DMGAN: Adversarial Learning-Based Decision Making for Human-Level Plant-Wide Operation of Process Industries Under Uncertainties

Nianzu Zheng, Jinliang Ding, *Senior Member, IEEE*, and Tianyou Chai, *Fellow, IEEE*

***Abstract*— To achieve plant-wide operational optimization and dynamic adjustment of operational index for an industrial process, knowledge-based methods have been widely employed over the past years. However, the extraction of knowledge base is a bottleneck for most existing approaches. To address this problem, we propose a novel framework based on the generative adversarial networks (GANs), termed as decision-making GAN (DMGAN), which directly learns from operational data and performs human-level decision making of the operational indices for plant-wide operation. In the proposed DMGAN, two adversarial criteria and three cycle consistency criteria are incorporated to encourage efficient posterior inference. To improve the generalization power of a generator with an increasing complexity of the industrial processes, a reinforced U-Net (RU-Net) is presented that improves the traditional U-Net by providing a more general combinator, a building block design, and drop-level regularization. In this article, we also propose three quantitative metrics for assessing the plant-wide operation performance. A case study based on the largest mineral processing factory in Western China is carried out, and the experimental results demonstrate the promising performance of the proposed DMGAN when compared with decision-making based on domain experts.**

***Index Terms*— Decision-making generative adversarial network (DMGAN), industrial processes, knowledge dissimilarity assessment, plant-wide operation, reinforced U-Net (RU-Net).**

I. Introduction

With the development of process industries, their operation becomes more challenging due to the presence of uncertainty, nonlinearity, and complex dynamics in industrial processes. In plant-wide operations, the dynamic adjustment of the operational indices of different process units in a plant requires high-level technical and domain knowledge, which hinders the development of process industries toward a green and intelligent direction [1]–[4]. Therefore, the role of intelligence systems for knowledge modeling and decision making of plant-wide operations has received considerable attention from both academics and industrial practitioners.

Several knowledge-based approaches, such as fuzzy cognitive maps (FCMs) [5], semantic networks [6], fuzzy rules [7], Bayesian networks [8], case-based reasoning (CBR) [4], and association rule mining (ARM) [3], [9], have been proposed to build intelligent knowledge systems. Chai *et al.* [1] suggested a hybrid knowledge system based on CBR, which combines multiobjective optimization, performance index prediction, and dynamic tuning techniques for plant-wide operational optimization. An ARM-based approach for dynamic adjustment of the operational index in industrial processes was developed in [3] and [10]. Although traditional knowledge-based methods for dynamic adjustment of the operational index have been applied in industrial processes, there is still a considerable room for further improvement as the system complexity and the number of decision variables increase. Iterative approach has the poor solving efficiency, and the ambiguity in experts' knowledge of decision-making systems is difficult to be addressed. In addition, the self-learning ability is very weak under new conditions, and the individual differences among experts in terms of problem-solving abilities and methods have certain negative effects on system performance. The abilities to update the knowledge base and to generate new knowledge are also quite limited. All these factors contribute to the "knowledge bottleneck," which limits the development of intelligent knowledge-based systems.

Recently, the generative adversarial network (GAN) [11]–[13] has offered a promising approach for modeling complex distributions based on a game-theoretic formulation. Motivated by the successes of GANs in various applications, a novel framework based on GAN, termed as the decision-making GAN (DMGAN), is proposed to solve the problem of knowledge bottlenecks in plant-wide operation of process industries. In the DMGAN, the plant-wide operational knowledge is treated as a directed probabilistic model. The proposed framework consists of an encoder, a generator, two discriminators in the latent space that are encoded by the encoder, and the decision space. Two adversarial discriminators aim to force the aggregated posterior distribution of encoder to be close to the prior distribution and match the distribution defined by generator with the prior knowledge of the industrial processes. To the best of our knowledge, the work reported in this article is original and represents the first time that GAN is applied for industrial processes.

Manuscript received April 27, 2018; revised November 20, 2018, May 11, 2019, September 14, 2019, and February 23, 2020; accepted March 5, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 61988101, Grant 61525302, and Grant 61590922, in part by the National Key Research and Development Program of China under Grant 2018YFB1701104, and in part by the Xingliao Plan of Liaoning Province under Grant XLYC1808001. *(Corresponding author: Jinliang Ding.)*

The authors are with the State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang 110819, China (e-mail: nianzuzheng.neu@gmail.com; jlding@mail.neu.edu.cn; tychai@mail.neu.edu.cn).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2020.2979800

The experimental results demonstrate that the proposed DMGAN performs efficiently and reasonably well in terms of approximate inference, self-learning, and knowledge discovery. The proposed approach can achieve human-level performance in plant-wide operations of process industries. Our technical contributions can be summarized as follows.

1) A novel DMGAN for decision making is proposed to solve the problem of knowledge bottlenecks in plant-wide operations. The DMGAN can learn a generator that utilizes both the latent code and conditional variables to output operational indices adjustments (OIAs) and predicts the distribution over the latent variables.
2) Two adversarial training criteria and three cycle consistency training criteria are introduced to encourage the bijection between the latent space and the decision space. Additionally, a conditional mapping is constructed from the target space to the decision space, which allows us to perform efficient posterior inference using simple sampling from the latent space with a conditional constraint.
3) An efficient reinforced U-Net (RU-Net) in the DMGAN is proposed to improve the generalization ability of the generator and match the complexities of industrial problems. In the RU-Net, the building block design and a general combinator are used to enhance local representation and feature propagation. A drop-level approach is also presented to prevent the RU-Net from overfitting.
4) A knowledge dissimilarity assessment based on Parzen window probability density estimation and mutual information (MIKDA) is defined to measure the match degree between a candidate solution in the decision space and the expert knowledge.

The remainder of this paper is organized as follows. Section II introduces the GAN concept and summarizes the development of deep neural networks (DNNs). Section III formulates the problem and describes the objectives of plant-wide operations for the industrial processes. The proposed DMGAN framework is discussed in Section IV. Section V presents the implementation details, including the RU-Net and evaluation metrics. Section VI reports the experimental results. This article is concluded in Section VII.

II. Preliminaries

A. Generative Adversarial Networks (GANs)

Recently, GANs have shown promising results in modeling complex multimodal data and synthesizing real-world data; thus, they have a wide variety of applications, such as image generation [14]–[19], representation learning [20]–[23], text-to-image tasks [24], [25], video application [26]–[29], and transfer learning [30], [31]. Before introducing the GAN into knowledge extraction of the industrial processes, the essential points of GAN are reviewed briefly.

The principle of GAN [11] is to establish a two-player min-max game between two models, namely, a generator G and a discriminator D. The generator G takes the input, which is a latent noise vector sampled from a prior distribution, and produces a sample. The discriminator D performs binary classification to determine whether the sample is generated from G or from training. Thus, two models are competed as follows. The generator G tries to fool the discriminator by modeling the data distribution, while the discriminator D tries to determine whether the given sample is sampled from the data distribution that we are trying to model rather than from our generator. The minmax game can be expressed as

$$\min_G \max_D \mathbb{E}_{x\sim p_{\text{data}}(x)}\big[\log D(x)\big] + \mathbb{E}_{z_r\sim p(z)}[1 - D(G(z_r))] \quad (1)$$

where z_r denotes a noise vector randomly sampled from a prior distribution $p(z)$ (e.g., Gaussian) and $p_{\text{data}}(x)$ denotes the distribution of training data. Both the generator G and the discriminator D are optimized alternately.

The conditional GANs (CGANs) are an extension of the GAN, where both D and G receive an additional vector of information ℓ as input [32]. Its objective thus becomes

$$\min_G \max_D \mathbb{E}_{x,\ell\sim p_{\text{data}}(x,\ell)}\big[\log D(x,\ell)\big] + \mathbb{E}_{z_r\sim p(z),\ell\sim p(\ell)}[1 - D(G(z_r,\ell),\ell)] \quad (2)$$

where $p(\ell)$ is the prior distribution over conditional variables ℓ. This model allows the output of the generative model to be controlled by conditional variables.

The adversarial autoencoder (AAE), which incorporates a probabilistic autoencoder with GAN, is proposed to perform variational inference. Let x be the input data and z be the latent code vector of the autoencoder. Let $p_\theta(z)$ be the prior distribution we want to impose on the code and $p_\theta(x|z)$ be the probabilistic decoder. Given a code z, a distribution over the possible corresponding value of x is produced. A variational autoencoder (VAE) [33] introduces a recognition model $q_\phi(z|x)$ (also referred to as the probabilistic encoder in autoencoder parlance) to estimate the intractable true posterior $p_\theta(z|x)$. To encourage the posterior $q_\phi(z|x)$ to be close to the true prior $p_\theta(x|z)$, VAE minimizes the following Kullback–Leibler (KL) divergence between the two distributions with respect to the variational parameter ϕ and the generative parameter θ as

$$\mathbb{E}_{x-p_d(x)} D_{\text{KL}}\big(q_\phi(z|x)||p_\theta(z|x)\big) = \mathcal{L}(\theta,\phi;x) + \mathbb{E}_x p_\theta(x) \quad (3)$$

where the second term $\mathbb{E}_x p_\theta(x)$ is the marginal likelihood, which can be viewed as a constant term. That is, the upper bound of the negative log likelihood of x is minimized as

$$\mathcal{L}(\theta,\phi;x) = D_{\text{KL}}\big(q_\phi(z|x)||p_\theta(z)\big) - \mathbb{E}_x\mathbb{E}_{q_\phi(z|x)} p_\theta(x|z) \quad (4)$$

where the first term $D_{\text{KL}}(q_\phi(z|x)||p_\theta(z))$, which is interpreted as the regularization term, can be integrated analytically; the second term $\mathbb{E}_x\mathbb{E}_{q_\phi(z|x)} p_\theta(x|z)$ is the expectation reconstruction error term. The AAE [34] is similar to the VAE but uses an adversarial training procedure to replace the KL divergence to encourage the posterior of the hidden code vector to match the arbitrary prior distribution. The implementation process of AAE consists of the reconstruction phase and the regularization phase. The AAE first updates its discriminator to discriminate the aggregated prior from the approximate posterior, and then the encoder is trained to confuse the discriminator. Both phases are trained alternately with a stochastic gradient descent algorithm.

B. Deep Convolutional Neural Networks (DCNNs)

The exploration of the network architectures of DCNNs has recently demonstrated impressive performance in complex machine learning tasks such as image classification [35]–[42] and segmentation [43], [44]. We summarized existing DCNN algorithms that were divided into multilevel feature learning algorithms and stacked building block ones.

Although different architectures vary in terms of network topologies and training algorithms, they all share a key characteristic. They create a skip architecture that combines appearance information from early layers with semantic information from later layers. Utilizing multilevel features through skip connections has generally returned performance dividends [44]–[46], and a new research problem emerges. How can high-level information be combined with low-level information to obtain the desired underlying mapping?

In the light of [42], we introduce a combination function, denoted as $g(\cdot,\cdot)$, which aims to merge any two or more feature blobs into the desired underlying mapping $\mathcal{H}(\mathbf{x})$. Recent approaches that attempt to address it or related problems can be grouped into five categories: the addition operations $g_1(\upsilon,u)=\upsilon+u$, the concatenation operations $g_2(\upsilon,u)=[\upsilon,u]$, the mean operations $g_3(\upsilon,u)=\upsilon\oplus u$, the Gaussian combination functions [39] [where the weight and prior are modeled as a function of the high-level signal u, as shown in (5)], and the autoassociative multilayer perceptron combination functions (where multilayer perceptron is used to map three scalars $[\upsilon,u,\upsilon\odot u]$ into a single output) [42]

$$\begin{aligned} g_4(\upsilon,u)&=(\upsilon-\mu(u))\odot\sigma(u)+\mu(u)\\ \mu(u)&=a_1\odot\text{sigmoid}(a_2\odot u+a_3)+a_4\odot u+a_5\\ \sigma(u)&=b_1\odot\text{sigmoid}(b_2\odot u+b_3)+b_4\odot u+b_5 \end{aligned}\tag{5}$$

where $\odot$ denotes an elementwise product, υ and u are the given features. The details of above equations can be found in [39] and [42].

Next, we further elaborate upon these categories. Highway networks [36] present short connections with gating functions, and a ResNet [37] bypasses the signal via identity connections. Both methods utilize the addition operation to combine features before they are passed into a layer, which is written as

$$\begin{aligned} \upsilon&=\mathbf{x},u=\mathcal{F}(\mathbf{x})\\ g_1(\upsilon,u)&=\mathcal{F}(\mathbf{x})+\mathbf{x} \end{aligned}\tag{6}$$

where $\mathcal{F}(\cdot)$ represents a nonlinear mapping network.

DenseNet [38] concatenates the feature maps of all preceding layers as inputs

$$\begin{aligned} \upsilon&=\left[\mathbf{x}_0,\mathbf{x}_1,\ldots,\mathbf{x}_{\ell-1}\right],\quad u=\mathcal{F}(\upsilon)\\ g_2(\upsilon,u)&=[\upsilon,\mathcal{F}(\upsilon)]. \end{aligned}\tag{7}$$

FractalNet [41] repeatedly combines several parallel layer sequences with a mean operation, which can be described as

$$\begin{aligned} \upsilon&=\mathcal{F}_1(\mathbf{x}),\quad u=\mathcal{F}_2(\mathbf{x})\\ g_3(\upsilon,u)&=\mathcal{F}_1(\mathbf{x})\oplus\mathcal{F}_2(\mathbf{x}) \end{aligned}\tag{8}$$

where $\mathcal{F}_1$ is a nonlinear function that consists of two truncated fractals and $\mathcal{F}_2$ is a single convolutional layer.

The above-combined functions require the same size of feature maps. However, as an essential part of DNN, the downsampling layer changes the size of the feature maps. To facilitate downsampling, the building block design is naturally taken into consideration and widely used in the construction of DCNN. The main idea of designing a network is that, according to the complexity of the problem under consideration, the network is divided into multiple building blocks. Then, an optimal local representational structure (also known as a block) is required to be designed in the network architecture. In addition, blocks such as residual blocks [37], dense blocks [38], and inception modules [40] are stacked (optionally followed by batch normalization and max-pooling) to construct the final desired network structure. Herein, we refer to the building block as a method to enhance the local representation.

Aiming to let the model have enough capacity to describe the dynamics of operational indices, the utilization of both a combination function and a building block is presented in the proposed RU-Net in Section V.

III. Problem Scenario

A. Plant-Wide Operation of the Production Line

A typical hierarchical control architecture in the process industries is illustrated in Fig. 1. At the top of this architecture, a planning and scheduling system determines the targets of plant production indices Q_k, denoted as $Q_k^*(k=1,2,\ldots)$, and their ranges $(Q_{k,\min},Q_{k,\max})$, where the production indices can be the product quality, yield, consumption, and so on; $Q_{k,\min}$ and $Q_{k,\max}$ denote the lower and upper boundaries of corresponding production targets, respectively. The outputs of this layer are sent to the technical departments, which generate the target values and ranges of the process automation systems (PASs) of each process unit, namely the operational indices r_i, as denoted by $r_i^*\sim\{r_{ij}^*\}$ and $(r_{ij,\min},r_{ij,\max})$, respectively, where $i=1,2,\ldots,I$ denotes the index of the process unit and $j=1,2,3$ represents the quality, yield, and consumption of the intermediate product of the unit processes. According to the target r_i^*, the optimal operation control (OOC) level of the PAS first computes the set points of the control loops, which are sent to the lower process control subsystem (PCS) of the PAS. The PCS then computes the control actions [1], which are necessary for the plant to reach the set points. The main drawback of the current information architecture is that, from the upper planning and scheduling system to the process control system, the skilled technicians are still required to accomplish a large amount of interactive adjustment, which cannot guarantee the optimization of production operation. In this context, we pay more attention on the upper plant-wide operation and dynamic adjustment of the process operation.

Plant-wide operation, in a narrow sense, is the dynamic adjustment of operational indices, which aims to suppress the effect of uncertainty on the production process. Due to the complexity and uncertainty of an industrial process, human experience is sometimes random and inaccurate, and it often causes production indices to deviate from the predetermined ranges. Thus, we introduce a knowledge framework to realize

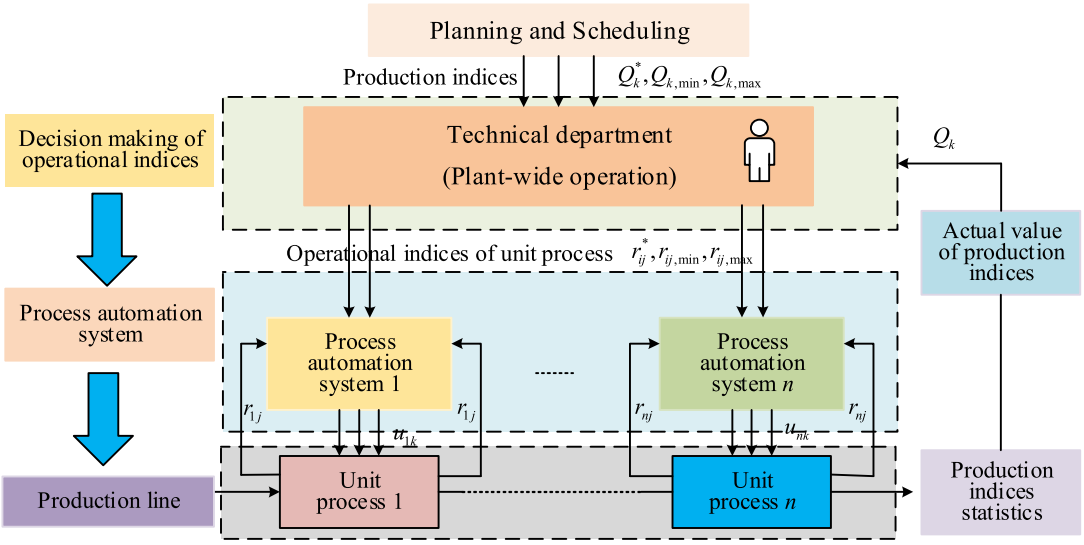

Fig. 1. Hierarchical control architecture of the whole production line for an industrial process.

knowledge modeling and decision making of plant-wide operations, which were realized previously using a human expert team to describe the system architecture and behavior under different conditions.

B. Objective of Plant-Wide Operation

Herein, we present three objectives for the plant-wide operation in industrial processes.

1) *Model adjustment performance (MAP):* These indices are used to represent the differences between the target values Q_k^* and the actual values $Q_k(t)$ of the production indices. Therefore, the smaller the values, the closer the actual production indices are to their target values

$$\min \sum_k |Q_k(t) - Q_k^*|^2, \quad k = 1, 2, \ldots. \tag{9}$$

2) *Energy consumption (EC) of OIAs:* The objective, taken as the auxiliary criterion, should be minimized to encourage the minimum adjustments needed to achieve plant-wide operation and to avoid drastic changes to the industrial processes to ensure their safe operation

$$\min \sum_i \sum_j |\Delta \hat{r}_{ij}(t)|^2, \quad i = 1, 2, \ldots, I, \quad j = 1, 2, \ldots \tag{10}$$

where $\Delta \hat{r}_{ij}(t)$ is the OIAs of operational indices r_i.

3) *MIKDA:* To assess the dissimilarity between knowledge built by the knowledge framework and the human knowledge, we propose a performance objective and it should be minimized so that the knowledge framework is similar to the thinking of human experts. It is formulated as follows:

$$\min D_{\mathrm{KDA}}\left(\Delta \hat{r}_{ij}(t) || \Delta r_{ij}^*(t)\right), \quad i = 1, 2, \ldots, I, \quad j = 1, 2, \ldots \tag{11}$$

where $\Delta r_{ij}^*(t)$ is the OIAs of the targets of operational indices r_i^*. D_{KDA} presents a mutual information-based dissimilarity, which is defined in Section V.

IV. Decision-Making GANs

Based on the foundations from Section III, we first present the framework of the proposed DMGAN for the plant-wide operation of industrial processes. Let $\{(\Delta r_{ij}(t), \Delta Q_k(t))\}_{t=1}^N$ be the historical data set, where $\Delta r_{ij}(t) \subset \mathcal{A}$ denotes OIA, $\Delta Q_k(t) \subset \mathcal{B}$ denotes the production indices changes (PICs), and $t = 1, 2, \ldots, N$ represents the data samples in the data set and the number of samples is N. Let $\{\Delta r_{ij}(t), \Delta d(t)\}_{t=1}^N$ be the incremental operating condition (IOC) data set, where $s = \{\Delta r_{ij}(t), \Delta d(t)\} \subset \mathcal{S}$ denotes the IOC variables and $\Delta d(t)$ is the incremental working condition. Let $\{r_{ij}^*(t-1), \Delta Q_k(t), d(t)\}_{t=1}^N$ be the conditional variable data set, where $\ell = \{r_{ij}^*(t-1), \Delta Q_k, d\} \subset \mathcal{L}$ is the conditional variables (the sampling time t is omitted for simplicity unless stated otherwise), including the last sampling instant of the targeted operational indices $r_{ij}^*(t-1)$, the desired adjustment values of the production indices ΔQ_k, and the working condition $d(t)$. Let $p(z)$ be the prior distribution that we want to randomly sample and $p(s|z, \ell)$ be the decoding distribution.

In the DMGAN, the strategy is divided into two stages: the training phase and the running one, in which the generator G takes the noise sample from the prior distribution and the conditioning variable ℓ as inputs and generates adjustments $\Delta \hat{r}_{ij}(t)$. Then they are added to the initial value of operational indices $\bar{r}_{ij}(t)$ to create the modified value $r_{ij}^*(t)$ and it is taken as the new target of operational indices, as illustrated in Fig. 2

$$\begin{aligned} r_{ij}^*(t) &= \bar{r}_{ij}(t) + \Delta \hat{r}_{ij}(t) \\ \Delta \hat{r}_{ij}(t) &= G\left(z, r_{ij}^*(t-1), \Delta Q_k, d\right). \end{aligned} \tag{12}$$

Our focus is the training phase. As shown in Fig. 3, the proposed DMGAN consists of a generator G, an encoder E, and two discriminators on the latent variables and OIAs,

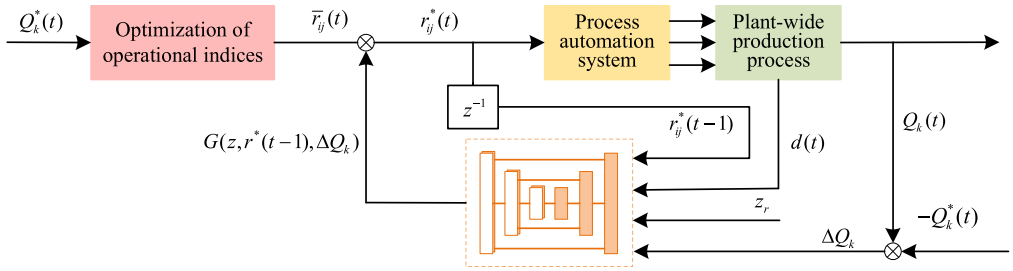

Fig. 2. Running phase of DMGAN for plant-wide operation of an industrial process.

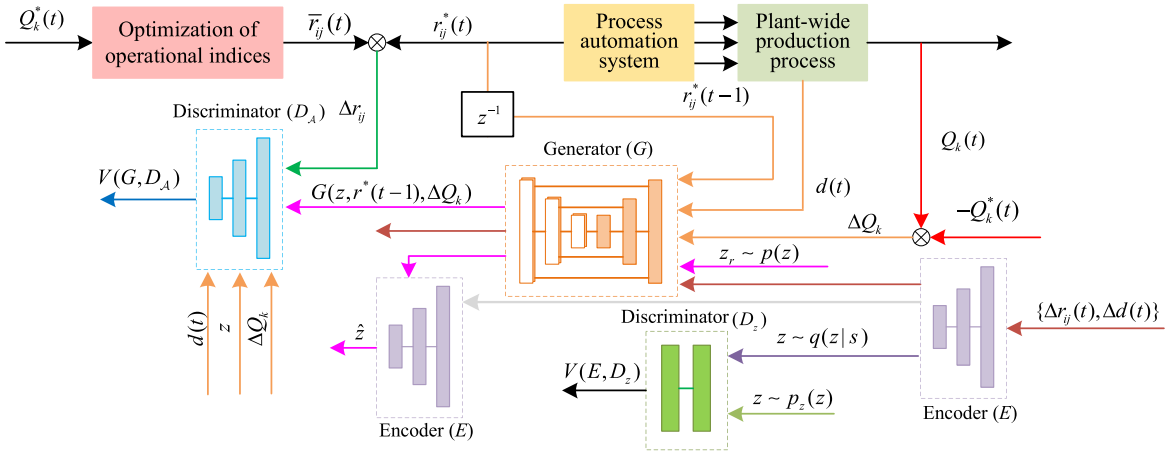

Fig. 3. Training phase of DMGAN for plant-wide operation of industrial process.

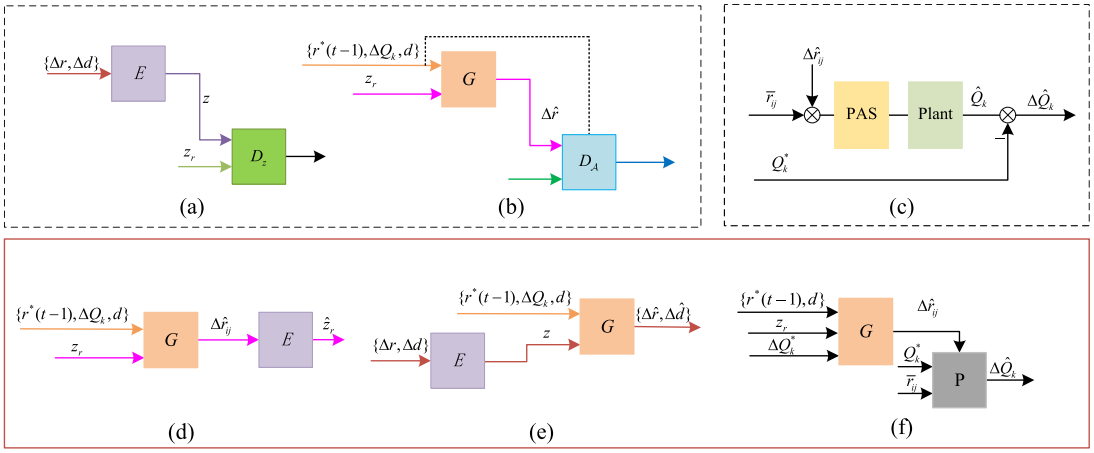

Fig. 4. Five training criteria of DMGAN for plant-wide operation of an industrial processes. (a) and (b) Adversarial training criteria. (d)–(f) Cycle consistency training criteria. (c) Structure of P in (f).

i.e., D_z and $D_{\mathcal{A}}$ respectively. Our goal is to learn a bijection between the decision space $\mathcal{A}$ and the latent space $\mathbb{Z}$, and a mapping from target space $\mathcal{B}$ to decision space $\mathcal{A}$. We explicitly encourage the bijective mapping between OIA and the latent variables z to ensure that generating from any part of the prior space $p(z)$ results in a meaningful OIA; that is, we not only learn a generator $p(s|z, \ell)$ that utilizes both the latent code z and conditional variable ℓ to output OIA but also jointly use a recognition network $q(z|s)$ (also called encoder E) to predict the distribution over the latent variables z. Therefore, the encoding distribution defines an aggregated posterior distribution $q(z) = \int_x q(z|s)p(s)ds$. The adversarial discriminators D_z aims to force the aggregated posterior distribution of encoder $q(z)$ to be close to the prior distribution $p(z)$, and $D_{\mathcal{A}}$ is used to match the distribution of OIA $p(\Delta r_{ij}|z, \ell)$ defined by generator G with the prior knowledge of the industrial processes. To describe the proposed approach clearly, as shown in Fig. 4, we decompose the approach into five training criteria, namely two adversarial training criteria and three cycle consistency training criteria,

which are also known as the reconstruction criteria. In each criterion, the networks and connections are the same color as the corresponding parts as shown in Fig. 3.

A. Adversarial Training Criterion

As shown in Fig. 4(a), to shape the output distribution of the latent code directly, we use an adversarial training procedure to perform variational inference by matching the aggregated posterior of the latent code z of the encoder with an arbitrary prior distribution $p(z)$ rather than the KL divergence [33]. A minmax objective function can be used to train E and D_z

$$\mathcal{L}^{z}_{\mathrm{GAN}}(E, D_z) = \mathbb{E}_{z\sim p(z)}\big[\log(D_z(z))\big] + \mathbb{E}_{s\sim p(s)}\big[1-\log(D_z(E(s)))\big] \quad (13)$$

where z is a latent vector sampled from a given distribution $p(z)$. To encourage robustness to small perturbations along the latent manifold, the reparameterization (RP) trick is used to predict the latent variable z from the encoder distribution $q(z|s)$ [33].

Conditioned on conditional variables ℓ, the generator $p(\Delta r_{ij}|z,\ell)$ takes a vector z as an input and outputs predicted OIA $\Delta\hat{r}_{ij} = G(z,\ell)$. The discriminator $D_{\mathcal{A}}$ takes a sample as an input stochastically chosen (with equal probability) to be either $\Delta\hat{r}_{ij}(t)$, as generated from the generator, or $\Delta r^*_{ij}(t)$, the manual adjustment computed by $\Delta r^*_{ij}(t) = r^*_{ij}(t) - \bar{r}_{ij}(t)$. In both real and generated cases, $D_{\mathcal{A}}$ also receives a condition ℓ and outputs a scalar, which is trained to be high if the input is real, and be low if it is generated from G. $D_{\mathcal{A}}$ and G are trained alternately to improve via competition. We illustrate the formulation in Fig. 4(b) and describe the minmax objective as follows:

$$\mathcal{L}^{\mathcal{A}}_{\mathrm{GAN}}(G, D_{\mathcal{A}}) = \mathbb{E}_{\Delta r^*_{ij},\ell\sim p(\mathcal{A},\ell)}\big[\log\big(D_{\mathcal{A}}\big(\Delta r^*_{ij},\ell\big)\big)\big] + \mathbb{E}_{z\sim p_{\mathcal{A}}(z),\ell\sim p(\ell)}\big[1-\log(D_{\mathcal{A}}(G(z,\ell),\ell))\big]. \quad (14)$$

The discriminator guarantees that the generator $p(\Delta r_{ij}|z,\ell)$ (abandoning auxiliary variables Δd) can learn a DNN model that is compatible with human experience in terms of structural information and relative importance among the operational indices r_{ij}.

B. Cycle Consistency Training Criterion

To further reduce the space of possible mapping functions [i.e., the encoder $q(z|s)$ and the generator $p(s|z,\ell)$], we argue that the learned mapping should be cycle consistent [14], [47]; in other words, composing the encoder and generator sequentially should result in the same input IOC s, and the opposite should produce the same latent code z. Thus, we minimize two cycle consistency losses, as illustrated in Fig. 4(d) and (e) and shown as follows:

$$\mathcal{L}^{\mathrm{AE}}_1(G, E) = \mathbb{E}_{s\sim p(s),\ell\sim p(\ell)}\|s - G(E(s),\ell)\|_1 \quad (15)$$

$$\mathcal{L}^{\mathrm{latent}}_1(E, G) = \mathbb{E}_{z\sim p(z),\ell\sim p(\ell)}\|z - E(G(z,\ell))\|_1. \quad (16)$$

Simultaneously, to guarantee that the output $\Delta\hat{r}_{ij}$ can improve the production index performance, as shown in Fig. 4(f), we minimize another cycle consistency loss as

$$\mathcal{L}^{\mathcal{B}}_1(P, G) = \mathbb{E}_{z\sim p(z),\ell\sim p(\ell)}\|\Delta Q^*_k - P(G(z,\ell))\|_1 \quad (17)$$

where the architecture of P is shown in Fig. 4(c) and ΔQ^*_k is the expected change in the global production indices Q_k. To make the process more stable and avoid drastic changes in operational indices $\bar{r}_{ij}(t)$, we prefer to tune the operational indices with minimal adjustments, which can be written as

$$\mathcal{L}^{\mathrm{TV}}_1(G) = \mathbb{E}_{z\sim p(z),\ell\sim p(\ell)}\|G(z,\ell)\|_1. \quad (18)$$

C. Objective of DMGAN

Our proposed approach encourages bijective consistency between the latent code z and IOC s, and a mapping from OIA $\Delta r_{ij}(t)$ to PIC $\Delta Q_k(t)$. Our objective finally becomes

$$\begin{aligned}\mathcal{L}(E, G, D_z, D_{\mathcal{A}}) &= \mathcal{L}^{\mathcal{A}}_{\mathrm{GAN}}(G, D_{\mathcal{A}}) + \lambda_z\mathcal{L}^{z}_{\mathrm{GAN}}(E, D_z) + \lambda_{AE}\mathcal{L}^{\mathrm{AE}}_1(G, E)\\ &\quad + \lambda_{\mathrm{latent}}\mathcal{L}^{\mathrm{latent}}_1(E, G) + \lambda_E\mathcal{L}^{\mathcal{B}}_1(P, G) + \lambda_{tv}\mathcal{L}^{\mathrm{TV}}_1(G) \quad (19)\end{aligned}$$

where λ_z, λ_{AE}, $\lambda_{\mathrm{latent}}$, λ_E, and λ_{tv} are the hyperparameters. Its solution is expressed as

$$G^*, E^*, D^*_{\mathcal{A}}, D^*_z = \arg\min_{G,E}\max_{D_{\mathcal{A}},D_z}\mathcal{L}(E, G, D_z, D_{\mathcal{A}}). \quad (20)$$

The result of training is that the encoder learns to convert the distribution of the IOC to a low-dimensional latent space, which allows the extraction of useful information about the IOC space, while the generator models learn the human experience in terms of structure information and thus improve the performance of the industrial processes.

V. Implementation Details

A. Model Description of RU-Net

The plant-wide operation requires the generator to take an input of arbitrary size and produce dense outputs with efficient learning and inference. Recently, U-Net [48] has demonstrated its efficiency on small data sets and produced strong results on the dense generation. Motivated by the above discussions in Section II and the success of U-Net, we propose a reinforced U-Net, termed as RU-Net, which is applicable to the complex plant-wide operation of operational indices.

The architecture of proposed RU-Net is shown in Fig. 5. We consider an RU-Net with $N_b = 2L + 1$ building blocks, where L is the level of the generator. Let $l \in \{1, 2, \ldots, L\}$ indices be the level of the network. Let $m^{(l)}$ and $y^{(l)}$ denote the feature maps of the inputs into layer l and the feature maps output by layer l, respectively. The RU-Net consists of a contracting path and an expansive path given as follows:

$$\begin{aligned} l: 1\to L: m^{(l+1)} &= d\big(y^{(l)}\big)\\ y^{(l+1)} &= f^{(l+1)}\big(m^{(l+1)}\big) \quad (21)\\ l: L\to 1: m^{(N_b+1-l)} &= g\big(y^{(l)}, u\big(y^{N_b-l}\big)\big)\\ y^{(N_b+1-l)} &= f^{(N_b+1-l)}\big(m^{(N_b+1-l)}\big) \quad (22)\end{aligned}$$

where $f^{(l)}$ is the lth building block function, d and u denote the downsampling operation and upsampling by deconvolution, respectively, and g is the combine function, which is defined in Section II.

The RU-Net has several compelling advantages. The network is widened using larger building blocks in place of

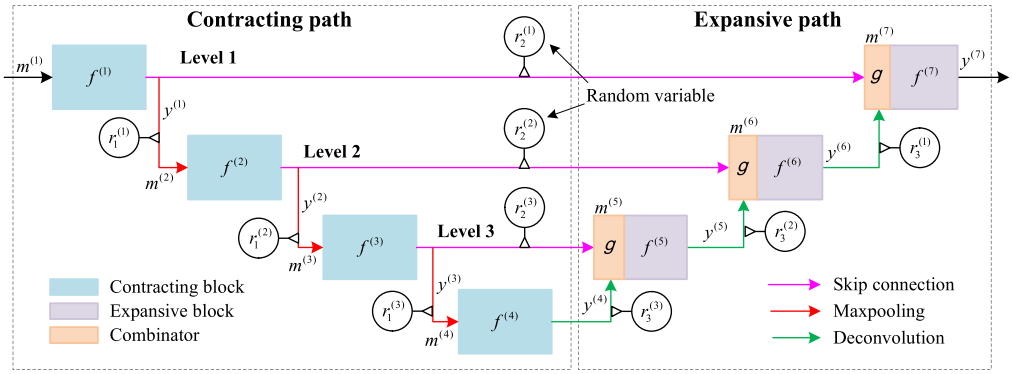

Fig. 5. Architecture of proposed RU-Net with drop-level regularization.

individual layers to enhance the local representation and to obtain performance gains. In addition, the skip connections between blocks and within blocks greatly alleviate the vanishing gradient problem, and encourages feature reuse and strength feature propagation.

There are several methods to prevent neural networks from overfitting and improve the generalization of the DCNN. For example, the dropout [49] and the DropConnect [50] modify interactions between sequential network layers to prevent coadaption. Huang *et al.* [51] dropped a subset of layers randomly to accelerate training. Larsson *et al.* [41] proposed a drop path to prevent subnetworks of different depths from coadapting. Inspired by these work, we propose a drop-level strategy to complement the RU-Net.

The layout of RU-Net with a drop-level strategy is illustrated schematically in Fig. 5 and the resulting RU-Net is

$$\begin{aligned} l:1\to L:\tilde{m}^{(l+1)} &= \lambda_1^{(l)}\odot d\left(y^{(l)}\right)\\ \tilde{y}^{(l+1)} &= f^{(l+1)}\left(\tilde{m}^{(l+1)}\right) \end{aligned} \tag{23}$$

$$\begin{aligned} l:L\to 1:\tilde{m}^{(N_b+1-l)} &= g\left(\lambda_2^{(l)}\odot y^{(l)},\lambda_3^{(l)}\odot u\left(y^{N_b-l}\right)\right)\\ \tilde{y}^{(N_b+1-l)} &= f^{(N_b+1-l)}\left(\tilde{m}^{(N_b+1-l)}\right) \end{aligned} \tag{24}$$

where $\lambda_w^{(l)}\in\mathbb{R}^{3l}$, $l=1,2,\ldots,L$ and $w=1,2,3$ are subject to an anticategorical distribution to control the passing-through of the connection line, which consists of sampling a subnetwork from a larger network. For training, the gradient passes through the subnetwork and the output of the final layer $\tilde{y}^{(N_b)}$ is scaled up by $1/p$ to keep the expected output value. Based on the hypothesis of information equivalence between the high level and the low level, the probability is calculated as $p=1-(1/2)^{\rho}$, where ρ is the level of the breaking connection of zero element. The resulting neural network without a drop level is used for testing.

B. Network Details

Specifically, the RU-Net consists of following types modules: VGG block, residual block, dense block, and fractal block. We propose an alternative combination function $g(v,u)=[v,u,\hbar(v,u)]$, and its derivatives are described as follows:

$$\begin{aligned} g_5(v,u) &= a\odot v+b\odot u\\ g_6(v,u) &= [v,u,(v-\mu(u))\odot\sigma(u)+\mu(u)]\\ g_7(z,u) &= [v,u,a\odot v+b\odot u] \end{aligned} \tag{25}$$

where $g_6(v,u)$ and $g_7(v,u)$ are the general Gaussian combination function and the general linear combination function (named GENLIN), respectively.

C. Comparison Methods and Evaluation Metrics

To quantitatively characterize the performance of the plant-wide operation of operational indices, we propose three metrics: the MAP, the EC, and the MIKDA. According to (9) and (10), MAP and EC are based on mean-squared error (MSE) to evaluate the performance as follows:

$$\mathrm{MAP}=\sum_k|Q_k(t)-Q_k^*|^2,\quad k=1,2,\ldots \tag{26}$$

$$\mathrm{EC}=\sum_i\sum_j|\Delta\hat{r}_{ij}(t)|^2,\quad i=1,2,\ldots I,\quad j=1,2,\ldots. \tag{27}$$

MIKDA is a dissimilarity-based assessment approach, in which we turn the measure of the dissimilarity between the training data $\Delta r_{ij}^*(t)\subset\mathcal{A}^+$ from an underlying distribution and the generated samples $\Delta\hat{r}_{ij}(t)\subset\mathcal{A}^-$ produced by the generator into using the mutual information between the feature variables ($f\subset\mathcal{A}$ and $\mathcal{A}=\mathcal{A}^++\mathcal{A}^-$) and the auxiliary label variables $c\subset C$(samples are from a real distribution when variables c are $+$, otherwise $-$). Therefore, the statistical dependence between the above two distributions can be measured. It is noted that the mutual information between f and c will decrease when the generated samples $\Delta\hat{r}_{ij}(t)$ are similar to the manual adjustments $\Delta r_{ij}^*(t)$. Let NC be the number of classes, the mutual information ($I(\mathcal{A},C)$) between f and c is

$$I(\mathcal{A},C)=H(C)-H(C|\mathcal{A}) \tag{28}$$

where

$$H(C)=-\sum_{c=1}^{NC}p(c)\log p(c). \tag{29}$$

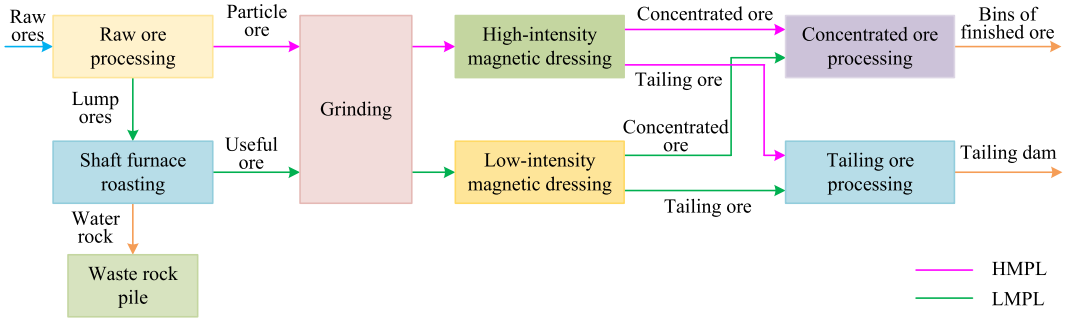

Fig. 6. Scheme of the mineral processing production process.

The calculation of $H(C|\mathcal{A})$ is based on Parzen window probability density estimation, as shown in the Appendix. Hence, the dissimilarity between training data $\mathcal{A}^+$ and generated data $\mathcal{A}^-$ can be measured as

$$D_{\mathrm{KDA}}\left(\mathcal{A}^+\|\mathcal{A}^-\right) = \frac{I(\mathcal{A}, C)}{H(C)}. \tag{30}$$

It is noted that the computational complexity in (37), given in the Appendix, is proportional to $N^2 \times d_f$, where N is the number of samples and d_f is the dimensions of the feature variables. However, to obtain a better estimation of probability density, the generated samples need to be sufficient. Thus, we divide the generated samples into multiple batches and then sample a batch without replacement. The sampled batch is combined with the training samples as a subset, and the batch size of the generative batches adopt the number of training samples to balance the number of samples from each class. Hence, the distance becomes

$$D_{\mathrm{KDA}}(\mathcal{A}\|C) = \frac{1}{M}\sum_{b=1}^{M}\sum_{\mathcal{S}_b\subset\mathcal{A}}\frac{I(\mathcal{S}_b, C)}{H(C)} \tag{31}$$

where M is the number of batches; $\mathcal{S}_b$ is the subset consisting of a batch of generated samples and the training samples.

VI. Case Study and Discussion

In this section, we present extensive experimental studies to demonstrate the effectiveness and efficiency of the proposed algorithms for the plant-wide operation of operational indices of mineral processing plants for hematite iron ore.

A. Real-World Scenarios

1) Production Line of Mineral Processing of Hematite Iron Ore: As shown in Fig. 6, the production process of the largest mineral processing factory of hematite iron ore in Western China, with a capacity of five million tons per year, consists of raw ore processing, shaft furnace roasting, grinding, high- and low-intensity magnetic dressing, concentrated ore processing, and tailing ore processing. For each process unit, the corresponding PAS is used to make their operational indices as close as possible to their targets.

Starting from raw ore processing, the raw ore is divided into particle ore, with size in the range from 0 to 15 mm, and lump ore, with size larger than 15 mm, which are processed by the high-intensity magnetic production line (HMPL) and the low-intensity magnetic production line (LMPL), respectively. For HMPL, the particle ore is first ground in the grinding unit, and ore pulp is generated and then sent into the high-intensity magnetic separator to be separated into concentrated ores and tailings. Simultaneously, the lump ore is sent into the roasting unit of the LMPL, where it is roasted in the shaft furnace. The roasted ore discharged from the furnace is then separated into useful ore and waste rock, and the former is then sent to the grinding unit to produce ore pulp of suitable particle size. Then, the ore pulp is sent into a low-intensity magnetic separator to be separated into concentrated ores and tailings. Finally, the mixed concentrated ores and tailings are dewatered to produce the finished concentrate and tailing.

2) Plant-Wide Operation of Mineral Processing for Hematite Iron Ore: The existing decision-making system of operational indices for hematite iron ore is composed of five PASs of process units, namely, the shaft furnace, two grinding units, and high- and low-intensity magnetic dressing units, which are used to create the actual operational indices to track their target values. The operational indices r_{ij} of each process unit are shown in Table I, and the production indices are the mixed concentrated grade Q_1 and the yield of concentrated ore Q_2. In this article, the targets $Q_1^*(t)$ and $Q_2^*(t)$ are set as 52.6% and 6868.5t/d, respectively, according to the practice.

B. Data and Experimental Setup

1) Data Sets: The data are collected from the production line of the largest mineral processing factory of hematite iron ore in China. In this article, 574 groups of data, which contain the operation indices $r_{ij}(t)$, the production indices $Q_k(t)$, and the corresponding operating condition $d(t)$, are collected and scaled to the range $[-1, 1]$ using shifting and scaling operations. Five hundred groups are used to train the proposed DMGAN, and the remaining 74 groups, almost a week of operational data with a sample time of 2 h, are used for the model validation. Moreover, the desired change value of the production indices is computed by the difference between the current production indices $Q_k(t)$ and the prescribed production indices, which are determined by the technical department.

2) Training Details: For all experiments, we set hyperparameters λ_z, λ_{AE}, λ_{latent}, λ_E, λ_{tv} to {0.01, 10, 0.5, 1, 0.01} for VGG&CONCAT and {0.01, 1, 0.5, 1, 0.01}

TABLE I

OPERATIONAL INDICES OF ROASTING PROCESS, GRINDING PROCESS, AND INTENSITY MAGNETIC DRESSING OF MINERAL PROCESS

LMPL			**HMPL**	
Roasting Process	Grinding Process	Low-intensity Magnetic dressing	Grinding Process	High-intensity Magnetic dressing
r_1 : Magnetic tube recovery rate (MTRR) (%)	r_2 : Grinding particle size (PSLMPL) (%)	r_{31} : Concentrate grade (CGLMPL) (%) r_{32} : Tailing grade (TGLMPL) (%)	r_4 : Grinding particle size (PSHMPL) (%)	r_{51} : Concentrate grade (CGHMPL) (%) r_{52} : Tailing Grade (TGHMPL) (%)

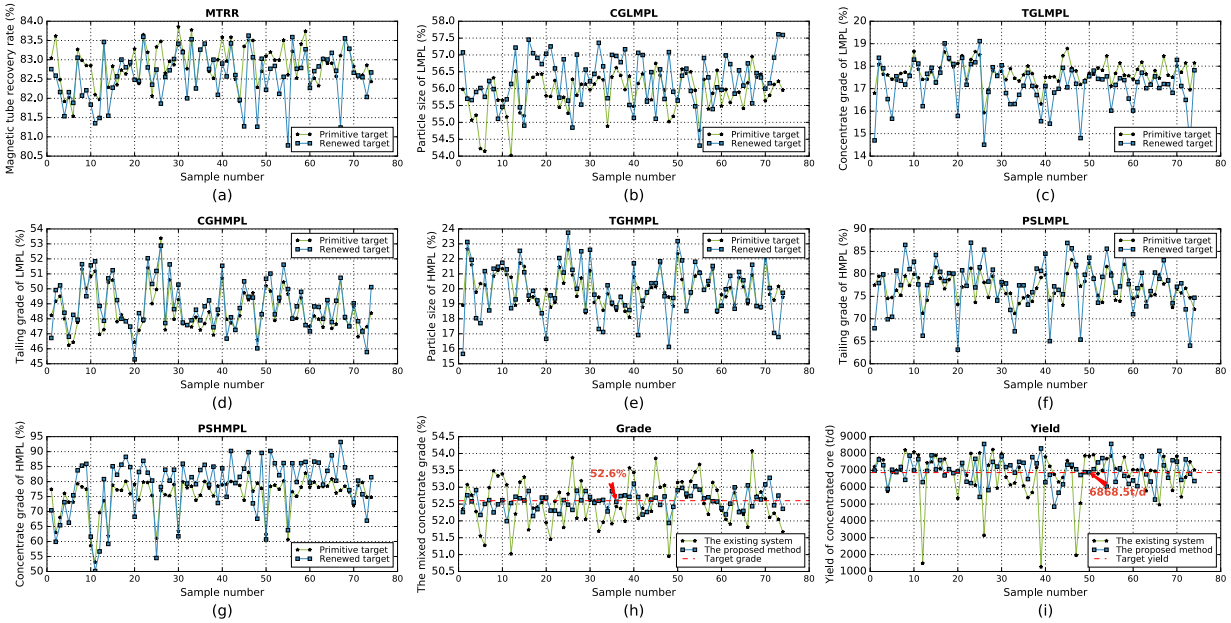

Fig. 7. Primitive values and renewed values of operational indices and the resulting performance. (a) MTRR. (b) PSLMPL. (c) CGLMPL. (d) TGLMPL. (e) PSHMPL. (f) TGHMPL. (g) CGHMPL. (h) Grade of concentrated ore. (i) Yield of concentrated ore.

for RES&GENLIN. To visualize the latent representation of the IOC, the dimension of the latent code is chosen as 2. Moreover, for injecting conditional variables into the generator G, we replicate conditional variables d_{IOC} times and simply concatenate it with the input data, where the width d_{IOC} is the dimension of the IOC. For the MIKDA metric, the number of generated samples per batch is set to 60, which is sufficient for evaluating dissimilarity. In addition, our emulation experiments are carried out in the simulation systems. Please see the Appendix for more details about the network architecture and training procedures.

3) Methods: To test the effectiveness of the proposed DMGAN, we compare it with several state-of-the-art GANs. Three baseline methods are included in our experiments.

1) *CGAN:* This approach uses a conditional adversarial framework. This method is well known as a pix2pix framework [15], [52] in the image generation domain.
2) *Conditional Adversarial Autoencoder GAN (CAAE-GAN):* In this conditional setup (similar to unconditional analog [34], [53]), the adversarial procedure can directly shape the latent code distribution by matching the aggregated posterior of the hidden code vector with an arbitrary prior distribution. This variation criterion is then optimized jointly with the reconstruction loss, which is related to VAEs [33].
3) *Conditional Information Maximizing GAN (CIM-GAN):* This approach adopts the encoder E to map inversely from the output of generator G to the latent code space, which is used to maximize the mutual information between them, and thus enforce the generator G to best use available random noise. This method can be referred to as "latent regressor" [54], [55] and is related to InfoGAN [56].
4) *DMGAN:* This method encourages the connection between the latent code and OIA in both directions jointly via cycle consistency training criteria and regularizes the encoder E and generator G by adversarial learning. In addition to the objective of DMGAN, we further study the performance with respect to different blocks, combination functions, and latent prior distributions.

C. Comparison With Existing Decision-Making Systems

The industrial experiment results over almost one week of operation are illustrated in Fig. 7, where the x-axis represents the sample numbers and the y-axis represents the corresponding index values. In Fig. 7(a)–(g), the blue and green lines

TABLE II
COMPARISON OF THE AVERAGE AND THE STANDARD DEVIATION OF MIKDA, EC, AND MAP OVER CGAN, CAAE-GAN, CIM-GAN, AND DMGAN

Experimental Setting	Model/Metrics	MIKDA	MAP	EC
-	Manual tuning	-	0.1616 ± 0.0000	-
VGG&CONCAT	CGAN	0.3555 ± 0.0060	0.0574 ± 0.0060	1.5354 ± 0.5572
	CAAE-GAN	0.3567 ± 0.0170	0.0497 ± 0.0137	0.8874 ± 0.3398
	CIM-GAN	0.2648 ± 0.0028	0.0706 ± 0.0175	1.0464 ± 0.9605
	DMGAN	0.2652 ± 0.0057	0.0497 ± 0.0143	0.9235 ± 0.2993
RES&GENLIN	CGAN+	0.3299 ± 0.0102	0.0516 ± 0.0167	7.7947 ± 3.8902
	CGAN	0.2519 ± 0.0113	0.0754 ± 0.0056	0.9849 ± 0.2316
	CAAE-GAN	0.3470 ± 0.0012	0.0492 ± 0.0080	0.6080 ± 0.1952
	CIM-GAN	0.2900 ± 0.0090	0.0505 ± 0.0040	1.1176 ± 0.3018
	DMGAN	**0.2300 ± 0.0017**	**0.0480 ± 0.0078**	**1.0708 ± 0.3157**

TABLE III
COMPARISON OF THE AVERAGE AND THE STANDARD DEVIATION OF MIKDA, EC, AND MAP OVER VGG, RES, DENSE, AND FRACTAL MODULES

Combinator	Building Block/Metrics	MIKDA	MAP	EC
CONCAT	VGG	0.2652 ± 0.0014	0.0497 ± 0.0143	0.9235 ± 0.2993
GENLIN	RES (*NL*=2)	0.2300 ± 0.0017	**0.0480 ± 0.0078**	**1.0708 ± 0.3157**
	DEN (*NL*=4)	0.3521 ± 0.0090	0.0486 ± 0.0046	0.8397 ± 0.1042
	FRA (*NL*=3)	**0.2221 ± 0.0016**	0.0597 ± 0.0118	0.8085 ± 0.2680

TABLE IV
COMPARISON OF THE AVERAGE AND THE STANDARD DEVIATION OF MIKDA, EC AND MAP OVER GAUSSIAN, LINEAR, CONCATENATE, GENERAL LINEAR, AND GENERAL GAUSSIAN

Building Blocks	Combinator/Metrics	MIKDA	MAP	EC
VGG	CONCAT	0.2652 ± 0.0057	0.0497 ± 0.0143	0.9235 ± 0.2993
RES	Linear	0.3506 ± 0.0134	0.0581 ± 0.0019	0.4181 ± 0.0110
	Gaussian	0.3593 ± 0.0011	0.0588 ± 0.0017	0.6194 ± 0.0196
	GENLIN	**0.2300 ± 0.0017**	0.0480 ± 0.0078	1.0708 ± 0.3157
	General Gaussian	0.3483 ± 0.0050	**0.0445 ± 0.0035**	0.5388 ± 0.0655

represent the primitive targets of operational indices (as shown in Table I) determined by the existing systems, and the renewed targets are generated by the DMGAN. The comparison results between the proposed approach and the existing decision-making methods are shown in Fig. 7(h) and (i). Therefore, the closer the production indices are to the targets, the more improvements can be achieved by the proposed method when compared with the existing systems. It is observed that the DMGAN performs favorably when compared with the existing decision-making systems. Although the significant variation of the production condition exists in the 1.5 months of collected data, the proposed DMGAN can still capture the common coordination characteristic, which is used to dampen the effects of dynamic uncertainties that is brought to industrial processes; thus, the target values are tracked as closely as possible by the actual values of the production indices.

D. Comparison Against GAN-Based Methods

The performance results of plant-wide operation are summarized in Table II. We adopt two settings, VGG with a concatenation combinator (CONCAT) and RES with GENLIN, as shown in (25). For both settings, our method yields better results than CGAN, CAAE-GAN, and CIM-GAN. We make several key observations.

1) The total variation loss should not be ignored, or it will result in the exploration of the EC metric as CGAN+.
2) CAAE-GAN efficiently exploits the knowledge existing in the operational data to achieve a better MAP. However, the MIKDA indicates that this approach cannot describe the whole decision space and is prone to model collapse.
3) CIM-GAN is capable of exploring the decision space by traversing the whole latent space, which helps produce higher MIKDA; however, this comes at a loss to the MAP.
4) DMGAN integrates the abovementioned approach and balances the exploration and the exploitation of knowledge. As a result, the DMGAN achieves the best MIKDA and MAP. The powerful self-learning and knowledge discovery abilities of DMGAN enable us to perform efficient approximate posterior inference using an industrial process data set of moderate quality.

E. Analysis of RU-Net

Table III shows the results of applying RU-Net with different building blocks in the DMGAN setting, where VGG is the plain network without skip connections and RES, DEN, and FRA are blocks of ResNet [37], DenseNet [38], and FractalNet [41], respectively. NL is the number of layers contained in the blocks. Due to the advantage of local representation, we found that the result of RU-Net with RES is better than the original setting with VGG under CONCAT. It is also found that those with DEN and FRA obtain better performance

TABLE V

COMPARISON OF THE AVERAGE AND THE STANDARD DEVIATION OF MIKDA, EC, AND MAP OVER DROP-LEVEL REGULARIZATION

Model/Metrics			MIKDA	MAP	EC
Combinator	Block	Regularization			
CONCAT	VGG	Drop-Level	0.2652 ± 0.0057	0.0497 ± 0.0143	0.9235 ± 0.2993
		NONE	0.4662 ± 0.0169	0.0792 ± 0.0019	0.4833 ± 0.0066
	RES	Drop-Level	0.3269 ± 0.0034	0.0488 ± 0.0077	1.2320 ± 0.3505
		NONE	0.3305 ± 0.0014	0.0721 ± 0.0030	0.6171 ± 0.0169
GENLIN		Drop-Level	$\mathbf{0.2300 \pm 0.0017}$	$\mathbf{0.0480 \pm 0.0078}$	1.0708 ± 0.3157
		NONE	0.4423 ± 0.0202	0.0491 ± 0.0028	0.6447 ± 0.0143

TABLE VI

COMPARISON OF THE AVERAGE AND THE STANDARD DEVIATION OF MIKDA, EC, AND MAP OVER GAUSSIAN, CIRCLE UNIFORM, STANDARD UNIFORM, GAUSSIAN MIXTURE, AND SWISS ROLL IN GENLIN AND RES SETTING

Prior Distribution	MIKDA	MAP	EC
Gaussian	$\mathbf{0.2652 \pm 0.0057}$	0.0497 ± 0.0143	0.9235 ± 0.5572
Circle Uniform	0.4552 ± 0.0021	$\mathbf{0.0440 \pm 0.0031}$	0.6661 ± 0.0168
Standard Uniform	0.2921 ± 0.0086	0.0655 ± 0.0037	1.1641 ± 0.0405
Gaussian-Mixture	0.3677 ± 0.0048	0.0502 ± 0.0043	1.3178 ± 0.0499
Swiss-Roll	0.3318 ± 0.0081	0.0604 ± 0.0051	1.3995 ± 0.0650

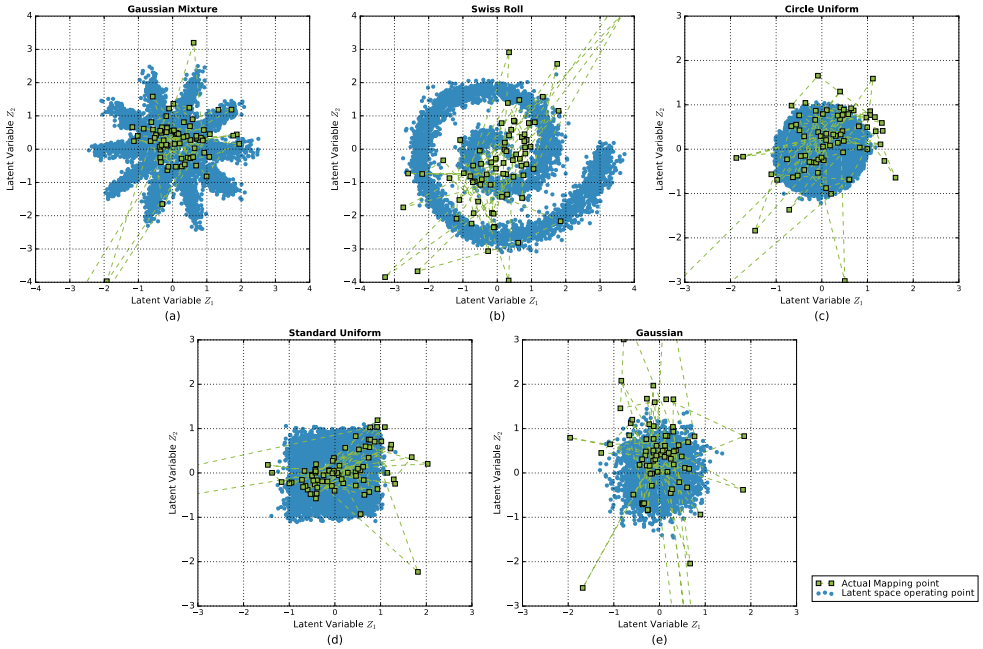

Fig. 8. Visualization of the latent data manifold with a 2-D latent space learned with DGGAN. For each of these testing samples, we plotted the corresponding latent code where the sample resides in the manifold. The prior latent spaces are (a) a 10-D Gaussian mixture, (b) Swiss roll, (c) uniform circle, (d) standard uniform, and (e) Gaussian distribution.

of MIKDA or MAP and are not able to balance them much. As shown in Table IV, for VGG, the performance of the proposed GENLIN and general Gaussian on MIKDA, MAP, and EC is superior to the existing function, Gaussian [39], [42], linear [39], and concatenation [48]. The reason may be that the general combinator can explicitly construct the intermediate feature representation, making the network be easily trained and get good results. To further demonstrate the efficiency of the drop level, the results of applying the drop-level method to different building blocks and combination functions are shown in Table V. Under different settings, the approach with the drop-level method achieves significantly better MIKDA and MAP than that without it [48]. It is worth mentioning that, due to the high degree of parameterization of the general Gaussian combinator, the drop level performs poorly in that setting.

F. Discovering the Latent Domain

Table VI presents the different prior latent distributions, which are imposed on the output of encoder E, including Gaussian, circle uniform, standard uniform, Gaussian mixture, and Swiss roll. In general, simple distributions, such as Gaussian and uniform distributions, are especially suitable for low data schemes because the number of regions in the complex latent space increases due to unavailable data covering the connection, while the simple distribution can be easily shaped and thus result in meaningful samples at the inference time. Meanwhile, complex distributions, such as the Gaussian mixture and Swiss roll, are unsuitable for constructing continuous space.

In this section, we also adopt the encoder E to learn a low-dimensional submanifold, where high-dimensional IOC

often reside, to facilitate the understanding of the industrial processes and to allow us to extract information about the IOC. Fig. 8 shows the coding space of the hold-out testing data resulting from DMGAN, where the blue points indicate the latent code from the generator G and the green dots indicate the testing data set. The blue points describe the prior distribution well, and most of the green dots are approximately subjected to the corresponding imposed distribution. The effects of industrial data qualities on the effectiveness of knowledge-based models are rarely considered in the previous approach. However, some outliers in the latent space can be seen clearly and will then be ignored in the generator, and thus, the DMGAN framework can autonomously filter out inaccurate knowledge.

VII. Conclusion

In process industries, it is very important to wisely set up a plant-wide operation to make the targets of the production indices followed by the actual values as closely as possible under uncertainties. To overcome the knowledge bottlenecks of the existing approaches, the DMGAN, a novel GAN-based decision-making framework, is proposed, where RU-Net is employed to improve the generalization of the generator. The proposed DMGAN with RU-Net is applied to real-world data collected from the largest mineral processing factory in Western China, and simulation results show the effectiveness of our approach in terms of three analysis metrics. The results indicate the potential of the proposed decision-making techniques for resolving the plant-wide operation problem in process industries, even though global optimality cannot be guaranteed in practice.

Directions of future work will focus on the following areas.

1) Extension of the DMGAN toward the semisupervised learning. A priority over the IOC statistics can be built and it can be leveraged by the generator to improve performance on data sets that are difficult to obtain in real time.
2) More rigorous theoretical understanding to ensure the global optimality of the DMGAN.
3) Incorporation with advanced deep-learning architectures such as attention mechanism [57], [58] to improve performance and discover the inherent relations between OIA and PIC.

Appendix

A. Training Details

All the networks were trained from scratch, with a learning rate of 0.0002 for 500 epochs. Weights were initialized via Xavier initialization. Biases were initialized to 0. Our proposed DMGAN was implemented in TensorFlow and conducted on an NVIDIA GeForce GTX 1080 GPU card with 8 GB of memory. We adopted the Adam solver [59] with a batch size of 64. It is noted that, for all settings, a 1×1 convolution was used as the bottleneck layer applied to building blocks to match the channels of the feature maps, which were inputs of the combination function; all the building blocks had the same network depth.

B. Network Architectures

Let c3s1-k denote a 1×3 convolution – batch normalization – rectified linear unit (ReLU) layer with k filters and stride 1. d2 denotes max-pooling with a kernel size of 2. Rk denotes a residual block that contains two 1×3 convolutional layers with the same number of filters on both layers. uk denotes 1×3 fractional – strided – convolution – batch normalization – ReLU layers with k filters and stride 1/2. fk denotes a fully connected layer of k hidden units.

1) Generator Architecture: The architecture of the generator with three levels and seven blocks consists of the following:

$$\text{c3s1} - 64, \text{R64}, \text{d2}, \text{R128}, \text{d2}, \text{R256}, \text{d2}, \text{R512}, \text{u256},$$
$$\text{R256}, \text{u128}, \text{R128}, \text{u64}, \text{R64}, \text{C1S1} - 1.$$

2) Discriminator Architectures: For discriminators, the last layer of both networks has a sigmoid activation. Let fr-k, fs-k denote a fully connected layer of k hidden units using LReLU with slope 0.2 and a sigmoid activation function. We can describe the architecture of the discriminator on the latent representation as fr-64, fr-64, fs-1 and the architecture of discriminator on the output of generator G as

$$\text{c3s1} - 64, \text{d2}, \text{c3s1} - 128, \text{d2}, \text{c3s1} - 128,$$
$$\text{d2}, \text{fr} - 512, \text{fs} - 1.$$

3) Encoder Architectures: To implement the RP [33], that is $z_m = \mu_m(s) + \sigma_m(s) \odot \varepsilon_m$ where the subscript m is the dimension of the latent code, and the mean $\mu(s)$ and the diagonal covariance matrix $\Sigma(s)$ are functions of IOC and $\varepsilon \sim \mathcal{N}(0, I)$, respectively, the last layer of the encoder has two heads using linear activation, which produce the mean $\mu(s)$ and diagonal variance matrix $\Sigma(s)$, with the number of each output equal to the dimension of latent variables. We obtained the latent codes randomly drawn from an independent Gaussian distribution $\mathcal{N}(\mu(s), \Sigma(s))$. The architecture is expressed as

$$\text{c3s1} - 64, \text{R64}, \text{d2}, \text{c3s1} - 64, \text{R128},$$
$$\text{d2}, \text{f256}, \text{(a) f2}, \text{(b) f2}$$

where (a) and (b) denote the two heads of the encoder E. To demonstrate the effectiveness of RP, the deterministic method (DM) is adopted. The latent code of the DM is generated by a deterministic function of data; thus, the only source of stochasticity in $q(z)$ is the data distribution. The results are shown in Tables VII and VIII, which are under the CONCAT setting and the GENLIN setting, respectively. RP is found to outperform the DM in all settings. One reason is that RP encourages robustness to small perturbations along the conditioning manifold and thus yields better results.

C. Solution of $H(C|\mathcal{A})$ Using Parzen Window Density Estimation and Bayesian Rules

According to the definition, $H(C|\mathcal{A})$ is expressed as

$$H(C|\mathcal{A}) = -\frac{1}{n_f} \sum_{j_f=1}^{n_f} \sum_{c=1}^{NC} p\left(c|f_j\right) \log p\left(c|f_{j_f}\right) \qquad (32)$$

TABLE VII

COMPARISON OF THE AVERAGE AND THE STANDARD DEVIATION OF MIKDA, EC, AND MAP OVER LATENT CODE GENERATION UNDER THE CONCATENATION SETTING

SETTING	MIKDA	MAP	EC
VGG&DM	0.3427±0.0082	0.1068±0.0211	0.8337±0.1613
VGG&RP	0.3445±0.0072	0.0723±0.0203	0.8571±0.3919
RES&DM	0.2432±0.0023	0.0703±0.0141	0.9978±0.2515
RES&RP	0.2154±0.0012	0.0674±0.0075	1.0634±0.4660

TABLE VIII

COMPARISON OF THE AVERAGE AND THE STANDARD DEVIATION OF MIKDA, EC, AND MAP OVER LATENT CODE GENERATION UNDER THE GENERAL LINEAR SETTING

SETTING	MIKDA	MAP	EC
RES&DM	0.3685±0.0021	0.0701±0.0113	0.8360±0.2657
RES&RP	0.2558±0.0096	0.0494±0.0058	1.2665±0.2532

$p(c|f)$ is the conditional probability of class $c \in [1, 2, \ldots, NC]$ given f. It can be computed by Bayesian rules as

$$p(c|f) = \frac{p(f|c)p(c)}{\sum_{c'=1}^{NC} p(f|c')p(c')}. \tag{33}$$

The conditional probability of f given class c can be estimated using Parzen window density estimation as [60]

$$p(f|c) = \frac{1}{n^c} \sum_{f_i \in I^c} \phi[f - f_i, h_c] \tag{34}$$

where n^c is the number of training samples belonging to class c; I_c is the set of the training samples belonging to class c; and

$$\phi(f - f_i, h_*) = \frac{1}{(2\pi)^{d/2} h_*^d |\Sigma|^{1/2}} \times \exp\left(-\frac{(f - f_i)^T \Sigma^{-1} (f - f_i)}{2h_*^2}\right). \tag{35}$$

The optimal parameter h_* of a Gaussian window is selected from candidates H_c by maximizing the log likelihood of $p(f_i|c)$ that is defined as

$$\mathcal{L}(f, c; h) = \sum_c \log p(f_i|c). \tag{36}$$

Thus, with (33)–(35), $p(c|f)$ can be computed by

$$p(c|f) = \frac{\sum_{f_i \in I_c} \exp\left(-\frac{(f-f_i)^T \Sigma^{-1} (f-f_i)}{2h_*^2}\right)}{\sum_{c'=1}^{NC} \sum_{\overline{f}_i \in I_{c'}} \exp\left(-\frac{(f-\overline{f}_i)^T \Sigma^{-1} (f-\overline{f}_i)}{2h_*^2}\right)} \tag{37}$$

where Σ is a covariance matrix of d_f-dimensional feature variables f, in which we set as the identity matrix.

REFERENCES

[1] T. Chai, J. Ding, and H. Wang, "Multi-objective hybrid intelligent optimization of operational indices for industrial processes and application," in *Proc. World Congr. IFAC*, Aug. 2011, pp. 10517–10522.

[2] Y. Xie, W. Gui, C. Yang, and X. Chen, "Knowledge automation and its industrial application," *Scientia Sinica Inf.*, vol. 46, no. 8, pp. 1016–1034, Aug. 2016.

[3] J. Ding, T. Chai, H. Wang, and X. Chen, "Knowledge-based global operation of mineral processing under uncertainty," *IEEE Trans Ind. Informat.*, vol. 8, no. 4, pp. 849–859, Nov. 2012.

[4] T. Chai, J. Ding, and F. Wu, "Hybrid intelligent control for optimal operation of shaft furnace roasting process," *Control Eng. Pract.*, vol. 19, no. 3, pp. 264–275, Mar. 2011.

[5] E. I. Papageorgiou and J. L. Salmeron, "A review of fuzzy cognitive maps research during the last decade," *IEEE Trans. Fuzzy Syst.*, vol. 21, no. 1, pp. 66–79, Feb. 2013.

[6] A. P. Prajapati and D. K. Chaturvedi, "Semantic network based knowledge representation for cognitive decision making in teaching electrical motor concepts," in *Proc. Int. Conf. Comput., Commun. Electron. (Comptelix)*, Jul. 2017, pp. 146–151.

[7] B. Zhang, C. Yang, H. Zhu, P. Shi, and W. Gui, "Controllable-domain-based fuzzy rule extraction for copper removal process control," *IEEE Trans. Fuzzy Syst.*, vol. 26, no. 3, pp. 1744–1756, Jun. 2018.

[8] A. Zagorecki and M. J. Druzdzel, "Knowledge engineering for Bayesian networks: How common are noisy-MAX distributions in practice?" *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 43, no. 1, pp. 186–195, Jan. 2013.

[9] B. Kamsu-Foguem, F. Rigal, and F. Mauget, "Mining association rules for the quality improvement of the production process," *Expert Syst. Appl.*, vol. 40, no. 4, pp. 1034–1045, Mar. 2013.

[10] J. Ding, H. Modares, T. Chai, and F. L. Lewis, "Data-based multi-objective plant-wide performance optimization of industrial processes under dynamic environments," *IEEE Trans Ind. Informat.*, vol. 12, no. 2, pp. 454–465, Apr. 2016.

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

[12] H. Hong, X. Li, and M. Wang, "GANE: A generative adversarial network embedding," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, doi: 10.1109/TNNLS.2019.2921841.

[13] A. Creswell and A. A. Bharath, "Inverting the generator of a generative adversarial network," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 7, pp. 1967–1974, Jul. 2019.

[14] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2223–2232.

[15] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier GANs," 2016, *arXiv:1610.09585*. [Online]. Available: http://arxiv.org/abs/1610.09585

[16] X. Wang and A. Gupta, "Generative image modeling using style and structure adversarial networks," in *Proc. Eur. Conf. Comput. Vis.*, Oct. 2016, pp. 318–335.

[17] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional GANs," 2017, *arXiv:1711.11585*. [Online]. Available: http://arxiv.org/abs/1711.11585

[18] S. Gurumurthy, R. K. Sarvadevabhatla, and R. V. Babu, "DeLiGAN: Generative adversarial networks for diverse and limited data," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4941–4949.

[19] R. Fergus *et al.*, "Deep generative image models using a Laplacian pyramid of adversarial networks," in *Proc. Adv. Neural Inf. Process Syst.*, Dec. 2015, pp. 1486–1494.

[20] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015, *arXiv:1511.06434*. [Online]. Available: http://arxiv.org/abs/1511.06434

[21] J. Wu *et al.*, "Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling," in *Proc. Adv. Neural Inf. Process Syst.*, Dec. 2016, pp. 82–90.

[22] M.-Y. Liu and O. Tuzel, "Coupled generative adversarial networks," 2016, *arXiv:1606.07536*. [Online]. Available: http://arxiv.org/abs/1606.07536

[23] C. Wan, T. Probst, L. V. Gool, and A. Yao, "Crossing nets: Combining GANs and VAEs with a shared latent space for hand pose estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 680–689.

[24] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," 2016, *arXiv:1605.05396*. [Online]. Available: http://arxiv.org/abs/1605.05396

[25] H. Zhang *et al.*, "StackGAN: Text to photo-realistic image synthesis with stacked generative adversarial networks," 2016, *arXiv:1612.03242*. [Online]. Available: http://arxiv.org/abs/1612.03242

[26] M. Mathieu, C. Couprie, and Y. LeCun, "Deep multi-scale video prediction beyond mean square error," 2015, *arXiv:1511.05440*. [Online]. Available: http://arxiv.org/abs/1511.05440

[27] C. Vondrick, H. Pirsiavash, and A. Torralba, "Generating videos with scene dynamics," in *Proc. Adv. Neural Inf. Process Syst.*, Dec. 2016, pp. 613–621.

[28] C. Finn, I. Goodfellow, and S. Levine, "Unsupervised learning for physical interaction through video prediction," in *Proc. Adv. Neural Inf. Process Syst.*, Dec. 2016, pp. 64–72.

[29] M. Ravanbakhsh, M. Nabi, E. Sangineto, L. Marcenaro, C. Regazzoni, and N. Sebe, "Abnormal event detection in videos using generative adversarial nets," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Beijing, China, Sep. 2017, pp. 1577–1581.

[30] T. Kim, M. Cha, H. Kim, J. Kwon Lee, and J. Kim, "Learning to discover cross-domain relations with generative adversarial networks," 2017, *arXiv:1703.05192*. [Online]. Available: http://arxiv.org/abs/1703.05192

[31] C.-C. Hsu, H.-T. Hwang, Y.-C. Wu, Y. Tsao, and H.-M. Wang, "Voice conversion from unaligned corpora using variational autoencoding wasserstein generative adversarial networks," 2017, *arXiv:1704.00849*. [Online]. Available: http://arxiv.org/abs/1704.00849

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

[33] D. P Kingma and M. Welling, "Auto-encoding variational Bayes," 2013, *arXiv:1312.6114*. [Online]. Available: http://arxiv.org/abs/1312.6114

[34] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial autoencoders," 2015, *arXiv:1511.05644*. [Online]. Available: http://arxiv.org/abs/1511.05644

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

[36] R. Kumar Srivastava, K. Greff, and J. Schmidhuber, "Highway networks," 2015, *arXiv:1505.00387*. [Online]. Available: http://arxiv.org/abs1505.00387

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

[38] G. Huang, Z. Liu, L. V. D. Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2261–2269.

[39] A. Rasmus *et al.*, "Semi-supervised learning with ladder networks," in *Proc. Adv. Neural Inf. Process Syst.*, Dec. 2015, pp. 3546–3554.

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

[41] G. Larsson, M. Maire, and G. Shakhnarovich, "FractalNet: Ultra-deep neural networks without residuals," 2016, *arXiv:1605.07648*. [Online]. Available: http://arxiv.org/abs/1605.07648

[42] M. Pezeshki *et al.*, "Deconstructing the ladder network architecture," in *Proc. ICML*, New York, NY, USA, Jun. 2016, pp. 2368–2376.

[43] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.

[44] B. Hariharan, P. Arbelaez, R. Girshick, and J. Malik, "Hypercolumns for object segmentation and fine-grained localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 447–456.

[45] P. Sermanet, K. Kavukcuoglu, S. Chintala, and Y. Lecun, "Pedestrian detection with unsupervised multi-stage feature learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 3626–3633.

[46] S. Yang and D. Ramanan, "Multi-scale recognition with DAG-CNNs," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1215–1223.

[47] T. Zhou, P. Krahenbuhl, M. Aubry, Q. Huang, and A. A. Efros, "Learning dense correspondence via 3D-guided cycle consistency," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 117–126.

[48] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Conput-Assist. Intervent.*, Oct. 2015, pp. 234–241.

[49] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," 2012, *arXiv:1207.0580*. [Online]. Available: http://arxiv.org/abs/1207.0580

[50] L. Wan, M. Zeiler, S. Zhang, Y. Le Cun, and R. Fergus, "Regularization of neural networks using dropconnect," in *Proc. Int. Conf. Mach. Learn.*, Jun. 2013, pp. 1058–1066.

[51] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, "Deep networks with stochastic depth," in *Proc. Eur. Conf. Comput. Vis.*, Oct. 2016, pp. 646–661.

[52] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," 2016, *arXiv:1611.07004*. [Online]. Available: http://arxiv.org/abs/1611.07004

[53] Z. Zhang, Y. Song, and H. Qi, "Age progression/regression by conditional adversarial autoencoder," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5810–5818.

[54] V. Dumoulin *et al.*, "Adversarially learned inference," 2016, *arXiv:1606.00704*. [Online]. Available: http://arxiv.org/abs/1606.00704

[55] J. Y. Zhu *et al.*, "Toward multimodal image-to-image translation," in *Proc. Adv. Neural Inf. Process Syst.*, Dec. 2017, pp. 465–476.

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

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

[58] K. Gregor, I. Danihelka, A. Graves, D. J. Rezende, and D. Wierstra, "DRAW: A recurrent neural network for image generation," 2015, *arXiv:1502.04623*. [Online]. Available: http://arxiv.org/abs/1502.04623

[59] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980

[60] N. Kwak and C.-H. Choi, "Input feature selection by mutual information based on Parzen window," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 12, pp. 1667–1671, Dec. 2002.

Nianzu Zheng received the B.S. degree from Northeastern University at Qinhuangdao, Qinhuangdao, China, in 2016. He is currently pursuing the M.S. degree with Northeastern University, Shenyang, China.

From 2016 to 2018, he was a Student of the State Key Laboratory of Synthetical Automation for Process Industry, Northeastern University. His current research interests include generative adversarial networks, artificial intelligence, and machine learning.

Jinliang Ding (Senior Member, IEEE) received the Ph.D. degree in control theory and control engineering from Northeastern University, Shenyang, China, in 2012.

He is currently a Professor with the State Key Laboratory of Synthetical Automation for Process Industry, Northeastern University. He has authored or coauthored over 100 refereed journal articles and refereed articles at international conferences. He is also the inventor or co-inventor of 17 patents. His current research interests include modeling, plant-wide control, and optimization for the complex industrial systems, machine learning, industrial artificial intelligence, and computational intelligence and application.

Dr. Ding was a recipient of the Young Scholars Science and Technology Award of China in 2016, the National Science Fund for Distinguished Young Scholars in 2015, the National Technological Invention Award in 2013, and three First-Prize of Science and Technology Awards of the Ministry of Education in 2006, 2012, and 2018, respectively. One of his articles published on *Control Engineering Practice* was selected for the Best Paper Award of 2011–2013.

Tianyou Chai (Fellow, IEEE) received the Ph.D. degree in control theory and control engineering from Northeastern University, Shenyang, China, in 1985.

He is currently a Professor with the State Key Laboratory of Synthetical Automation for Process Industry, Northeastern University. He has authored more than 120 peer-reviewed international journal articles and around 224 international conference articles. His current research interests include adaptive control, intelligent decoupling control, and the development of control technologies with applications to various industrial processes.

Dr. Chai is also a member of the Chinese Academy of Engineering, an Academician of the International Eurasian Academy of Sciences, and a fellow of the International Federation of Automatic Control. He is also a Distinguished Visiting Fellow of the Royal Academy of Engineering, U.K., and an Invitation Fellow of the Japan Society for the Promotion of Science. He received the 2002 Technological Science Progress Award from the Ho Leung Ho Lee Foundation, the 2007 Industry Award for Excellence in Transitional Control Research from the IEEE Control Systems Society, and the 2010 Yang Jia-Chi Science and Technology Award from the Chinese Association of Automation.