*, vol. 42, no. 8, pp. 3868–3874, 2015.

[8] Z. Han, Y. Liu, J. Zhao, and W. Wang, "Real time prediction for converter gas tank levels based on multi-output least square support vector regressor," *Control Eng. Pract.*, vol. 20, no. 12, pp. 1400–1409, Dec. 2012.

[9] Y. Bengio and Y. LeCun, "Scaling learning algorithms towards AI," *Large-Scale Kernel Mach.*, vol. 34, no. 5, pp. 1–41, 2007.

[10] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

[11] L. Zhang, L. Zhang, and B. Du, "Deep learning for remote sensing data: A technical tutorial on the state of the Art," *IEEE Geosci. Remote Sens. Mag.*, vol. 4, no. 2, pp. 22–40, Jun. 2016.

[12] X. X. Zhu *et al.*, "Deep learning in remote sensing: A comprehensive review and list of resources," *IEEE Geosci. Remote Sens. Mag.*, vol. 5, no. 4, pp. 8–36, Dec. 2017.

[13] J. C. B. Gamboa, "Deep learning for time-series analysis," 2017, *arXiv:1701.01887*. [Online]. Available: https://arxiv.org/abs/1701.01889

[14] T. V. Gestel *et al.*, "Financial time series prediction using least squares support vector machines within the evidence framework," *IEEE Trans. Neural Netw.*, vol. 12, no. 4, pp. 809–821, Jul. 2001.

[15] B. Doucoure, K. Agbossou, and A. Cardenas, "Time series prediction using artificial wavelet neural network and multi-resolution analysis: Application to wind speed data," *Renew. Energy*, vol. 92, pp. 202–211, Jul. 2016.

[16] X. An, D. Jiang, M. Zhao, and C. Liu, "Short-term prediction of wind power using EMD and chaotic theory," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 17, no. 2, pp. 1036–1042, 2012.

[17] L. Zhang, W.-D. Zhou, P.-C. Chang, J.-W. Yang, and F.-Z. Li, "Iterated time series prediction with multiple support vector regression models," *Neurocomputing*, vol. 99, pp. 411–422, Jan. 2013.

[18] A. Miranian and M. Abdollahzade, "Developing a local least-squares support vector machines-based neuro-fuzzy model for nonlinear and chaotic time series prediction," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 2, pp. 207–218, Feb. 2013.

[19] C.-H. Lee, F.-Y. Chang, and C.-M. Lin, "An efficient interval type-2 fuzzy CMAC for chaos time-series prediction and synchronization," *IEEE Trans. Cybern.*, vol. 44, no. 3, pp. 329–341, Mar. 2014.

[20] S. B. Taieb, G. Bontempi, A. F. Atiya, and A. Sorjamaa, "A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition," *Expert Syst. Appl.*, vol. 39, no. 8, pp. 7067–7083, Jun. 2012.

[21] A. G. Parlos, O. T. Rais, and A. F. Atiya, "Multi-step-ahead prediction using dynamic recurrent neural networks," *Neural Netw.*, vol. 13, no. 7, pp. 765–786, Sep. 2000.

[22] E. Ramasso, M. Rombaut, and N. Zerhouni, "Joint prediction of continuous and discrete states in time-series based on belief functions," *IEEE Trans. Cybern.*, vol. 43, no. 1, pp. 37–50, Feb. 2013.

[23] C. H. Aladag, U. Yolcu, E. Egrioglu, and A. Z. Dalar, "A new time invariant fuzzy time series forecasting method based on particle swarm optimization," *Appl. Soft Comput.*, vol. 12, no. 10, pp. 3291–3299, 2012.

[24] P. C. Young, *Recursive Estimation and Time-Series Analysis: An Introduction*. Berlin, Germany: Springer, 2012.

[25] D. T. Mirikitani and N. Nikolaev, "Recursive Bayesian recurrent neural networks for time-series modeling," *IEEE Trans. Neural Netw.*, vol. 21, no. 2, pp. 262–274, Feb. 2010.

[26] H. Liu, H.-Q. Tian, and Y.-F. Li, "Comparison of two new ARIMA-ANN and ARIMA-Kalman hybrid methods for wind speed prediction," *Appl. Energy*, vol. 98, pp. 415–424, Oct. 2012.

[27] J. Zhao, Z. Y. Han, and W. Pedrycz, "Granular model of long-term prediction for energy system in steel industry," *IEEE Trans. Cybern.*, vol. 46, no. 2, pp. 388–400, Jul. 2016.

[28] Z. Y. Han, J. Zhao, Q. Liu, and W. Wang, "Granular-computing based hybrid collaborative fuzzy clustering for long-term prediction of multiple gas holders levels," *Inf. Sci.*, vol. 330, pp. 175–185, Feb. 2016.

[29] Z. Han, J. Zhao, W. Wang, and Y. Liu, "A two-stage method for predicting and scheduling energy in an oxygen/nitrogen system of the steel industry," *Control. Eng. Pract.*, vol. 52, pp. 35–45, Jul. 2016.

[30] A. Pankratz, *Forecasting With Univariate Box-Jenkins Models: Concepts and Cases*. Hoboken, NJ, USA: Wiley, 2009.

[31] M. Khashei and M. Bijari, "A novel hybridization of artificial neural networks and ARIMA models for time series forecasting," *Appl. Soft Comput.*, vol. 11, no. 2, pp. 2664–2675, Mar. 2011.

[32] E. Cadenas, W. Rivera, R. Campos-Amezcua, and C. Heard, "Wind speed prediction using a univariate ARIMA model and a multivariate NARX model," *Energies*, vol. 9, no. 2, p. 109, 2016.

[33] C. Yuan, S. Liu, and Z. Fang, "Comparison of China's primary energy consumption forecasting by using ARIMA (the autoregressive integrated moving average) model and GM (1,1) model," *Energy*, vol. 100, pp. 384–390, Apr. 2016.

[34] S. Johansen, "Estimation and hypothesis testing of cointegration vectors in Gaussian vector autoregressive models," *Econometrica*, vol. 59, no. 6, pp. 1551–1580, 1991.

[35] A. Y. Ng and M. I. Jordan, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes," in *Proc. Adv. Neural Inf. Process. Syst.*, 2002, pp. 841–848.

[36] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer, 1995.

[37] F. E. Tay and L. Cao, "Application of support vector machines in financial time series forecasting," *Omega*, vol. 29, no. 4, pp. 309–317, Aug. 2001.

[38] M. das Chagas Moura, E. Zio, I. D. Lins, and E. Droguett, "Failure and reliability prediction by support vector machines regression of time series data," *Rel. Eng. Syst. Saf.*, vol. 96, no. 11, pp. 1527–1534, 2011.

[39] A. Mellit, A. M. Pavan, and M. Benghanem, "Least squares support vector machine for short-term prediction of meteorological time series," *Theor. Appl. Climatol.*, vol. 111, nos. 1–2, pp. 297–307, 2013.

[40] N. I. Sapankevych and R. Sankar, "Time series prediction using support vector machines: A survey," *IEEE Comput. Intell. Mag.*, vol. 4, no. 2, pp. 24–38, May 2009.

[41] T. Quan, X. Liu, and Q. Liu, "Weighted least squares support vector machine local region method for nonlinear time series prediction," *Appl. Soft Comput.*, vol. 10, no. 2, pp. 562–566, Mar. 2010.

[42] J. C. Platt, "12 fast training of support vector machines using sequential minimal optimization," in *Proc. Adv. Kernel Methods*, 1999, pp. 185–208.

[43] J. A. K. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Process. Lett.*, vol. 9, no. 3, pp. 293–300, Jun. 1999.

[44] X. Wang, J. Wu, C. Liu, S. Wang, and W. Niu, "A hybrid model based on singular Spectrum analysis and support vector machines regression for failure time series prediction," *Qual. Rel. Eng. Int.*, vol. 32, no. 8, pp. 2717–2738, 2016.

[45] Y. Zhou, F. J. Chang, and L. C. Chang, "Multi-output support vector machine for regional multi-step-ahead $PM_{2.5}$ forecasting," *Sci. Total Environ.*, vol. 651, pp. 230–240, Feb. 2019.

[46] A. Mellit and A. M. Pavan, "A 24-h forecast of solar irradiance using artificial neural network: Application for performance prediction of a grid-connected PV plant at Trieste, Italy," *Sol. Energy*, vol. 84, no. 5, pp. 807–821, 2010.

[47] G. Lachtermacher and J. D. Fuller, "Back propagation in time-series forecasting," *J. Forecast.*, vol. 14, no. 4, pp. 381–393, 1995.

[48] L. Wang, Y. Zeng, and T. Chen, "Back propagation neural network with adaptive differential evolution algorithm for time series forecasting," *Expert Syst. Appl.*, vol. 42, no. 2, pp. 855–863, 2015.

[49] D. S. K. Karunasinghe and S. Y. Liong, "Chaotic time series prediction with a global model: Artificial neural network," *J. Hydrol.*, vol. 323, nos. 1–4, pp. 92–105, 2006.

[50] N. K. Kasabov and Q. Song, "DENFIS: Dynamic evolving neural-fuzzy inference system and its application for time-series prediction," *IEEE Trans. Fuzzy Syst.*, vol. 10, no. 2, pp. 144–154, Feb. 2002.

[51] F. S. Wong, "Time series forecasting using backpropagation neural networks," *Neurocomputing*, vol. 2, no. 4, pp. 147–159, 1991.

[52] K. Chakraborty, K. Mehrotra, C. K. Mohan, and S. Ranka, "Forecasting the behavior of multivariate time series using neural networks," *Neural Netw.*, vol. 5, no. 6, pp. 961–970, 1992.

[53] E. S. Chng, S. Chen, and B. Mulgrew, "Gradient radial basis function networks for nonlinear and nonstationary time series prediction," *IEEE Trans. Neural Netw.*, vol. 7, no. 1, pp. 190–194, Jan. 1996.

[54] Z. Ramedani, M. Omid, A. Keyhani, S. Shamshirband, and B. Khoshnevisan, "Potential of radial basis function based support vector regression for global solar radiation prediction," *Renew. Sustain. Energy Rev.*, vol. 39, pp. 1005–1011, Nov. 2014.

[55] H. Leung, T. Lo, and S. Wang, "Prediction of noisy chaotic time series using an optimal radial basis function neural network," *IEEE Trans. Neural Netw.*, vol. 12, no. 5, pp. 1163–1172, Sep. 2001.

[56] G. Sideratos and N. D. Hatziargyriou, "Probabilistic wind power forecasting using radial basis function neural networks," *IEEE Trans. Power Syst.*, vol. 27, no. 4, pp. 1788–1796, Nov. 2012.

[57] C. Harpham and C. W. Dawson, "The effect of different basis functions on a radial basis function network for time series prediction: A comparative study," *Neurocomputing*, vol. 69, nos. 16–18, pp. 2161–2170, 2006.

[58] R. Chandra, "Competition and collaboration in cooperative coevolution of Elman recurrent neural networks for time-series prediction," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 12, pp. 3123–3136, Dec. 2015.

[59] R. Chandra and M. Zhang, "Cooperative coevolution of Elman recurrent neural networks for chaotic time series prediction," *Neurocomputing*, vol. 86, pp. 116–123, Jun. 2012.

[60] S. Anbazhagan and N. Kumarappan, "Day-ahead deregulated electricity market price forecasting using recurrent neural network," *IEEE Syst. J.*, vol. 7, no. 4, pp. 866–872, Dec. 2013.

[61] E. Egrioglu, U. Yolcu, C. H. Aladag, and E. Bas, "Recurrent multiplicative neuron model artificial neural network for non-linear time series forecasting," *Neural Process. Lett.*, vol. 41, no. 2, pp. 249–258, 2015.

[62] E. Guresen, G. Kayakutlu, and T. U. Daim, "Using artificial neural network models in stock market index prediction," *Expert Syst. Appl.*, vol. 38, no. 8, pp. 10389–10397, 2011.

[63] S.-X. Lun, X.-S. Yao, H.-Y. Qi, and H.-F. Hu, "A novel model of leaky integrator echo state network for time-series prediction," *Neurocomputing*, vol. 159, pp. 58–66, Jul. 2015.

[64] L. Wang, Z. Wang, and S. Liu, "An effective multivariate time series classification approach using echo state network and adaptive differential evolution algorithm," *Expert Syst. Appl.*, vol. 43, pp. 237–249, Jan. 2016.

[65] D. Li, M. Han, and J. Wang, "Chaotic time series prediction based on a novel robust echo state network," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 5, pp. 787–799, May 2012.

[66] N. Chouikhi, B. Ammar, N. Rokbani, and A. M. Alimi, "PSO-based analysis of echo state network parameters for time series forecasting," *Appl. Soft Comput.*, vol. 55, pp. 211–225, Jun. 2017.

[67] T. N. Wiesel and D. H. Hubel, "Receptive fields of single neurones in the cat's striate cortex," *J. Physiol.*, vol. 148, no. 3, pp. 574–591, 1959.

[68] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," in *The Handbook of Brain Theory and Neural Networks*, vol. 3361, no. 10. Cambridge, MA, USA: MIT Press, 1995.

[69] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1–9.

[70] H. Lee, R. Grosse, R. Ranganath, and A. N. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, 2009, pp. 609–616.

[71] S. Hershey *et al.*, "CNN architectures for large-scale audio classification," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2017, pp. 131–135.

[72] A. Oord, S. Dieleman, and H. Zen, "WaveNet: A generative model for raw audio," 2016, *arXiv:1609.03499*. [Online]. Available: https://arxiv.org/abs/1609.03499

[73] Y. Zheng, Q. Liu, E. Chen, Y. Ge, and J. L. Zhao, "Time series classification using multi-channels deep convolutional neural networks," in *Proc. Int. Conf. Web-Age Inf. Manage.* Cham, Switzerland: Springer, 2014, pp. 298–310.

[74] A. Borovykh, S. Bohte, and C. W. Oosterlee, "Conditional time series forecasting with convolutional neural networks," 2017, *arXiv:1703.04691*. [Online]. Available: https://arxiv.org/abs/1703.04691

[75] J. Yang, M. N. Nguyen, P. P. San, X. Li, and S. Krishnaswamy, "Deep convolutional neural networks on multichannel time series for human activity recognition," in *Proc. IJCAI*, vol. 15, 2015, pp. 3995–4001.

[76] S. Hoermann, M. Bach, and K. Dietmayer, "Dynamic occupancy grid prediction for urban autonomous driving: A deep learning approach with fully automatic labeling," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 2056–2063.

[77] X. Ding, Y. Zhang, T. Liu, and J. Duan, "Deep learning for event-driven stock prediction," in *Proc. IJCAI*, 2015, pp. 2327–2333.

[78] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," in *Proc. AAAI*, 2017, pp. 1655–1661.

[79] H. Z. Wang, G.-Q. Li, G.-B. Wang, J.-C. Peng, H. Jiang, and Y.-T. Liu, "Deep learning based ensemble approach for probabilistic wind power forecasting," *Appl. Energy*, vol. 188, pp. 56–70, Feb. 2017.

[80] J. You, X. Li, M. Low, D. Lobell, and S. Ermon, "Deep Gaussian process for crop yield prediction based on remote sensing data," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 1–7.

[81] I. Wallach, M. Dzamba, and A. Heifets, "AtomNet: A deep convolutional neural network for bioactivity prediction in structure-based drug discovery," 2015, *arXiv:1510.02855*. [Online]. Available: https://arxiv.org/abs/1510.02855

[82] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, and Y. Wang, "Learning traffic as images: A deep convolutional neural network for large-scale transportation network speed prediction," *Sensors*, vol. 17, no. 4, p. 818, 2017.

[83] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1097–1105.

[84] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," Istituto Dalle Molle di Studi sull'Intelligenza Artificiale, Manno, Switzerland, Tech. Rep. IDSIA-01-99, 1999.

[85] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[86] A. Graves, N. Jaitly, and A.-R. Mohamed, "Hybrid speech recognition with deep bidirectional LSTM," in *Proc. IEEE Workshop Autom. Speech Recognit. Understand. (ASRU)*, Dec. 2013, pp. 273–278.

[87] K. Greff, R. K. Srivastava, J. Koutnìk, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2222–2232, Oct. 2017.

[88] Z. C. Lipton, D. C. Kale, C. Elkan, and R. Wetzel, "Learning to diagnose with LSTM recurrent neural networks," 2015, *arXiv:1511.03677*. [Online]. Available: https://arxiv.org/abs/1511.03677

[89] R. Fu, Z. Zhang, and L. Li, "Using LSTM and GRU neural network methods for traffic flow prediction," in *Proc. Youth Acad. Annu. Conf. Chin. Assoc. Automat. (YAC)*, Nov. 2016, pp. 324–328.

[90] P. Filonov, A. Lavrentyev, and A. Vorontsov, "Multivariate industrial time series with cyber-attack simulation: Fault detection using an LSTM-based predictive data model," 2016, *arXiv:1612.06676*. [Online]. Available: https://arxiv.org/abs/1612.06676

[91] K. Ma, H. Leung, E. Jalilian, and D. Huang, "Fiber-optic acoustic-based disturbance prediction in pipelines using deep learning," *IEEE Sensors Lett.*, vol. 1, no. 6, Dec. 2017, Art. no. 6001404.

[92] Y. Bengio and F. Gingras, "Recurrent neural networks for missing or asynchronous data," *Adv. Neural Inf.Processing Syst.*, vol. 1996, pp. 395–401.

[93] D. Neil, M. Pfeiffer, and S. C. Liu, "Phased LSTM: Accelerating recurrent network training for long or event-based sequences," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3882–3890.

[94] F. A. Gers and J. Schmidhuber, "Recurrent nets that time and count," in *Proc. IEEE-INNS-ENNS Int. Joint Conf. Neural Netw. (IJCNN)*, vol. 3, Jul. 2000, pp. 189–194.

[95] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An empirical exploration of recurrent network architectures," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 2342–2350.

[96] K. Cho *et al.*, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," 2014, *arXiv:1406.1078*. [Online]. Available: https://arxiv.org/abs/1406.1078

[97] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang, and G. Cottrell, "A dual-stage attention-based recurrent neural network for time series prediction," 2017, *arXiv:1704.02971*. [Online]. Available: https://arxiv.org/abs/1704.02971

[98] Y. Liang, S. Ke, J. Zhang, X. Yi, and Y. Zheng, "GeoMAN: Multi-level attention networks for geo-sensory time series prediction," in *Proc. IJCAI*, 2018, pp. 3428–3434.

[99] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," 2014, *arXiv:1409.1259*. [Online]. Available: https://arxiv.org/abs/1409.1259

[100] A. Graves, S. Fernández, and J. Schmidhuber, "Bidirectional LSTM networks for improved phoneme classification and recognition," in *Proc. Int. Conf. Artif. Neural Netw.* Berlin, Germany: Springer, 2005, pp. 799–804.

[101] A. Zhao, L. Qi, and J. Li, "LSTM for diagnosis of neurodegenerative diseases using gait data," in *Proc. 9th Int. Conf. Graph. Image Process. (ICGIP)*, vol. 10615, 2018, Art. no. 106155B.

[102] D. Zhang, G. Lindholm, and H. Ratnaweera, "Use long short-term memory to enhance Internet of Things for combined sewer overflow monitoring," *J. Hydrol.*, vol. 556, pp. 409–418, Jan. 2018.

[103] J. Cowton, I. Kyriazakis, T. Plötz, and J. Bacardit, "A combined deep learning gru-autoencoder for the early detection of respiratory disease in pigs using multiple environmental sensors," *Sensors*, vol. 18, no. 8, p. 2521, 2018.

[104] A. X. Wang, C. Tran, N. Desai, D. Lobell, and S. Ermon, "Deep transfer learning for crop yield prediction with remote sensing data," in *Proc. 1st ACM SIGCAS Conf. Comput. Sustain. Soc.*, 2018, p. 50.

[105] X. Jia, A. Khandelwal, and G. Nayak, "Predict land covers with transition modeling and incremental learning," in *Proc. SIAM Int. Conf. Data Mining*, 2017, pp. 171–179.

[106] X. Jia *et al.*, "Incremental dual-memory LSTM in land cover prediction," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2017, pp. 867–876.

[107] Q. Zhang, L. T. Yang, Z. Chen, and P. Li, "A survey on deep learning for big data," *Inf. Fusion*, vol. 42, pp. 146–157, Jul. 2018.

[108] J. Zheng and L. Peng, "An autoencoder-based image reconstruction for electrical capacitance tomography," *IEEE Sensors J.*, vol. 18, no. 13, pp. 5464–5474, Jul. 2018.

[109] J. Snoek, R. P. Adams, and H. Larochelle, "Nonparametric guidance of autoencoder representations using label information," *J. Mach. Learn. Res.*, vol. 13, pp. 2567–2588, Sep. 2012.

[110] A. P. S. Chandar *et al.*, "An autoencoder approach to learning bilingual word representations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 1853–1861.

[111] Z. Shao, L. Zhang, and L. Wang, "Stacked sparse autoencoder modeling using the synergy of airborne LiDAR and satellite optical and SAR data to map forest above-ground biomass," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 10, no. 12, pp. 5569–5582, Dec. 2017.

[112] J. Geng, H. Wang, J. Fan, and X. Ma, "Deep supervised and contractive neural network for SAR image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 4, pp. 2442–2459, Apr. 2017.

[113] X. Zhang, G. Chen, W. Wang, Q. Wang, and F. Dai, "Object-based land-cover supervised classification for very-high-resolution UAV images using stacked denoising autoencoders," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 10, no. 7, pp. 3373–3385, Jul. 2017.

[114] P. Zhang *et al.*, "Multimodal fusion for sensor data using stacked autoencoders," in *Proc. IEEE 10th Int. Conf. Intell. Sensors, Sensor Netw. Inf. Process. (ISSNIP)*, Apr. 2015, pp. 1–2.

[115] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 865–873, Apr. 2015.

[116] Q. Yang, Y. Zhou, Y. Yu, J. Yuan, X. Xing, and S. Du, "Multi-step-ahead host load prediction using autoencoder and echo state networks in cloud computing," *J. Supercomput.*, vol. 71, no. 8, pp. 3037–3053, 2015.

[117] C. Li, Z. Ding, D. Zhao, J. Yi, and G. Zhang, "Building energy consumption prediction: An extreme deep learning approach," *Energies*, vol. 10, no. 10, p. 1525, 2017.

[118] L. Deng and D. Yu, "Deep learning: Methods and applications," *Found. Trends Signal Process.*, vol. 7, nos. 3–4, pp. 197–387, 2014.

[119] C. Sun, M. Ma, Z. B. Zhao, and X. Chen, "Sparse deep stacking network for fault diagnosis of motor," *IEEE Trans. Ind. Informat.*, vol. 14, no. 7, pp. 3261–3270, Jul. 2018.

[120] H. Palangi, R. Ward, and L. Deng, "Convolutional deep stacking networks for distributed compressive sensing," *Signal Process.*, vol. 131, pp. 181–189, Feb. 2017.

[121] J. Li, H. Chang, and J. Yang, "Sparse deep stacking network for image classification," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 1–7.

[122] Z. Q. Wang and D. L. Wang, "Recurrent deep stacking networks for supervised speech separation," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2017, pp. 71–75.

[123] L. Deng, B. Hutchinson, and D. Yu, "Parallel training for deep stacking networks," in *Proc. 13th Annu. Conf. Int. Speech Commun. Assoc.*, 2012, pp. 1–4.

[124] B. Hutchinson, L. Deng, and D. Yu, "Tensor deep stacking networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1944–1957, Dec. 2013.

[125] P. S. Huang, L. Deng, M. Hasegawa-Johnson, and X. He, "Random features for kernel deep convex network," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 3143–3147.

[126] W. Cohen and R. V. de Carvalho, "Stacked sequential learning," in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, 2005, pp. 671–676.

[127] M. Das and S. K. Ghosh, "Deep-STEP: A deep learning approach for spatiotemporal prediction of remote sensing data," *IEEE Geosci. Remote Sens. Lett.*, vol. 13, no. 12, pp. 1984–1988, Dec. 2016.

[128] M. Das and S. K. Ghosh, "A deep-learning-based forecasting ensemble to predict missing data for remote sensing analysis," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 10, no. 12, pp. 5228–5236, Dec. 2017.

[129] P. Romeu, F. Zamora-Martínez, and P. Botella–Rocamora, "Stacked denoising auto-encoders for short-term time series forecasting," in *Artificial Neural Networks*. Cham, Switzerland: Springer, 2015, pp. 463–486.

[130] Y. Zhao, J. Li, and L. Yu, "A deep learning ensemble approach for crude oil price forecasting," *Energy Econ.*, vol. 66, pp. 9–16, Aug. 2017.

[131] G. Song and Q. Dai, "A novel double deep ELMs ensemble system for time series forecasting," *Knowl.-Based Syst.*, vol. 134, pp. 31–49, Oct. 2017.

[132] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA, USA: MIT Press, 2006.

[133] J. Hu and J. Wang, "Short-term wind speed prediction using empirical wavelet transform and Gaussian process regression," *Energy*, vol. 93, pp. 1456–1466, Dec. 2015.

[134] R. Palm, "Multiple-step-ahead prediction in control systems with Gaussian process models and TS-fuzzy models," *Eng. Appl. Artif. Intell.*, vol. 20, no. 8, pp. 1023–1035, 2007.

[135] W. Yan, H. Qiu, and Y. Xue, "Gaussian process for long-term time-series forecasting," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Jun. 2009, pp. 3420–3427.

[136] D. Lee and R. Baldick, "Short-term wind power ensemble prediction based on Gaussian processes and neural networks," *IEEE Trans. Smart Grid*, vol. 5, no. 1, pp. 501–510, Jan. 2014.

[137] S. Brahim-Belhouari and A. Bermak, "Gaussian process for nonstationary time series prediction," *Comput. Statist. Data Anal.*, vol. 47, no. 4, pp. 705–712, 2004.

[138] A. Girard, C. E. Rasmussen, J. Q. Candela, and R. Murray-Smith, "Gaussian process priors with uncertain inputs application to multiple-step ahead time series forecasting," in *Proc. Adv. Neural Inf. Process. Syst.*, 2003, pp. 545–552.

[139] A. Damianou and N. D. Lawrence, "Semi-described and semi-supervised learning with Gaussian processes," 2015, *arXiv:1509.01168*. [Online]. Available: https://arxiv.org/abs/1509.01168

[140] M. Lázaro-Gredilla and M. K. Titsias, "Variational heteroscedastic Gaussian process regression," in *Proc. Int. Conf. Mach. Learn.*, 2011, pp. 841–848.

[141] P. Kou, D. Liang, L. Gao, and J. Lou, "Probabilistic electricity price forecasting with variational heteroscedastic Gaussian process and active learning," *Energy Convers. Manage.*, vol. 89, pp. 298–308, Jan. 2015.

[142] J. Han, X.-P. Zhang, and F. Wang, "Gaussian process regression stochastic volatility model for financial time series," *IEEE J. Sel. Topics Signal Process.*, vol. 10, no. 6, pp. 1015–1028, Sep. 2016.

[143] D. Heckerman, "A tutorial on learning with Bayesian networks," in *Learning in Graphical Models*. Dordrecht, The Netherlands: Springer, 1998, pp. 301–354.

[144] K. P. Murphy and S. Russell, "Dynamic Bayesian networks: Representation, inference and learning," Ph.D. dissertation, Dept. Comput. Sci., Graduate Division, Univ. California, Berkeley, Berkeley, CA, USA, 2002.

[145] Q. Xiao, C. Chaoqin, and Z. Li, "Time series prediction using dynamic Bayesian network," *Optik*, vol. 135, pp. 98–103, Apr. 2017.

[146] L. Chen, Y. Liu, J. Zhao, W. Wang, and Q. Liu, "Prediction intervals for industrial data with incomplete input using kernel-based dynamic Bayesian networks," *Artif. Intell. Rev.*, vol. 46, no. 3, pp. 307–326, 2016.

[147] M. Naili, M. Bourahla, M. Naili, and A. K. Tari, "Stability-based dynamic Bayesian network method for dynamic data mining," *Eng. Appl. Artif. Intell.*, vol. 77, pp. 283–310, Jan. 2019.

[148] M. Zou and S. D. Conzen, "A new dynamic Bayesian network (DBN) approach for identifying gene regulatory networks from time course microarray data," *Bioinformatics*, vol. 21, no. 1, pp. 71–79, Aug. 2004.

[149] K. Kourou, C. Papaloukas, and D. I. Fotiadis, "Integration of pathway knowledge and dynamic Bayesian networks for the prediction of oral cancer recurrence," *IEEE J. Biomed. Health Inform.*, vol. 21, no. 2, pp. 320–327, Dec. 2017.

[150] M. Das and S. K. Ghosh, "Spatio-temporal prediction of meteorological time series data: An approach based on spatial Bayesian network (SpaBN)," in *Proc. Int. Conf. Pattern Recognit. Mach. Intell.* Cham, Switzerland: Springer, 2017, pp. 615–622.

[151] H. Guo, X. Liu, and Z. Sun, "Multivariate time series prediction using a hybridization of VARMA models and Bayesian networks," *J. Appl. Statist.*, vol. 43, no. 16, pp. 2897–2909, 2016.

[152] N. Lethanh, K. Kaito, and K. Kobayashi, "Infrastructure deterioration prediction with a Poisson hidden Markov model on time series data," *J. Infrastruct. Syst.*, vol. 21, no. 3, 2014, Art. no. 04014051.

[153] S. Bhardwaj, S. Srivastava, S. Vaishnavi, and J. R. P. Gupta, "Chaotic time series prediction using combination of hidden Markov model and neural nets," in *Proc. IEEE Int. Conf. Comput. Inf. Syst. Ind. Manage. Appl. (CISIM)*, Oct. 2010, pp. 585–589.

[154] A. M. Mikaeil, B. Guo, X. Bai, and Z. Wang, "Primary user channel state prediction based on time series and hidden Markov model," in *Proc. 2nd IEEE Int. Conf. Syst. Inform. (ICSAI)*, Nov. 2014, pp. 866–870.

[155] K. Wakabayashi and T. Miura, "Data stream prediction using incremental hidden Markov models," in *Proc. Int. Conf. Data Warehousing Knowl. Discovery*. Berlin, Germany: Springer, 2009, pp. 63–74.

[156] H. Guo, W. Pedrycz, and X. Liu, "Hidden Markov models based approaches to long-term prediction for granular time series," *IEEE Trans. Fuzzy Syst.*, vol. 26, no. 5, pp. 2807–2817, Oct. 2018.

[157] R. Hassan, B. Nath, and M. Kirley, "HMM based fuzzy model for time series prediction," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, Jan. 2006, pp. 2120–2126.

[158] M. Dong, D. Yang, Y. Kuang, D. He, S. Erdal, and D. Kenski, "$PM_{2.5}$ concentration prediction using hidden semi-Markov model-based times series data mining," *Expert Syst. Appl.*, vol. 36, no. 5, pp. 9046–9055, 2009.

[159] G. E. Hinton and T. J. Sejnowski, "Learning and relearning in Boltzmann machines," in *Parallel Distributed Processing–Explorations in the Microstructure of Cognition*, vol. 1. Cambridge, MA, USA: MIT Press, 1986, pp. 282–317.

[160] N. Lu, T. Li, X. Ren, and H. Miao, "A deep learning scheme for motor imagery classification based on restricted Boltzmann machines," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 25, no. 6, pp. 566–576, Jun. 2017.

[161] I. Sutskever, G. E. Hinton, and G. W. Taylor, "The recurrent temporal restricted Boltzmann machine," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 1601–1608.

[162] V. Mnih, H. Larochelle, and G. E. Hinton, "Conditional restricted Boltzmann machines for structured output prediction," 2012, *arXiv:1202.3748*. [Online]. Available: https://arxiv.org/abs/1202.3748

[163] K. Sohn, G. Zhou, C. Lee, and H. Lee, "Learning and selecting features jointly with point-wise gated Boltzmann machines," in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. 217–225.

[164] T. Kuremoto, S. Kimura, and K. Kobayashi, "Time series forecasting using restricted Boltzmann machine," in *Proc. Int. Conf. Intell. Comput.* Berlin, Germany: Springer, 2012, pp. 17–22.

[165] T. Osogami, "Boltzmann machines for time-series," 2017, *arXiv:1708.06004*. [Online]. Available: https://arxiv.org/abs/1708.06004

[166] C.-Y. Zhang, C. L. P. Chen, M. Gan, and L. Chen, "Predictive deep Boltzmann machine for multiperiod wind speed forecasting," *IEEE Trans. Sustain. Energy*, vol. 6, no. 4, pp. 1416–1425, Oct. 2015.

[167] S. Dasgupta and T. Osogami, "Nonlinear dynamic Boltzmann machines for time-series prediction," in *Proc. AAAI*, 2017, pp. 1833–1839.

[168] N. Le Roux and Y. Bengio, "Representational power of restricted Boltzmann machines and deep belief networks," *Neural Comput.*, vol. 20, no. 6, pp. 1631–1649, Jun. 2008.

[169] T. Kuremoto, M. Obayashi, K. Kobayashi, T. Hirata, and S. Mabu, "Forecast chaotic time series data by DBNs," in *Proc. 7th Int. Congr. Image Signal Process. (CISP)*, Oct. 2014, pp. 1130–1135.

[170] T. Kuremoto, S. Kimura, K. Kobayashi, and M. Obayashi, "Time series forecasting using a deep belief network with restricted Boltzmann machines," *Neurocomputing*, vol. 137, pp. 47–56, Aug. 2014.

[171] W. Huang, G. Song, H. Hong, and K. Xie, "Deep architecture for traffic flow prediction: Deep belief networks with multitask learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 5, pp. 2191–2201, Oct. 2014.

[172] H. Z. Wang, G. B. Wang, G. Q. Li, J. C. Peng, and Y. T. Liu, "Deep belief network based deterministic and probabilistic wind speed forecasting approach," *Appl. Energy*, vol. 182, pp. 80–93, Nov. 2016.

[173] A. Dedinec, S. Filiposka, A. Dedinec, and L. Kocarev, "Deep belief network based electricity load forecasting: An analysis of macedonian case," *Energy*, vol. 115, pp. 1688–1700, Nov. 2016.

[174] J. Chen, Q. Jin, and J. Chao, "Design of deep belief networks for short-term prediction of drought index using data in the Huaihe river basin," *Math. Problems Eng.*, vol. 2012, Feb. 2012, Art. no. 235929.

[175] P. Jiang, C. Chen, and X. Liu, "Time series prediction for evolutions of complex systems: A deep learning approach," in *Proc. IEEE Int. Conf. Control Robot. Eng. (ICCRE)*, Apr. 2016, pp. 1–6.

[176] R. Soua, A. Koesdwiady, and F. Karray, "Big-data-generated traffic flow prediction using deep learning and dempster-shafer theory," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2016, pp. 3195–3202.

[177] X. Qiu, Y. Ren, P. N. Suganthan, and G. A. J. Amaratunga, "Empirical mode decomposition based ensemble deep learning for load demand time series forecasting," *Appl. Soft Comput.*, vol. 54, pp. 246–255, May 2017.

[178] X. Qiu, L. Zhang, Y. Ren, P. N. Suganthan, and G. Amaratunga, "Ensemble deep learning for regression and time series forecasting," in *Proc. IEEE Symp. Comput. Intell. Ensemble Learn. (CIEL)*, Dec. 2014, pp. 1–6.

[179] T. Hirata, T. Kuremoto, M. Obayashi, S. Mabu, and K. Kobayashi, "Time series prediction using DBN and ARIMA," in *Proc. Int. Conf. Comput. Appl. Technol. (CCATS)*, Apr. 2015, pp. 24–29.

[180] I. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[181] I. Goodfellow, "NIPS 2016 tutorial: Generative adversarial networks," 2016, *arXiv:1701.00160*. [Online]. Available: https://arxiv.org/abs/1701.00160

[182] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 2014, *arXiv:1411.1784*. [Online]. Available: https://arxiv.org/abs/1411.1784

[183] X. Chen, Y. Duan, and R. Houthooft, "InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2172–2180.

[184] L. Yu, W. Zhang, J. Wang, J. Schulman, I. Sutskever, and P. Abbeel, "Seqgan: Sequence generative adversarial nets with policy gradient," in *Proc. 21st AAAI Conf. Artif. Intell.*, 2017, pp. 2172–2180.

[185] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 214–223.

[186] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley, "Least squares generative adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 2794–2802.

[187] X. Yi, E. Walia, and P. Babyn, "Generative adversarial network in medical imaging: A review," 2018, *arXiv:1809.07294*. [Online]. Available: https://arxiv.org/abs/1809.07294

[188] X. Zhou, Z. Pan, G. Hu, S. Tang, and C. Zhao, "Stock market prediction on high-frequency data using generative adversarial nets," *Math. Problems Eng.*, vol. 2018, 2018, Art. no. 4907423.

[189] C. Tian, C. Li, G. Zhang, and Y. Lv, "Data driven parallel prediction of building energy consumption using generative adversarial nets," *Energy Buildings*, vol. 186, pp. 230–243, Mar. 2019.

[190] Y. Lv, Y. Chen, L. Li, and F. Wang, "Generative adversarial networks for parallel transportation systems," *IEEE Intell. Transp. Syst. Mag.*, vol. 10, no. 3, pp. 4–10, 2018.

[191] Y. Liang, Z. Cui, Y. Tian, H. Chen, and Y. Wang, "A deep generative adversarial architecture for network-wide spatial-temporal traffic-state estimation," *Transp. Res. Rec.*, vol. 2672, pp. 87–105, Jan. 2018.

[192] C. Bowles *et al.*, "GAN augmentation: Augmenting training data using generative adversarial networks," 2018, *arXiv:1810.10863*. [Online]. Available: https://arxiv.org/abs/1810.10863

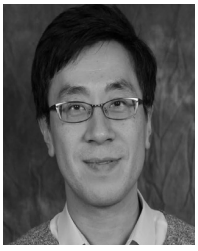
[193] Y. K. Bang and C. H. Lee, "Fuzzy time series prediction using hierarchical clustering algorithms," *Expert Syst. Appl.*, vol. 38, no. 4, pp. 4312–4325, 2011.
[194] V. A. Gromov and E. A. Borisenko, "Predictive clustering on non-successive observations for multi-step ahead chaotic time series prediction," *Neural Comput. Appl.*, vol. 26, no. 8, pp. 1827–1838, 2015.
[195] T. Wang, Z. Han, J. Zhao, and W. Wang, "Adaptive granulation-based prediction for energy system of steel industry," *IEEE Trans. Cybern.*, vol. 48, no. 1, pp. 127–138, Nov. 2018.
[196] W. Lu, J. Yang, X. Liu, and W. Pedrycz, "The modeling and prediction of time series based on synergy of high-order fuzzy cognitive map and fuzzy c-means clustering," *Knowl.-Based Syst.*, vol. 70, pp. 242–255, Nov. 2014.
[197] C. Smith and D. Wunsch, "Time series prediction via two-step clustering," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2015, pp. 1–4.
[198] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vol. 13, no. 1, pp. 21–27, Jan. 1967.
[199] H. Zhang, A. C. Berg, M. Maire, and J. Malik, "SVM-KNN: Discriminative nearest neighbor classification for visual category recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, vol. 2, Jun. 2006, pp. 2126–2136.
[200] J. Myung, D.-K. Kim, S.-Y. Kho, and C.-H. Park, "Travel time prediction using k-nearest neighbor method with combined data from vehicle detector system and automatic toll collection system," *Transp. Res. Rec.*, vol. 2256, no. 1, pp. 51–59, 2011.
[201] Z. Xing, J. Pei, and S. Y. Philip, "Early prediction on time series: A nearest neighbor approach," in *Proc. IJCAI*, 2009, pp. 1297–1302.
[202] L. Zhang, Q. Liu, W. Yang, N. Wei, and D. Dong, "An improved k-nearest neighbor model for short-term traffic flow prediction," *Procedia-Social Behav. Sci.*, vol. 96, pp. 653–662, Nov. 2013.
[203] F. H. Al-Qahtani and S. F. Crone, "Multivariate k-nearest neighbour regression for time series data—A novel algorithm for forecasting UK electricity demand," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Aug. 2013, pp. 1–8.
[204] B. Yu, X. Song, F. Guan, Z. Yang, and B. Yao, "K-nearest neighbor model for multiple-time-step prediction of short-term traffic condition," *J. Transp. Eng.*, vol. 142, no. 6, 2016, Art. no. 04016018.
[205] Y. Bai, Z. Chen, J. Xie, and C. Li, "Daily reservoir inflow forecasting using multiscale deep feature learning with hybrid models," *J. Hydrol.*, vol. 532, pp. 193–206, Jan. 2016.
[206] H. Liu, X.-W. Mi, and Y.-F. Li, "Wind speed forecasting method based on deep learning strategy using empirical wavelet transform, long short term memory neural network and Elman neural network," *Energy Convers. Manage.*, vol. 156, pp. 498–514, Jan. 2018.
[207] N. Kussul, A. Shelestov, M. Lavreniuk, I. Butko, and S. Skakun, "Deep learning approach for large scale land cover mapping based on remote sensing data fusion," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2016, pp. 198–201.
[208] Y. Wu and H. Tan, "Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework," 2016, *arXiv:1612.01022*. [Online]. Available: https://arxiv.org/abs/1612.01022
[209] A. Gensler, J. Henze, and B. Sick, "Deep learning for solar power forecasting—An approach using AutoEncoder and LSTM neural networks," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2016, pp. 2858–2865.
[210] A. Grover, A. Kapoor, and E. Horvitz, "A deep hybrid model for weather forecasting," *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2015, pp. 379–386.
[211] I. Colak, S. Sagiroglu, and M. Yesilbudak, "Data mining and wind power prediction: A literature review," *Renew. Energy*, vol. 46, pp. 241–247, Oct. 2012.
[212] T. Chai and R. R. Draxler, "Root mean square error (RMSE) or mean absolute error (MAE)?—Arguments against avoiding RMSE in the literature," *Geosci. Model Develop.*, vol. 7, no. 3, pp. 1247–1250, 2014.
[213] S. P. Singh, S. Urooj, and A. Lay-Ekuakille, "Breast cancer detection using PCPCET and ADEWNN: A geometric invariant approach to medical X-ray image sensors," *IEEE Sensors J.*, vol. 16, no. 12, pp. 4847–4855, Jun. 2016.
[214] J. Zhao and X. Yu, "Adaptive natural gradient learning algorithms for Mackey–Glass chaotic time prediction," *Neurocomputing*, vol. 157, pp. 41–45, Jun. 2015.
[215] V. A. Gromov and A. N. Shulga, "Chaotic time series prediction with employment of ant colony optimization," *Expert Syst. Appl.*, vol. 39, no. 9, pp. 8474–8478, 2012.
[216] T. Miller, "Explanation in artificial intelligence: Insights from the social sciences," *Artif. Intell.*, vol. 267, pp. 1–38, Feb. 2018.
[217] A. Adadi and M. Berrada, "Peeking inside the black-box: A survey on explainable artificial intelligence (XAI)," *IEEE Access*, vol. 6, pp. 52138–52160, 2018.
[218] T. Miller, P. Howe, and L. Sonenberg, "Explainable AI: Beware of inmates running the asylum or: How I learnt to stop worrying and love the social and behavioural sciences," in *Proc. IJCAI Workshop Explainable AI (XAI)*, 2017, p. 36.
[219] O. Biran and C. Cotton, "Explanation and justification in machine learning: A survey," in *Proc. IJCAI Workshop Explainable AI (XAI)*, 2017, p. 8.
[220] P. W. Koh and P. Liang, "Understanding black-box predictions via influence functions," 2017, *arXiv:1703.04730*. [Online]. Available: https://arxiv.org/abs/1703.04730
[221] S. Gu, T. Lillicrap, I. Sutskever, and S. Levine, "Continuous deep q-learning with model-based acceleration," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 2829–2838.
[222] C. Zhang, G. Sun, Z. Fang, P. Zhou, P. Pan, and J. Cong, "Caffeine: Towards uniformed representation and acceleration for deep convolutional neural networks," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, to be published.
[223] T. Gokmen and Y. Vlasov, "Acceleration of deep neural network training with resistive cross-point devices: Design considerations," *Frontiers Neurosci.*, vol. 10, p. 333, Jul. 2016.
[224] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.
[225] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. AAAI*, vol. 2, 2016, p. 5.
[226] V. Mnih *et al.*, "Playing atari with deep reinforcement learning," 2013, *arXiv:1312.5602*. [Online]. Available: https://arxiv.org/abs/1312.5602
[227] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," 2017, *arXiv:1703.03400*. [Online]. Available: https://arxiv.org/abs/1703.03400
[228] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, "Meta-learning with memory-augmented neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1842–1850.
[229] A. Dosovitskiy and V. Koltun, "Learning to act by predicting the future," 2016, *arXiv:1611.01779*. [Online]. Available: https://arxiv.org/abs/1611.01779

Zhongyang Han (M'18) received the B.S. degree from the City Institute, Dalian University of Technology, Dalian, China, and the Ph.D. degree in engineering from the Dalian University of Technology, in 2010 and 2016, respectively.

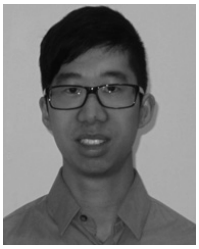
He is currently an Associate Professor with the School of Control Science and Engineering, Dalian University of Technology. His current research interests include computer-integrated manufacturing systems, artificial intelligence, data mining, and machine learning.

Jun Zhao (M'10) received the B.S. degree in control theory from Dalian Jiaotong University, Dalian, China, and the Ph.D. degree in engineering from the Dalian University of Technology, Dalian, in 2003 and 2008, respectively.

He is currently a Professor with the School of Control Science and Engineering, Dalian University of Technology. His current research interests include industrial production scheduling, computer-integrated manufacturing, intelligent optimization, and machine learning.



Henry Leung (F'15) was with the Department of National Defence (DND) of Canada as a Defense Scientist. He is a Professor with the Department of Electrical and Computer Engineering, University of Calgary. His current research interests include information fusion, machine learning, IoT, nonlinear dynamics, robotics, and signal and image processing. He is a Fellow of SPIE. He is an Associate Editor of the *IEEE Circuits and Systems Magazine* and the IEEE TRANSACTIONS ON AEROSPACE AND ELECTRONIC SYSTEMS. He is the Topic Editor on Robotic Sensors of the *International Journal of Advanced Robotic Systems*. He is an Editor of the Springer book series on *Information Fusion and Data Science*.



King Fai Ma received the B.S. (Hons.) degree in electrical engineering and the M.S. degree in electrical engineering from the University of Calgary in 2017, where he is currently pursuing the Ph.D. degree in electrical engineering with the Sensor Networks and Robotics Laboratory.

His research interests include big data analytics, deep learning, applied signal processing, computer vision, and pipeline monitoring.

Wei Wang (SM'13) received the B.S., M.S., and Ph.D. degrees from Northeastern University, Shenyang, China, in 1982, 1986, and 1988, respectively, all in industrial automation.

He was a Post-Doctoral Fellow with the Division of Engineering Cybernetics, Norwegian Science and Technology University, Trondheim, Norway, from 1990 to 1992, a Professor and Vice Director of the National Engineering Research Center of Metallurgical Automation of China, Beijing, China, from 1995 to 1999, and a Research Fellow with the Department of Engineering Science, University of Oxford, Oxford, U.K., from 1998 to 1999. He is currently a Professor with the School of Control Sciences and Engineering, Dalian University of Technology, Dalian, China. His current research interests include adaptive controls, computer-integrated manufacturing, and computer controls of industrial processes.