EgPDE-Net: Building Continuous Neural Networks for Time Series Prediction With Exogenous Variables

Penglei Gao, Xi Yang, Rui Zhang, Ping Guo, Senior Member, IEEE, John Y. Goulermas, and Kaizhu Huang, Senior Member, IEEE

Abstract—While exogenous variables have a major impact on performance improvement in time series analysis, interseries correlation and time dependence among them are rarely considered in the present continuous methods. The dynamical systems of multivariate time series could be modeled with complex unknown partial differential equations (PDEs) which play a prominent role in many disciplines of science and engineering. In this article, we propose a continuous-time model for arbitrary-step prediction to learn an unknown PDE system in multivariate time series whose governing equations are parameterized by self-attention and gated recurrent neural networks. The proposed model, exogenous-guided PDE network (EgPDE-Net), takes account of the relationships among the exogenous variables and their effects on the target series. Importantly, the model can be reduced into a regularized ordinary differential equation (ODE) problem with specially designed regularization guidance, which makes the PDE problem tractable to obtain numerical solutions and feasible to predict multiple future values of the target series at arbitrary time points. Extensive experiments demonstrate that our proposed model could achieve competitive accuracy over strong baselines: on average, it outperforms the best baseline by reducing 9.85% on RMSE and 13.98% on MAE for arbitrary-step prediction.

Index Terms—Arbitrary-step prediction, continuous time, partial differential equation (PDE), time series analysis.

I. INTRODUCTION

TIME series analysis is an essential topic in diverse real-world scenarios, such as power prediction [1], financial investment [2], air quality assessment [3], clinical analysis [4], and traffic forecasting [5]. Accurate prediction of the future evolution helps people make important decisions for benefit maximization. With more demanding scenarios, one challenging and meaningful task is to forecast continuous multiple future values of one specific target series at arbitrary time points with multivariate time series. Most deep learning structures, including recurrent neural networks (RNNs), are interpreted as a discrete approximation to sequence prediction, which could only forecast the fixed step size of future values [6], [7]. This discretization typically breaks down for arbitrary-step prediction [8]. When dealing with this arbitrary-step prediction problem, a more practical approach is to build continuous models for the dynamical behavior of multivariate data. Fig. 1 shows the different predictive between traditional discrete networks and our proposed continuous exogenous-variable-guided framework.

Building continuous networks has drawn much attention in academic research fields in a variety of applications, such as time series analysis, node classification with graph neural network, set modeling, and normalizing flows [9], [10], [11], [12], [13], [14]. The neural ordinary differential equation (ODE) is one popular mainstream proposed in [9] to model time series with a continuous-time approach, in which neural networks parameterize the derivative of the hidden states. While multivariate data are input for arbitrary-step prediction, the exogenous variables have different impacts on the target series for the prediction performance. Modeling the interaction among the exogenous variables could provide extra information and more accurate predictions. Despite the benefit, interseries correlation and time dependence among the exogenous variables are rarely considered in the present continuous methods.

To better cope with both building continuous networks for the arbitrary-step prediction and modeling the impacts of the exogenous variables jointly, we propose a general continuous-time method called exogenous-guided partial differential equation network (EgPDE-Net) in this article. The exogenous
variables contain a rich data structure and information, especially the interactions among variables, which could further benefit the prediction performance. As a critical contribution, the proposed partial differential equation (PDE) framework can better model the influence of exogenous variables, which is rarely studied in the existing ODE-based methods. Unlike current mainstreams for continuous-time modeling, that is, the ODE-based networks, EgPDE-Net is the first model built on a more general PDE framework. It describes the multivariate time series by referring to the Cauchy problem in the heat conduction equation

$$\dot{z}(y, x, t) := F(y, x, z, t, \nabla_x z, \nabla^2_x z, \ldots).$$ \hspace{1cm} (1)$$

Physically, PDEs are the fundamental equations of many important disciplines exploring the mysteries of the universe [15]. Mathematically, they describe the changes that can be expressed continuously in the system through partial derivatives of multiple independent variables. More importantly, PDEs enjoy an elegant mechanism to model the derivatives of input multiple variables and latent states; this is more theoretically appealing than ODE systems which only describe the evolution of latent states w.r.t. time. Consequently, whilst making accurate continuous predictions, PDEs offer a better capability to model the impact of exogenous variables.

On the practical front, we design a tractable way to solve the PDE problem for arbitrary-step prediction by utilizing two ODE nets to achieve the transition between PDE and ODE. Specifically, the exogenous variables are processed with a self-attention block extracting the global and local information simultaneously. The correlation among exogenous variables could be captured in the encoding phase. The attention mechanism takes advantage of allocating different weights representing each series, which could provide interpretability to distinguish the different contributions among different driving time series [16], [17]. The first ODE network is applied to generate the partial differential weights in both the time and variable domains to guide the predicted target series trajectory. On the other hand, the target series is processed with a GRU block to capture the temporal dependence. The second ODE network could generate the final latent states with the hidden representation of the target series and partial differential weights.

It is noted that some multivariate time series prediction methods can implicitly model the impact of exogenous variables by forecasting all the input series simultaneously [18], [19], [20]. On the one hand, these methods lack the capacity of forecasting continuous multiple future values and can only be applied to traditional discrete cases. On the other hand, they could not pinpoint the influence of exogenous variables on the specific target series.

The main contributions of this article could be summarized as follows.

1) We propose a novel continuous method to consider both the intraseries temporal patterns for the target series and the interseries correlations among the exogenous variables for multivariate time series analysis.

2) To the best of our knowledge, the designed general PDE framework, called EgPDE-Net, is the first work to build continuous-time representation for multivariate time series as a PDE problem. The specially designed architecture could transform the PDE problem into the ODE problem with the tool of an ODE solver, which makes the PDE problem easier to solve and feasible to conduct arbitrary-step prediction.

3) Experiments show that EgPDE-Net performs better than the strong baselines over four multivariate time series. On average, it outperforms the best baseline by reducing 9.85% on RMSE and 13.98% on MAE for arbitrary-step prediction.

II. RELATED WORK

A. Multivariate Time Series

Classical methods, including the autoregressive (AR) model [21] and Vector AR model [22], have shown their effectiveness for various real-world applications based on a linear behavior given the past values of the series. However, the linearity limits them to model complex nonlinear characteristics in multivariate time series. In the recent decade, deep learning methods have experienced booming development for nonlinear high-dimensional time-varying problems [23], [24], [25] with the capability of handling nonlinear problems in multistep time series prediction task [6], [17]. In the work of [26], the authors built a model with long-short term memory (LSTM) architecture to forecast multiple values on Web traffic data. Convolution neural networks could also be applied for multivariate time series prediction. Binkowski et al. [19] proposed a novel convolutional network for predicting multivariate asynchronous time series called significance-offset convolutional neural network (SOCNN). This model was designed to combine the AR model and CNN. There are
two convolutional parts in this model architecture. One captures the local significance of observed data, while the other represents the predictors entirely independent of position in time. Methods in [18] and [20] combined CNN and RNN, which aim to predict the future value of each individual variable for multivariate time series. Both of them generated forecasting values of all the series simultaneously, taking their historical data as input. In the work of [27], the authors added the temporal change information to the objective function to capture the impact of the abrupt and slow change information. These approaches have limitations in representing the contributions and influences to the target series with continuously changing information. In contrast, our method focuses on the specific target series by capturing the continuous trends and the network can make predictions using other exogenous variables’ information.

B. Variant of ODE Net

With the emergence of neural ODE, researchers have begun to focus on building continuous models to solve complicated problems. The ODE net is applied for irregularly sampled time series classification in [10]. The authors added the ODE network to the loop of the RNN network. The previous hidden states were modified by an ODE network before being updated in the next step. They designed an encoder with ODE-RNN to process the irregularly sampled data instead of fixing the sampled gap by imputation. In the work of [28], the authors considered the influence of subsequent observations when adjusting the trajectory of the latent states. They modified the ODE net through the controlled differential equations (CDE). The derivative of the input data w.r.t. time $t$ is multiplied to the ODE function when integrating the latent states. Jhin et al. [29] aimed to enhance neural controlled differential equation (NCDE) and extended the interpolation-based NCDE model with both extrapolation and interpolation algorithms. They built another latent continuous path from a discrete time-series input by using an encoder-decoder architecture. Finlay et al. [30] stated that training neural ODEs on large datasets had not been tractable due to the necessity of allowing the adaptive numerical ODE solver to refine its step size to minimal values. They proposed a theoretically grounded combination of both optimal transport and stability regularization to tackle this problem, which encouraged neural ODEs to prefer simpler dynamics. They added a kinetic energy regularization term and a Jacobian Frobenius norm regularization term to the loss function. The results show that this framework could achieve acceptable performance and great reductions in time consumed. The existing ODE related methods rarely consider the influence of other exogenous variables, which could improve the forecasting performance. In our method, the specially designed PDE framework could utilize the impact of multivariate time series and provide a better prediction.

C. Variant of PDE Net

In the objective world, PDEs are extensively applicable to describe many propagative systems modeling the relationship of the partial derivatives w.r.t. time and space, such as heat dissipation, the behavior of sound waves, disease progression, fluid dynamics, weather patterns, or cellular kinetics [31], [32]. Such PDE models are considered the cornerstone of natural science and are utilized to describe most of the fundamental laws in Physics. Recently, Long et al. [33] has proposed a deep feed-forward method PDE-Net to discover the hidden PDE dynamical behavior with a convolutional neural network. Further works have extended the approach with symbolic neural networks [34] and graph neural networks [32]. These models are designed to solve the specific dynamical systems of PDEs, which could be considered as a simulation of the underlying systems. They lack the contribution of the exogenous variables when making predictions for the target series. Luo et al. [35] proposed a new learning framework to automatically model the differential operators for multivariate time series. They design a polynomial-block with convolutional layers to learn the potential derivatives. Unlike the existing PDE methods, we aim to model multivariate time series for arbitrary-step prediction, assuming that the dynamical system is governed by an unknown PDE and supports continuous evolution over time. Our method treats the driving series as a guided term and globally learns the derivatives of exogenous variables.

D. Attention Mechanism in Time Series Prediction

Besides making predictions, providing interpretability is also important for multivariate time series models. The attention mechanism is an appropriate method to provide interpretability. Two types of attention generated by two independent RNNs are applied in [36] and [16] to provide interpretable insights into the data. They leveraged two RNNs generating alpha and beta attention representing the importance of time and variables. Guo et al. [17] designed a parallel network to enhance the interpretability with a tensorized LSTM structure for single-step prediction. The network first generated temporal attention given the input data. Then variable attention is generated according to the temporal attention and the hidden states in RNN. Self-attention is proposed in [37] for language modeling. The authors developed a transformer framework by using an Encoder-Decoder structure. The network generates query, key and value given the input data. Then a Softmax operation is conducted on the matrix multiplication of query and key to generate the attention weights. The self-attention in the transformer could help the network focus on more relevant latent states at specific time steps. Another work of [38] improved the transformer architecture by proposing a ProbSparse Self-attention mechanism which allows each key only to interact with the dominant queries. This framework could leverage the most important queries, reducing the network parameters.

III. METHODOLOGY

In this section, we will introduce the problem statement of arbitrary-step prediction and model details of our EgPDE-Net. We leverage two ODE nets dealing with exogenous variables and target series, respectively, with two pipelines. As shown
in Fig. 2, the exogenous variables are processed with a self-attention block generating a summarized hidden representation $hx_0$. The first ODE net is used to obtain the regularized partial derivatives weights of the hidden representation w.r.t. $X_T$. The hidden representation $hy_0$ is generated using a GRU block with the input of target series. The second ODE net is leveraged combining the hidden representation of the target series and the regularized partial derivatives weights to guide the generation of the final latent states $z_t$, where $x_t = [x_t^1, x_t^2, \ldots, x_t^N], t = 1, \ldots, T$ and $N$ is the number of exogenous variables.

Fig. 2. Model structure of EgPDE-Net. The hidden representations of $hx_0$ and $hy_0$ are generated with the self-attention block and GRU block, respectively. Then the above ODE net is leveraged to obtain the latent states representing the partial derivative weights guiding the generation of the trajectory of the final latent state $z_t$ in the below ODE net. $t_1, t_2, \ldots, t_M$ are the forecasting time points.

A. Arbitrary-Step Prediction

In time series analysis, most deep learning models aim to forecast the future value of time $T+1$ given the historical data of previous $T$ time steps. However, the effective decision often requires forecasting multiple future values for multivariate time series data in many real-world problems. For instance, knowing the demand for electricity in the next few hours could help to devise a better-energy use plan, and forecasting the stock market in the near or distant future could produce more profits [20]. This work aims to forecast multiple future values at arbitrary time points by building a continuous model. We follow the work of [8] for the definition of arbitrary-step prediction.

Given a multivariate time series, it consists of a target series that we want to predict and exogenous variables that impact the target series in the predicting decision. With the length $T$ of historical data, the aim is to forecast the multiple future values of the target series represented as

$$\hat{y}_{T+m_1}, \ldots, \hat{y}_{T+m_K} = F(X_T, Y_T)$$  \hspace{1cm} (2)

where $K$ is the number of predicted future values. The forecasting time interval $[T + m_1, \ldots, T + m_K]$ could be set as any continuous value, for example, $[T + 0.8, \ldots, T + 2.2]$. Here, $F(\cdot)$ is achieved by the proposed continuous framework EgPDE-Net. $X_T$ is the exogenous variables input data of length $T$, and $Y_T$ is the target series input data denoted as

$$X_T = \{x_1, x_2, \ldots, x_T\} \in \mathbb{R}^{T \times N}$$  \hspace{1cm} (3)

$$Y_T = \{y_1, y_2, \ldots, y_T\}^T \in \mathbb{R}^T$$  \hspace{1cm} (4)

B. Neural ODE Net

Neural ODE proposed in [9] is a continuous-time model to overcome the limitations of requiring discrete observation and emission intervals in RNNs. A hidden state $h$ in an RNN is modeled as

$$h_{t+1} = h_t + f(h_t, \theta_t).$$  \hspace{1cm} (5)

When adding more layers and taking smaller steps in (5), the continuous dynamics of hidden units are parameterized through an ODE specified by a neural network

$$\frac{dh(t)}{dt} = f_\theta(h(t), t).$$  \hspace{1cm} (6)

In this form, the ODE network parameterizes the derivative of the hidden states w.r.t. time $t$ with parameters $\theta_t$ rather than directly parameterizing the hidden states. The hidden states $h(t)$ could be evaluated at desired time points by integrating the ODE function over the specific time interval with an initial value, such as

$$h(t_{end}) = h(t_{start}) + \int_{t_{start}}^{t_{end}} f_\theta(h(s), s)ds.$$  \hspace{1cm} (7)

Neural ODE models the incremental changes in time series, bringing more smooth and accurate estimation for prediction tasks, requiring a constant memory cost without storing any intermediate quantities of the forward pass.

C. Reducing PDE Problem With Regularized Guided ODE

For some physics, chemistry, and biology problems, it is necessary to establish various mathematical models. Most of them are described by Reaction-Diffusion Equation for quantitative or qualitative analysis. Reaction-Diffusion Equation could be derived from many natural phenomena, such as heat conduction in physics, substance concentration change in the chemical reaction, and species invasion process in biology. The
matrix-valued functions $A(U(x, t)), B(U(x, t))$ can be used to define the elliptic operator $O_L$ according to [15, 39]

$$O_L U = -\nabla (A(U(x))) \nabla U) + B(U(x)) \nabla U. \quad (8)$$

Matrix-valued functions $A$ and $B$ can be considered the coefficient functions and are determined according to the characteristics of the specific problems. They are weight matrices to represent the PDE system. We will use the designed networks to estimate these weight matrices. In mathematics, Laplace operator is expressed as

$$\nabla^2 U = \Delta U, \quad \Delta U = \sum_i \frac{\partial^2}{\partial x_i^2} U. \quad (9)$$

In some cases, it is meaningful to apply the relatively physical equation to the existing neural methods to improve neural network model performance. The Cauchy problem of the 1-D heat conduction equation $f(x, t)$ with the constrained mapping $\varphi(\cdot)$ is expressed as follows:

$$u_t - a^2 u_{xx} = f(x, t), \quad -\infty < x < +\infty, \quad t > 0$$

subject to $u(x, 0) = \varphi(x), \quad -\infty < x < +\infty \quad (10)$

where $u_t$ is the first order derivative w.r.t. time $t$ and $u_{xx}$ is the second order derivative w.r.t. variable $x$. Many problems could be modeled converging the heat conduction equation in real-world applications. In time series analysis, the sequence changes could be considered as a delivery process similar to the diffusion equation. The future values of the sequence will converge with time evolving given the historical data. When processing with multivariate time series, the target series is influenced by the historical information of itself and other exogenous variables. Inspired from the idea of general neural PDE in the work of [15], the PDE problem for multivariate time series analysis could be defined as follows:

$$\frac{\partial z}{\partial t} = F(y, x_1, \ldots, x_n, t, \frac{\partial^2 z}{\partial x_i^2}, \ldots, \frac{\partial^2 z}{\partial x_n^2}, A(z), B(z)) \quad (11)$$

where $z$ represents the trajectory of target series in latent space, $x_i$ represents the individual exogenous series, and $A(z), B(z)$ are the function of latent state $z$.

The defined nonlinear PDE problem is complicated and usually has no explicit closed-form solution. Inspired from the previous works [32], [40], [41], [42], [43], [44] approximating the PDEs with weak solutions, we reduce the PDE problem to an exogenous-guided ODE problem parameterized by using two neural networks as the weak solutions. The complicated PDE problem could be transformed into a simpler modeling problem and is easier to get the numerical results which also improves the forecasting performance with the weak solutions. In this case, we define the following lemma to transform the PDE problem into a regularized exogenous-guided ODE problem.

**Lemma 1**: For simplicity, we use $\nabla^2$ to represent $\sum_i (\frac{\partial^2 z}{\partial x_i^2})$ according to (9). In (11), referring to the reaction-diffusion equations and quasi-linear approximations for multivariate time series by leveraging (8), the defined PDE problem of (11) could be substituted with

$$\frac{\partial z}{\partial t} = O_L z = -\nabla (A(z)) \nabla z) + B(z) \nabla z. \quad (12)$$

Here, $A(z)$ is the diffusion matrix and $B(z) \nabla z$ is the convection vector. Then we rewrite the (12) according to the differential criterion as follows:

$$\frac{\partial z}{\partial t} = -A(z) \nabla^2 z + B(z) \nabla z = -A(z) \nabla^2 z + B(z) \nabla z \quad (13)$$

We merge $\nabla A(z)$ and $B(z)$ into new parameter $B'(z) = B(z) - \nabla A(z)$. Then we could have the representation of $\frac{\partial z}{\partial t}$ as follows:

$$\frac{\partial z}{\partial t} = -A(z) \nabla^2 z + B'(z) \nabla z \quad (14)$$

When we take the exponential operation on both side of (14), we could have the variant of the derivative of $z$

$$\exp \left( \frac{\partial z}{\partial t} \right) = \exp \left( -A(z) \nabla^2 z + B'(z) \nabla z \right)$$

$$\exp \left( -A(z) \nabla^2 z \right) \cdot \exp \left( B'(z) \nabla z \right) \quad (15)$$

The two parts of the right side of (15) are estimated with two neural networks $G(\cdot)$ and $f(\cdot)$

$$\exp \left( -A(z) \nabla^2 z \right) =: G(z_X) \quad (16)$$

$$\exp \left( B'(z) \nabla z \right) =: f(z_Y) \quad (17)$$

With the approximation and transition by the two neural networks, we could obtain the partial derivative of the latent states $z$

$$\frac{\partial z}{\partial t} = \ln(G(z_X) \ast f(z_Y)) \quad (18)$$

which is an example of Neural ODE [9]. Briefly, we name the ODE in (18) the exogenous guidance ODE, which could be numerically solved using the black-box ODE solver. The graphic illustration of the transition of Lemma 1 is shown in Fig. 3.

The function $G(\cdot)$ is achieved with the first ODE network to get the regularized partial representation of exogenous variables. In this task, the target series is influenced by many exogenous variables. The element-wise product is used in (18) and $G(\cdot)$ is like a weight matrix representing the weight ratio that affects and guides the generated trajectory of the target series for final prediction in the latent space. The first ODE net is applied as follows:

$$z_t = G(z(X)) = \text{ODESolve}(z_{0}, \theta_g, t) \quad (19)$$
\[ z_{y0} = \text{Attblock}(X_T). \] (20)

In (20), \text{Attblock}(\cdot) is achieved by a self-attention block, which could capture the temporal correlation and variable-wise correlation among the exogenous variables. The attention weights could be learned automatically in the self-attention block. In (18), \( f \) is parameterized with another neural network and \( G \) is the regularized term to adjust the ODE function in generation phrase. The final latent states \( z_t \) are generated with the second ODE net conditioned on \( z_e \):

\[ z_{y0} = \text{GRU}(Y) \] (21)
\[ z_t = \text{ODESolve}(z_{y0}, z_t, h_t, t). \] (22)

For \( t \in (t_0, t_n] \), the solution of \( z_{y_t} \) given the initial value \( z_{y_{t_0}} \) could be computed as

\[ z_{y_t} = z_{y_0} + \int_{t_0}^{t_n} \ln(G_{y_0}(z_X)[f_0(z_{y_s})]) ds. \] (23)

It is tractable to solve the PDE problem using the same techniques as for Neural ODE. In the experiments, we use the existing torchdiffeq package [9] with modifying the computation of the ODE function.

D. Self-Attention Block

The basic self-attention proposed in [37] leverages the scaled dot-product to compute the attention with the tuple input (query, key, value) derived from raw data

\[ \text{Attention}(Q, K, V) = \text{softmax} \left( \frac{QK^T}{\sqrt{d_k}} \right) V \] (24)

where \( d_k \) is the dimension of one query or key. The Softmax operation is used to weights into a probability distribution. Matrices \( Q, K, \) and \( V \) are the hidden representations of raw exogenous variables parameterized with neural networks.

Instead of using one single \( d_{model} \)-dimensional attention module processing the exogenous variables by linearly projecting the queries, keys and values \( h \) times with differently learned linear projections. Multihead attention could allow the model to capture information from different representation subspaces at different positions jointly. The self-attention block has an attractive capability of capturing the intrarelationship in a single sequence and the intercorrelation among different sequences

\[ \text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^Q \]

where \( \text{head}_i = \text{Attention}(QW^Q_i, KW^K_i, VW^V_i) \) (25)

with parameter matrices \( W^Q_i \in \mathbb{R}^{d_{model} \times d_k}, W^K_i \in \mathbb{R}^{d_{model} \times d_k}, W^V_i \in \mathbb{R}^{d_{model} \times d_k}, \) and \( W^Q \in \mathbb{R}^{d_{model} \times d_{model}} \).

E. Loss Function

The objective function is the mean square error (MSE) between the predicted values and the true values

\[ L_{mse} = \frac{1}{L} \sum_{i=1}^{L} \frac{1}{K} \sum_{j=1}^{K} (y_i^j - \hat{y}_i^j)^2 \] (26)

where \( L \) is the number of training samples and \( K \) is the number of predicted future values. The goal in the training stage is to minimize the loss function given the parameter set of our model by using gradient descent methods for optimization, such as the Adam algorithm [45].

F. Complexity Analysis

Assume that the hidden size of the self-attention block is \( d_{model} \) and the number of exogenous variables is \( N \), the attention module processing the exogenous variables has computation complexity \( d_{model}^2 + N \cdot d_{model} \). As for the update process of the ode function, the complexity is \( d_{rnn}^2 \), where \( d_{rnn} \) is the hidden dimension of the ode function. Overall, the computation complexity is \( O(d_{model}^2 + N \cdot d_{model} + d_{rnn}^2) \). In general, though it may take more parameters and computations, the extra overhead does not affect our models application in real scenarios. More importantly, our method could model the impacts of the exogenous variables effectively and lead to significant improvements in performance. On average, our method outperforms the best baseline by reducing 9.85% on RMSE and 13.98% on MAE for arbitrary-step prediction.

F. Complexity Analysis

Assume that the hidden size of the self-attention block is \( d_{model} \) and the number of exogenous variables is \( N \), the attention module processing the exogenous variables has computation complexity \( d_{model}^2 + N \cdot d_{model} \). As for the update process of the ode function, the complexity is \( d_{rnn}^2 \), where \( d_{rnn} \) is the hidden dimension of the ode function. Overall, the computation complexity is \( O(d_{model}^2 + N \cdot d_{model} + d_{rnn}^2) \). In general, though it may take more parameters and computations, the extra overhead does not affect our models application in real scenarios. More importantly, our method could model the impacts of the exogenous variables effectively and lead to significant improvements in performance. On average, our method outperforms the best baseline by reducing 9.85% on RMSE and 13.98% on MAE for arbitrary-step prediction.

Taking the Electricity dataset as one illustrative example, we demonstrate the number of parameters, the training time for one epoch, and the testing time on the Electricity dataset in Table I. The results show that our method has a competitive inference time with a relatively small number of parameters.

| Model          | \# of parameters | train time (s) | test time (s) |
|----------------|-----------------|----------------|---------------|
| LSTNet         | 72,175          | 1.82           | 0.20          |
| Latest ODE-RNN | 62,541          | 43.19          | 2.08          |
| MTGODE         | 70,901          | 55.45          | 2.46          |
| STG-NCDB       | 2,543,301       | 171.49         | 6.19          |
| STGODE         | 610,741         | 24.75          | 0.77          |
| ETN-ODE        | 7,302           | 33.36          | 1.64          |
| EgPDE-Net      | 50,859          | 55.76          | 1.85          |

IV. EXPERIMENTS

In this section, we conduct extensive experiments on four real-world datasets for arbitrary-step and standard multistep predictions of multivariate time series against three continuous-time methods and two noncontinuous methods. The code is available at https://github.com/PengleiGao/EgPDE-Net.

A. Datasets

As summarized in Table II, the four datasets are publicly available. The train/validation/test sets are obtained with a split ratio of 8:1:1 following the previous works [8], [17].

1) SML2010 [46]: It is a public dataset used for indoor temperature forecasting sampled every minute. The room temperature is taken as the target series, and another 13 time series are exogenous variables containing approximately 40 days of monitoring data.
features and unknown graph structures. Then it leverages ODE net to produce multiple future values at arbitrary time tandem attention to encode the raw time series and applies the as the encoder for time series classification and regression. The the RNN internal structure and uses the ODE-RNN framework are both RNNs.

2) Electricity [47]: This is a public dataset for electricity consumption prediction in Homestead, U.S. The consumption is chosen as the target series sampled hourly, while the other 15 time series are exogenous variables containing weather features.

3) Electricity Transformer Temperature (ETT) [38]: ETT is a crucial indicator in electric power deployment. This dataset collected two-year data from two separate counties in China. The data are sampled hourly. According to the two counties, it consists of two separated datasets as (ETTH1, ETTH2). There are 6 power load features, and the oil temperature is selected as the target series.

B. Experimental Settings

For all the datasets, the window size $T$ is chosen as 20 following the baseline methods. Our proposed model EgPDE-Net is implemented in PyTorch with a mini-batch size of 128, and a learning rate of 0.001 for SML2010 and 0.01 for the other datasets. For EgPDE-Net, the hidden size of RNN is chosen from {32, 64, 128}, and the dimension of self-attention is chosen from {15, 30, 45, 60}. The number of heads in self-attention is chosen from {2, 4, 6, 8}. Each method is trained five times to report the average performance for comparison. Two standard evaluation metrics are chosen for the prediction task to evaluate the performance of all the methods.

1) Root Mean Squared Error: $\text{RMSE} = \sqrt{(1/p) \sum_{i=1}^{p} (\hat{y}_i - y_i)^2}$.

2) Mean Absolute Error: $\text{MAE} = (1/p) \sum_{i=1}^{p} |\hat{y}_i - y_i|$.

C. Baselines

Three continuous-time deep learning methods and two noncontinuous methods are chosen for the comparisons.

Latent ODE [9]: It uses a variational autoencoder structure with reversed time inputs. The encoder and decoder networks are both RNNs.

Latent ODE-RNN [10]: This method adds the ODE net into the RNN internal structure and uses the ODE-RNN framework as the encoder for time series classification and regression. The hidden states are generated with the ODE net before updated in the RNN cell.

ETN-ODE [8]: This method leverages tensorized GRU and tandem attention to encode the raw time series and applies the ODE net to produce multiple future values at arbitrary time points.

MTGODE [48]: This method first abstracts multivariate time series into dynamic graphs with time-evolving node features and unknown graph structures. Then it leverages neural ODE to process the graph features with a continuous encoder.

STG-NCDE [49]: This method extends the concept of NCDE and designs two NCDEs: one for temporal processing and the other for spatial processing. The two NCDEs are combined into a single framework for traffic forecasting.

STGODE [50]: This method captures spatial-temporal dynamics through a tensor-based ODE and uses a temporal dilated convolution structure to represent long-term temporal dependencies for traffic forecasting.

IMV-Tensor [17]: It employs a tensorized LSTM to capture different dynamics in multivariate time series and mixture attention to model the generative process of the target series for the next value prediction.

LSTNet [18]: It combines the CNN and RNN to extract short-term local dependency patterns among variables and to discover long-term patterns for time series trends. The aim is to forecast the multistep future value of each individual variable for multivariate time series. We add one linear layer to output all the future values for multistep prediction.

D. Results of Arbitrary-Step Prediction

For the arbitrary-step prediction task, we follow the settings of [8]. The basic idea for arbitrary-step prediction is to forecast multiple future values at arbitrary time steps which are not recorded in the original time series. The output could be arbitrary multiple values between two observations of a fixed sample gap in the test stage. For instance, the electricity consumption data is sampled hourly. Given the historical data, our model could output the future values in the next thirty minutes or the next one and a half hours. We could adjust the integrated time interval to obtain desired future values based on (23). In the experimental parts, since there are no public datasets specifically adapted to forecasting arbitrary future values, we reshape the dataset to half of its original size by taking twice the sampling gap to better and more conveniently demonstrate and verify the effectiveness of arbitrary-step prediction quantitatively. The model only outputs three future values at integral time points sharing the same sample gap as the input data during the training stage, for example, $T + 1$; $T + 2$; $T + 3$. In the testing stage, the model would output two additional future values at continuous steps, for example, $T + 1.5$ and $T + 2.5$, which are not involved during training.

Tables III and IV show the RMSE and MAE of arbitrary-step prediction on the four datasets compared with both continuous and noncontinuous baseline methods, respectively. The tables contain the error of each step with integral time points “Step1,” “Step2,” “Step3,” and continuous-time points “Step1.5” and “Step2.5.” The column “Average” represents the mean error of the five steps. For the IMV-tensor and LSTNet methods, we did not report the results of continuous-time points because of its noncontinuous model limitation. The results demonstrate that our proposed model EgPDE-Net achieves the best performance among continuous and noncontinuous methods. EgPDE-Net obtains the smallest RMSE and MAE on each predicted time step on the four datasets. Latent-ODE and Latent ODE-RNN have relatively large errors on

| Dataset | SML2010 | Electricity | ETTH1 | ETTH2 |
|---------|---------|-------------|-------|-------|
| Distances | 4,137.0 | 22,201.0 | 17,420.0 | 17,420.0 |
| #Features | 13 | 15 | 6 | 6 |
| Sample rate | 1min | 1h | 1h | 1h |
| Train size | 80% | 80% | 80% | 80% |
| Valid size | 10% | 10% | 10% | 10% |
| Test size | 10% | 10% | 10% | 10% |
RMSE and MAE, in which the encoder structure has limitations for capturing the relationship among multivariate time series. Our proposed model EgPDE-Net outperforms the best-base model ETN-ODE by achieving an average decrease of the five-time steps of 16.39%, 4.24%, 11.95%, and 5.73% on RMSE and 24.64%, 8.35%, 16.63%, and 7.04% on MAE for SML2010, ETTh1, ETTh2, and Electricity datasets, respectively. The designed architecture successfully transforms the PDE problem into the ODE problem which could be solved by the ODE black-box solver tractably. The first ODE net captures the interseries correlation among the exogenous variables, which is considered as the regularization term. The second ODE net captures the local information for the target series conditioned on the regularized partial derivatives. Our proposed model EgPDE-Net both utilizes the global relative information among different series and local temporal information in each individual series.

In Figs. 4–7, we visualize the two extra predicted values on-time point $T+1.5$ and $T+2.5$ of the target series on the four datasets. The rectangular regions are enlarged to show the prediction effects of each method more clearly. The red dashed line represents the forecasting values of EgPDE-Net. The original target series of SML2010 and Electricity datasets has a periodic tendency. Figs. 4 and 7 show that both EgPDE-Net and ETN-ODE can capture the periodicity well, but EgPDE-Net shows more accurate prediction bounds. The proposed model EgPDE-Net both utilizes the global relative information among different series and local temporal information in each individual series.

In Figs. 4–7, we visualize the two extra predicted values on-time point $T+1.5$ and $T+2.5$ of the target series on the four datasets. The rectangular regions are enlarged to show the prediction effects of each method more clearly. The red dashed line represents the forecasting values of EgPDE-Net. The original target series of SML2010 and Electricity datasets has a periodic tendency. Figs. 4 and 7 show that both EgPDE-Net and ETN-ODE can capture the periodicity well, but EgPDE-Net shows more accurate prediction bounds. The proposed model EgPDE-Net both utilizes the global relative information among different series and local temporal information in each individual series.
and ETN-ODE fit the target series perfectly. However, our proposed model EgPDE-Net forecasts the target series better in the crests and troughs on SML2010 and Electricity datasets. In Figs. 5 and 6, we could recognize more clearly that the red dashed line is closer to the target series described by the solid blue line, which means the EgPDE-Net model has better-fitting results. For example, the red dashed line successfully captures the stable period on time intervals 30 to 40 in Fig. 5. Furthermore, we visualize the variable contribution in Fig. 8 on SML and Electricity datasets for arbitrary-step prediction. The overall results show that different variable has a different impact on the target series, which is consistent with our life experience. For the SML dataset, variables “Humid.room” and “CO2.diningR” have larger impacts on the target series during training. In our lives, the amount of carbon dioxide will affect the temperature of the room. More carbon dioxide will make the temperature of the room higher. Humidity affects the efficiency of indoor appliances, such as air conditioners. When the air humidity is high, the air conditioner requires more energy to remove moisture from the air, which affects the temperature change in the room. For the electricity dataset, variables “FeelsLike,” “Cloudcover,” and “DewPoint” have larger impacts on the target series “electricity consumption.” In the real world, “FeelsLike” is a comprehensive consideration of humidity, wind speed, temperature and other factors to describe the actual temperature that people feel. When the perceived temperature is higher than the actual temperature, people may use air conditioning more frequently to lower the temperature, resulting in increased electricity consumption. Conversely, when the perceived temperature is lower than the actual temperature, people may use more heating, which also increases electricity consumption. Cloud cover affects...
temperature and sunlight exposure, which indirectly affects household electricity consumption. For example, cloud cover reduces sunlight exposure and can lead to the need for more indoor lighting, which increases electricity consumption. The dew point temperature is the temperature at which air reaches saturation and begins to condense at a certain pressure. High-dew points may prompt people to use dehumidifiers, while low-dew points may prompt people to use humidifiers. Both devices will increase electricity consumption.

### E. Results of Standard Multistep Prediction

In this section, we compare the performance of various methods on three different standard multistep prediction tasks, forecasting the next 1, 5 and 10 future values testifying the ability of predicting short and long-term sequences. Table V shows the RMSE and MAE of all the methods on the four datasets. The best results are displayed in boldface. In most cases, we could observe that the proposed model EgPDE-Net achieves the smallest errors on both RMSE and MAE metrics. It indicates the success of two ODE nets modeling the interseries correlation among exogenous variables and intraseries relationship in the temporal aspect. Specifically, EgPDE-Net has better performance on long-term forecasting tasks than other methods. This result might be contributed to the representation of the correlation of exogenous variables in EgPDE-Net, which could influence the prediction of the target series in the long-term period. Focusing on predicting one target series and modeling the influence of other exogenous variables could improve the performance of multistep prediction. Although LSTNet achieves smaller RMSE and MAE, our method also produces competitive results on task $M = 1$ in the SML2010 dataset. LSTNet aims to forecast the desirable future value ahead of the current timestamp at a specific time point. This model outputs one value at a time, which has an advantage in predicting one-step future value by capturing the local information with CNN structure. It focuses on one-step short-term forecasting without considering the continuous changes among the multiple future values. For long-term forecasting tasks, LSTNet has unsatisfactory performance. This model outputs multiple future values with a linear layer and has limitations in processing accumulative errors without modeling the forecasting incremental information. Our EgPDE-Net has the advantage of forecasting one target series with the input of multivariate time series other than outputting each individual variable. Neither Latent ODE nor Latent ODE-RNN are performing well in multistep prediction, both of which focus more on the reconstruction of time series and temporal relationships. The RNN encoder in Latent ODE brings limitations in processing multivariate time series. The Latent ODE-RNN model adds the ODE net in the RNN basic cell and has advantages in dealing with irregular sampling data. However, this framework performs worse when the predicted target series is affected by other complex exogenous variables. Compared with these graph-based continuous methods equipped with neural ODE, our method achieves promising results and reflects the success of capturing interacting information among exogenous variables with the PDE framework. These graph-based methods all leverage neural ODE as an encoder, and the decoder is normal linear networks, which has limitations for multistep prediction and has unsatisfactory performance in modeling the incremental change with small cumulative errors. Compared with the noncontinuous method IMV-tensor, our proposed method keeps the advantage of predicting short and long term period future values. These results indicate that building continuous networks utilizing the information of exogenous variables could benefit arbitrary-step prediction and improve model performance for standard multistep prediction. We further conduct extra experiments on the forecasting step $M = 20$ to testify the performance of long-term forecasting with several competitive models. The results are shown in Table VI. We can see that our method is also effective and achieves the best performance on most metrics of the four datasets for long-term forecasting.

### F. Ablation Study

In this section, we design two variants of EgPDE-Net to demonstrate the effectiveness and importance of our model components.

1. **w/o Self-Attr**: Replace the self attention component with GRU layer when embedding the exogenous variables.
2. **w/o zx_ode**: Remove the first ODE net and use an LSTM layer to obtain the weight for each forecasting step.

We conduct the ablation experiments on the arbitrary-step prediction task on the four datasets, and the results of RMSE is shown in Table VII. In general, the complete EgPDE-Net achieves the best performance on all the datasets. Removing any component of EgPDE-Net will increase the forecasting...
TABLE V

| Dataset       | SML   | Electricity | ETTH1 | ETTH2 |
|---------------|-------|-------------|-------|-------|
| Metrics       | RMSE  | MAE         | RMSE  | MAE   |
| STGODE        | 0.514 ± 0.031 | 0.364 ± 0.029 | 4.587 ± 0.061 | 3.999 ± 0.079 |
| MTGODE        | 1.146 ± 0.040 | 0.843 ± 0.058 | 4.238 ± 0.110 | 3.799 ± 0.072 |
| STG-NCDE      | 0.423 ± 0.023 | 0.299 ± 0.024 | 4.238 ± 0.110 | 3.799 ± 0.072 |
| ETN-ODE       | 0.508 ± 0.060 | 0.369 ± 0.054 | 4.238 ± 0.110 | 3.799 ± 0.072 |
| EgPDE-Net     | 0.396 ± 0.038 | 0.261 ± 0.017 | 4.933 ± 0.126 | 3.999 ± 0.079 |

G. Discussion and Future Work

We conduct discussions on several possible topics which may deserve our future exploration. First, for continuous modeling, it could cost more time in the training stage and inference. We will investigate more into the internal structure of ODE net to accelerate training and the theoretical exploration on the precision of weak solutions for PDEs to improve the model performance. Second, in the arbitrary-step prediction, the impact of different resampling sizes on prediction performance could be further investigated. Third, in the design of the encoding network, we consider the RNN which is naturally adaptive to the time series. The encoding network is built as discrete models, which may limit the feature representation with different data types. In the future, we will explore building continuous encoding networks to deal with richer data types. Finally, dealing with informative missingness and partial observations is also one important research topic and we will consider exploring this systematically in the future.
Moreover, similar to [27], our defined PDE problem can be assumed to find a solution to transform the original discrete series into a continuous series and make predictions by locating the desired time point in the continuous space, which contains the changing information, including abrupt and slow change. The temporal change information can be implicitly learned in our framework. This could be equated to a hypothesis testing problem of continuous modeling and discrete modeling. We will leave the exploration of this topic in our future work.

V. CONCLUSION

In this work, we proposed the EgPDE-Net, which aims to solve the PDE modeling problem in multivariate time series analysis. We developed a neural network to estimate the partial derivative and considered it as a regularized term to guide the generation trajectory of the specific target series. The two ODE networks applied in this framework take advantage of capturing the intraseries temporal patterns and the interseries correlations jointly among the target series and the exogenous variables. Focusing on the specific target series prediction with multivariate input could take advantage of the influence of the exogenous variables. Experiments on four real-world datasets demonstrated improvements over the baseline methods.

REFERENCES

[1] D. Chen, L. Chen, Y. Zhang, B. Wen, and C. Yang, “A multiscale interactive recurrent network for time-series forecasting,” IEEE Trans. Cybern., vol. 52, no. 9, pp. 8793–8803, Sep. 2022.

[2] J. Wang, T. Sun, B. Liu, Y. Cao, and H. Zhu, “CLVSA: A convolutional LSTM based variational sequence-to-sequence model with attention for predicting trends of financial markets,” in Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., (KDD), 2018, pp. 1387–1395.

[3] X. Xu and M. Yoneda, “Multitask air-quality prediction based on LSTM-autoencoder model,” IEEE Trans. Cybern., vol. 51, no. 5, pp. 2577–2586, May 2021.

[4] A. W. Mulyadi, E. Jun, and H.-I. Suk, “Uncertainty-aware variational-recurrent imputation network for clinical time series,” IEEE Trans. Cybern., vol. 23, no. 2, pp. 9684–9694, Sep. 2022.

[5] Z. Xu, Y. Kang, Y. Cao, and Z. Li, “Spatiotemporal graph convolution multifusion network for urban vehicle emission prediction,” IEEE Trans. Neural Netw. Learn. Syst., vol. 32, no. 8, pp. 3342–3354, Aug. 2021.

[6] R. Yu, S. Zheng, A. Anandkumar, and Y. Yue, “Long-term forecasting using higher order tensor RNNS,” 2019, arXiv:1711.00073.

[7] I. Fox, L. Ang, M. Jaiswal, R. Pop-Busui, and J. Wiens, “Deep multi-output forecasting: Learning to accurately predict blood glucose trajectories,” in Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., (KDD), 2018, pp. 1387–1395.
Z. Long, Y. Lu, and B. Dong, “PDE-Net 2.0: Learning PDEs from data with a numeric-symbolic hybrid deep network,” J. Comput. Phys., vol. 399, Dec. 2019, Art. no. 108925.

Y. Luo, C. Xu, Y. Liu, W. Liu, S. Zheng, and J. Bian, “Learning differential operators for interpretable time series modeling,” in Proc. 28th ACM SIGKDD Conf. Knowl. Discov. Data Min., 2022, pp. 1192–1201.

E. Choi, M. T. Bahadori, J. Sun, J. Kulas, A. Schuetz, and W. Stewart, “REIN: An interpretable predictive model for healthcare using reverse time attention mechanism,” in Proc. 30th Adv. Neural Inf. Process. Syst., 2016, pp. 3504–3512.

A. Vaswani et al., “Attention is all you need,” in Proc. 31st Adv. Neural Inf. Process. Syst., 2017, pp. 5998–6008.

H. Zhou et al., “Informer: Beyond efficient transformer for long sequence time-series forecasting,” 2021, arXiv:2012.07436.

R. O. Wells, Elliptic Operator Theory, New York, NY, USA: Springer, 2008, pp. 108–153.

G. Bao, X. Ye, Y. Zang, and H. Zhou, “Numerical solution of inverse problems by weak adversarial networks,” Inverse Problems, vol. 36, no. 11, 2020, Art. no. 115003.

C. Beck, S. Becker, P. Cheridito, A. Jentzen, and A. Neufeld, “Deep splitting method for parabolic PDEs,” SIAM J. Sci. Comput., vol. 43, no. 5, pp. A3135–A3154, 2021.

J. Sirigniano and K. Spiliopoulos, “DGM: A deep learning algorithm for solving partial differential equations,” J. Comput. Phys., vol. 375, pp. 1339–1364, Dec. 2018.

F. Chen, J. Huang, C. Wang, and H. Yang, “Friedrichs learning: Weak solutions of partial differential equations via deep learning,” 2022, arXiv:2012.08023.

E. Weinan, J. Han, and A. Jentzen, “Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations,” Commun. Math. Statist., vol. 5, no. 4, pp. 349–380, 2017.

D. P. Kingma and J. L. Ba, “ADAM: A method for stochastic optimization,” in Proc. 3rd Int. Conf. Learn. Represent., 2015, pp. 1–15.

F. Zamora-Martinez, P. Romeo, P. Botella-Rocamora, and J. Pardo, “Online learning of indoor temperature forecasting models towards energy efficiency,” Energy Build., vol. 83, pp. 162–172, Nov. 2014.

A. Jain, “HomeStead(US) electricity consumption.” Accessed: Aug. 2022. [Online]. Available: https://www.kaggle.com/datasets/unajlhe/homestead-electricity-consumption

J. Chen, Y. Zheng, Y. Li, C. Chen, B. Yang, and S. Pan, “Multivariate time series forecasting with dynamic graph neural ODEs,” IEEE Trans. Knowl. Data Eng., vol. 33, no. 9, pp. 1968–1980, Sep. 2023.

K. Choi, H. Choi, J. Hwang, and N. Park, “Graph neural controlled differential equations for traffic forecasting,” in Proc. AAAI Conf. Artif. Intell., 2022, pp. 6367–6374.

Z. Fang, Q. Long, G. Song, and K. Xie, “Spatial-temporal graph ODE networks for traffic flow forecasting,” in Proc. 27th ACM SIGKDD Conf. Knowl. Discov. Data Min., 2021, pp. 364–373.

Penglei Gao received the B.Sc. degree in applied statistics from the Shanghai University of Finance and Economics, Shanghai, China, in 2015, and the M.Sc. degree in financial mathematics from Xi’an Jiaotong–Liverpool University, Suzhou, China, in 2017. He is currently pursuing the Ph.D. degree with the University of Liverpool, Liverpool, U.K. His research interests include time series and deep learning.

Xi Yang received the B.Sc. degree (with Finance) in mathematics, the M.Sc. degree (with Distinction) in computer science, and the Ph.D. degree in engineering from the University of Liverpool, Liverpool, U.K., in 2011, 2013, and 2019, respectively. She is currently an Assistant Professor with Xi’an Jiaotong–Liverpool University, Suzhou, China. Her research interests lie in the field of Bayesian probabilistic approaches to machine learning, pattern recognition, and data mining.

Rui Zhang received the B.Sc. degree (First Class Hons.) in telecommunication engineering from Jilin University, Changchun, China, in 2001, and the Ph.D. degree in computer science and mathematics from the University of Ulster, Coleraine, U.K., in 2007. After finishing Ph.D. degree, she was a Research Associate with the University of Bradford, Bradford, U.K., and the University of Bristol, Bristol, U.K., for 5 years. She was with Xi’an Jiaotong–Liverpool University, Suzhou, China, in 2012, and currently an Associate Professor. Her research interests include machine learning, data mining, and statistical analysis.

Ping Guo (Senior Member, IEEE) received the M.Sc. degree in optics from the Department of Physics, Peking University, Beijing, China, in 1983, and the Ph.D. degree from the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, in 2001. He is a Professor with Beijing Normal University, Beijing. He holds six patents and has published more than 300 papers, and two books: Computational Intelligence in Software Reliability Engineering and Image Semantic Analysis. His research interests include computational intelligence theory and its applications in pattern recognition, image processing, software reliability engineering, and astronomical data processing.

Prof. Guo received the 2012 Beijing Municipal Government Award of Science and Technology (Third Rank) entitled “Regularisation Method and its Application.” He is a CCF Senior Member and the Chair of IEEE CIS Beijing Chapter from 2015 to 2016.

John Y. Goulermas received the B.Sc. degree (First Class) in computation from The University of Manchester (UMIST), Manchester, U.K., in 1994, and the M.Sc. and Ph.D. degrees from the Control Systems Center, UMIST, in 1996 and 2000, respectively. He was a Professor with the Department of Computer Science, University of Liverpool, Liverpool, U.K. He had worked with various applications, including image analysis, biomedical engineering, industrial monitoring, and security. His research interests included mathematical modeling, data analytics, and machine learning.

Kaizhu Huang (Senior Member, IEEE) received the Ph.D. degree from The Chinese University of Hong Kong (CUHK), Hong Kong, in 2004. He is currently a Full Professor with Data Science Research Center, Duke Kunshan University, Suzhou. He holds six patents and has published more than 300 papers, and two books: Computational and Applied Technology. He was with Fujitsu Research Centre, CUHK; University of Bristol, Bristol, U.K.; and the National Laboratory of Pattern Recognition, Chinese Academy of Sciences, Beijing, China, from 2004 to 2012. His research interests include machine learning, neural information processing, and pattern recognition. Prof. Huang was the recipient of the 2011 Asia Pacific Neural Network Society Young Researcher Award. He received the Runner-Up Best Paper or Book Award eight times. He serves as an Associated Editor/Advisory Board Member for eight journals and book series. He was invited as a Keynote Speaker in more than 40 international conferences or workshops.