IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON COLLABORATION FOR INTERNET OF THINGS

Received February 12, 2019, accepted February 27, 2019, date of publication March 4, 2019, date of current version March 26, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2902865

Modeling IoT Equipment With Graph Neural Networks

WEISHAN ZHANG[1], YAFEI ZHANG[1], LIANG XU[2,3], JIEHAN ZHOU[4], YAN LIU[5], MU GU[6,7], XIN LIU[1], AND SU YANG[8]

[1]School of Computer and Communication Engineering, China University of Petroleum, Qingdao 266580, China
[2]School of Computer and Communication Engineering, Beijing University of Science and Technology, Beijing 100083, China
[3]Qingdao Deep Intelligence Information Technology Co., Ltd., Qingdao 266200, China
[4]Faculty of Information Technology and Electrical Engineering, University of Oulu, 90014 Oulu, Finland
[5]Faculty of Engineering and Computer Science, Concordia University, Montreal, QC H3G 1M8, Canada
[6]Beijing Aerospace Smart Manufacturing Technology Development Co., Ltd., Beijing 100854, China
[7]School of Mechatronics Engineering, Harbin Institute of Technology, Harbin 150006, China
[8]College of Computer Science and Technology, Fudan University, Shanghai 200433, China

Corresponding authors: Weishan Zhang (zhangws@upc.edu.cn), Jiehan Zhou (jiehan.zhou@oulu.fi), and Xin Liu (lx@upc.edu.cn)

This work was supported in part by the Key Research Program of Shandong Province under Grant 2017GGX10140, in part by the Fundamental Research Funds for the Central Universities under Grant 2015020031, and in part by the National Natural Science Foundation of China under Grant 61309024.

ABSTRACT Traditional neural networks usually concentrate on temporal data in system simulation, and lack of capabilities to reason inner logic relations between different dimensions of data collected from embedded sensors. This paper proposes a graph neural network-based modeling approach for IoT equipment (called GNNM-IoT), which considers both temporal and inner logic relations of data, in which vertices denote sensor data and edges denote relationships between vertices. The GNNM-IoT model's relationships between sensors with neural networks to produce nonlinear complex relationships. We have evaluated the GNNM-IoT using air-conditioner data from a world leading IoT company, which demonstrates that it is effective and outperforms ARIMA and LSTM methods.

INDEX TERMS Graph neural networks, deep learning, simulation, time series prediction, IoT.

I. INTRODUCTION

Modern Internet of Things (IoT) equipment can be complex. Various sensors are usually embedded in the IoT equipment. For example, a central air conditioner is deployed with a number of pressure, temperature, voltage, and other sensors. However, for many IoT applications, effective data are not usually available, e.g., the fault data on central air conditioning equipment, and this hinders equipment fault diagnosis and prediction.

The above issues motivate studies on approaches for simulating the equipment operation. Wu et al. applied neural networks into optimizing simulation model performance [1]. Kim et al. presented a supervised learning method to learn relationships between pilot assignment and user's location patterns [2]. Taki *et al.* [3] used artificial neural networks to estimate greenhouse parameters. These classical neural networks simply use the approximation capability for a complex function from a neural network, without much consideration of inner logic relations between IoT sensor data [4].

In recent years, deep learning has been applied in different fields including system simulation. Yeo and Melnyk [5] proposed a deep learning algorithm for data-driven simulation of noisy dynamical system to predict probability distribution and simulate a stochastic process. Wang *et al.* [6] designed a stacked auto-encoder to properly extract nonlinear and non-stationary features in smart grids. The performance of deep learning can be improved by expert knowledge, as exemplified by the well-known attention mechanism [7]. These existing deep learning based approaches still did not model inner logic relations of IoT sensors.

In summary, all these reviewed efforts are focusing on abstracting features layer by layer in a neural network, without much consideration of associations between features. As a result, as the relations of units in the same layer are barely disposed in these neural networks.

From another aspect, IoT equipment can be described with an undirected graph, in which a vertex denotes sensor data while an edge denotes relations between data. The graph

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

neural networks (GNNs) [8], inherited from the graph, are good tools for graph based tasks. The representational power of GNNs has been studied theoretically [9], which improves the reasoning on logic relations between different objects [10]. Schlichtkrul *et al.* [11] introduced an unsupervised model, named neural relational inference (NRI) model for inferring the interactions of particles. These works motivate us to use GNNs for modeling and simulating the IoT equipment operation, in terms of predicting equipment states based on the internal relationships between different embedded sensors effectively.

This paper proposes a graph neural network approach for modeling IoT systems (GNNM-IoT), which introduces the encoder-decoder pattern, where the encoder learns the potential relationships between sensor data, and the decoder predicts the system states. The accuracy of the learned relationships is assessed by the predicted data quality. The evaluation shows that the GNNM-IoT is effective. It has better performance than the Long-Short Term Memory (LSTM) [12] and Auto-Regressive Integrated Moving Average (ARIMA) [13] on the air conditioner time series data.

The contributions of this paper include:

- Modeling relationships between embedded sensors in IoT equipment using graph neural networks. The relationships between the sensors are denoted by the graph edges, which are modeled by multi-layer full-connection neural networks.
- Reconstructing the input data based on the Variational Autoencoder [14]. An encoder is designed to learn the relations between the vertices. Then a decoder is used to reconstruct the input data based on the vertices at first moment and the relational functions. Meanwhile, we introduce the Gumbel-sampling to boost the performance of GNNM-IoT.
- Introducing the residual structure [15] to learn the differences of sensor states at different moments, which is beneficial for the graph neural network to concentrate on state changes. This can further improve the accuracy of GNNM-IoT.

The remainder of this paper is organized as follows: Section 2 reviews the work. Section 3 presents the design of the GNNM-IoT model, especially its network structure and data processing. Section 4 evaluates the GNNM-IoT model with practical data from air-conditioner for its performance and compares it with the LSTM and ARIMA models. Section 5 concludes the paper and discusses future work to do.

II. RELATED WORK

A lot of efforts have been dedicated on using association rule [16], [17] to discover association relationships between data. However, these solutions only identify there exists a certain relationship without mining how these relationships guide the running of a system, and have very weak capabilities of predicting system operational conditions. There are works on using deep learning approaches to predict system statuses, such as LSTM based IoT status prediction [18] or DBN (Deep Belief Network) based approach [19]. However, these works fail to address the inner logic relations of data, which limits the complete simulation of IoT equipment.

System simulation consists of processes of system modeling, simulation modeling and simulation experiments [20]–[23]. There are three methodologies for system simulation, including statistical methods [24], physical modeling [25], and soft-computing based approaches [26]. They require a pre-knowledge of the dynamical system [27]. Jaeger & Haas [28] proposed an echo state network (ESN), which randomly generated and predicted states and was used in some dynamic systems [29], [30]. However, it is often the case that we do not quite know the principles behind some physical processes, or the system might be too complex to use a classical simulation model [31].

Deep learning has the power of fitting both linear and nonlinear functions [32]–[34], and it has been used for modeling complex structure [35], and for data-driven reconstruction of dynamic systems [36]–[38]. Jin *et al.* [39] proposed a Deep Reconstruction Model (DRM) that combined the deep learning and Elman neural network for manifesting the memory effect of nonlinear systems. Wu and Rahman [40] studied the optimized machine learning framework for modeling the water distribution network management by DBNs. Taki *et al.* [3] applied artificial intelligence (MLP, RBF and SVM models with k-fold cross-validation) to control climate conditions as well as energy consumption for greenhouse simulation. Wang *et al.* [41] introduced a Stack Auto-Encoder (SAE) to narrow down the width of state variables for wind power forecasting. Yeo and Melnyk [5] presented a RNN-based model without any assumption to directly predict probability density for simulation of noisy nonlinear dynamic systems.

The previous studies have confirmed the effectiveness of neural networks for modeling nonlinear problems. However, we still face some challenges in reasoning potential relationships between data [42]. To address this, we apply GNNs to improve the current neural networks' reasoning capabilities.

III. GNNM-IoT MODEL DESIGN

Our goal is to model the relationships between sensor data and simulate the running of the IoT equipment. The first question is how to design the model. Our solution is to use an encoder to learn the relationships between data. Undirected graph could be one of the best tools for representing these relationships. Therefore, we introduce graph neural networks to learn the relationships between data to build a relational model. The second question is how to evaluate the learning. Since we only have observational data, the relationships between the data are hidden behind the observation. Therefore, the accuracy of the learned relationships should be evaluated by predicting the states of IoT equipment. We connect a decoder to the encoder as a prediction model, which is also realized as GNNs. In this way, the observed data can be directly used to evaluate the relationships learned by

the encoder. The more accurate the prediction is, the more accurate the learned relationships are.

The existing neural networks (e.g. LSTM and DBN), as black box solutions, lack of a structure to represent relations between data features, and just simply believe that a neural network was able to learn everything by itself. Our proposed GNNM-IoT uses GNNs to restrict the encoder network structure, and guide neural networks to reach the minimum loss value by the GNN network structure. This design leads to a better performance than the existing neural networks.

Figure 1 presents the GNNM-IoT model with an Encoder-Decoder pattern, which inputs the preprocessed data into a Multi-Layer Perception (MLP)-Encoder. The potential relationships learned by the MLP-Encoder and the sampling data from Gumbel-Sampling are input to the MLP-Decoder. Then the MLP-Decoder reconstructs the output data. The GNNM-IoT training depends on the design of a loss function, which is used to calculate deviation between the truth and the output data. The calculation is based on MSE (mean squared error) and KL (Kullback-Leibler) divergence.

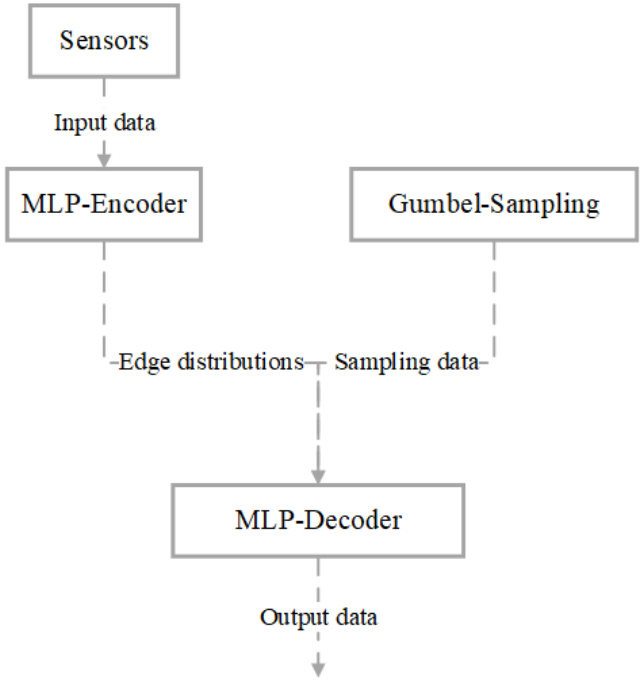


FIGURE 1. GNNM-IoT model overview.

A. SENSORS

Sensors are widely used in IoT equipment. The sensed data are related to the historical state of the IoT equipment. Meanwhile, these data are also related to the system parameters configuration. These data will be measured by the entropy function as defined in (1), where c is the threshold. If $H(x)$ is lower than c, the corresponding x will not be used as the input data.

$$H(x) = -\sum_{x \in X} p(x) \log(p(x)) \tag{1}$$

$$H(x) > c \tag{2}$$

B. MLP-ENCODER

We assume that sensors have relationships with each other. Therefore, an IoT system can be represented by a fully connected graph $G = (N, E)$ as shown in Fig. 2. $N = \{n_1, n_2, n_3, \ldots n_n\}$ denotes n sensors, which represents the states of the corresponding equipment components. The MLP-Encoder is designed with objectives to learn the relational functions of directions in the graph.

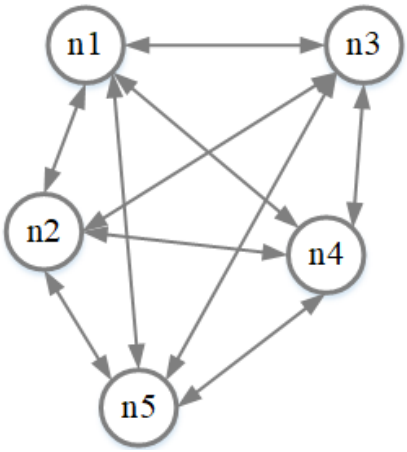


FIGURE 2. Graph G representing sensors and their relationships.

The GNNM-IoT MLP-Encoder uses message passing [43] method, which consists of two phases, message passing and readout defined as Equation (3) and (4). This method updates graph from state S_{i-1} to state S_i. Each node n_i is assigned to a multi-layer fully-connected neural network. Each node n_i in S_{i-1} is connected to other nodes in S_i, as shown in Fig. 3.

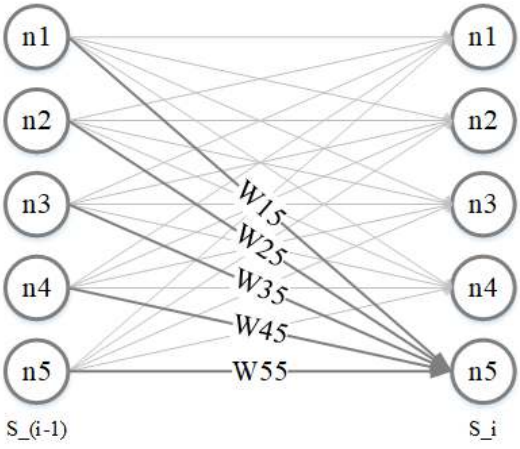


FIGURE 3. Layer assignment of the graph.

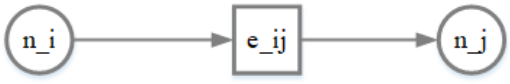


FIGURE 4. Node - neural network block - node.

Figure 4 presents a neural network block for representing complex relationships between nodes, where n_i denotes

i^{th} node in the previous layer, n_j denotes j^{th} node in the next layer, e_{ij} denotes the edge between n_i and n_j.

Therefore, given a graph $G = (N, E)$, N denotes a set of nodes, $n \in N$, E represents a set of e, $e \in E$, $N = \{n_1, n_2, n_3, \ldots n_n\}$, $E = \{e_{ij}, i, j = 1, 2, \ldots n\}$. We use the message passing [43] to transmit the information from a node to an edge (using Equation (3)), then the information is input to the node in the next layer (using Equation (4)).

$$n \to e: \; h^l_{(i,j)} = f^l_e([h^l_i, h^l_j, x_{(i,j)}]) \tag{3}$$

$$e \to n: \; h^{l+1}_j = f^l_n([\sum_{i \in N_j} h^l_{(i,j)}, x_j]) \tag{4}$$

where h^l_i denotes the embedding of n_i, l represent the current layer, $h^l_{(i,j)}$ stands for the embedding of e_{ij}, f denotes a function. Here the potential relationships are learned by three mappings: the first mapping from n to e, the second mapping form e to n, and the third mapping from n to e.

C. GUMBEL-SAMPLING

A normal distribution is applied to the Variational auto-encoder (VAE) [14] to limit the distribution of the mean vector and the variance vector. Similar to VAE, the GNNM-IoT introduces the Gumbel re-parameter trick [44] to optimize model performance, so that the potential relationships are Gumbel distributions defined as Equation (5). Thus, the purpose of the MLP-Encoder is to learn the parameters of a Gumbel distribution, then to conduct sampling from the Gumbel distribution. For example, we can get the normal Gumbel distribution sampling data by sampling the discrete stochastic variable x_π in the probability vector π with m dimensions of Gumbel noise.

$$p(x) = \frac{1}{\beta} e^{-z-e^{-z}}, \quad z = \frac{x-\mu}{\beta} \tag{5}$$

G_i is the stochastic variable of the Gumbel distributions that are independently identically distributed. G_i can be generated from uniform distribution by the Gumbel distribution inverse: $G_i = -\log(-\log(U_i))$, $U_i \sim U(0, 1)$.

As shown in Fig. 5, re-parameterization moves the sampling steps away from the computation graph for gradient backpropagation. Here, f, z, x, ϕ are the deterministic nodes, and ε is the stochastic node. If z includes ε, it is impossible for z to back propagate the gradient. So ε is moved out of z, which is regarded as an input without the weight variable. Then z becomes a deterministic node. The stochastic distribution represented by ε can be added to the forward propagation but ε is not changed by backpropagation. ε obeys the Gumbel distribution in this paper.

D. MLP-DECODER

The GNNM-IoT MLP-Decoder predicts system states based on the previous system states and the relational functions learned by the MLP-Encoder using Equations (6) and (7). In the actual environment, if the time interval is short, a single-step prediction would be useless for users. Therefore,

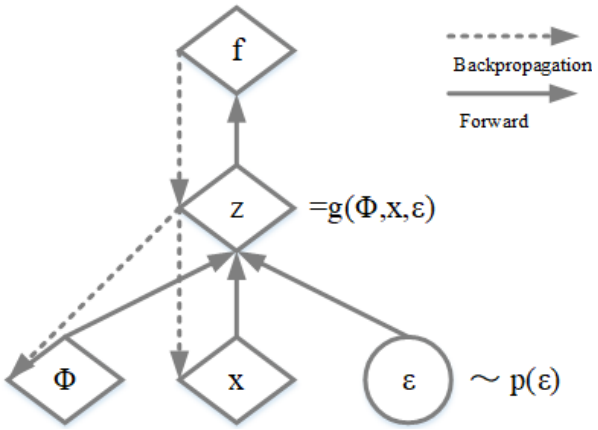


FIGURE 5. Re-parameterization.

a multi-step prediction is applied to accumulate the prediction error for backpropagation and to update the parameters in the GNNM-IoT model in training phase.

$$n \to e: \tilde{h}^t_{(i,j)} = \sum_k z_{ij,k} \tilde{f}^k_e([x^t_i, x^t_j]) \tag{6}$$

$$e \to n: \mu^{t+1}_j = x^t_j + \tilde{f}_n(\sum_{i \neq j)} \tilde{h}^t_{(i,j)}) \tag{7}$$

where z denotes the latent ground truth graph, $z_{ij,k}$ denotes k^{th} element in the vector z_{ij}, x^t_i denotes the state of x_i in time t, $\tilde{f}^k_e$ denotes the mapping function between the nodes' state in time t and the edge $\tilde{h}^t_{(i,j)}$, $\tilde{f}_n$ transfers the edges $\tilde{h}^t_{(i,j)}$ into the nodes μ^{t+1}_j. x^t_j is to make the MLP-Decoder learn the differences between the previous and the next system states.

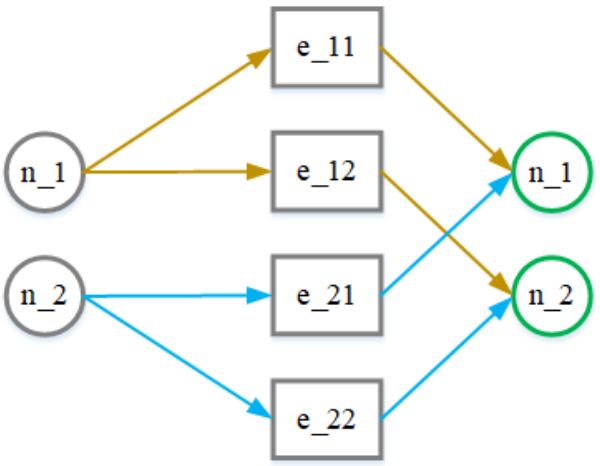


FIGURE 6. Single-step prediction.

Fig. 6 presents the single-step prediction, $[e_{11}, e_{12}, e_{21}, e_{22}]$ are the relational functions learned by the MLP-Encoder. The initial state of each sensor is n_0. The target of MLP-Decoder is to predict the next state n_i using a single-step prediction. A long time prediction is completed by stacking multiple

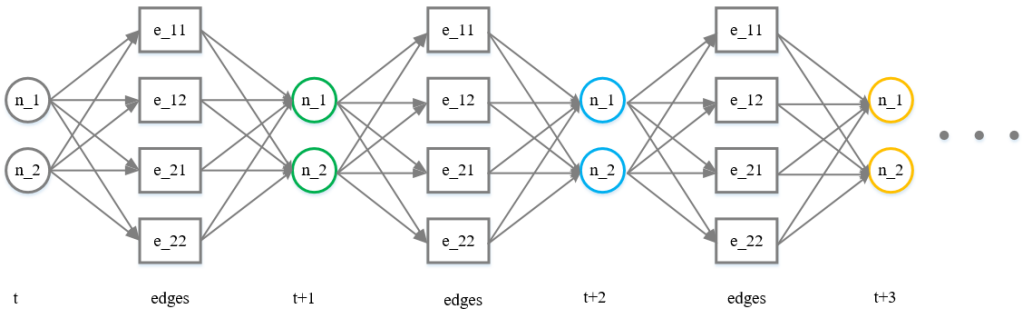


FIGURE 7. Multi-step prediction.

single-step predictions, so that the previous prediction result is the input for next step prediction, as shown in Fig. 7.

The multi-step prediction can predict the system state series. When the MLP-Decoder predicts the system states to follow based on the first moment system states in every sample, the multi-step prediction accumulates prediction errors, which is the basement of loss function. Then the error is used to update weights in the GNNM-IoT model by gradient backpropagation.

E. LOSS FUNCTION

The GNNM-IoT model defines the loss function as Equation (8) based on KL divergence of the relational functions and the MSE between the prediction and ground truth, where c denotes the proportion of KL divergence, e denotes the relations learning by the encoder, x denotes the input data, and y denotes the prediction result.

$$Loss = cKL(e) + (1 - c)\|x - y\|^2 \tag{8}$$

IV. EVALUATION

The GNNM-IoT implements the MLP-Encoder and MLP-Decoder with fully-connected networks and applies the Adam algorithm [45] as the optimization algorithm. The experiment uses Ubuntu 16.04 and PyTorch 0.3 [46] as the software platform and a GTX1070 GPU for training neural networks.

A. AIR-CONDITIONER SIMULATION

We are working with a world leading IoT company to conduct the analysis and prediction of their central air conditioning systems using our developed GNNM-IoT model.

The experiment first removes those data whose entropy is lower than the given threshold. The input data are then cleaned, filled and normalized by data preprocessing. The experiment chooses 55 data dimensions from the original 182 dimensions.

The experiment describes data matrix as (t,55), in which t denotes the time length. Due to the features of GNN and the sampling frequency of sensors (twice a minute), there is a requirement for data prediction of half an hour in advance. Therefore, we transfer the data matrix into the matrix (m,55,60), in which m is the number of samples. Then the data are given to the GNNM-IoT, ARIMA and LSTM for training and predicting future states.

Figure 8 illustrates the results with the comparison between the LSTM and GNNM-IoT loss curves. This shows that the LSTM loss reaches the lowest value of 0.04787 after 70000 iterations, whereas the GNNM-IoT loss reaches the lowest value of 0.00605 after 500 iterations. The accuracy of the GNNM-IoT is seven times higher than that of LSTM, whereas the iterations by the GNNM-IoT are only 1/400 by the LSTM. In addition, the GNNM-IoT convergent speed is faster than that of LSTM. And the GNNM-IoT loss curve has two obvious jumps during the iterations. Then we use the MSE loss as the standard to estimate the accuracy of the model because the MSE loss is computed based on the similarity between the prediction data and the ground truth data.

TABLE 1. Predictive accuracy statistics of the LSTM and the GNNM-IoT measured with MSE.

Algorithm	Train	Validation
LSTM	0.0478699133	0.0952380896
GNNM-IoT	0.0060533981	0.0006080393

Table 1 shows the prediction accuracy of the LSTM and GNNM-IoT measured with MSE based on the historical data. Comparing with the LSTM, the GNNM-IoT MSE is only 12% of the LSTM MSE loss during the training phase. The GNNM-IoT MSE is far less than that of LSTM during the validation phase.

We predict a longer time series and add the ARIMA as the third model for a comparison. The results are shown in Fig. 9 and 10.

The yellow, green and purple curves denote the prediction data in Figure 9. The blue curve denotes the ground truth data. The LSTM and the GNNM-IoT use the same data and training mode. In Figure 9, The 60 points represent a 30 minutes period, as the sampling frequency of the sensors is twice a minute. We use single-step prediction to reconstruct the input data. When the models predict 30 minutes' data,

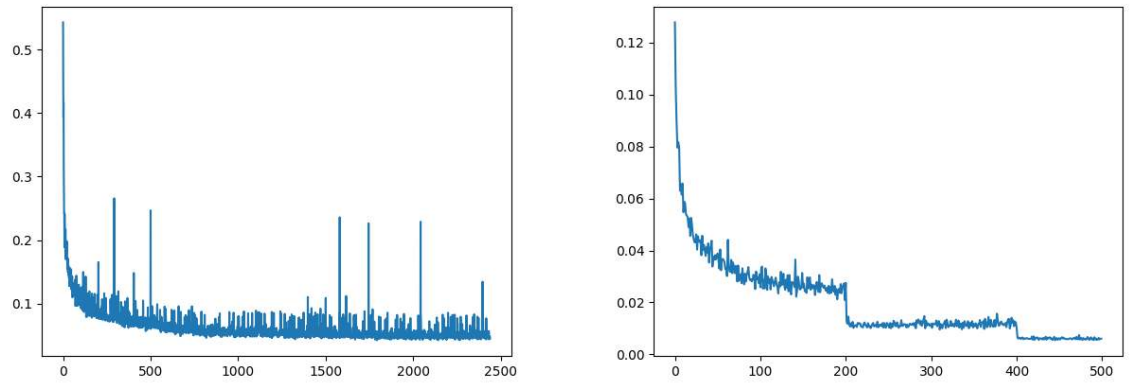


FIGURE 8. LSTM loss curve (left) and GNNM-IoT loss curve (right). (a) LSTM MSE loss. (b) GNNM-IoT MSE loss.

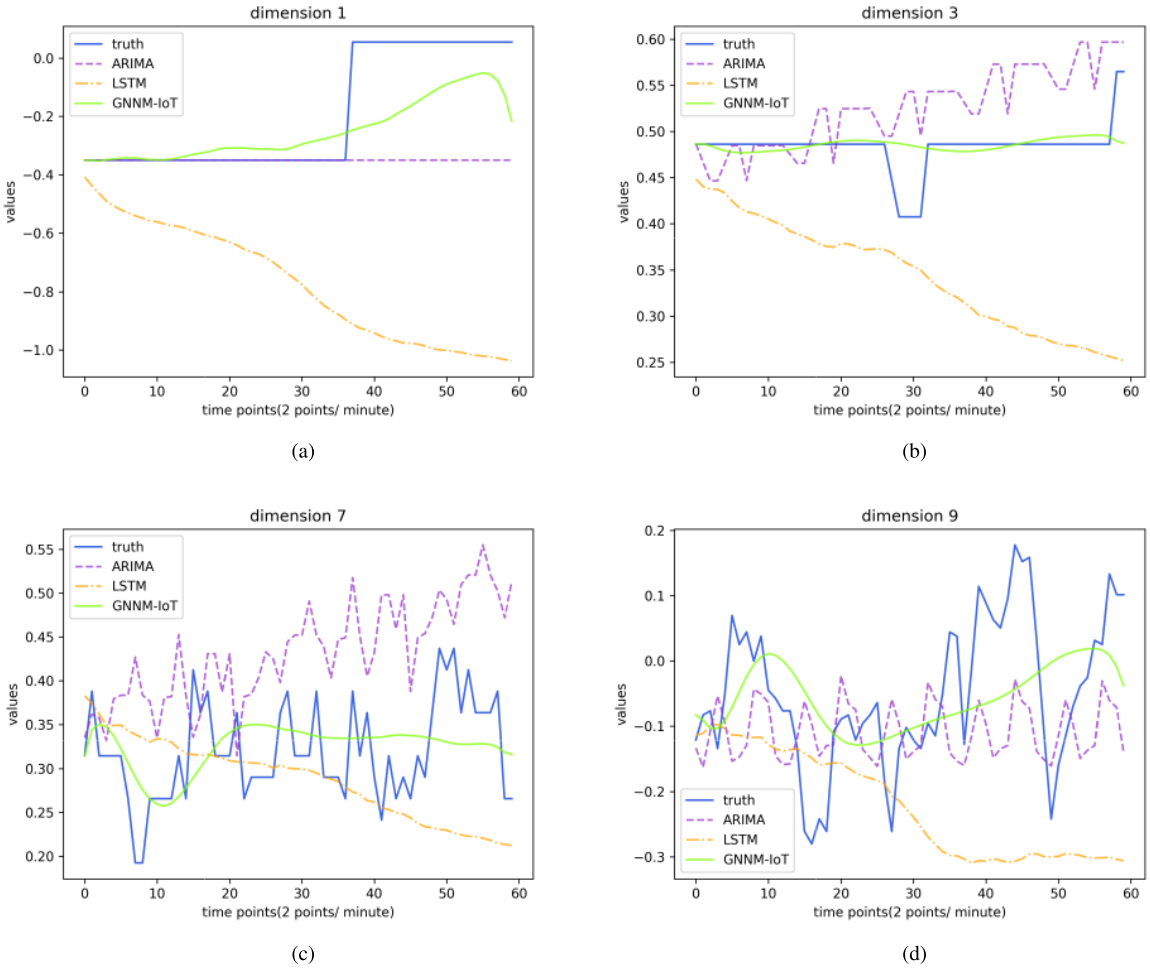
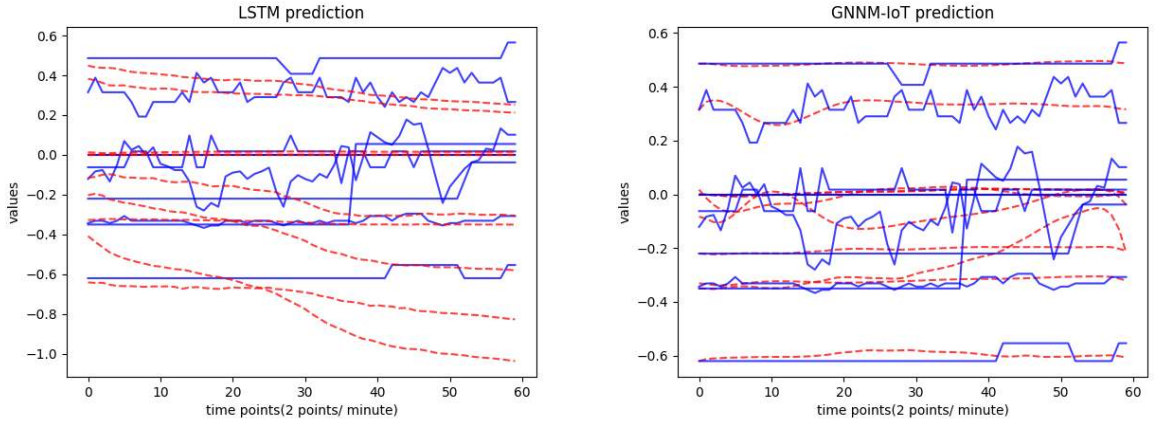


FIGURE 9. 30 minutes prediction using ARIMA, LSTM and GNNM-IoT. (a) dimension1. (b) dimension3. (c) dimension7. (d) dimension9.

the second point is predicted based on the first prediction point, until it reaches the final point. The results in Fig. 9 shows that the GNNM-IoT has the best performance in terms of prediction accuracy. We can see that from the start of 20 points, the GNNM-IoT keeps the trends of data whereas the ARIMA and LSTM cannot.



(a) (b)

FIGURE 10. Multi-dimension prediction. (a) LSTM. (b) GNNM-IoT.

The red curve denotes the prediction data in Fig. 10. The blue curve denotes the ground truth data. We can see that the GNNM-IoT can fit every dimensions' data well, which means that the GNNM-IoT has a satisfactory prediction performance. The LSTM demonstrates bad performance for the dimensions with fluctuation, but performs well for the dimensions with a smooth curve.

The experiments above have shown that the GNNM-IoT has obvious advantages in predicting future system states compared with other methods. The GNNM-IoT model described above is implemented by a one-step prediction. However, it also can be implemented by a multi-step prediction. For this reason, the following experiments are conducted to assess the multi-step prediction with the same data and the same GNNM-IoT model but with different prediction steps, namely 5 steps (points), 10 steps (points) and 15 steps (points). All predictions are for 60 points (30 minutes) repeating the experiment 12, 6 or 4 times respectively. It can predict the long term trend of system states based on the decoder, while the prediction module in the decoder is the same MLP used for mapping the base states and the future states.

In the following experiment, the base predictions are 5 steps, 10 steps and 15 steps, respectively. This means that 5 steps need to be repeated 12 times, 10 steps be repeated 6 times and 15 steps be repeated 4 times to predict 60 points. Then we make comparisons between 5 steps, 10 steps and 15 steps for choosing the best prediction steps. The loss descent curves are shown in Fig. 11, 12 and 13, where the results demonstrate the model performance with different prediction steps. We calculate the MSE of different prediction steps, which means that the MSE of 5 steps prediction has an average error of 5 points, and the same applies to the 10 and 15 steps experiments.

We can see that the training loss curves of 5 steps (Fig. 11) and 10 steps (Fig. 12) are convergent, while the training of 15 steps (Fig. 13) prediction is divergent with concussion. Figure 11 and 12 show that there exist two sudden drops,

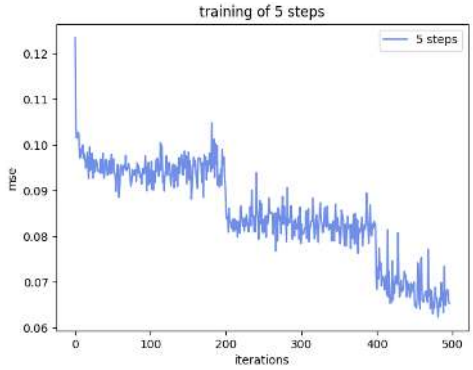


FIGURE 11. MSE loss descent curve of 5 steps.

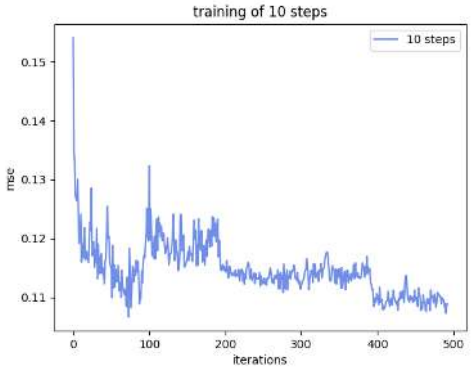


FIGURE 12. MSE loss descent curve of 10 steps.

which means that these models have a rapid descent during training. In addition, the total iterations are fewer than other methods but the time consumption is longer than other methods in a single iteration.

The validation loss descent curves of different steps are shown in Fig. 14, 15 and 16. We can see that the MSE curves of the validation data set are slightly instable in the

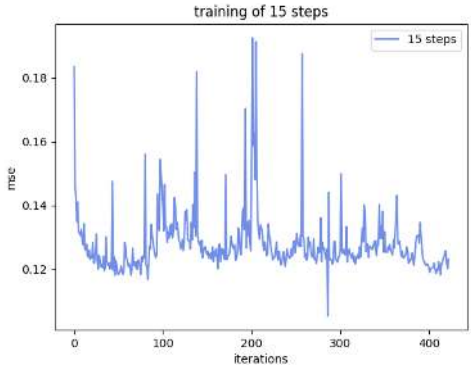


FIGURE 13. MSE loss descent curve of 15 steps.

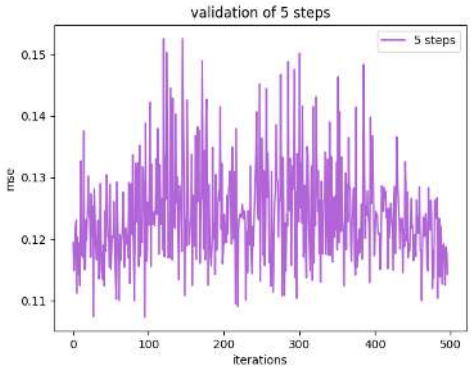


FIGURE 14. validation loss descent curve of 5 steps.

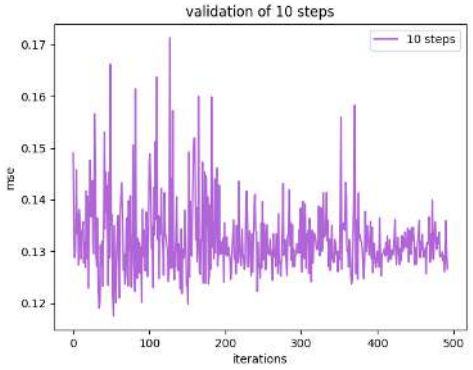


FIGURE 15. validation loss descent curve of 10 steps.

training stage. The fluctuations of validation loss descent curves tend to become less and convergent in the late period. However, the validation curve of 15 steps model consistently fluctuates and does not converge. That means that the long-term prediction performance is not good of the 15 steps prediction. The reason is that for the current model setting, the MLP, mapping the function relationships, doesn't effectively learn the long-term intrinsic mapping. As a result, the model performance based on 15 steps is poor, which eventually leads to the situation where the curve does not converge.

Figure 17 shows the comparison using GNNM-IoT with 1 step, 5 steps, 10 steps and 15 steps and four dimensions

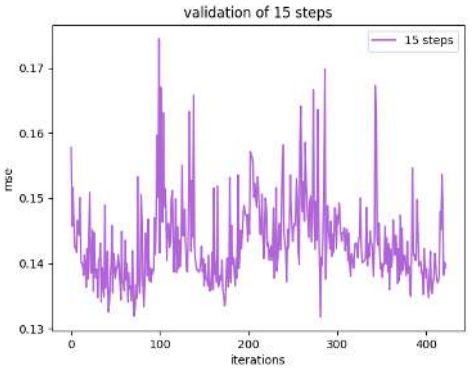


FIGURE 16. validation loss descent curve of 15 steps.

data (1, 3, 7, and 9). We can see that the overall prediction with 1 step is more accurate and can effectively reflect the state trends. The prediction with 5 steps, 10 steps and 15 steps cannot learn trends well. Figure 17 shows that the real data has a sudden rise in the late stage, while the results with 5, 10, and 15 steps are transited to a high point smoothly with no sudden changes and details of data changes are lost.

Table 2 further verifies the above analysis with the MSE data, where the prediction of 1 step has a better performance than those predictions of 5, 10 and 15 steps.

TABLE 2. MSE in 30 minutes prediction with 1 step, 5 steps, 10 steps and 15 steps.

dimension	1 step	5 steps	10 steps	15 steps
dimension1	0.0177443	0.0150794	0.012957	0.0209351
dimension3	0.0006556	0.0099382	0.0007769	0.00082072
dimension7	0.0028725	0.0134935	0.0033587	0.00278901
dimension9	0.0112787	0.0163024	0.0200642	0.0149668

B. DISCUSSION

The fluctuation of real data is fast. From the predictions results with 1, 5, 10, and 15 steps, we can find that the 1 step based model has the best results and can reflect data changes. Multi-step predictions do not perform well in our experimental conditions. Therefore, the prediction step is usually decided based on data characteristics. If data changes quickly, then the prediction step needs to be chosen smaller.

The above evaluations show that GNNM-IoT has a good performance compared with that of the ARIMA and LSTM. This states that GNNM-IoT is good at modeling inner logic relations between system components. The ARIMA model only obtains the data features but the principle of a system. In addition, the long-term prediction performance (longer than 30 minutes) may be bad due to the designed decoder with a single step prediction in this experiment.

Our goal is to model the relationships between data. The encoder is used to learn these relationships, which is the core of GNNM-IoT. The decoder is used to evaluate the accuracy

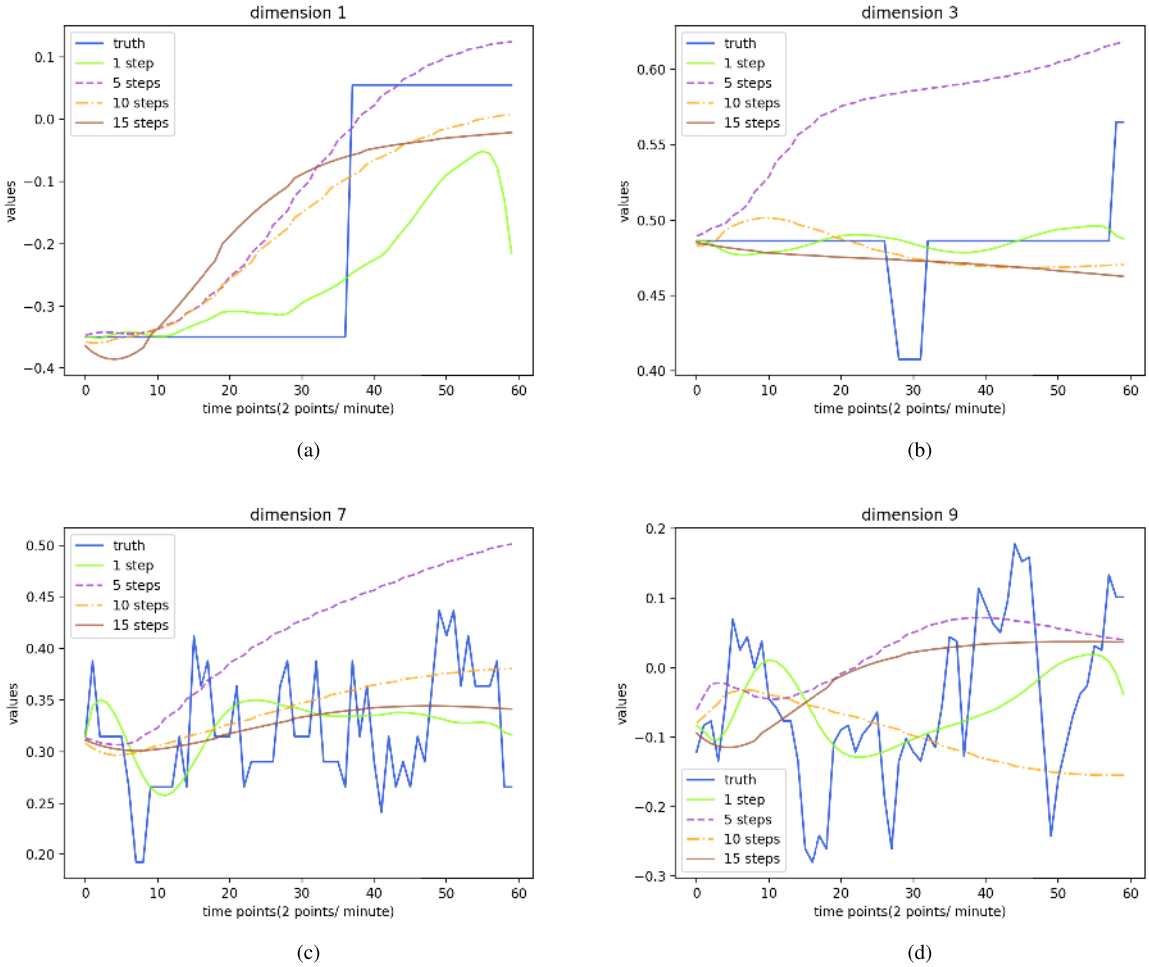


FIGURE 17. 30 minutes prediction using GNNM-IoT with 1 step, 5 steps, 10 steps and 15 steps. (a) dimension1. (b) dimension3. (c) dimension7. (d) dimension9.

of the learned relationships by the encoder, which can be understood as predicting the data of the next moment by using the data relationships of the previous moment. And how to assess the correctness or accuracy of the relationships? Since all we have are the observational data, the relationships between the data are unknown, so the accuracy of the learned relationships are verified in a predictive way. If the prediction is accurate, then the learned relationships are correct. These evaluations show that GNNM-IoT is effective to model the inner relationships.

V. CONCLUSION AND FUTURE WORK

System simulation of industrial IoT equipment needs to consider temporal information, but more importantly it needs to provide the ability of reasoning inner logic relations between dimensions of data. This paper proposes a Graph Neural Network based modeling approach for IoT equipment, called GNNM-IoT, which models relationships between sensors for obtaining relationships by neural networks. We have evaluated the performance of GNNM-IoT with air-conditioner data from a world leading IoT company, and compared with that of ARIMA and LSTM. It shows that our GNNM-IoT produces a higher performance than the other two approaches.

In the experiment involving the GNNM-IoT model, we find that it consumes quite some computing resources when it builds the data relations. With the sensor number increases, resources consumption becomes higher. The future work is to prune the relations based on expert knowledge to further reduce data dimensions in order to boost the processing speed. The second direction is to abstract an advanced vertex from the vertexes with close relationships by introducing a pooling module, in order to further reduce resource consumption.

REFERENCES

[1] S. Wu, X. Liu, Y. Shao, F. Zhang, and M. Yang, "Optimization via simulation based on neural network," *J. Syst. Simul.*, vol. 30, pp. 36–44, Jan. 2018.

[2] K. Kim, J. Lee, and J. Choi, "Deep learning based pilot allocation scheme (DL-PAS) for 5G massive MIMO system," *IEEE Commun. Lett.*, vol. 22, no. 4, pp. 828–831, Apr. 2018.

[3] M. Taki, S. A. Mehdizadeh, A. Rohani, M. Rahnama, and M. Rahmati-Joneidabad, "Applied machine learning in greenhouse simulation; new application and analysis," *Inf. Process. Agricult.*, vol. 5, no. 2, pp. 253–268, 2018.

[4] C. Zhu, V. C. M. Leung, J. J. P. C. Rodrigues, L. Shu, L. Wang, and H. Zhou, "Social sensor cloud: Framework, greenness, issues, and outlook," *IEEE Netw.*, vol. 32, no. 5, pp. 100–105, Sep./Oct. 2018.

[5] K. Yeo and I. Melnyk. (2018). "Deep learning algorithm for data-driven simulation of noisy dynamical system." [Online]. Available: https://arxiv.org/abs/1802.08323

[6] H. Wang *et al.*, "Deep learning-based interval state estimation of AC smart grids against sparse cyber attacks," *IEEE Trans. Ind. Informat.*, vol. 14, no. 11, pp. 4766–4778, Nov. 2018.

[7] V. Mnih, N. Heess, A. Graves, and K. Kavukcuoglu, "Recurrent models of visual attention," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2204–2212.

[8] M. Henaff, J. Bruna, and Y. LeCun. (2015). "Deep convolutional networks on graph-structured data." [Online]. Available: https://arxiv.org/abs/1506.05163

[9] R. Herzig, M. Raboh, G. Chechik, J. Berant, and A. Globerson. (2018). "Mapping images to scene graphs with permutation-invariant structured prediction." [Online]. Available: https://arxiv.org/abs/1802.05451

[10] P. W. Battaglia *et al.* (2018). "Relational inductive biases, deep learning, and graph networks." [Online]. Available: https://arxiv.org/abs/1806.01261

[11] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *Proc. Eur. Semantic Web Conf.* Cham, Switzerland: Springer, 2018, pp. 593–607.

[12] 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.

[13] G. E. P. Box and D. A. Pierce, "Distribution of residual autocorrelations in autoregressive-integrated moving average time series models," *J. Amer. Statist. Assoc.*, vol. 65, no. 332, pp. 1509–1526, Apr. 1970.

[14] D. P. Kingma and M. Welling. (2013). "Auto-encoding variational Bayes." [Online]. Available: https://arxiv.org/abs/1312.6114

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[16] R. Agrawal, T. Imieli ski, and A. N. Swami, "Mining association rules between sets of items in large databases," in *Proc. Int. Conf. Manage. Data*, 1993, vol. 22, no. 2, pp. 207–216.

[17] X. Liu *et al.*, "PARMTRD: Parallel association rules based multiple-topic relationships detection," in *Proc. Web Services (ICWS), 25th Int. Conf., Held Services Conf. Fed. (SCF)*, Seattle, WA, USA, Jun. 2018, pp. 422–443. doi: 10.1007/978-3-319-94289-6_27.

[18] W. Zhang *et al.*, "LSTM-based analysis of industrial IoT equipment," *IEEE Access*, vol. 6, pp. 23551–23560, 2018. doi: 10.1109/ACCESS.2018.2825538.

[19] W. Zhang *et al.*, "Resource requests prediction in the cloud computing environment with a deep belief network," *Softw., Pract. Exper.*, vol. 47, no. 3, pp. 473–488, 2017. doi: 10.1002/spe.2426.

[20] R. Abdelhady, "Modeling and simulation of a micro grid-connected solar PV system," *Water Sci.*, vol. 31, no. 1, pp. 1–10, 2017.

[21] L. Exel and G. Frey, "Modeling and simulation of local flexibilities and their effect to the entire power system," *Comput. Sci.-Res. Develop.*, vol. 33, pp. 49–60, Feb. 2018.

[22] D. Weintrop *et al.*, "Defining computational thinking for mathematics and science classrooms," *J. Sci. Edu. Technol.*, vol. 25, no. 1, pp. 127–147, 2016.

[23] J. Cisonni, A. Van Hirtum, X. Pelorson, and J. Willems, "Theoretical simulation and experimental validation of inverse quasi-one-dimensional steady and unsteady glottal flow models," *J. Acoust. Soc. Amer.*, vol. 124, no. 1, pp. 535–545, 2008.

[24] Y. Chakhchoukh, P. Panciatici, and L. Mili, "Electric load forecasting based on statistical robust methods," *IEEE Trans. Power Syst.*, vol. 26, no. 3, pp. 982–991, Aug. 2011.

[25] L. Li, K. Ota, and M. Dong, "When weather matters: IoT-based electrical load forecasting for smart grid," *IEEE Commun. Mag.*, vol. 55, no. 10, pp. 46–51, Oct. 2017.

[26] K. Zor, O. Timur, and A. Teke, "A state-of-the-art review of artificial intelligence techniques for short-term electric load forecasting," in *Proc. IEEE 6th Int. Youth Conf. Energy (IYCE)*, Jun. 2017, pp. 1–7.

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

[28] H. Jaeger and H. Haas, "Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication," *Science*, vol. 304, no. 5667, pp. 78–80, Apr. 2004.

[29] M. Inubushi and K. Yoshimura, "Reservoir computing beyond memory-nonlinearity trade-off," *Sci. Rep.*, vol. 7, no. 1, 2017, Art. no. 10199.

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

[31] F. Hamilton, T. Berry, and T. Sauer, "Ensemble Kalman filtering without a model," *Phys. Rev. X*, vol. 6, no. 1, Jan. 2016, Art. no. 011021.

[32] D. Wang, M. Wang, and X. Qiao, "Support vector machines regression and modeling of greenhouse environment," *Comput. Electron. Agricult.*, vol. 66, no. 1, pp. 46–52, Apr. 2009.

[33] A. Dariouchy, E. Aassif, K. Lekouch, L. Bouirden, and G. Maze, "Prediction of the intern parameters tomato greenhouse in a semi-arid area using a time-series model of artificial neural networks," *Measurement*, vol. 42, no. 3, pp. 456–463, Apr. 2009.

[34] R. Abdi, M. Taki, and M. Akbarpour, "An analysis of energy input-output and emissions of greenhouse gases from agricultural productions," *Int. J. Natural Eng. Sci.*, vol. 6, no. 3, pp. 73–79, May 2012.

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

[36] A. E. Ruano, E. M. Crispim, E. Z. Conceição, and M. M. J. Lúcio, "Prediction of building's temperature using neural networks models," *Energy Buildings*, vol. 38, no. 6, pp. 682–694, Jun. 2006.

[37] R. ABDI, M. TAKI, and A. JALALI, "Study on energy use pattern, optimization of energy consumption and CO_2 emission for greenhouse tomato production," *Int. J. Natural Eng. Sci.*, vol. 7, no. 1, pp. 42–49, Jan. 2013.

[38] W. Zou, F. Yao, B. Zhang, C. He, and Z. Guan, "Verification and predicting temperature and humidity in a solar greenhouse based on convex bidirectional extreme learning machine algorithm," *Neurocomputing*, vol. 249, pp. 72–85, Aug. 2017.

[39] X. Jin, J. Shao, X. Zhang, W. An, and R. Malekian, "Modeling of nonlinear system based on deep learning framework," *Nonlinear Dyn.*, vol. 84, no. 3, pp. 1327–1340, 2016.

[40] Z. Y. Wu and A. Rahman, "Optimized deep learning framework for water distribution data-driven modeling," *Procedia Eng.*, vol. 186, pp. 261–268, Jan. 2017.

[41] 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.

[42] A. L. Yuille and C. Liu. (2018). "Deep nets: What have they ever done for vision?" [Online]. Available: https://arxiv.org/abs/1805.04025

[43] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. (2017). "Neural message passing for quantum chemistry." [Online]. Available: https://arxiv.org/abs/1704.01212

[44] E. Jang, S. Gu, and B. Poole. (2016). "Categorical reparameterization with gumbel-softmax." [Online]. Available: https://arxiv.org/abs/1611.01144

[45] D. P. Kingma and J. Ba. (2014). "Adam: A method for stochastic optimization." [Online]. Available: https://arxiv.org/abs/1412.6980

[46] A. Paszke *et al.*, "Automatic differentiation in pytorch," in *Proc. 31st Conf. Neural Inf. Process. Syst.*, 2017, pp. 1–4.

WEISHAN ZHANG received the Ph.D. degree from Northwestern Polytechnical University, China, in 2001. He is currently a Full Professor and the Deputy Head for research of the Department of Software Engineering, China University of Petroleum. He has published more than 100 papers. His current H-index according to Google scholar is 17. His current research interests are big data platforms, pervasive cloud computing, and service oriented computing.

YAFEI ZHANG is currently pursuing the master's degree in big data processing, deep learning, and software engineering.

LIANG XU received the master's degree in software engineering from the China University of Petroleum. He is currently pursuing the Ph.D. degree with the Beijing University of Science and Technology. His current research interests include deep learning, software engineering, and big data processing. He has published more than ten papers in these fields.

JIEHAN ZHOU received the Ph.D. degree in computer engineering from the University of Oulu, Finland, and the Ph.D. degree in manufacturing automation from the Huazhong University of Science and Technology, China. He is currently an Associate Professor with the University of Oulu. He has published more than 100 papers. His current research interests include big data platforms, the Internet of Things, and pervasive cloud computing.

YAN LIU was a Senior Research Scientist with the Pacific Northwest National Laboratory, WA, USA, delivering high performance and scalable data analysis platforms for domains of power systems, scientific computing, and engineering simulation. She is currently an Associate Professor with the Faculty of Engineering and Computer Science, Concordia University. He has more than 15 years research experience of developing data intensive algorithms on distributed and parallel computing systems. Her current research interests include parallel and distributed machine learning, and automatically scaling back-end computing resources also by means of machine learning.

MU GU received the master's degree in control engineering from the Graduate School of the Second Aerospace Academy. He is currently pursuing the Ph.D. degree with the Harbin Institute of Technology. He is also a Senior Engineer with Beijing Aerospace Smart Manufacturing Technology Development Co., Ltd. His current research interests include cloud manufacturing, knowledge engineering, and big data platforms. He has published more than ten papers in these fields.

XIN LIU received the Ph.D. degree from Nankai University, China, in 2012. She is currently an Associate Professor with the College of Computer and Communication Engineering, China University of Petroleum, Qingdao, China. Her research interests include social networks, data mining, and big data processing.

SU YANG is currently a Professor with the Department of Computer Science and Engineering, Fudan University. He is also the Principal Investigator of a number of NSFC projects, including graphical symbol recognition in natural scenes and detection of abnormal collective behaviors via movement and communication pattern analysis. His research interests include pattern recognition, social computing, machine vision, and data mining.