A Novel Deep Learning-Based Robust Dual-Rate Dynamic Data Modeling for Quality Prediction

Xiangan Meng, Qiang Liu, *Senior Member, IEEE*, Chao Yang, *Graduate Student Member, IEEE*, Le Zhou, *Member, IEEE*, and Yiu-Ming Cheung, *Fellow, IEEE*

***Abstract*—Traditional data-driven quality prediction methods are mainly built from static models using clean data with a slow sampling rate, leaving the process dynamics unused. To make full use of dynamic process data collected at a fast sampling rate, this article proposes a novel deep learning-based robust dual-rate dynamic data modeling method for quality prediction of dynamic nonlinear processes. A new dynamic data denoising generative adversarial imputation network is first proposed for the missing value imputation among the dynamic process data. Then, a new hint convolutional neural network (HCNN) is established for dual-rate data based quality prediction. The proposed HCNN incorporates the information hint mechanism of channel expansion into the convolutional neural network to extract the dynamic features with definitive time and variable information. Finally, the proposed method is verified using the Dow distillation process dataset and Beijing multisite air quality dataset.**

***Index Terms*—Data-driven quality prediction, dual-rate data modeling, dynamic data denoising generative adversarial imputation network (DDGAIN), dynamic process.**

I. Introduction

QUALITY prediction is essential to product quality improvement, energy saving, and emission reduction of industrial processes. With the wide use of distributed control systems and Industrial Internet, a large number of process data and quality data are collected [1]. Data-driven quality prediction methods that find out the relations between process data and quality data have attracted much attention in the academia and industry in the last decades [2], [3], [4].

Traditional data-driven quality prediction methods mainly use linear models, such as partial least squares (PLS) [5] to extract the relations between process data and quality data. However, they have difficulties in establishing nonlinear relations between process data and quality data. For this reason, some sophisticated models, such as support vector regression [6], decision tree models [7], and artificial neural network [8], have been used for quality prediction. To address complex dynamic nonlinear relations in industrial processes, deep learning models have been used in the field of data-driven quality prediction [9], [10]. A viable class of approaches is to build prediction models using extracted features by applying transform learning [11], [12], [13], [14], [15], variational autoencoders (VAE) [16], stacked autoencoders (SAE) [17], and etc. Although multilayer perceptron (MLP) [18] can extract quality-related features, process data, and quality data are required to be collected at the same sampling rate. From the practical perspective, industrial processes usually collect two-level data, including process data at a fast sampling rate and quality data at a slow sampling rate. The quality data are used to promote the process operation performance and product quality, but they are usually obtained by offline lab-analysis rather than online measurement. Quality prediction using the real-time process measurements is essential to advanced process control and operation optimization for the quality improvement and cost reduction of the industrial processes. This motivates dual-rate data-based quality prediction.

There are three main solutions for data-driven quality prediction using dual-rate data, i.e., up-sampling, down-sampling, and dual-rate data modeling methods. Down-sampling and up-sampling techniques transform the original data into the ones with the same sampling rate [19]. Using the transformed data, data-driven quality prediction models are subsequently developed by applying PLS, MLP, SAE, etc. While down-sampling inevitably leaves the dynamics unused, up-sampling aims to use the measurement with the highest sampling rate. However, the prediction errors may accumulate with the increase of prediction

Manuscript received 31 December 2022; revised 9 April 2023; accepted 28 April 2023. Date of publication 12 May 2023; date of current version 19 January 2024. This work was supported in part by the National Key Research and Development Project under Grant 2020YFB1710003, in part by the National Natural Science Foundation of China under Grant 61991401, Grant 62161160338, and Grant U20A20189, in part by the National Natural Science Foundation of China (NSFC)/Research Grants Council (RGC) Joint Research Scheme N_HKBU214/21, in part by the General Research Fund (GRF) of RGC under Grant 12202622, and in part by the 111 Project 2.0 under Grant B08015. Paper no. TII-22-5287. *(Corresponding author: Qiang Liu.)*

Xiangan Meng, Qiang Liu, and Chao Yang are with the State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang 110819, China (e-mail: 2102036@stu.neu.edu.cn; liuq@mail.neu.edu.cn; yangc1109@163.com).

Le Zhou is with the School of Automation and Electrical Engineering, Zhejiang University of Science and Technology, Hangzhou 310023, China (e-mail: zhoule@zust.edu.cn).

Yiu-Ming Cheung is with the Department of Computer Science, Hong Kong Baptist University, Hong Kong, SAR, China (e-mail: ymc@comp.hkbu.edu.hk).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TII.2023.3275700.

Digital Object Identifier 10.1109/TII.2023.3275700

steps [20]. Tipping and Bishop proposes an up-sampling method, namely probabilistic principal component analysis model, by which missing values can be estimated through the probabilistic model [21]. Both down-sampling and up-sampling manners, however, may alter the correlation structure of the dual-rate data. By contrast, dual-rate data modeling methods can utilize all of the data samples at a fast sampling rate, thus retaining the dynamics among the original process data to establish a more accurate model. For example, paper [22] develops a principal component regression model for quality prediction. Also paper [23] incorporates a cooperative training strategy with PLS to develop a dual-rate data-based quality prediction model. However, the abovementioned methods cannot effectively find out the dynamic nonlinear relations among process data for data-driven quality prediction.

In recent years, deep learning-based methods have attracted much attention in data-driven quality prediction [24]. Although recurrent neural networks and long short-term memory (LSTM) can extract dynamic nonlinear relations, the models are overly complex [25]. To take advantage of the dynamic nonlinear feature extraction through convolution kernel operator, convolutional neural network (CNN)-based quality prediction has been studied recently [24], [26]. The convolution kernel operation on the windowed fast sampling process data in the form of matrix is used to extract the latent features for quality prediction. For example, a finite impulse response-CNN is developed for quality prediction [24]. This method applies finite impulse responses to generate the input of the CNN, in order to overcome the defect of traditional CNN that equally treats historical process data samples, leaving the sequence information of the samples unused. A multichannel CNN-based quality prediction model is proposed in [26] to find out dynamic relations among process data. Furthermore, a multidimensional CNN (MDCNN) that extracts variable correlation, temporal feature, and spatial feature through three convolutional kernels is proposed for quality prediction [27]. The work of [24] considers the disadvantages of CNNs due to shared weights, but it does not consider the representative information contained in each variable. As far as we know, dynamic modeling of dual-rate data is still a challenge for CNN-based methods.

In addition, the abovementioned data-driven quality prediction methods require complete dynamic data samples. However, dynamic process data at a fast sampling rate may be corrupted with missing values and outliers caused by transmission anomaly, hardware sensor failure or maintenance, etc. [28]. Direct leaving out the anomaly samples by traditional PauTa criterion [24] may significantly reduce the number of samples and corrupt the dynamics among the successive process data samples. This will make it impossible to extract the latent dynamic features for data-driven quality prediction. Missing data imputation (MDI) that reconstructs complete samples by minimizing the estimation error provides a new way for static data imputation. Typical MDI models include principal component analysis [29], expectation maximization [30], and MissForest [31]. However, most of the samples with missing values in practical industrial processes may be incomplete. In recent years, denoising-based autoencoders have been used for incomplete data imputation. However, these methods merge the data samples with missing values and the ones with no missing values in the hidden layer. To address the incomplete data imputation, Yoon et al. proposes a generative adversarial imputation network (GAIN) that uses a mask matrix to indicate the positions of the missing values [32]. Although GAIN shows advantages compared to MissForest and Autoencoder in [32], it is static data imputation in nature. To the best of authors' knowledge, dynamic data imputation remains unsolved.

In this article, to make full use of the fast sampling process data with corrupted values, a novel deep learning-based robust dual-rate dynamic data modeling method is proposed for quality prediction of dynamic nonlinear processes. First, to capture dynamic features among multiple samples of multiple moments in a moving window, one-dimensional CNN (1D-CNN) is incorporated with GAIN to establish a novel dynamic data denoising generative adversarial imputation network (DDGAIN). During the data generator stage, data samples generated from the dynamic features are reconstructed using a VAE to exclude the data noises. Second, a hint CNN (HCNN) with channel extension is proposed for dual-rate dynamic data modeling. Inspired by [33] that applies CNN to encode position information, the proposed HCNN incorporates a dual information hint mechanism that separately organizes temporal flag and variable flag for the original process data. Two separate convolution kernels are used to extract the hint information contained in the samples at each moment and the hint information contained in each variable during the time period. The hint information and the process data matrix are then concatenated and transferred to the next layer for dynamic relationship extraction. These two aspects constitute the major contributions of this work.

The rest of this article is organized as follows. Traditional GAIN, VAE, and CNN models are briefly reviewed in Section II. Then, the proposed robust dual-rate data modeling method that is composed of DDGAIN and HCNN is presented in Section III. In Section IV, Dow distillation column process dataset and the Beijing multisite air quality dataset are used to verify the effectiveness of the proposed method. Finally, Section V concludes this article.

II. Preliminaries

A. Generative Adversarial Imputation Network

GAIN used for data imputation is proposed by Yoon et al. [32]. GAIN is derived from the generative adversarial network, with the main structure composed of a generator, a hint generator, and a discriminator, as shown in Fig. 1.

The generator observes the real components of the original data vector and uses the observations to estimate the missing values. The mask matrix is used to hint at the position of the missing elements in the generator. The discriminator discriminates whether the input is the real component or the interpolated component. The hint generator displays partial information to the discriminator on which components are missing from the original sample. This hint mechanism makes the generator learn the real data distribution. The generator and discriminator compete with each other to coevolve and reach Nash equilibrium.

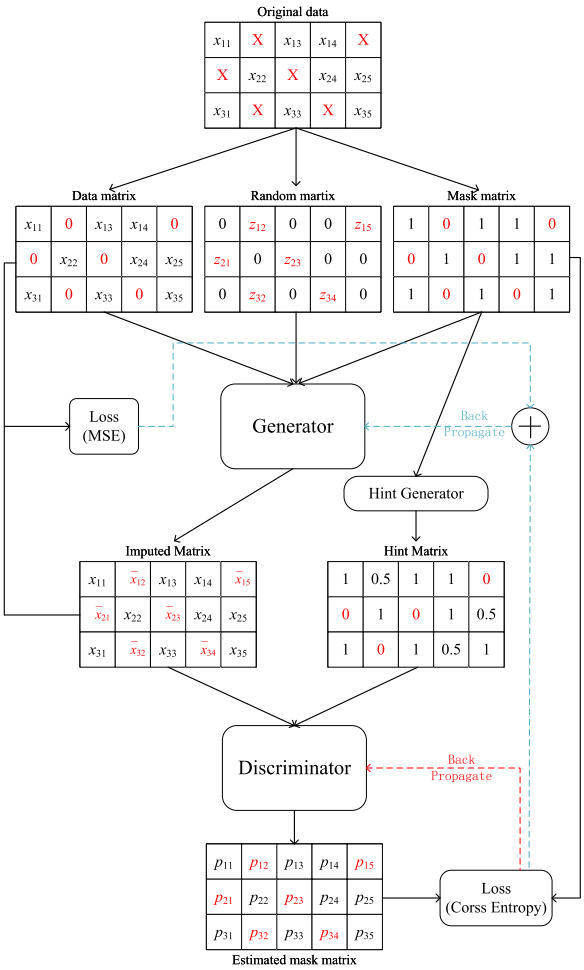

Fig. 1. Structure of GAIN [32].

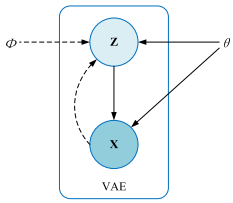

Fig. 2. Graphical model of the VAE.

B. Variational Autoencoder

VAE, as a typical deep generative model, is established based on variational Bayesian inference proposed by Kingma et al. [16]. VAE consists of an input layer, an encoder layer, a latent variable layer, a decoder layer, and an output layer. The structure of the probability graph of VAE is shown in Fig. 2, where the solid line indicates the generative model $q_\phi(\mathbf{z})$ $p_\theta(\mathbf{x} \mid \mathbf{z})$. The dashed line indicates the inference model $q_\phi(\mathbf{z} \mid \mathbf{x})$, which is an approximation of the intractable true posterior distribution $p_\theta(\mathbf{z} \mid \mathbf{x})$. Here, $\mathbf{x}$ represents the observed variable, $\mathbf{z}$ represents the latent variable, and the variational parameter θ is learned jointly with the generative model parameter ϕ.

The marginal likelihood of $\mathbf{x}$ can be given as

$$\log p(\mathbf{x}) = D_{\mathrm{KL}}\left(q_\phi(\mathbf{z} \mid \mathbf{x}) \| p_\theta(\mathbf{z} \mid \mathbf{x})\right) + \mathcal{L}(\theta, \phi; \mathbf{x}) \tag{1}$$

where D_{KL} denotes the Kullback–Leibler divergence of the approximation from the true posterior. $\mathcal{L}(\theta, \phi; \mathbf{x})$ is called the (variational) lower bound on the marginal likelihood of data sample $\mathbf{x}$. It could also be written as

$$\begin{aligned} \log p_\theta(\mathbf{x}) &\geq \mathcal{L}(\theta, \phi; \mathbf{x}) \\ &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[-\log q_\phi(\mathbf{z} \mid \mathbf{x}) + \log p_\theta(\mathbf{x}, \mathbf{z})\right] \\ &= -D_{\mathrm{KL}}\left(q_\phi(\mathbf{z} \mid \mathbf{x}) \| p_\theta(\mathbf{z})\right) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x} \mid \mathbf{z})\right]. \end{aligned} \tag{2}$$

The loss function used to train the VAE is given as

$$\mathcal{L} = -\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x} \mid \mathbf{z})\right] + D_{\mathrm{KL}}\left(q_\phi(\mathbf{z} \mid \mathbf{x}) \| p_\theta(\mathbf{z})\right). \tag{3}$$

The VAE algorithm that reconstructs the input through the decoder shows an advantage on noise suppression, it is incorporated into the dynamic data imputation in Section III-A.

C. Convolutional Neural Network

Two convolution operations of the popular two-dimensional CNN (2D-CNN) are as follows:

$$H_2 = \frac{H_1 - K_h + 2p}{S} + 1 \tag{4}$$

$$W_2 = \frac{W_1 - K_w + 2p}{S} + 1 \tag{5}$$

where the size of the original feature map is $(H_1 \times W_1)$, the size of the convolution kernel is (K_w, K_h). p is the filling size. The size of the feature obtained by convolution operation is $(H_2 \times W_2)$.

The work of [24], [26], [34] demonstrates the advantage of 2D-CNN on local feature extraction. However, 2D-CNN equally treats the historical process data samples. 1D-CNN that commonly used for sequence modeling and natural language processing is applicable to dynamic data modeling.

III. Proposed DDGAIN-HCNN for Quality Prediction Using Dual-Rate Dynamic Data

A. Dynamic Data Denoising GAIN

Traditional GAIN does not take into account the dynamic relations among the process data. To make use of the dynamic relations for dynamic data imputation, this section proposes a novel DDGAIN. The DDGAIN is built from the framework in Fig. 1, while the dynamic relations and denoising are taken into account. The overall structure of DDGAIN consists of a data moving window processing, an 1D-CNN-based generator, an 1D-CNN-based discriminator, a hint generator, and a loss function.

1) Data Moving Window Processing: The moving window technique is first applied to preprocess the data. The moving window is a window with a fixed size. There are two main parameters, that is, the size of the window in the time dimension l and the step size s. The size of the other side of the moving window is equal to the dimension of the variable. The

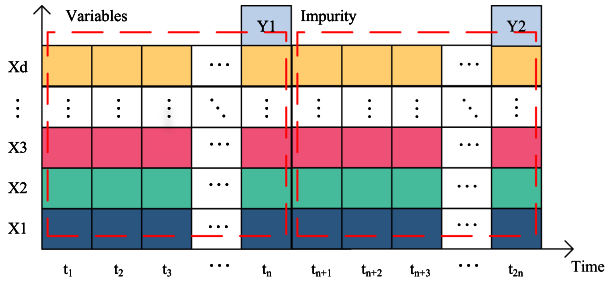

Fig. 3. Moving window diagram of process data in dual rate data.

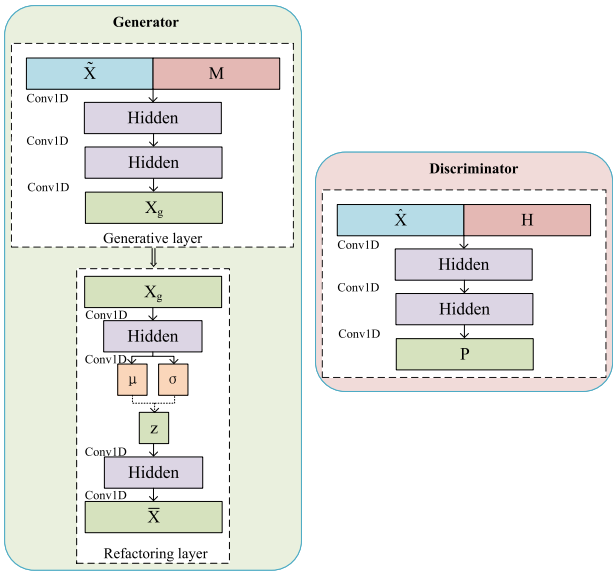

Fig. 4. Generator and discriminator in DDGAIN. The base component is 1D-CNN.

subset after the moving window processing can be represented as $\mathbf{X}_l=\{\mathbf{x}_{t-l+1}, \mathbf{x}_{t-l+2}, \ldots, \mathbf{x}_t\}$, where $\mathbf{x}_t$ is the sample of the moment $\mathbf{x}_t=[x_t^1, x_t^2, \ldots, x_t^d] \in \mathbb{R}^d$. The dual-rate data and the moving window processing are shown in Fig. 3. It can be seen that process variables are sampled at every moment, while quality variables are sampled at every t_n moments ($t_n > 1$). Therefore, the length of the moving window is set to $l = t_n$, so that each subset contains all the process data samples between two adjacent quality samples.

2) Generator in DDGAIN: The generator (G) in DDGAIN is shown on the left in Fig. 4. G is composed of two parts: the generation layer and the reconstruction layer. Both parts are built using 1D-CNN. Internally the generation layer will generate data $\mathbf{X}_g$ based on the real data and the mask matrix. The reconstruction layer reconstructs $\mathbf{X}_g$ into $\overline{\mathbf{X}}$ by VAE to depress the noises in the generated data. Externally, the generator takes $\tilde{\mathbf{X}}$ and $\mathbf{M}$ as the inputs, where $\tilde{\mathbf{X}}$ is obtained from the following

$$\tilde{\mathbf{X}} = \mathbf{M} \odot \mathbf{X} + (\mathbf{1} - \mathbf{M}) \odot \mathbf{Z}. \tag{6}$$

In the abovementioned equation, $\odot$ denotes an element-wise multiplication. $\mathbf{X}$ is the original data matrix, $\mathbf{M}$ is the mask matrix, and $\mathbf{Z}$ is the noise. Each element in $\mathbf{M}$ denotes whether the corresponding value is missing or not. "0" represents missing, while "1" represents not missing.

In practice, noise is filled into the $\tilde{\mathbf{X}}$, generating a new $\overline{\mathbf{X}}$. The missing values in $\tilde{\mathbf{X}}$ are filled with the newly generated data in $\overline{\mathbf{X}}$ to form the interpolated data $\hat{\mathbf{X}}$

$$\overline{\mathbf{X}} = G(\tilde{\mathbf{X}}, \mathbf{M}) \tag{7}$$

$$\hat{\mathbf{X}} = \mathbf{M} \odot \tilde{\mathbf{X}} + (\mathbf{1} - \mathbf{M}) \odot \overline{\mathbf{X}}. \tag{8}$$

3) Discriminator in DDGAIN: The discriminator (D) is modeled by 1D-CNN, as shown in Fig. 4. It is expected to distinguish, which components are observed or imputed. Discriminator takes $\mathbf{H}$ and $\hat{\mathbf{X}}$ as the inputs, with $\mathbf{P}$ as the outputs. $\mathbf{H}$ is the hint matrix generated by the hint generator. $\hat{\mathbf{X}}$ is the interpolated matrix. $\mathbf{P}$ (i, j) corresponding to the probability that $\hat{\mathbf{X}}$ (i, j) is observed rather than imputed. $\mathbf{P}$ that varies from 0 to 1 is computed from (9). The objective of the discriminator is to distinguish the generated components and real components. In other words, the mask matrix $\mathbf{M}$ determined by the dataset is reconstructed

$$\mathbf{P} = D(\hat{\mathbf{X}}, \mathbf{H}). \tag{9}$$

4) Hint Generator in DDGAIN: The hint generator transfers partial information on the binary mask matrix $\mathbf{M}$ to D to ensure that G generates data samples according to the real data distribution. The hint generator takes the mask matrix $\mathbf{M}$ as the input, with $\mathbf{H}$ as the output. $\mathbf{H}$ controls the information content of the mask matrix $\mathbf{M}$, which is transferred to D. If $\mathbf{H}$ does not contain "sufficient" information about $\mathbf{M}$, it cannot guarantee that G learns the real data distribution [35]. We thus define a matrix $\mathbf{B}$, whose element $\mathbf{B}$ (i, j) is randomly sampled from $[0, 1]$. $\mathbf{H}$ is calculated as

$$\mathbf{H} = \mathbf{B} \odot \mathbf{M} + 0.5(\mathbf{1} - \mathbf{B}). \tag{10}$$

As for h, it can be concluded $h(i, j) = m(i, j)$ when $b(i, j) = 1$. When $b(i, j) = 0$, $h(i, j) = 0.5$ and no useful information from $m(i, j)$ is transmitted.

5) Loss Function of DDGAIN: The loss function of the discriminator is computed from (11). The cross-entropy loss is involved only when $b(i, j) = 0$, with B_0 the number of 0 elements in $\mathbf{B}$

$$\mathcal{L}_D(\mathbf{M}, \mathbf{P}, \mathbf{B}) = -\frac{1}{B_0} \sum_{i=1}^{I} \sum_{j=1;\mathbf{B}(i,j)=0}^{J} [\mathbf{M}(i, j) \log(\mathbf{P}(i, j)) + (1 - \mathbf{M}(i, j)) \log(1 - \mathbf{P}(i, j))]. \tag{11}$$

The loss function of the generator is computed from (12) and consists of two terms. The first term, similar to D loss, determines whether the interpolated values cheat G. The second term is the mean square error of the actual values and interpolated values at the nonmissing position

$$\mathcal{L}_G(\mathbf{M}, \mathbf{P}, \mathbf{B}, \overline{\mathbf{X}}, \mathbf{X}) = -\frac{1}{B_0} \sum_{i=1}^{I} \sum_{j=1;\mathbf{B}(i,j)=0}^{J} (1 - \mathbf{M}(i, j)) \log(1 - \mathbf{P}(i, j)) - \alpha \frac{1}{M_0} \sum_{i=1}^{I} \sum_{j=1}^{J} (\mathbf{M}(i, j)\overline{\mathbf{X}}(i, j) - \mathbf{M}(i, j)\mathbf{X}(i, j))^2 \tag{12}$$

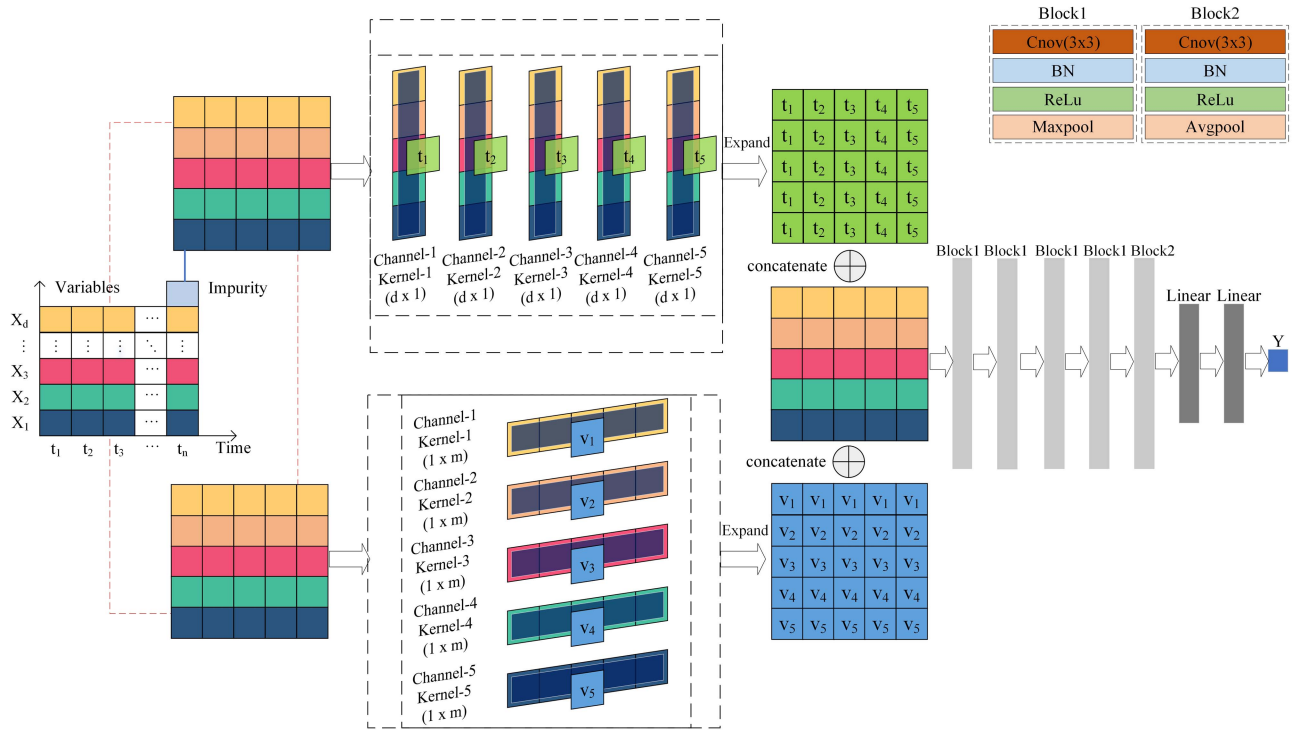

Fig. 5. Structure of the HCNN with a temporal information hint mechanism and a variable information hint mechanism.

where M_0 is the number of 0 elements in $\mathbf{M}$ and α is a hyperparameter.

B. HCNN With Channel Expansion for Quality Prediction

Traditional CNN uses convolution kernels to extract space features of the images. Convolution kernels have the property of weight sharing, with the same convolution kernel parameters. The benefit of weight sharing is that the number of parameters is greatly reduced and the depth of the network can be increased. However, process data are normally dynamic with temporal information. CNN treats historical process data samples equally, rather than assigns different weights based on their temporal snapshots. Since each variable has representative information on quality variables, it is necessary to extract the difference between each variable. To address this, we propose to overlay temporal information hint and variable information hint on the feature maps of the process data to find the dynamics in process data.

The input of CNN is usually four dimensions (batch size, channel, height, and width). In general, image data occupies 3 channels and process data occupies 1 channel. As shown in Fig. 5, the proposed HCNN with a dual information hint mechanism extracts the representation information of time and variables by expanding the process data on channel dimension.

Temporal information hint mechanism: The input data matrix $\mathbf{X}$ with a dimension (batch size, 1, d, n) is expanded along the time axis as (batch size, n, d, 1). The representation information of time in n channels is extracted using n different convolution kernels of size $(d \times 1)$ to form a temporal information vector, which contains features unique to n different moments. Then, the temporal information vector is expanded into a matrix $\mathbf{T}$ with the same dimension as the input.

Variable information hint mechanism: Variable information is extracted similarly to temporal information. By expanding $\mathbf{X}$ along the variable axis as (batch size, d, 1, n), the representation information in the d channels is extracted using d different convolution kernels of size $(1 \times n)$. A vector of variable information is constructed, which encapsulates features that are unique to d different variables within the time window. The vector of variable information is thereafter expanded into a matrix $\mathbf{V}$ with the same dimension as the input.

The data matrices $\mathbf{X}$, $\mathbf{T}$, and $\mathbf{V}$ are superimposed to obtain

$$\mathbf{X}_{\text{input}} = \mathbf{X} \oplus \mathbf{T} \oplus \mathbf{V} \tag{13}$$

where $\oplus$ represents the sum of each element of the matrix. Each element in $\mathbf{X}_{\text{input}}$ is superimposed with different information as $(T_{(i,j)}, V_{(i,j)})$.

In order to extend the receptive field for global information extraction, multilayer convolutions, and pooling layers are applied, as shown in Fig. 5. Five convolutional layers are used to extract sample-related dynamic features. The Avgpool can extract the overall features to prevent losing much global information, and the Maxpool can filter out useless features and extract far apart features relevant to quality prediction. Two blocks including Block 1 and Block 2 are first established. Each block contains a convolution layer, a batch normalization (BN) layer, and a rectified linear unit (ReLU) layer. BN is a commonly used adaptive reparameterization method, which can alleviate the gradient disappearance or explosion phenomenon in CNN training, and accelerate the training. ReLU in the following is to enable the

hierarchical nonlinear mapping learning:

$$\mathrm{ReLU}(x) = \max(0, x). \tag{14}$$

Block 1 extracts the maximum value of adjacent features by maximum pooling. Block 2 maps the feature values to feature vectors by average pooling. The first four layers of the network use Block 1 to extract the dynamic. After each convolution, most of the information on the feature map is retained through maximum pooling. The fifth layer uses Block 2 to integrate the extracted dynamics into vectors. The last two layers use the fully connected layer to obtain the predicted value. The mean squared error is used as the loss function of the HCNN to converge the algorithm [36]. The loss function is defined as follows:

$$\mathrm{Loss} = \frac{1}{n}\sum_{i=1}^{n} \|y_i - \hat{y}_i\|^2 \tag{15}$$

where n is the number of samples, $\hat{y}_i$ is the predicted value of the ith sample, y_i is the true value of the ith sample. The loss function that measures the distance between the predicted value and the true value can be minimized using backpropagation algorithm.

Note that the proposed method works well only when the relations between the dual-rate data remain unchanged. While there are significant difference on data distribution between the training dataset and the test dataset, one can incorporate the idea of transfer learning and adaptive modeling. In addition, the data quality is essential to data-driven modeling. Whenever there are a large number of missing values, data-driven modeling including the proposed quality prediction method may fail. For the situation when there are a small number of missing values for the process data, the proposed DDGAIN-HCNN can be robust to the missing values. In this work, the validation set is used to select the appropriate parameters based on the grid search. When there is a significant prediction performance degradation caused by model mismatch, it is necessary to collect a new training dataset to train the model or develop an adaptive modeling method similar to the recursive PLS in [37].

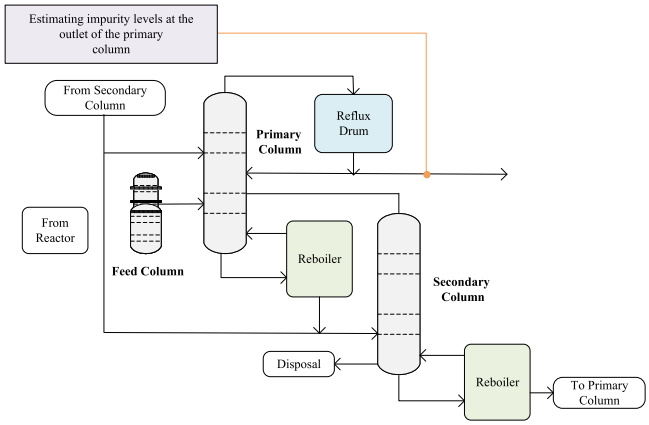

Fig. 6. Block diagram of the Dow challenge problem [38], [39].

TABLE I
DEFINITION OF DOW PROCESS VARIABLES AND QUALITY VARIABLE

Primary Column (PC)	Secondary Column (SC)
x1: PC Reflux Flow	x22: SC Base Concentration
x2: PC Tails Flow	x23: Flow from Input to SC
x3: Input to PC Bed 3 Flow	x24: SC Tails Flow
x4: Input to PC Bed 2 Flow	x25: SC Tray Drum Pressure
x5: PC Feed Flow from FC	x26: SC Head Pressure
x6: PC Make Flow	x27: SC Base Pressure
x7: PC Base Level	x28: SC Base Temperature
x8: PC Reflux Drum Pressure	x29: SC Tray 3 Temperature
x9: PC Condenser Reflux Drum Level	x31: SC Bed 2 Temperature
x10: PC Bed1 Drum Pressure	x32: SC Tray 2 Temperature
x11: PC Bed2 Drum Pressure	x33: SC Tray 1 Temperature
x12: PC Bed3 Drum Pressure	x34: SC Tails Temperature
x13: PC Bed4 Drum Pressure	x35: SC Tails Concentration
x14: PC Base Pressure	
x15: PC Head Pressure	Feed Column (FC)
x16: PC Tails Temperature	x36: FC Recycle Flow
x16: PC Tails Temperature	x37: FC Tails Flow to PC
x18: PC Bed 4 Temperature	x38: FC Calculated Drum Pressure
x19: PC Bed 3 Temperature	x39: FC Steam Flow
x20: PC Bed 2 Temperature	x40: FC Tails Flow
x21: PC Bed 1 Temperature	
x41: Avg_Reactor_Outlet_Impurity	
x42: Avg_Delta_Composition PC	
x43: PC Reflux/Feed Ratio	
x44: PC Make/Reflux Ratio	
y: Impurity	

IV. EXPERIMENTAL STUDIES

A. Dow Distillation Tower Experiment

1) Dow Process Description: The Dow challenge problem in [38] and [39] is used to demonstrate the proposed DDGAIN-HCNN model. The refining system as shown in Fig. 6 is composed of three distillation columns: a primary column, a feed column, and a secondary column. The primary column is controlled according to the reflux feed ratio. Accumulation of impurities leads to accelerated catalyst aging and degrades the operation of the refining system. The process has to operate in a suboptimal manner to remove impurities that can affect the quality of the final product, evenly making the operation unsaleable. It is, thus, necessary to online measure the concentration of impurities.

The impurity concentration at the top of the primary column is selected as the quality variable. A total of 44 process variables x1–x44 and a quality variable y are listed in Table I. According to [39], data samples during operating condition change from August 22, 2016 to December 16, 2016 are excluded. In addition, x16 and x17 with seasonal variations are replaced by two newly generated variables [39].

It is further observed that real measurements of the quality variable are taken at a slow sampling interval, which is typical for lab-test measurements. In this work, process data and quality data are collected at dual-rates (1 h for process variables and 5 h for the quality variable). Moreover, the missing values are taken into account during the modeling stage. These two aspects motivate the application of the proposed robust dual-rate data modeling method. For this study, the numbers of samples for the training dataset, verification dataset, and test dataset are 1200, 203, and 1133, respectively.

2) Modeling and Result Analysis: The adaptive moment estimation (Adam) optimizer using the exponentially weighted averaging technique is adopted for the proposed method and the baselines. The hyperparameters of the models are optimized by grid search to tradeoff model complexity against generalization error. The learning rate is chosen from [0.1, 0.01, 0.001, 0.0001], batch size is chosen from [15, 25, 35, 45], α is chosen from [1,

TABLE II
COMPARISON RESULTS OF DYNAMIC DATA IMPUTATION FOR DOW

Missing rate	RMSE				
	KNN [40]	MICE [41]	GAIN [32]	DGAIN	DDGAIN
0.1	0.04532	0.04082	0.07646	0.05261	**0.03855**
0.2	0.05171	0.04442	0.11813	0.07082	**0.04031**
0.3	0.05572	0.04837	0.14757	0.08069	**0.04731**
0.4	0.06033	**0.05313**	0.15870	0.11045	0.08196
0.5	0.07078	**0.05626**	0.16037	0.11751	0.08595

The bold entities mean the optimal performance of all models on the corresponding index under the same conditions.

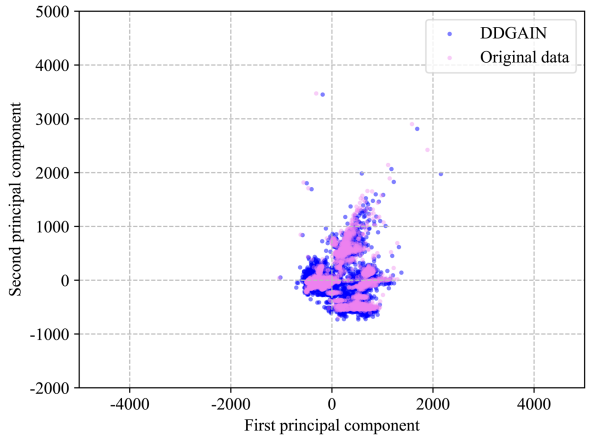

Fig. 7. DDGAIN imputation data distribution for Dow with the original data at 0.5 missing rates.

10, 30, 50], the DDGAIN epoch is chosen from [15 000, 30 000, 45 000], and the HCNN epoch is chosen from [50, 75, 100, 125, 150]. The optimal parameters for the GAIN, DGAIN, and DDGAIN models are set as follows: batch size is 25, learning rate is 0.001, α is 10, and the number of epoch is 30 000. The optimal parameters of the prediction model for HCNN are set as follows: batch size is 25, learning rate is 0.0001, and the number of epoch is 100.

In the imputation experimental part, the training data are randomly missing at a rate of 0.1 to 0.5. GAIN, dynamic data GAIN (DGAIN) without VAE denoising, and the proposed DDGAIN with VAE denoising are used to compare the imputation performance. It is compared with the traditional k-nearest neighbors (KNN) [40] and multivariate imputation by chained equations (MICE) [41] methods. The results are shown in Table II. It can be seen that the dynamic data imputation methods including DGAIN and DDGAIN outperform the traditional GAIN, with DDGAIN achieving the most excellent performance. Meanwhile, DDGAIN outperforms the KNN and MICE methods in the case of 0.1–0.3 missing rates, achieving the best performance. The data distributions using the proposed DDGAIN and with no data imputation at 0.5 missing rates are shown in Fig. 7.

Using the clean data that retain the process dynamics by dynamic imputation, data-driven quality prediction is performed by the proposed HCNN, and the baseline models including MLP, LSTM, and CNN, as well as the state-of-the-art models including temporal convolutional networks (TCN) [42] and MDCNN. In order to achieve a fair comparison, the parameters for each model are optimally selected. The moving window length is

TABLE III
DEFINITION OF BEIJING MULTISITE AIR QUALITY DATASET VARIABLES AND QUALITY VARIABLE

Air Quality Data	Meteorological Data
x1: PM_{10} Concentration	x6: Temperature
x2: SO_2 Concentration	x7: Pressure
x3: NO_2 Concentration	x8: Dew Point Temperature
x4: CO Concentration	x9: Precipitation
x5: CO_3 Concentration	x10: Wind Direction
y: $PM_{2.5}$(Particulate Matter)	x11: Wind Speed

TABLE IV
COMPARISON RESULTS OF DYNAMIC DATA IMPUTATION FOR AIR QUALITY DATA

Missing rate	RMSE				
	KNN [40]	MICE [41]	GAIN [32]	DGAIN	DDGAIN
0.1	0.08417	0.07628	0.13085	0.09304	**0.05983**
0.2	0.11935	0.08165	0.13453	0.11773	**0.06081**
0.3	0.14426	0.09445	0.13581	0.12855	**0.06566**
0.4	0.14804	0.09817	0.13664	0.13056	**0.06960**
0.5	0.14976	0.10420	0.14568	0.13627	**0.07235**

The bold entities mean the optimal performance of all models on the corresponding index under the same conditions.

set to $l = 5$. MLP uses fine-grained data modeled with each sample dimension of (44×1), and the network contains two hidden layers (22, 12). LSTM, CNN, TCN, and MDCNN use dynamic data with a window size of (44×5). The LSTM has a hidden layer (22), the TCN uses two temporal blocks of channel size 20, where each temporal block contains two convolutional layers with convolution kernel size 3. The network structures of CNN and MDCNN are consistent with HCNN, with a 5-CNN layer convolutional kernel of (3×3) and three layers of fully connected networks (40, 16, 1). Note that the MDCNN uses three convolutional kernels (3×3), (1×3), (3×1) for parallel computation.

The root mean square error (RMSE), the coefficient of determination index R^2, and the mean absolute error (MAE) are used to evaluate the prediction accuracy, with the results listed in Table V. From this table, the prediction with DDGAIN imputation performs better than the one with GAIN imputation. Although the imputation accuracy of MICE is slightly higher than that of DDGAIN at 0.4 and 0.5 missing rates, the prediction using DDGAIN is significantly better than that of MICE, indicating that DDGAIN exploits the dynamics of the data. The interpolated values are more consistent with the distribution of the original data. Moreover, the performance of DDGAIN is more significant in the case of a large missing rate. It is worth noting that MLP and LSTM achieve a good prediction performance when mean imputation is applied, but process dynamics used for quality prediction are missing. MLP and LSTM display lower prediction accuracy since the process dynamics are not extracted well.

Compared with the traditional MLP, LSTM, CNN, as well as TCN and MDCNN models, HCNN achieves the best prediction. Although CNN, TCN, MDCNN, and HCNN are all CNN-based dual-rate models, they are sensitive to the imputation accuracy. The predictions using CNN, TCN, MDCNN, and HCNN with mean imputation perform worse than the ones with GAIN, DGAIN, and DDGAIN. The TCN achieves a further improvement compared to the LSTM model. The MDCNN model

TABLE V
COMPARISON RESULTS OF QUALITY PREDICTION FOR THE DOW PROCESS

Missing rate	Imputation	MLP [18]			LSTM [25]			CNN [24]			TCN [42]			MDCNN [27]			HCNN		
		RMSE	R^2	MAE	RMSE	R^2	MAE	RMSE	R^2	MAE	RMSE	R^2	MAE	RMSE	R^2	MAE	RMSE	R^2	MAE
0.1	Mean	0.17517	0.15928	0.09410	0.16826	0.20852	0.08677	0.19641	-0.05815	0.12709	0.16811	0.22645	0.09835	0.44984	-4.65077	0.33906	0.19368	-0.02983	0.09921
	KNN	0.18763	0.03463	0.10786	0.17737	0.13912	0.09471	0.17936	0.11931	0.10252	0.16846	0.2233	0.09769	0.18269	0.07197	0.09871	0.16132	0.28793	0.08471
	MICE	0.18802	0.03061	0.10841	0.17738	0.13903	0.09521	0.18425	0.07003	0.10938	0.16607	0.24519	0.09375	0.18556	0.05140	0.10202	0.16144	0.28682	0.08476
	GAIN	0.18821	0.02849	0.10862	0.17790	0.13396	0.09582	0.17962	0.11687	0.10281	0.16764	0.23094	0.09643	0.17476	0.16207	0.09347	0.16190	0.28275	0.08607
	DGAIN	0.18781	0.03267	0.10809	0.17745	0.13835	0.09500	0.17933	0.13610	0.10245	0.16865	0.22162	0.09824	0.17861	0.12232	0.09222	0.16190	0.28276	0.08508
	DDGAIN	0.18762	0.03472	0.10780	0.17729	0.13987	0.09498	0.17895	0.12346	0.10191	0.16706	0.23616	0.09558	0.17262	0.18413	0.09293	**0.16129**	**0.28816**	**0.08464**
0.2	Mean	0.17712	0.13984	0.09673	0.16902	0.21821	0.08796	0.19990	-0.09524	0.12677	0.17408	0.17058	0.10706	0.44558	-4.57353	0.30692	0.20583	-0.16229	0.10798
	KNN	0.18789	0.05881	0.10820	0.17776	0.13533	0.09569	0.17918	0.12119	0.10225	0.16718	0.23512	0.09525	0.18432	0.06476	0.10212	0.16288	0.27387	0.08858
	MICE	0.18894	0.02111	0.10905	0.17803	0.13265	0.09596	0.17878	0.12518	0.10098	0.16785	0.22900	0.09739	0.19190	-0.01461	0.11519	0.16162	0.28525	0.08591
	GAIN	0.18815	0.02892	0.10851	0.17799	0.13300	0.09587	0.18139	0.09935	0.10493	0.16672	0.23914	0.09504	0.18570	0.04630	0.10766	0.16279	0.27488	0.08778
	DGAIN	0.18776	0.03312	0.10798	0.17761	0.13673	0.09542	0.17908	0.12848	0.10201	0.16736	0.23356	0.09630	0.18098	0.09697	0.09817	0.16283	0.27430	0.08724
	DDGAIN	0.18768	0.03413	0.10789	0.17754	0.13741	0.09532	0.17884	0.12450	0.10187	0.16592	0.24653	0.09314	0.17777	0.20051	0.09559	**0.16132**	**0.28792**	**0.08456**
0.3	Mean	0.17923	0.11861	0.09941	0.17020	0.20722	0.08974	0.22432	-0.42180	0.11531	0.18052	0.10772	0.11656	0.44012	-4.49619	0.31388	0.22256	-0.36744	0.11890
	KNN	0.18755	0.03547	0.10778	0.17752	0.13767	0.09506	0.17796	0.13323	0.08246	0.16802	0.22749	0.09708	0.18246	0.08456	0.10411	0.16229	0.27917	0.08726
	MICE	0.19007	0.00936	0.11054	0.18000	0.11336	0.09858	0.18780	0.03205	0.11300	0.16828	0.21206	0.09734	0.19363	-0.03485	0.11425	0.16223	0.27977	0.08692
	GAIN	0.18902	0.01973	0.10959	0.17825	0.13038	0.09609	0.18020	0.11083	0.10201	0.16912	0.21704	0.10008	0.18182	0.08904	0.10575	0.16210	0.28099	0.08690
	DGAIN	0.18790	0.03164	0.10816	0.17773	0.13554	0.09560	0.17885	0.12427	0.10071	0.16866	0.22159	0.09804	0.17807	0.13005	0.10131	0.16374	0.26621	0.09008
	DDGAIN	0.18648	0.04626	0.10619	0.17611	0.15125	0.09320	0.17662	0.14619	0.09851	0.16708	0.23611	0.09552	0.17568	0.15350	0.09494	**0.16203**	**0.28157**	**0.08655**
0.4	Mean	0.18171	0.09301	0.10258	0.17114	0.19837	0.09089	0.26644	-1.02316	0.13662	0.17996	0.11330	0.11421	0.43321	-4.55815	0.26915	0.25492	-0.79746	0.13248
	KNN	0.18764	0.03460	0.10827	0.17761	0.13672	0.09522	0.17944	0.11859	0.10269	0.16795	0.22811	0.09716	0.19486	-0.05959	0.11249	**0.16146**	**0.28667**	**0.08452**
	MICE	0.19097	-0.00010	0.11184	0.18137	0.09977	0.10049	0.19157	-0.00577	0.11635	0.16850	0.22308	0.09826	0.19290	-0.01878	0.11179	0.16263	0.27626	0.08621
	GAIN	0.18915	0.01795	0.10967	0.17801	0.13265	0.09553	0.18020	0.11083	0.09968	0.16632	0.24311	0.09533	0.18472	0.05782	0.10547	0.16691	0.23766	0.09582
	DGAIN	0.18805	0.03731	0.10853	0.17871	0.12571	0.09602	0.18054	0.10789	0.10298	0.16794	0.22825	0.09717	0.17932	0.11445	0.10057	0.16368	0.26678	0.09056
	DDGAIN	0.18724	0.03868	0.10734	0.17724	0.14028	0.09498	0.18042	0.10913	0.10393	0.16554	0.25018	0.09323	0.17326	0.17801	0.09508	0.16166	0.28491	0.08540
0.5	Mean	0.18568	0.05163	0.10762	0.17294	0.18124	0.09306	0.24793	-0.75292	0.12502	0.19143	-0.00328	0.13020	0.48765	-5.94488	0.30237	0.27642	-1.11351	0.16807
	KNN	0.18938	0.01639	0.10982	0.17950	0.11820	0.09797	0.18127	0.10025	0.10497	0.16851	0.22293	0.09860	0.18799	0.02148	0.09977	0.16233	0.27892	0.08632
	MICE	0.19145	-0.00521	0.11247	0.18209	0.09254	0.10149	0.18909	0.02120	0.11520	0.16337	0.26961	0.08998	0.20214	-0.12487	0.12914	0.16272	0.27547	0.08835
	GAIN	0.19334	-0.00559	0.11551	0.18318	0.08153	0.10272	0.18507	0.06184	0.10868	0.17546	0.15630	0.11005	0.17733	0.13628	0.10137	0.17446	0.16675	0.10717
	DGAIN	0.19042	0.00569	0.11110	0.17919	0.12098	0.09725	0.18309	0.08220	0.10377	0.16591	0.24679	0.09436	0.18402	0.07087	0.10054	0.16303	0.27209	0.08869
	DDGAIN	0.18738	0.03711	0.10739	0.17688	0.14380	0.09426	0.18251	0.08788	0.10603	0.16512	0.25391	0.09204	0.18403	0.06842	0.10315	**0.16127**	**0.28838**	**0.08409**

The bold entities mean the optimal performance of all models on the corresponding index under the same conditions.

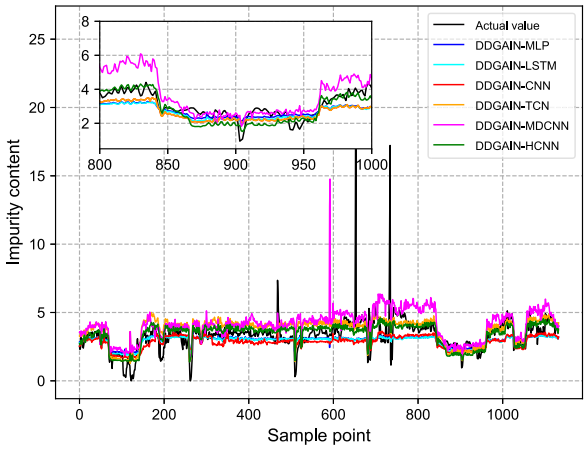

Fig. 8. Prediction results of Dow data at 0.5 missing rates. The prediction methods including MLP, LSTM, CNN, TCN, MDCNN, and HCNN, are trained by the DDGAIN imputation data.

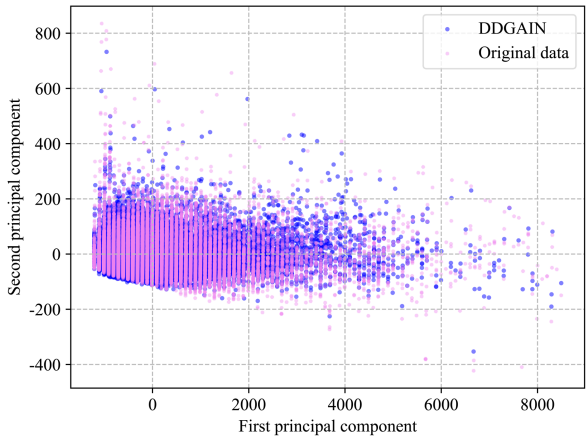

Fig. 10. DDGAIN imputation data distribution for air quality data with the original data at 0.5 missing rates.

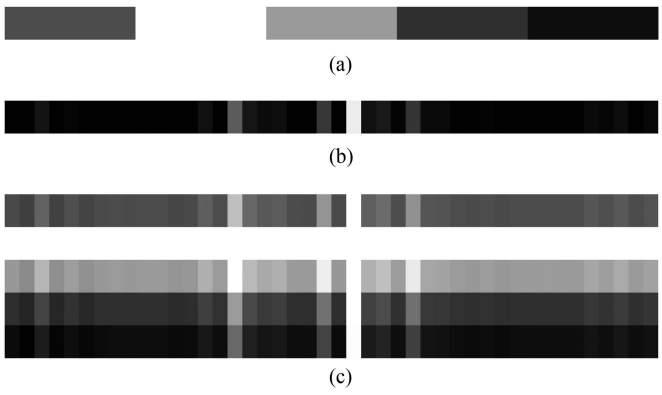

Fig. 9. Information hint mechanism grayscale for Dow process. (a) Temporal information vector within the window ($l = 5$). (b) Variable information vector extracted from 44 variables within the window. (c) T $\oplus$ V information matrix within the window.

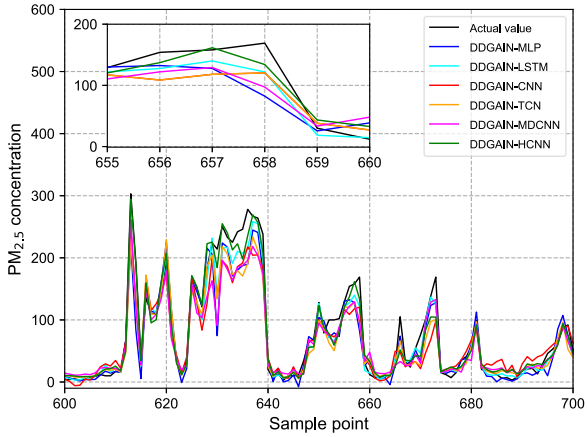

Fig. 11. Prediction results of air quality data at 0.1 missing rates. The prediction methods including MLP, LSTM, CNN, TCN, MDCNN, and HCNN, are trained by the data after DDGAIN imputation.

TABLE VI
COMPARISON RESULTS OF QUALITY PREDICTION FOR AIR QUALITY DATA

Missing rate	Imputation	MLP [18]			LSTM [25]			CNN [24]			TCN [42]			MDCNN [27]			HCNN		
		RMSE	R^2	MAE	RMSE	R^2	MAE	RMSE	R^2	MAE	RMSE	R^2	MAE	RMSE	R^2	MAE	RMSE	R^2	MAE
0.1	Mean	0.44669	0.91352	0.28288	0.46720	0.90632	0.25663	0.46419	0.90750	0.28463	0.45604	0.91283	0.29206	0.44117	0.91520	0.27688	0.43776	0.91770	0.27315
	KNN	0.44705	0.91333	0.25486	0.46320	0.90792	0.23284	0.45101	0.91251	0.28639	0.45743	0.91227	0.27815	0.41962	0.92425	0.25369	0.41583	0.92578	0.26216
	MICE	0.43894	0.91642	0.24858	0.45206	0.91223	0.24188	0.45735	0.90992	0.28534	0.45924	0.91157	0.28266	0.41908	0.92449	0.24617	0.40381	0.92995	0.25778
	GAIN	0.44547	0.91314	0.25318	0.46283	0.90768	0.23435	0.45894	0.90917	0.28162	0.45618	0.91274	0.26966	0.42936	0.92053	0.25762	0.41548	0.92593	0.25389
	DGAIN	0.44289	0.91499	0.25033	0.45880	0.90967	0.23631	0.45697	0.91005	0.28426	0.45492	0.91323	0.27057	0.42690	0.92146	0.25622	0.41482	0.92618	0.25923
	DDGAIN	0.43547	0.91766	0.24020	0.43960	0.91705	0.23412	0.44473	0.91514	0.26026	0.45295	0.91399	0.27875	0.41393	0.92639	0.25301	**0.38012**	**0.93790**	**0.22774**
0.2	Mean	0.47254	0.90372	0.31970	0.49355	0.89550	0.26274	0.49515	0.89478	0.30883	0.48582	0.90081	0.30953	0.47868	0.90151	0.28797	0.46556	0.90699	0.28764
	KNN	0.46778	0.90571	0.29441	0.51142	0.88762	0.26478	0.49645	0.89393	0.30349	0.47319	0.90600	0.29543	0.46918	0.90498	0.26404	0.46104	0.90878	0.28129
	MICE	0.45590	0.90981	0.26564	0.54989	0.87012	0.26352	0.48178	0.89984	0.27900	0.45803	0.91204	0.27918	0.46356	0.90759	0.28521	0.41516	0.92590	0.26102
	GAIN	0.46886	0.90518	0.30073	0.50916	0.88859	0.26249	0.49791	0.89345	0.30986	0.46953	0.90758	0.28580	0.45326	0.91143	0.26177	0.49184	0.89615	0.31225
	DGAIN	0.49619	0.89373	0.29460	0.51760	0.88492	0.25685	0.50309	0.89132	0.30405	0.46202	0.91053	0.29990	0.44807	0.91383	0.25963	0.46109	0.90879	0.28252
	DDGAIN	0.45302	0.91016	0.26549	0.49704	0.89400	0.24944	0.44393	0.91538	0.27299	0.45170	0.91445	0.27409	0.42988	0.92055	0.25033	**0.38559**	**0.93620**	**0.23781**
0.3	Mean	0.50896	0.88821	0.35534	0.54229	0.87367	0.32304	0.52168	0.88302	0.32727	0.52062	0.88631	0.34446	0.51439	0.88606	0.29123	0.55212	0.86897	0.36176
	KNN	0.51676	0.88452	0.33417	0.53934	0.87503	0.30331	0.52018	0.88371	0.32442	0.52372	0.88474	0.31248	0.50912	0.88829	0.27527	0.51482	0.88622	0.33503
	MICE	0.46556	0.90540	0.26956	0.55910	0.86567	0.27205	0.48715	0.89815	0.29004	0.47221	0.90653	0.28286	0.55164	0.86915	0.31738	0.45223	0.91223	0.28446
	GAIN	0.49281	0.89521	0.31327	0.55318	0.86849	0.29583	0.52868	0.88002	0.32148	0.50541	0.89269	0.30427	0.53475	0.87702	0.28017	0.51567	0.88582	0.32179
	DGAIN	0.50856	0.88835	0.33731	0.53392	0.87749	0.29147	0.52373	0.88198	0.32595	0.50401	0.89328	0.30535	0.52488	0.88312	0.28850	0.49513	0.89478	0.31816
	DDGAIN	0.46465	0.90598	0.26127	0.53016	0.87937	0.26421	0.47374	0.90369	0.28406	0.46560	0.90911	0.28582	0.50839	0.88868	0.29634	**0.42469**	**0.92258**	**0.25235**
0.4	Mean	0.53681	0.87569	0.40458	0.47916	0.87126	0.31946	0.50420	0.89079	0.33620	0.63781	0.82879	0.43417	0.52921	0.87857	0.31845	0.61992	0.83458	0.43432
	KNN	0.51490	0.88849	0.34911	0.52219	0.88301	0.32564	0.50189	0.89187	0.34327	0.57434	0.86117	0.37592	0.52365	0.88168	0.31577	0.50374	0.89105	0.29860
	MICE	0.50118	0.88975	0.31411	0.60573	0.84124	0.31696	0.52114	0.88313	0.32526	0.47912	0.90376	0.28529	0.51975	0.88305	0.29521	0.45540	0.91101	0.28839
	GAIN	0.49465	0.89436	0.35491	0.55901	0.86533	0.35654	0.51689	0.88536	0.32424	0.56312	0.86659	0.36510	0.52323	0.88225	0.32240	0.59306	0.84877	0.40304
	DGAIN	0.49528	0.89399	0.34495	0.54470	0.87242	0.32303	0.50523	0.89044	0.30148	0.55195	0.87119	0.37889	0.54218	0.87297	0.32779	0.53808	0.87548	0.34981
	DDGAIN	0.47027	0.90347	0.27745	0.54167	0.87413	0.29657	0.46438	0.90744	0.28227	0.47269	0.90631	0.28793	0.51722	0.88507	0.31069	**0.43252**	**0.91968**	**0.26955**
0.5	Mean	0.60547	0.84171	0.46558	0.56462	0.86256	0.38730	0.63940	0.82427	0.44272	0.78485	0.74133	0.52881	0.55977	0.86273	0.35966	0.74210	0.76225	0.54587
	KNN	0.51839	0.87706	0.36195	0.57013	0.86054	0.38032	0.57171	0.86161	0.41824	0.63149	0.82399	0.43503	0.57933	0.85347	0.36101	0.57773	0.85951	0.34554
	MICE	0.48719	0.89388	0.30716	0.64423	0.82154	0.35837	0.56964	0.85932	0.35584	0.50990	0.89086	0.30044	0.59016	0.85030	0.36459	0.51759	0.88504	0.33522
	GAIN	0.55990	0.86437	0.40724	0.58893	0.85088	0.39031	0.50460	0.89074	0.32264	0.77215	0.74795	0.51563	0.55252	0.86891	0.34261	0.75809	0.75221	0.53151
	DGAIN	0.57920	0.85570	0.41224	0.55864	0.86601	0.36291	0.56175	0.86311	0.38867	0.75020	0.76196	0.49576	0.55025	0.87008	0.36453	0.58465	0.85330	0.36004
	DDGAIN	0.48562	0.89650	0.30468	0.59654	0.84726	0.32839	0.49286	0.89533	0.31116	0.48962	0.89944	0.31140	0.53304	0.87803	0.32675	**0.48053**	**0.90090**	**0.30426**

The bold entities mean the optimal performance of all models on the corresponding index under the same conditions.

achieves the worst prediction in the case of mean imputation, since three parallel networks lead to error accumulation in the case of poor imputation accuracy. By contrast, HCNN which efficiently takes into account the temporal and variable information of dynamic data performs the best among the four models. HCNN outperforms CNN, TCN, and MDCNN in all imputation methods except mean imputation, reflecting the robustness of the proposed method. Especially, the more accurate the imputation, the better the model performance. The prediction performance of the robust dual-rate data-driven quality prediction with the combination of DDGAIN and HCNN is the best. In the case of 0.5 missing rates, the prediction performance still performs well. Moreover, the prediction accuracy of DDGAIN-HCNN is much higher than that of GAIN-HCNN, indicating the strong robustness of the DDGAIN model.

The prediction results of five models with DDGAIN imputation under the 0.5 missing rate condition are shown in Fig. 8. It can be seen that MLP, LSTM, and CNN do not predict well during the 800 to 1000 samples. The TCN performs better than LSTM, but still does not predict the trend of impurity values well. Whereas MDCNN-based prediction fluctuates significantly, HCNN that makes use of intersample temporal features and variable spatial features achieves a more accurate prediction. The grayscale image with superimposed information is shown in Fig. 9. Fig. 9(a) displays the image temporal information vector, while Fig. 9(b) displays the variable spatial information vector, with Fig. 9(c) the information matrix to be superimposed, i.e., $\mathbf{T} \oplus \mathbf{V}$. The difference of information for each moment and each variable can be seen in Fig. 9(a) and (b), respectively. From Fig. 9(c), the information to be superimposed is different for each element of the data matrix $\mathbf{X}$, indicating the dynamic information of the process data used for prediction.

B. Experiments on Beijing Multisite Air Quality Dataset

1) Beijing Multisite Air Quality Dataset Description: The Beijing multisite air quality dataset [43] includes hourly pollution data from 12 air quality monitoring sites. Since the quality variable $PM_{2.5}$ is usually influenced by meteorological conditions, meteorological data from around the stations are included. That is, the data for each station consisted of air quality data and meteorological data, for a total of 12 valid variables, as shown in Table III. The data from the Wanshugong site are selected for the validation of the proposed method, with the nonnumerical wind data discarded. The remaining 10 process variables are sampled on an hourly basis and the quality data are sampled every four hours. Each sample consists of fine-grained data within a moving window, as shown in the red box in Fig. 3. The window length is 4. A number of 7000 samples are used for model training, 500 samples for model verification, and 707 samples for model testing.

TABLE VII
ABLATION EXPERIMENTAL RESULTS OF HCNN

Method	RMSE	R^2	MAE
CNN [24]	0.44473	0.91514	0.26026
HCNN (T)	0.40923	0.92813	0.25122
HCNN (V)	0.39408	0.93332	0.24185
HCNN (T+V)	**0.38012**	**0.93790**	**0.22774**

The bold entities mean the optimal performance of all models on the corresponding index under the same conditions.

2) Modeling and Result Analysis: The Adam optimizer is adopted for the proposed method and the baselines. The optimal parameters for the GAIN, DGAIN, and DDGAIN models are as follows: batchsize is 25, learning rate is 0.001, α is 100, and the number of epoch is 30 000. The optimal parameters of HCNN are set as follows: batch size is 75, learning rate is 0.001, and the number of epoch is 100.

In the imputation experimental part, the window lengths for DGAIN and DDGAIN are 4. The results are shown in Table IV. From this table, DDGAIN achieves the best performance for 0.1 to 0.5 missing rates. The imputation accuracy is higher than that of MICE. It also illustrates that in data with dynamic relationships, DDGAIN not only taps into the dynamic relationships, but also senses the time-series changes of variables. The data

distributions using the proposed DDGAIN at 0.5 missing rates are shown in Fig. 10.

In the prediction experimental part, MLP uses a (10, 5, 1) structure, LSTM hidden layer has a number of 5 features, TCN uses two temporal Blocks with a channel number of 5, and the convolution kernel size is 3. The basic structure and convolution kernel size of the CNN, MDCNN, and HCNN remain unchanged, and the fully connected layer is (200, 1). The dynamic window size of LSTM, CNN, TCN, MDCNN, and HCNN is set to (10×4). The prediction results are shown in Table VI. The DDGAIN shows significant advantages. The proposed HCNN is better than the other five methods, especially at the missing rate of 0.1 to 0.3. Taking 0.1 missing rates as an example, HCNN is superior to MLP, LSTM, CNN, TCN, and MDCNN under Mean, KNN, GAIN, DGAIN, and DDGAIN imputation methods. Meanwhile, the DDGAIN-HCNN method performs the best at various missing rates. The prediction curves for the six prediction models using DDGAIN imputation at 0.1 missing rates is shown in Fig. 11. It can be seen that the green curve (HCNN) fits the actual values well.

Ablation experiments for the HCNN are carried out using DDGAIN imputation method with a 0.1 missing rates. The results are shown in Table VII. It can be seen that HCNN achieves a better performance than CNN using temporal information T alone and variable information V. HCNN with a dual hint mechanism (T+V) performs the best.

V. Conclusion

In this article, a robust dual-rate dynamic data modeling method, i.e., DDGAIN-HCNN, has been proposed for quality prediction of dynamic nonlinear processes with missing values. The highlights of this article are two-fold: 1) The proposed DDGAIN provides a new way for dynamic data imputation of missing values; 2) The proposed HCNN quality prediction model addresses the limitations of traditional methods by superimposing the temporal and variable information of process data in the original data through an information hint mechanism. Future work will focus on the following three aspects:

1) incorporating attention mechanism to extract far apart dynamics;
2) developing quality-relevant data imputation method;
3) extending dual-rate data modeling to multirate data modeling for quality prediction.

References

[1] F. A. Souza, R. Araújo, and J. Mendes, "Review of soft sensor methods for regression applications," *Chemometrics Intell. Lab. Syst.*, vol. 152, pp. 69–79, 2016.

[2] Z. Yang and Z. Ge, "Industrial virtual sensing for big process data based on parallelized nonlinear variational Bayesian factor regression," *IEEE Trans. Instrum. Meas.*, vol. 69, no. 10, pp. 8128–8136, Oct. 2020.

[3] L. Yao et al., "Virtual sensing f-CaO content of cement clinker based on incremental deep dynamic features extracting and transferring model," *IEEE Trans. Instrum. Meas.*, vol. 70, Jul. 2020, Art. no. 2500610.

[4] Y. Liu, C. Yang, M. Zhang, Y. Dai, and Y. Yao, "Development of adversarial transfer learning soft sensor for multigrade processes," *Ind. Eng. Chem. Res.*, vol. 59, no. 37, pp. 16330–16345, 2020.

[5] J. Zheng, Z. Song, and Z. Ge, "Probabilistic learning of partial least squares regression model: Theory and industrial applications," *Chemometrics Intell. Lab. Syst.*, vol. 158, pp. 80–90, 2016.

[6] H. Kaneko and K. Funatsu, "Application of online support vector regression for soft sensors," *AIChE J.*, vol. 60, no. 2, pp. 600–612, 2014.

[7] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 785–794.

[8] J. Gonzaga, L. A. C. Meleiro, C. Kiang, and R. Maciel Filho, "Ann-based soft-sensor for real-time process monitoring and control of an industrial polymerization process," *Comput. Chem. Eng.*, vol. 33, no. 1, pp. 43–49, 2009.

[9] C. Shang, F. Yang, D. Huang, and W. Lyu, "Data-driven soft sensor development based on deep learning technique," *J. Process Control*, vol. 24, no. 3, pp. 223–233, 2014.

[10] Q. Sun and Z. Ge, "A survey on deep learning for data-driven soft sensors," *IEEE Trans. Ind. Inform.*, vol. 17, no. 9, pp. 5853–5866, Sep. 2021.

[11] J. Maggu and A. Majumdar, "Supervised kernel transform learning," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, 2019, pp. 1–8.

[12] J. Maggu, A. Majumdar, E. Chouzenoux, and G. Chierchia, "Deep convolutional transform learning," in *Proc. Int. Conf. Neural Inf. Process.*, 2020, pp. 300–307.

[13] J. Maggu and A. Majumdar, "Robust transform learning," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2017, pp. 1467–1471.

[14] P. Gupta, J. Maggu, A. Majumdar, E. Chouzenoux, and G. Chierchia, "Deconfuse: A deep convolutional transform-based unsupervised fusion framework," *EURASIP J. Adv. Signal Process.*, vol. 20, no. 1, pp. 1–32, 2020.

[15] J. Maggu and A. Majumdar, "Greedy deep transform learning," in *Proc. IEEE Int. Conf. Image Process.*, 2017, pp. 1822–1826.

[16] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," 2014. [Online]. Available: http://arxiv.org/abs/1312.6114

[17] Q. Sun and Z. Ge, "Gated stacked target-related autoencoder: A novel deep feature extraction and layerwise ensemble method for industrial soft sensor application," *IEEE Trans. Cybern.*, vol. 52, no. 5, pp. 3457–3468, May 2022.

[18] J. Davis, "An introduction to neural networks," *J. Cogn. Neurosci.*, vol. 8, no. 4, pp. 383–383, 1996.

[19] L. Zhou, Y. Wang, Z. Ge, and Z. Song, "Multirate factor analysis models for fault detection in multirate processes," *IEEE Trans. Ind. Inform.*, vol. 15, no. 7, pp. 4076–4085, Jul. 2019.

[20] Z. Chai, C. Zhao, and B. Huang, "Variational progressive-transfer network for soft sensing of multirate industrial processes," *IEEE Trans. Cybern.*, vol. 52, no. 12, pp. 12882–12892, Dec. 2022.

[21] M. E. Tipping and C. M. Bishop, "Probabilistic principal component analysis," *J. Roy. Statist. Soc.: Ser. B. (Statist. Methodol.)*, vol. 61, no. 3, pp. 611–622, 1999.

[22] Z. Ge, B. Huang, and Z. Song, "Mixture semisupervised principal component regression model and soft sensor application," *AIChE J.*, vol. 60, no. 2, pp. 533–545, 2014.

[23] L. Bao, X. Yuan, and Z. Ge, "Co-training partial least squares model for semi-supervised soft sensor development," *Chemometrics Intell. Lab. Syst.*, vol. 147, pp. 75–85, 2015.

[24] K. Wang, C. Shang, L. Liu, Y. Jiang, D. Huang, and F. Yang, "Dynamic soft sensor development based on convolutional neural networks," *Ind. Eng. Chem. Res.*, vol. 58, no. 26, pp. 11521–11531, 2019.

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

[26] X. Yuan, S. Qi, Y. A. Shardt, Y. Wang, C. Yang, and W. Gui, "Soft sensor model for dynamic processes based on multichannel convolutional neural network," *Chemometrics Intell. Lab. Syst.*, vol. 203, 2020, Art. no. 104050.

[27] X. Jiang and Z. Ge, "Augmented multidimensional convolutional neural network for industrial soft sensing," *IEEE Trans. Instrum. Meas.*, vol. 70, Apr. 2021, Art no. 2508410.

[28] Z. Yao and C. Zhao, "FIGAN: A. missing industrial data imputation method customized for soft sensor application," *IEEE Trans. Automat. Sci. Eng.*, vol. 19, no. 4, pp. 3712–3722, Oct. 2022.

[29] J. Josse and F. Husson, "Multiple imputation in principal component analysis," *Adv. Data Anal. Classification*, vol. 5, no. 3, pp. 231–246, 2011.

[30] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Statist. Soc.: Ser. B. (Methodological)*, vol. 39, no. 1, pp. 1–22, 1977.

[31] D. J. Stekhoven and P. Bühlmann, "Missforest—non-parametric missing value imputation for mixed-type data," *Bioinformatics*, vol. 28, no. 1, pp. 112–118, 2012.
[32] J. Yoon, J. Jordon, and M. Schaar, "GAIN: Missing data imputation using generative adversarial nets," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 5689–5698.
[33] M. A. Islam, S. Jia, and N. D. Bruce, "How much position information do convolutional neural networks encode?," 2020. [Online]. Available: https://openreview.net/forum?id=rJeB36NKvB
[34] W. Zhu, Y. Ma, Y. Zhou, M. Benton, and J. Romagnoli, "Deep learning based soft sensor and its application on a pyrolysis reactor for compositions predictions of gas phase components," *Comput. Aided Chem. Eng.*, vol. 44, pp. 2245–2250, 2018.
[35] X. Liu, X. Wang, L. Zou, J. Xia, and W. Pang, "Spatial imputation for air pollutants data sets via low rank matrix completion algorithm," *Environ. Int.*, vol. 139, 2020, Art. no. 105713.
[36] J. Maggu and A. Majumdar, "Alternate formulation for transform learning," in *Proc. 10th Indian Conf. Comput. Vis., Graph. Image Process.*, 2016, pp. 1–8.
[37] S. J. Qin, "Recursive PLS algorithms for adaptive data modeling," *Comput. Chem. Eng.*, vol. 22, no. 4/5, pp. 503–514, 1998.
[38] B. Braun et al., "Data science challenges in chemical manufacturing," in *Proc. IFAC World Congress*, Berlin, Germany, 2020.
[39] S. J. Qin et al., "Integration of process knowledge and statistical learning for the Dow data challenge problem," *Comput. Chem. Eng.*, vol. 153, 2021, Art. no. 107451.
[40] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vol. IT-13, no. 1, pp. 21–27, Jan. 1967.
[41] S. Van Buuren and K. Groothuis-Oudshoorn, "MICE: Multivariate imputation by chained equations in R," *J. Statist. Softw.*, vol. 45, pp. 1–67, 2011.
[42] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018. [Online]. Available: http://arxiv.org/abs/1803.01271
[43] S. Zhang, B. Guo, A. Dong, J. He, Z. Xu, and S. X. Chen, "Cautionary tales on air-quality improvement in Beijing," *Proc. Roy. Soc. A: Math., Phys. Eng. Sci.*, vol. 473, no. 2205, 2017, Art. no. 20170457.

Xiangan Meng received the B.Eng. degree in automation from the Institute of Electrical Engineering, Yanshan University, Qinhuangdao, China, in 2021. He is currently working toward the M.S. degree in control science and engineering with the State Key Laboratory of Synthetical Automation for Process Industry, Northeastern University, Shenyang, China.

His current research interests include deep learning, data driven quality prediction, process monitoring, and process data analytics.

Qiang Liu (Senior Member, IEEE) received the B.S., M.S., and Ph.D. degrees in control theory and engineering from Northeastern University, Shenyang, China, in 2003, 2006, and 2012, respectively.

He is currently a Full Professor with the State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang, China. He was a Research Associate with the Department of Chemical Engineering, University of Southern California, Los Angeles, CA, USA from September 2014 to October 2016. He has authored or coauthored more than 70 peer-reviewed papers. His research interests include big data analytics, machine learning, statistical process monitoring, and fault diagnosis of complex industrial processes.

Prof. Liu was the recipient of the Outstanding Young Scholar of Liaoning Revitalization Talents Program, China. His paper titled "Perspectives on Big Data Modeling of Process Industries" was selected as one of the F5000-Top academic papers in Chinese top-quality Science Citation Index (SCI) tech Journals in 2019.

Chao Yang (Graduate Student Member, IEEE) received the M.Eng. degree in power engineering and engineering thermophysics from the Zhejiang University of Technology, Hangzhou, China, in 2019. He is currently working toward the Ph.D. degree in control science and engineering with the State Key Laboratory of Synthetical Automation for Process Industry, Northeastern University, Shenyang, China.

His current research interests include deep learning, transfer learning, process monitoring, fault diagnosis, and process data analytics.

Le Zhou (Member, IEEE) received the B.Eng. and Ph.D. degrees in control science and engineering from the Department of Control Science and Engineering, Zhejiang University, Hangzhou, China, in 2010 and 2015, respectively.

He was a Visiting Scholar with the Viterbi School of Engineering, University of Southern California, Los Angeles, Los Angeles, CA, USA, from December 2013 to June 2014, and a Visiting Scholar with the Department of Chemical Engineering, National Tsinghua University, Hsinchu, Taiwan, from January 2019 to March 2019. He is currently a Professor with the School of Automation and Electrical Engineering, Zhejiang University of Science and Technology, Hangzhou, China. His research interests include industrial process modeling, monitoring and fault diagnosis, soft sensor modeling, and deep learning.

Yiu-Ming Cheung (Fellow, IEEE) received the Ph.D. degree in computer science from the Department of Computer Science and Engineering, Chinese University of Hong Kong, Hong Kong, in 2000.

He is currently a Chair Professor of the Department of Computer Science, Hong Kong Baptist University, Hong Kong. His research interests include machine learning and visual computing, as well as their applications in data science, pattern recognition, and optimization.

Dr. Cheung is also a fellow of American Association for the Advancement of Science (AAAS), The Institution of Engineering and Technology (IET), and British Computer Society (BCS). Since 2023, he has been the Editor-in-Chief of IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTATIONAL INTELLIGENCE. Also, he is currently an Associate Editor for several prestigious journals, including IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS ON COGNITIVE AND DEVELOPMENTAL SYSTEMS, IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS (2014–2020), *Pattern Recognition*, to name a few.