DLUP: A Deep Learning Utility Prediction Scheme for Solid-State Fermentation Services in IIoT

Min Wang, Shanchen Pang, Tong Ding, Sibo Qiao, Xue Zhai, Shuo Wang, Neal N. Xiong, *Senior Member, IEEE*, and Zhengwen Huang

***Abstract*—At present, solid-state fermentation (SSF) is mainly controlled by artificial experience, and the product quality and yield are not stable. Therefore, predicting the quality and yield of SSF is of great significance for improving the utility of SSF. In this article, we propose a deep learning utility prediction (DLUP) scheme for the SSF in the Industrial Internet of Things, including parameters collection and utility prediction of the SSF process. Furthermore, we propose a novel edge-rewritable Petri net to model the parameters collection and utility prediction of the SSF process and further verify their soundness. More importantly, DLUP combines the generating ability of least squares generative adversarial network with the predicting ability of fully connected neural network to realize the utility prediction (usually use the alcohol concentration) of SSF. Experiments show that the proposed method predicts the alcohol concentration more accurately than the other joint prediction methods. In addition, the method in our article provides evidences for setting the ratio of raw materials and proper temperature through numerical analysis.**

***Index Terms*—Fully connected neural network, least squares generative adversarial network, Petri net, solid-state fermentation, utility prediction.**

Manuscript received June 30, 2021; revised July 27, 2021; accepted August 9, 2021. Date of publication August 20, 2021; date of current version February 2, 2022. This work was supported in part by the Major Science and Technology Innovation Project of Shandong Province under Grant 2019TSLH0214 and in part by Tai Shan Industry Leading Talent Project under Grant tscy20180416. Paper no. TII-21-2737. *(Corresponding author: Shanchen Pang.)*

Min Wang is with the College of Control Science and Engineering, China University of Petroleum (East China), Qingdao 266555, China (e-mail: minwang2121@163.com).

Shanchen Pang, Sibo Qiao, and Xue Zhai are with the College of Computer Science and Technology, China University of Petroleum (East China), Qingdao 266555, China (e-mail: pangsc@upc.edu.cn; siboqiao@126.com; 2417792534@qq.com).

Tong Ding is with the College of Software, Shandong University, Shandong 250014, China (e-mail: 793394086@qq.com).

Shuo Wang is with the Department of Computer Science and Technology, Tongji University, Shanghai 200000, China (e-mail: 836208454@qq.com).

Neal N. Xiong is with the Department of Computer Science and Mathematics, Sul Ross State University, Alpine, TX 79830 USA (e-mail: xiongnaixue@gmail.com).

Zhengwen Huang is with the Department of Electronic and Computer Engineering, Brunel University London, UB8 3PH London, U.K. (e-mail: Zhengwen.huang@brunel.ac.uk).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TII.2021.3106590.

Digital Object Identifier 10.1109/TII.2021.3106590

I. Introduction

SOLID-STATE fermentation (SSF) has been defined as the fermentation process which involves solid matrix and is carried out in absence or near absence of free water [1]. The purpose of SSF is to accumulate the target metabolites. SSF takes a certain proportion of raw materials and an appropriate cellar-entry temperature as the main preconditions for cellar-entry fermentation [2]. Once the raw material is sent into the fermentation cellar, no operation can be applied to the fermentation cellar until the end of fermentation. The yield and quality of SSF are always unstable through traditional method, which depends on artificial expertise to control the proportion of raw materials. Relationship among raw material parameters should be analyzed to ensure the quality and yield of SSF. Industrial Internet of Things (IIoT) provides the high-performance computing technology for the industrial mass production [3]–[6]. Therefore, traditional industry of SSF should be further enhanced with the technology of IIoT to realize the intelligent manufacturing.

Now, changing raw materials and adjusting cellar temperature are the two main methods to improve the quality and yield of SSF. The influence of different raw materials on the quality is analyzed [7], but the different proportions of materials on liquor quality are not studied. Microbial is another factor that affects the quality of SSF. The quality of liquor with different microbial community leads to different yield [8]. Main microorganisms affecting the quality of rice-liquor are analyzed, and the optimum temperature of the microorganisms is studied in [9]. Temperature has a significant impact on the growth and metabolism of microorganisms, which in turn affect the yield and quality of SSF. Xiang and Zhang [10] conclude that SSF temperature curve of high-quality liquor should be with the trend of rise slowly in the early stage, rise rapidly in the middle stage, and drop slowly in the later stage. In addition, year of liquor is predicted by the forecasting model based on the 3-D florescence spectra [11], but this method cannot further improve the quality and yield of liquor.

Nevertheless, different products of SSF have different microbial communities. The impact of microbes on SSF is an essential research field. However, research of real-time monitoring and control of the fermentation temperature ignores that fermentation cellar cannot be opened during the SSF process. In order to improve the yield and quality of SSF products without

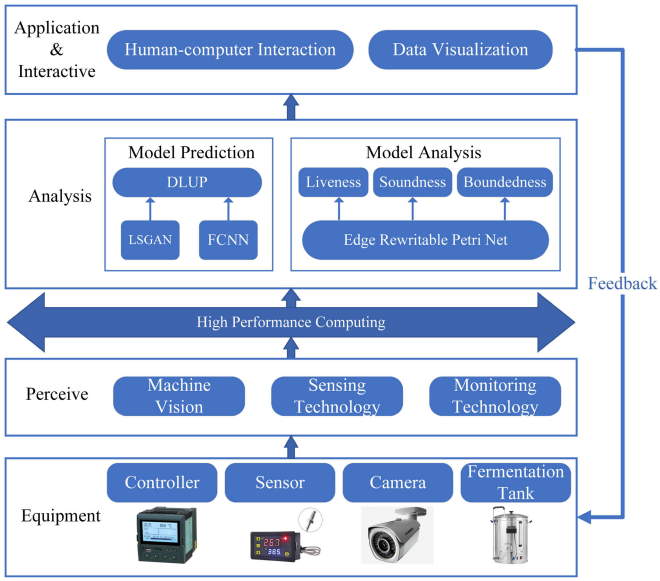

Fig. 1. Parameter collection and utility prediction of SSF process based on IIoT.

controlling the fermentation process, another effective method is to control the key preconditions of SSF in advance, because the proportion of preconditions is a vital factor affecting the quality and yield of SSF products [10].

It is a technical approach that improves the quality and yield of SSF by optimizing the key preconditions, so we need to collect SSF parameters to analyze the relationships among parameters. IIoT provides the methods for collecting data and analyzing data relationships. Parameters collection and utility prediction of SSF process based on IIoT are shown in Fig. 1. SSF equipments include controllers, sensors, cameras, and fermentation tanks. The equipment data is collected by the sensing and monitoring technology through perceive layer. High-performance computing layer provides the computing resources for analyzing the collected data [12], [13]. Collected data will be analyzed by the deep learning methods such as data generation algorithm of least squares generative adversarial network (LSGAN) and prediction algorithm fully connected neural network (FCNN). In addition, the liveness, soundness, and other properties of the system logic can be analyzed. Analysis results will be sent to equipment through the interactive layer.

Since the parameters of SSF process are obtained by regular or irregular collection, which is a typical dynamic discrete event, the collection time for parameters is dynamic. Hence, the parameters collection of SSF process is dynamically reconstructed. Petri net is a graphic modeling tool with a strict mathematical definition and is well applied to describe the discrete, synchronous, asynchronous, and concurrent processes [14]. Moreover, rewritable Petri nets [15] are proposed to solve the formal description and modeling in dynamic system reconstruction. Rewritable Petri nets provide the better analysis and verification method for dynamic discrete systems with structural reconstruction. Therefore, rewritable Petri nets can be well applied to model and verify the system framework of the parameters collection and analysis of SSF process. To predict the quality and yield of SSF, we use deep learning to analyze the hidden relationship between parameters. Deep learning is a tool for data analysis and mining [16], which needs numerous data for neural network training. The more sufficient data, the more accurate the data relationship is mined.

In this article, our contributions are as follows: 1) establish a system framework of parameters collection and utility prediction of SSF process; 2) propose an edge-rewritable Petri net to model and verify the soundness of the system framework; and 3) design a joint prediction model by combining the generating ability of LSGAN and the predicting ability of FCNN.

The rest of this article is organized as follows. Related works are introduced in Section II. Section III introduces the system framework of parameters collection and utility prediction of SSF process, and builds an edge-rewritable Petri net model of the system framework to verify the soundness of the model. Section IV describes the deep learning utility prediction (DLUP) scheme. Comparison of performance analysis is in Section V. Finally, Section VI concludes this article.

II. Related Work

The main raw material of SSF is grain. In 2019, the consumption of grain in the liquor industry by SSF reached 30 million tons. The sauce, vinegar, and other products in our daily life are all the products of SSF. According to statistics, about 10%–15% of grain is converted into food through SSF. This year, the impact of COVID-19 gives high priority to food shortage, which leads to the global food crisis. Therefore, it is of great significance to predict the quality and yield to avoid the failure of SSF.

In order to enhance the yield and quality of SSF, we apply the IIoT to SSF. Billions of smart devices are connected to create actual massive IoT, and the smart devices mutually interact and share data without any human assistance [3]. The process of parameters collection and utility prediction of SSF is a workflow. It is necessary to model the workflow to ensure the correctness of the workflow. Traditional workflow models include event–condition–action (ECA), business process execution language (BPEL), yet another workflow language (YAWL), and so on. These modeling approaches describe business process as a set of activities executed by a fixed control flow. Mansour and Jelassi [17] propose an ECA-based structure to control intelligent manufacturing systems, which is to divide the manufacturing system into three parts: event, condition, and action. In addition, using YAWL to analyze the properties of the model requires constructing the model with YAWL and using the YAWL engine [18]. In [19], the serverless application workflow is defined as a weighted directed graph. The performance and cost of the serverless application can be predicted through the analysis of the workflow model. However, these models do not scale well and have some limitations when modeling dynamic systems. Petri net is a model developed to describe distributed system. Fan *et al.* [20] use the Petri net to model the applications over the heterogeneous clouds to ensure the reliability of cloud applications. The authors propose a place-rewritable Petri net which is used to model the reconfiguration systems in [15]. According to the process of collecting the parameters of SSF

process, we propose an edge-rewritable Petri net to model the SSF process system.

Due to the insufficient data in SSF, we need to generate more effective data for analyzing. Data generated from the process of SSF is in 1-D format. Generative adversarial network (GAN) was proposed by Goodfellow in 2014 [21], which generates effective data by using few-shot data. GAN does not relay on any prior assumptions in data generation. The data generated by GAN conforms to the distribution of actual data through the max–min game between generator (G) and discriminator (D). Now, GAN is widely applied in image data generation, voice data generation, text data generation, and so on. For example, the image data augmentation method of pedestrians of small scale or in heavy occlusions is proposed in [22]; image generated in this article holds good visual quality as well as attributes. The aging image is repaired by GAN and high-quality image is generated [23]. In [24], classification enhancement GAN (CEGAN) is proposed to solve the problem of data imbalance in classification, which enhances the accuracy of target prediction in the case of data imbalance. Wasserstein GAN (WGAN) is used to repair broken teeth [26]. LSGAN adopts the least squares loss in G and D. The image quality generated by LSGAN is higher than that of GAN, and the learning process is also more stable [25]. In view of the successful application of LSGAN in image data generation, we adopt LSGAN to generate 1-D data.

After obtaining sufficient data, the relationship among different attributes needs to be analyzed. There are many prediction models such as Markov model (MM) and pairwise trust prediction through matrix factorization (PTP-MF). However, these methods mainly depend on the latest state to predict the target value, and the important characteristics of many state nodes are ignored, resulting in low accuracy of prediction. FCNN is a computational model that imitates biological neural network [27]. FCNN is a deep learning technology that is widely used in the field of classification, prediction, and so on. Long short-term memory is proposed to simulate the variation of PM2.5 pollution and then use FCNN to predict the air quality [28]. The alternating current arc faults are predicted by FCNN in [29] and the heartbeats can be classified correctly by FCNN to detect the abnormal heartbeats [30]. Based on these researches about the FCNN, it is concluded that FCNN can be well applied in analyzing and predicting data, so we adopt the FCNN to predict the quality and yield of SSF.

III. Edge-Rewritable Petri Net Model of the System Framework

System framework of DLUP for SSF is modeled by edge-rewritable Petri net in this section, and the soundness of the system framework is also verified.

A. System Framework

System framework includes parameters collection of SSF process and analysis of parametric relations to predict the quality and yield. The main parameters in SSF of liquor include acidity, starch content, humidity, maternal draff, daqu, bran shell, original cellar temperature, and ground temperature. These parameters affect the quality and yield of liquor. These factors of acidity, starch content, humidity, maternal draff, daqu, and bran shell affect each other in pairs. In order to obtain the mathematical relationship among these parameters and realize the intellectualization of SSF industry, we apply the deep learning to the SSF industry to realize the utility prediction of SSF process. The system framework of parameters collection and utility prediction of SSF process is shown in Fig. 2.

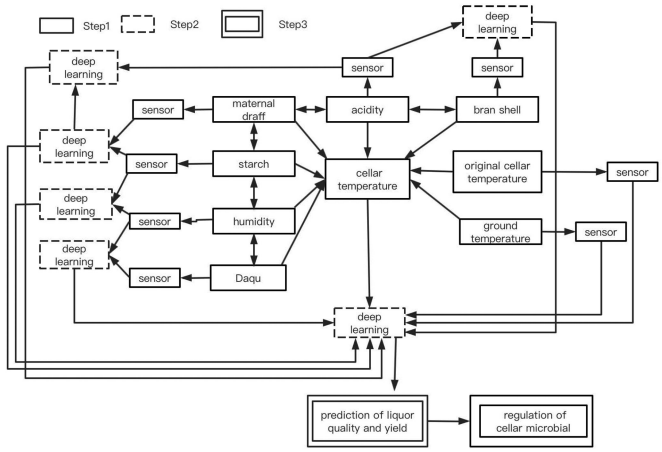

Fig. 2. System framework of parameters collection and utility prediction of SSF process.

Step1: Parameters will be collected several times by the sensors.

Step2: The relationship among parameters will be studied by deep learning algorithms.

Step3: The utility prediction model of SSF is obtained.

B. Edge-Rewritable Petri Net

Petri net is a workflow modeling and analysis tool. Since the parameters of SSF process need to be collected by several times, the edge-rewritable Petri net is proposed to model the framework. More details are given in [32].

Definition 1 (Edge-Rewritable Petri net): A 7-tuple $EN=(P, T, F, W, K, M, W_v)$ is an edge-rewritable Petri net, where (P, T, F) is a basic Petri net [14].

1) P is a set of places. Token is contained in the places.

2) T is a finite set of transition. $P\cap T=\varnothing$.

3) $F\subseteq(P\times T)\cup(T\times P)$ is a set of arcs.

4) $W: F\rightarrow\{0, 1, 2,...\}$ is a weight function, $K: P\rightarrow\{0, 1, 2,...\}$ is a capacity function, $M: P\rightarrow\{0, 1, 2,...\}$ is a marking function of EN and meets the condition of $\forall p \in P: M(p)\leq K(p)$.

5) $\exists f \in F$ is a rewritable edge, when $t_i\in T$ and t_i is a vertex of f, $\#(t_i/\sigma)=W_v\rightarrow F=F\text{-}\{f\}$. $W_v\in\mathbb{N}^*$ is the rewritable restriction of f. σ is a sequence of transitions, $\#(t_i/\sigma)$ denotes the number of t_i occurrence in σ. The rewritable edges are represented by dashed lines.

6) The firing rule of transition t in EN is the same as the general weighted Petri net [14].

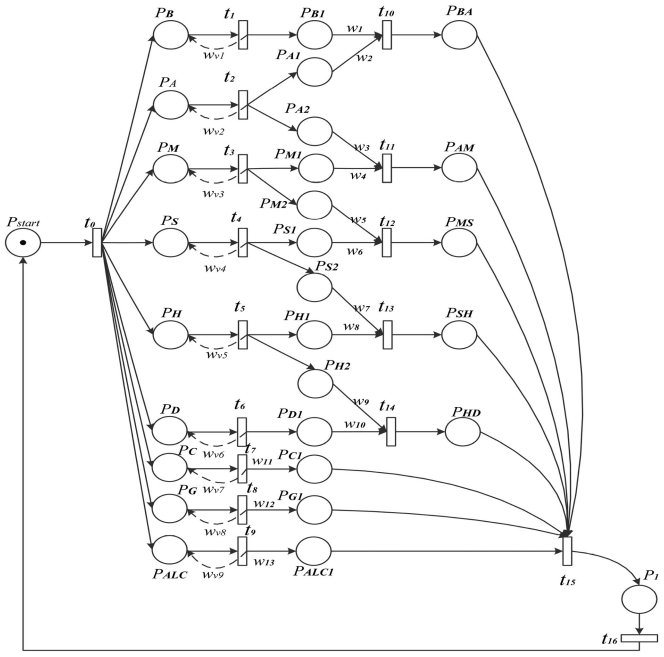

Fig. 3. Edge-rewritable Petri net model of the system framework.

C. Edge-Rewritable Petri Net Modeling the System Framework

The system framework of parameters collection and utility prediction of SSF process is a workflow. We use the edge-rewritable Petri net to model the system framework to verify the soundness of the system framework. The place P denotes the parameters and relation model of parameters; meanwhile, the transition T denotes the operation of collecting and studying the parameters relationships. The rewritable edges realize the multiple random collection of parameters.

According to the system framework, we use the edge-rewritable Petri net to model the system framework of parameters collection and utility prediction of SSF. Capital letters are used to indicate the parameters of liquor SSF. ***M***: master grains, ***S***: starch content, ***D***: daqu, ***H***: humidity, ***A***: acidity, ***B***: bran shell, ***C***: cellar temperature, ***G***: ground temperature, and ***ALC***: alcohol concentration. The edge-rewritable Petri net model of the system framework is shown in Fig. 3. The meanings of all the elements in Fig. 3 are shown in Tables I and II.

In this model, P_{start} is a start place with a token, and transition t_0 denotes the operation of taking materials into the cellar. First, t_0 is fired and then parameters places of P_B, P_A, P_M, P_S, P_H, P_D, P_C, P_G, and P_{ALC} will obtain a token. Now, the collected transitions t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, and t_9 can be fired. If t_1 is fired, then the collected parameters place P_{B1} will get a token, which means the parameter of ***B*** is obtained; meanwhile, P_B also gets a token and transition t_1 will be fired again. t_1 will be fired several times, which is a loop structure that represents the process of collecting data several times. When the number of times that t_1 is fired is equal to W_{v1}, the rewritable edge f=(t_1, P_B) will disappear. Naturally, when transition t_2 is fired, the acidity collecting parameters places of P_{A1} and P_{A2} will get a token, respectively, at the same time. In addition, the acidity parameter place P_A also gets a token and transition t_2 will be fired again. When the times that t_2 is fired is equal to W_{v2}, the rewritable edge f=(t_2, P_A) will disappear. If the number of tokens in P_{B1} and P_{A1} is equal to weight function W_1 and W_2, respectively, the study transition t_{10} will be fired. Next, the place P_{BA} of relation model for parameter B and parameter A will get a token. Similarly, the study transition t_{11}, t_{12}, t_{13}, and t_{14} are fired in the same way. Finally, we acquire the prediction model of liquor quality and yield. At this time, the workflow model of the system framework is completed. In addition, we add an adjust transition t_{16} and the edges (P_1, t_{16}) and (t_{16}, P_{start}) to form an extended network, which is used to readjust the precondition parameters ***B***, ***A***, ***M***, and so on.

TABLE I
MEANING OF EACH PLACE IN FIG. 3

Places	Meaning
P_{start}	Start
P_B, P_A, P_M, P_S	The parameters of **B**ran shell quality, **A**cidity, **M**aternal draff quality and **S**tarch content respectively.
P_H, P_D, P_C, P_G, P_{ALC}	The parameters of **H**umidity, **D**aqu quality, the **C**ellar-temperature, **G**round temperature, **ALC**ohol concentration respectively.
P_{B1}, P_{D1}, P_{C1}, P_{G1}, P_{ACL1}	The collected parameters of **B**ran shell quality, **D**aqu quality, the original **C**ellar-temperature, **G**round temperature, **ALC**ohol concentration respectively.
P_{A1}, P_{A2}	The collected parameter of **A**cidity
P_{M1}, P_{M2}	The collected parameter of **M**aternal draff quality.
P_{S1}, P_{S2}	The collected parameter of **S**tarch content
P_{H1}, P_{H2}	The collected parameter of **H**umidity
P_{BA}	The relational model of **B**ran shell quality and **A**cidity
P_{AM}	The relational model of **A**cidity and **M**aternal draff quality
P_{MS}	The relational model of **M**aternal draff quality and **S**tarch content
P_{SH}	The relational model of **S**tarch content and **H**umidity
P_{HD}	The relational model of **H**umidity and **D**aqu quality
P_1	The relationship model of quality and yield of **ALC**ohol concentration

TABLE II
MEANING OF EACH TRANSITION AND WEIGHT FUNCTION IN FIG. 3

T	Meaning	W	Meaning
t_0	Into the cellar	$W_{v1}-W_{v9}$	The weight function of rewritable edge
t_1-t_9	Collect	W_1-W_{10}	The weight function
$t_{10}-t_{15}$	Study		
t_{16}	Adjust		

D. Soundness of the System Framework

The soundness of the system framework guarantees the logic correctness of the parameters collection and utility prediction of SSF process. If a Petri net is soundness, it will meet the following definitions.

Definition 2 (Soundness): EN=(P, T, F, K, W, M, W_v) is soundness [14] if the following conditions are all satisfied.

1) For each marking M which is reached by the start marking M_0 has transition sequences σ_1 and σ_2 to the end marking M_{end}, that is, $\forall M$: $M_0[\sigma_1\rangle M \rightarrow M[\sigma_1\rangle M_{\text{end}}$.

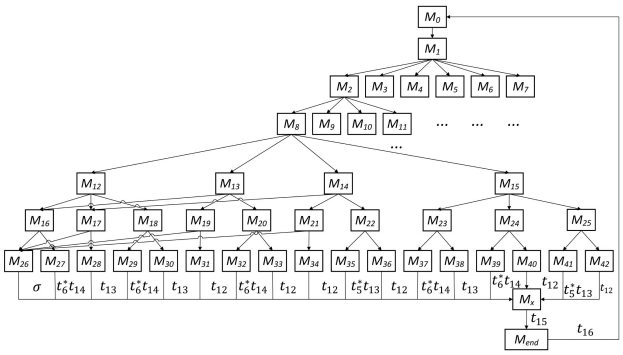

Fig. 4. Compressed reachability graph of the edge-rewritable Petri net model of system framework.

2) The end marking M_{end} (the end place p_{end} contains token) is reachable from M. M_{end} is the only one end marking, that is, $\forall M$: $(M_0[\sigma\rangle M \cap M \geq M_{\text{end}}) \rightarrow (M{=}M_{\text{end}})$.

3) There is no dead marking in EN, that is, $\forall t \in T$, $\exists M$ and M' have $M_0[\sigma\rangle M[t\rangle M'$.

Theorem 1: The edge-rewritable Petri net EN is soundness if the extended workflow [31] of EN is liveness and boundedness [14].

Reachability graph is a main tool to analyze the property of Petri net [19]. Each point is a marking in the reachability graph. We use the analysis software of Petri net to build the reachability graph for the edge-rewritable Petri net of the system framework. The result is that the reachability graph includes 1866 reachability status and 8750 arcs. More details are given in [32]. However, this reachability graph is too huge to show visually. Naturally, the reachability graph needs to be compressed, which meets the principle that the paths will not lost. The compressed reachability graph is shown in Fig. 4.

Definition 3 (Homogenous sequence): σ_1 and σ_2 are two transition sequences; if $M \xrightarrow{\sigma_1} M'$, $M \xrightarrow{\sigma_2} M'$, $\forall t \in \sigma_1 \wedge t \in \sigma_2$: $\#(t/\sigma_1){=}\#(t/\sigma_2)$, then σ_1 and σ_2 are the homogenous sequence. We use (σ_1) or (σ_2) to represent homogenous sequence. $\#(t/\sigma_1)$ and $\#(t/\sigma_2)$ denote the number of transition t in σ_1 and σ_2, respectively.

Property 1: According to Definition 3, two homogeneous sequences start from the same marking M and reach another equal marking M' through different transition sequences. We compress the paths between M and M', and the reachability [14] of the original reachability graph remains unchanged after compression.

We compress the reachability graph of Fig. 3 according to the principle of homogeneous sequence. Transitions t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, and t_9 will be fired many times according to the collection times in Fig. 3. $t_i{}^*$ denotes the transition sequence $t_i t_i ... t_i$ that t_i has been fired many times continuously. Meanwhile, transition sequences $t_7{}^*$, $t_8{}^*$, and $t_9{}^*$ are compressed as σ in Fig. 4. Each arc of the reachable graph has a transition sequence, and the marking is transformed into another marking by the transition sequence on the arc. For example, marking M_{27} is transformed into M_x through the transition sequence $t_6{}^* t_{14}$. The transition sequences among the markings in Fig. 4 are shown in Table III.

TABLE III
TRANSITION SEQUENCES AMONG THE MARKINGS

Markings	Transition sequences	Markings	Transition sequences
$M_1 \rightarrow M_2$	$(t_1{}^* t_2{}^*) t_{10}$	$M_{12} \rightarrow M_{16}$	$t_5{}^* t_{13}$
$M_1 \rightarrow M_3$	$(t_2{}^* t_3{}^*) t_{11}$	$M_{12} \rightarrow M_{17}$	$(t_5{}^* t_6{}^*) t_{14}$
$M_1 \rightarrow M_4$	$(t_3{}^* t_4{}^*) t_{12}$	$M_{12} \rightarrow M_{18}$	(σ)
$M_1 \rightarrow M_5$	$(t_4{}^* t_5{}^*) t_{13}$	$M_{13} \rightarrow M_{16}$	t_{12}
$M_1 \rightarrow M_6$	$(t_5{}^* t_6{}^*) t_{14}$	$M_{13} \rightarrow M_{19}$	$t_6{}^* t_{14}$
$M_1 \rightarrow M_7$	(σ)	$M_{13} \rightarrow M_{20}$	(σ)
$M_2 \rightarrow M_8$	$t_3{}^* t_{11}$	$M_{14} \rightarrow M_{17}$	t_{12}
$M_2 \rightarrow M_9$	$(t_3{}^* t_4{}^*) t_{12}$	$M_{14} \rightarrow M_{21}$	$t_5{}^* t_{13}$
$M_2 \rightarrow M_{10}$	$(t_4{}^* t_5{}^*) t_{13}$	$M_{14} \rightarrow M_{22}$	(σ)
$M_2 \rightarrow M_{11}$	$(t_5{}^* t_6{}^*) t_{14}$	$M_{15} \rightarrow M_{23}$	$t_4{}^* t_{12}$
$M_8 \rightarrow M_{12}$	$t_4{}^* t_{12}$	$M_{15} \rightarrow M_{24}$	$(t_4{}^* t_5{}^*) t_{13}$
$M_8 \rightarrow M_{13}$	$(t_4{}^* t_5{}^*) t_{13}$	$M_{15} \rightarrow M_{25}$	$(t_5{}^* t_6{}^*) t_{14}$
$M_8 \rightarrow M_{14}$	$(t_5{}^* t_6{}^*) t_{14}$	$M_{16} \rightarrow M_{26}$	$t_6{}^* t_{14}$
$M_8 \rightarrow M_{15}$	(σ)	$M_{16} \rightarrow M_{27}$	(σ)
$M_{17} \rightarrow M_{26}$	t_{13}	$M_{24} \rightarrow M_{39}$	t_{12}
$M_{17} \rightarrow M_{28}$	(σ)	$M_{24} \rightarrow M_{40}$	$t_6{}^* t_{14}$
$M_{18} \rightarrow M_{29}$	$t_5{}^* t_{13}$	$M_{25} \rightarrow M_{41}$	t_{12}
$M_{18} \rightarrow M_{30}$	$t_6{}^* t_{14}$	$M_{25} \rightarrow M_{42}$	$t_5{}^* t_{13}$
$M_{19} \rightarrow M_{26}$	t_{12}	$M_{21} \rightarrow M_{34}$	(σ)
$M_{19} \rightarrow M_{31}$	(σ)	$M_{22} \rightarrow M_{35}$	t_{12}
$M_{20} \rightarrow M_{32}$	t_{12}	$M_{22} \rightarrow M_{36}$	$t_5{}^* t_{13}$
$M_{20} \rightarrow M_{33}$	$t_6{}^* t_{14}$	$M_{23} \rightarrow M_{37}$	$t_5{}^* t_{13}$
$M_{21} \rightarrow M_{26}$	t_{12}	$M_{23} \rightarrow M_{38}$	$(t_5{}^* t_6{}^*) t_{14}$

The original workflow becomes an extended workflow when adding a new transition t_{16} in Fig. 3. If t_{16} is fired, the end marking will be transformed to the initial marking M_0. Since $\forall t \in T$ and $\forall M \in R(M_0)$ are all $\exists M' \in R(M)$, $M'[t\rangle$, the extended workflow with t_{16} is liveness [14]. The rewritable edges, such as (t_1, P_B), (t_2, P_A), (t_3, P_M), and so on, will disappear when the collection times are equal to the edge weight, which guarantees the boundedness of the places in the edge-rewritable Petri net model of system framework.

In summary, the extended work flow is liveness and boundedness through the analysis. Therefore, the system framework is soundness.

IV. Our Proposed DLUP Scheme

In this section, we introduce the DLUP scheme for predicting the ***ACL***. DLUP combines the generating ability of the LSGAN with the predicting ability of FCNN.

A. Motivations

The main component of maternal draff is starch. Daqu is mainly composed of microorganisms. Bran shell provides more aerobic breathing space for microbial metabolism. Cellar temperature is a necessary environment condition for SSF and microbial metabolism. As a result, the main parameters of SSF process are ***H***, ***S***, ***A***, and ***C***. These four parameters affect the quality and yield of ***ALC***.

There are only 34 real data samples. Real sample data is sparse. The depth of FCNN is the key to the accuracy of prediction. With the increasing depth of FCNN, the accuracy degradation is caused by overfitting. Based on these problems, we propose a supervised learning with few-shot learning. This method expands the training data set by generating more data,

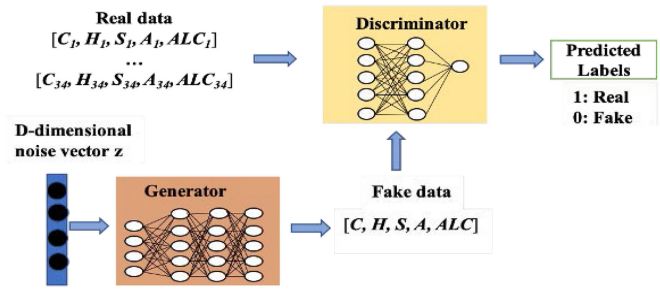

Fig. 5. Framework of LSGAN to generate one-dimensional data based on few-shot.

which is similar to the real data distribution, then we train the generated data by FCNN with five layers, so that FCNN has a stronger perception of data distribution.

First, we use the LSGAN to generate more data. Through the continuous max–min game between D and G, the data generated by G will become more consistent with the real data distribution. Second, we fix the learning parameters of G when the performance of G is no longer improved, and then input latent code z into G to generate more data. Latent code z is sampled from the standard Gaussian distribution. G perceives the feature of Gaussian distribution by sampling from the Gaussian distribution continuously, and synthesizes the generated data with Gaussian characteristics based on the prior knowledge in the game between D and G. All the data with Gaussian characteristics are not valid, so we define a filter to ensure the validity of the generated data. The filter is that the generated data is valid if the root-mean-squared error (RMSE) of the generated data and real data is less than a threshold. We obtain the final effective generated data by the filter. Finally, these effective generated data will be trained by FCNN to predict ***ALC***.

B. Data Generation Model of LSGAN

We generate more data of ***H***, ***S***, ***A***, ***C***, and ***ALC*** for the prediction model. Due to the lack of data features, we use FCNN in G and D. z$\sim P_z$ is the latent code which is input to G. θ_g is the trainable parameter of G network. G (z, θ_g) is a multilayer perception which learns to map the latent code z to the data space $\mathbb{R}^{n\times 5}$, then G will generate the data that is closer to the real distribution. The input of D is real data or the fake data. x$\sim P_x$ is the input data of D, θ_d is the trainable parameter of D network. D (x, θ_d) is another perception which discriminates whether the input data x is real data. The value of D is from 0 to 1, which indicates the real probability of the input data. G generates new data and D determines whether the generated data is getting closer to the real data. The data generated by G will become closer to the real data through the game between the D and G. The framework of LSGAN to generate one-dimensional data based on few-shot is shown in Fig. 5. The loss function of the LSGAN is as follows:

$$arg \min_G \max_D = E_{x\sim P_x}[D(x)]^2 + E_{z\sim P_z}[1-D(G(z))]^2. \quad (1)$$

C. Joint Prediction Model of FCNN and LSGAN

Training the prediction model FCNN with the few-shot real data will lead to overfitting. Therefore, we combine the LSGAN with FCNN to predict the ALC. We use the trained G in LSGAN to generate more data; the framework of the joint prediction model DLUP is shown in Fig. 6. Similarly, G (z, θ_g) maps the latent code z to the real data space $\mathbb{R}^{n\times 5}$ to generate more data. In order to reduce the difference between the generated data and the real data, we make the RMSE between the generated data and the real data. The generated data whose RMSE is less than the threshold will be preserved and the other generated data will be deleted. The amount of generated data is different because of the random latent code z. Therefore, we set a buffer to hold the generated data to ensure the stability of FCNN. δ is the size of buffer and γ is the batch size of FCNN. We set δ to be greater than γ. If the amount of data in the buffer is less than δ, then the generated data will be put into the buffer until the buffer is full. The data in buffer will be replaced by the generated data randomly when the buffer is full. The DLUP algorithm is shown in Algorithm 1. Time complexity of Algorithm 1 is O(mn^2). DLUP algorithm includes three steps.

Step 1: We use the trained G of LSGAN to generate data with latent code, and then make the RMSE between the generated data and real data. If the RMSE is less than a threshold μ_1, then the generated data will be added into the array select_data.

Step 2: Data in the select_data will be put into the buffer if the length of buffer is less than the buffer size δ. In addition, data in the buffer will be replaced randomly by the data in the select_data according to the index of the buffer when the buffer is full.

Step 3: In this step, the data in buffer will be trained by FCNN. First, the length of buffer should be judged. If the length of buffer is less than the batch size γ, then go back to the Step 1 to obtain more data for training. Second, FCNN will be trained when the length of buffer is equal to the batch size γ. Loss function of FCNN is mean-squared error (MSE). If the loss value is greater than the loss threshold value μ_2, then we use the gradient descent function Δ to refresh the network weight. Finally, we input any group of ***H***, ***S***, ***A***, and ***C*** to FCNN to get the predicted ***ALC***.

V. Performance Analysis

In this section, we design several experiments to verify the effectiveness of DLUP. 1) We joined the GAN and WGAN with FCNN, respectively, and joined the LSGAN with multiple linear regression (MLR) [33]. We compare these three joint prediction methods with our method DLUP. 2) We design four nonjoint prediction methods and compare with our method DLUP.

A. Data Preprocessing

All of our experimental data comes from the three wine cellars with similar environment A1, A2, and A3. We record ***C***, ***H***, ***S***, ***A***, and ***ALC*** every two days and get 11, 11, and 12 groups of data samples from these wine cellars, respectively. Partial SSF real parameters are shown in Table IV. We can see that the

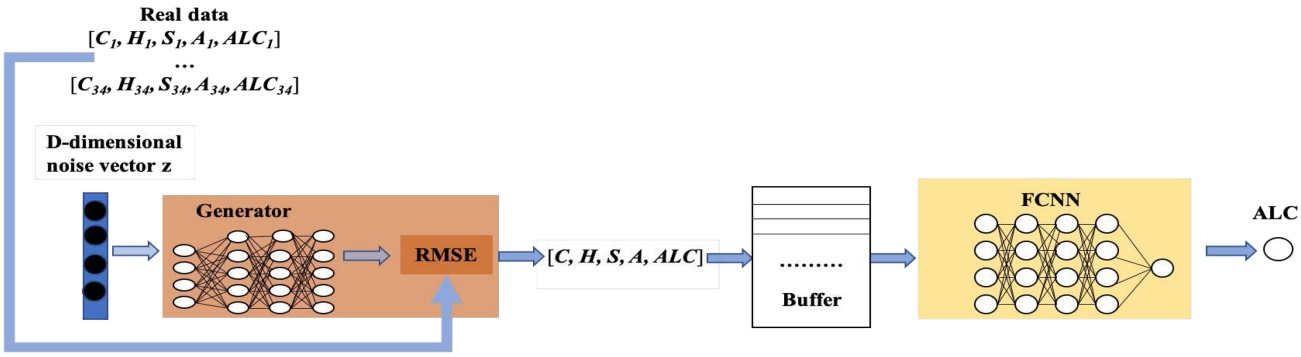

Fig. 6. Framework of the joint prediction model DLUP.

Algorithm 1: DLUP Algorithm.

Input: random noise z; generated data in G(x); real data set; random parameters S, A, C and H; buffer-size δ; batch-size γ; threshold value μ_1, μ_2; gradient descent function Δ; network weights W; learning rate α;

Output: ALC

Step 1: filter the generated data;
select_data[];
for *data (i) in G(x)* **do**
  for *data (r) in real data set* **do**
    if *RMSE of data (i) and data(r)* $<\mu_1$ **then**
      add the data (i) in the array of select_data;
    end
  end
end
Step 2: put the filtered data into buffer;
for *element in select_data* **do**
  if *length (buffer)* $<\delta$ **then**
    put the element into buffer ;
  else
    index=range[0, δ-1];
    replace the data with index in the buffer with element;
  end
end
Step 3: predict ALC;
if *length (buffer)*$<\gamma$ **then**
  jump to **Step 1**;
else
  while *loss=[train*(S, A, C, H)-ALC$]^2 > \mu_2$ **do**
    $\Delta \leftarrow$ -$g((F(S, A, C, H)\text{-}ALC)^2)$;
    $W = W + \alpha\Delta$;
  end
end
ALC=predict (S, A, C, H);
return ALC;

range of values for each dimension attribute varies greatly. In order to accelerate the speed of network convergence, we adopt a min–max method to scale all the original data into [0,1]; the

TABLE IV
REAL PARAMETERS OF SSF

Cellar Temperature	Humidity%	Starch content%	Acidity%
40	45.31	35.17	1.42
40.5	45.4	34.85	1.55
43	45.4	34.25	1.69
42	45.42	34.12	1.69

TABLE V
KLD OF THE THREE DATA GENERATION ALGORITHMS

Parameters	GAN [21]	WGAN [26]	DLUP
C	0.0061	0.0061	0.0058
H	0.0056	0.0049	0.0055
S	0.0366	0.0255	0.0335
A	0.0316	0.0320	0.0310
ALC	0.161	0.239	0.153
Total	0.2409	0.3075	0.2288

method is described as follows:

$$X = \frac{X - Min}{Max - Min}. \tag{2}$$

First, we input the real data into LSGAN to train the G and D until generated data is consistent with the real data. Next, we input the latent code into the trained G to generate more data for FCNN to predict. Furthermore, we make the RMSE between generated data and real data, and then we filter the generated data with RMSE greater than 0.15 through several experiments. We set the epoch of LSGAN as 20 and set the epoch of FCNN as 20 000.

We also generate data by GAN and WGAN and compare with our proposed method DLUP. We use Kullback–Leibler divergence (KLD) to calculate the similarity between the distribution of real data and generated data. The smaller the KLD value, the closer the generated data is to the real data. The KLD of C, H, S, A, and ALC is shown in Table V. The results show that the data generated by DLUP has the total minimum KLD.

B. Comparison and Analysis of Experimental Results

We joined FCNN with GAN (GAN-FCNN) and WGAN (WGAN-FCNN), respectively, and LSGAN with MLR (LSGAN-MLR) as well, and compare the prediction results with our method DLUP. We design a FCNN with five layers, and the structure of FCNN is shown in Fig. 7. The prediction results are

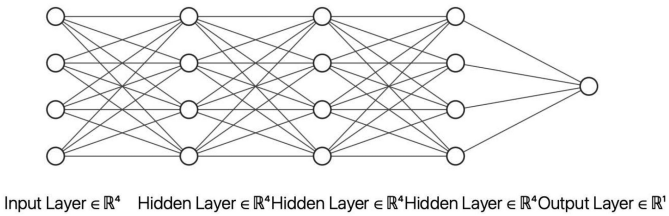

Fig. 7. Structure of FCNN.

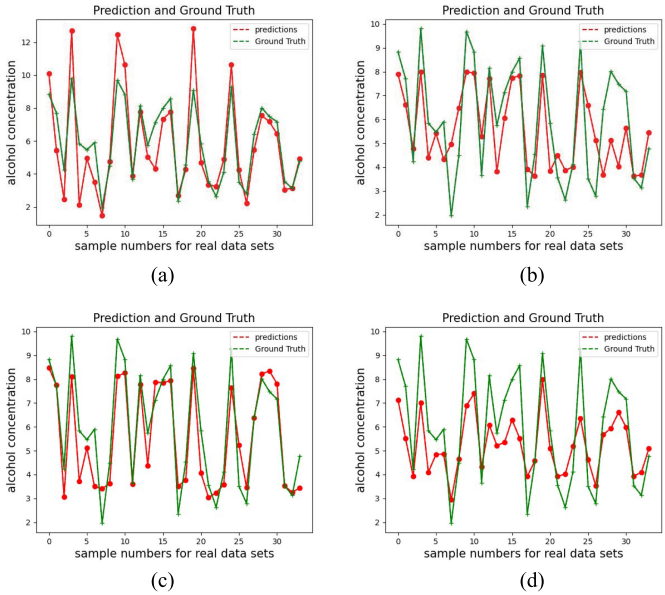

Fig. 8. Prediction results of ***ALC*** based on the GAN-FCNN, WGAN-FCNN, DLUP, and LSGAN-MLR.

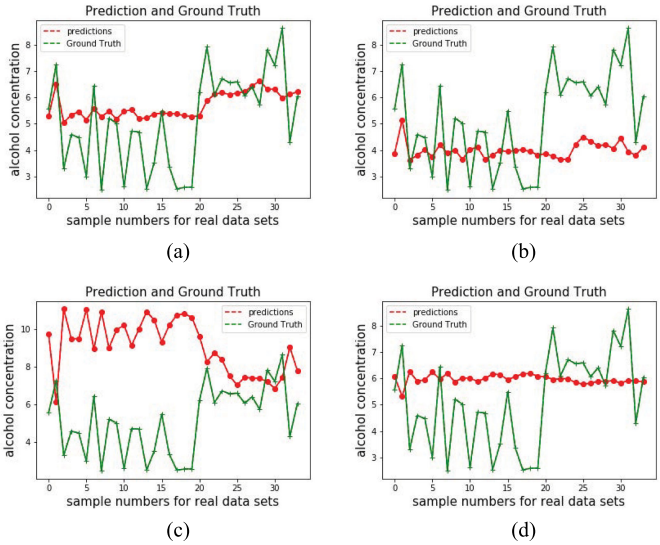

Fig. 9. ***ALC*** prediction results based on the nonjoint GAN-FCNN [21], [27], nonjoint WGAN-FCNN [26], [27], nonjoint DLUP [25], [27], and nonjoint LSGAN-MLR [25], [33].

TABLE VI
MSE AND RUNNING TIME OF EACH PREDICTION MODEL

Prediction models	MSE	Running time
GAN-FCNN	1.535	4.2h
WGAN-FCNN	1.6618	4.3h
DLUP	1.0681	3.9h
LSGAN-MLR	1.505	3.5h
Nonjoint GAN-FCNN [21], [27]	1.579	0.21h
Nonjoint WGAN-FCNN [26], [27]	2.043	0.25h
Nonjoint DLUP [25], [27]	4.986	0.33h
Nonjoint LSGAN-MLR [25], [33]	2.062	0.2h

verified by using 34 sets of real data. The results are shown in Fig. 8. X-axis represents the sample numbers (sample number is random) of the data sets and Y-axis represents the ***ALC***. The green line represents the real ***ALC***, and the red line represents the ***ALC*** predicted by the joint prediction models.

In addition, we generate data by LSGAN and filter the generated data with RMSE greater than 0.15, then we use the filtered data to train FCNN. In this method, G trained by LSGAN does not continue to generate data for FCNN, which is a nonjoint training method. Similarly, we construct the nonjoint model by combining GAN with FCNN and combining WGAN with FCNN. Moreover, we generate data by LSGAN and filter the generated data with RMSE greater than 0.15, then the filtered data will be trained by MLR to predict ***ALC***. The prediction results are shown in Fig. 9.

We get MSE between the real ***ALC*** and the predicted ***ALC***, and the MSE of each method is shown in Table VI. The smaller the MSE is, the more accurate the prediction is. Each algorithm is tested 10 times, and the average running time of each algorithm is shown in Table VI. We can see that our method DLUP has the minimum MSE 1.0681. However, the running time of DLUP is higher than the other nonjoint prediction methods.

In summary, our DLUP scheme combines the generating ability of LSGAN with the predicting ability of FCNN. DLUP has higher prediction accuracy through the experiments. However, the running time of DLUP is longer than the other nonjoint prediction methods in this article.

C. Engineering Application

SSF has a good application in enzyme preparation, organic acid flavor, and other industry fields. SSF puts the preproportion raw materials into the fermentation tanks, and the fermentation process cannot be controlled or changed any more. The production of liquor mainly relies on SSF. Using deep learning to predict the quality and yield of SSF can reduce the failure probability in SSF. Fig. 10 shows the data processing of liquor SSF.

However, the proportion of raw material depends on the artificial expertise, which leads to unstable yield and quality. To improve the utilization in SSF, we predict the quality and yield in advance according to the proportion of raw materials before SSF. In Fig. 11, collected data will be analyzed by our proposed DLUP scheme. Specifically, we use the G of trained LSGAN to generate more data, and then joined the trained G of LSGAN with FCNN to predict the yield and quality of SSF.

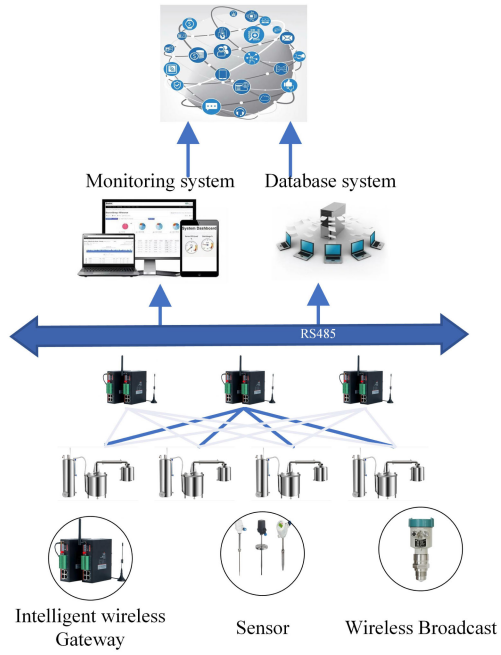

Fig. 10. Data processing of liquor SSF based on IIoT.

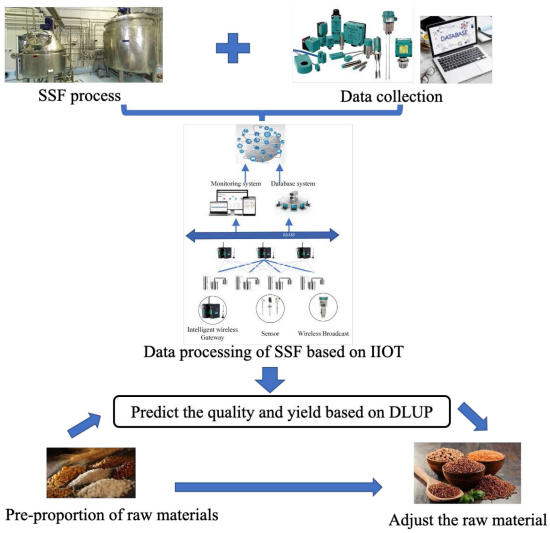

Fig. 11. Application of SSF prediction based on DLUP in IIoT.

This method predicts the quality and yield of raw materials in advance, which reduces the failure probability of SSF.

VI. Conclusion

We designed a system framework for collecting parameters and predicting the quality and yield of liquor SSF in IIoT. System framework of liquor SSF was modeled by the edge-rewritable Petri net and the soundness of the system framework was proved. Since the real data only has 34 groups, we generated data by trained G of LSGAN, then these generated data was used to predict yield and quality by FCNN. Trained G of LSGAN generates more data that is close to the true distribution. FCNN was able to analyze the relationship of ***C***, ***H***, ***S***, ***A***, and ***ALC***. Experiments showed that the prediction accuracy of the joint prediction algorithms was higher than that of the other nonjoint prediction algorithms, and DLUP outperformed the other joint models GAN-FCNN, WGAN-FCNN, and LSGAN-MLR.

We will focus on developing more efficient ways to enhance the quality and yield for SSF. Influences of ***C***, ***H***, ***S***, and ***A*** on ***ALC*** are discussed in this article. However, microorganism is another factor that affects the quality of SSF. In the future, the influence of microorganism on SSF will be studied.

References

[1] R. R. Singhania, A. K. Patel, and R. C. Soccol, "Recent advances in solid-state fermentation," *Biochem. Eng. J.*, vol. 44, no. 1, pp. 13–18, 2009.
[2] W. Zou, C. Q. Zhao, and H. Luo, "Diversity and function of microbial community in Chinese strong-flavor Baijiu ecosystem: A review," *Front. Microbiol.*, vol. 9, no. 671, pp. 1–15, 2018.
[3] L. D. Xu, W. He, and S. Li, "Internet of Things in industries: A survey," *IEEE Trans. Ind. Informat.*, vol. 10, no. 4, pp. 2233–2243, Nov. 2014.
[4] N. Xiong *et al.*, "A self-tuning failure detection scheme for cloud computing service," in *Proc. IEEE 26th Int. Parallel Distrib. Process. Symp.*, 2012, pp. 668–679.
[5] Y. Zeng, N. Xiong, J. H. Park, and G. Zheng, "An emergency-adaptive routing scheme for wireless sensor networks for building fire hazard monitoring," *Sensors*, vol. 10, no. 6, pp. 6128–6148, 2010.
[6] H. Li, J. Liu, R. W. Liu, N. Xiong, K. Wu, and T. Kim, "A dimensionality reduction-based multi-step clustering method for robust vessel trajectory analysis," *Sensors*, vol. 17, no. 8, pp. 1792.1–1792.26, 2017.
[7] Y. Li, "Production of Feng–Luzhou-flavor combined liquor by strengthening technical research," *Liquor-Making*, vol. 35, no. 1, pp. 23–26, 2008.
[8] L. J. Chai *et al.*, "Profiling the clostridia with butyrate-producing potential in the mud of Chinese liquor fermentation cellar," *Int. J. Food Microbiol.*, vol. 297, pp. 41–50, 2019.
[9] Y. Xuan, Y. Yumiko, M. Ikenaga, and X. Han, "Manufactural impact of the solid-state saccharification process in riceflavor Baijiu production," *J. Biosci. Bioeng.*, vol. 129, no. 3, pp. 315–321, 2019.
[10] S. Q. Xiang and Z. G. Zhang, "The relations between the production factors in solid-state fermentation of traditional Nongxiang Baijiu," *Liquor-Making Sci. Technol.*, vol. 1, no. 1, pp. 56–59, 2016.
[11] J. Gu *et al.*, "Year prediction and flavor classification of Chinese liquors based on fluorescence spectra," *Measurement*, vol. 134, pp. 48–53, 2019.
[12] B. Lin *et al.*, "A time-driven data placement strategy for a scientific workflow combining edge computing and cloud computing," *IEEE Trans. Ind. Informat.*, vol. 15, no. 7, pp. 4254–4265, Jul. 2019.
[13] W. Fang, X. Yao, X. Zhao, J. Yin, and N. Xiong, "A stochastic control approach to maximize profit on service provisioning for mobile cloudlet platforms," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 48, no. 4, pp. 522–534, Apr. 2018.
[14] T. Murata, "Petrinets: Properties, analysis and applications," *Proc. IEEE*, vol. 77, no. 4, pp. 541–580, Apr. 1989.
[15] S. C. Pang and C. Lin, "Rewritable petri nets: Rewritable place and properties analysis," *Chin. J. Comput.*, vol. 35, no. 10, pp. 202–213, 2012.
[16] B. Yi *et al.*, "Deep matrix factorization with implicit feedback embedding for recommendation system," *IEEE Trans. Ind. Informat.*, vol. 15, no. 8, pp. 4591–4601, Aug. 2019.
[17] W. Mansour and K. Jelassi, "RFID technology to control manufacturing systems using OPC server," in *Proc. Int. Conf. Elect. Sci. Technol.*, 2014, pp. 1–4.
[18] W. M. Van Der Aalst and A. H. Ter Hofstede, "YAWL: Yet another workflow language," *Inf. Syst.*, vol. 30, no. 4, pp. 245–275, 2005.
[19] C. Lin and H. Khazaei, "Modeling and optimization of performance and cost of serverless applications," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 3, pp. 615–632, Mar. 2021.
[20] G. Fan, H. Yu, and L. Chen, "Formally modeling and analyzing the reliability of cloud applications," *Int. J. Softw. Eng. Knowl. Eng.*, vol. 26, no. 2, pp. 273–305, 2016.
[21] I. Goodfellow, J. P. Abadie, M. Mehdi, and B. Xu, "Generative adversarial nets," in *Proc. Conf. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.
[22] S. Liu *et al.*, "A novel data augmentation scheme for pedestrian detection with attribute preserving GAN," *Neurocomputing*, vol. 401, pp. 123–132, 2020.

[23] F. Konidaris, T. Tagaris, M. Sdraka, and A. Stafylopatis, "A generative adversarial networks as an advanced data augmentation technique for MRI data," *IEEE Trans. Med. Imag.*, vol. 37, no. 3, pp. 673–679, Feb. 2018.
[24] S. Suh, H. Lee, P. Lukowicz, and Y. O. Lee, "CEGAN: Classification enhancement generative adversarial networks for unraveling data imbalance problems," *Neural Netw.*, vol. 133, pp. 69–86, 2021.
[25] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z.Wang, and S. P. Smolley, "On the effectiveness of least squares generative adversarial networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 12, pp. 2947–2960, Dec. 2018.
[26] S. Tian *et al.*, "Efficient computer-aided design of dental inlay restoration: A deep adversarial framework," *IEEE Trans. Med. Imag.*, vol. 40, no. 9, pp. 2415–2427, Sep. 2021.
[27] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *Proc. IEEE Int. Conf. Acoust.*, 2015, pp. 4580–4584.
[28] J. Zhao, F. Deng, Y. Cai, and J. Chen, "Long short-term memory-fully connected (LSTM-FC) neural network for PM(2.5) concentration prediction," *Chemosphere*, vol. 220, pp. 486–492, 2019.
[29] Y. Wang, F. Zhang, X. Zhang, and S. Zhang, "Series AC ARC fault detection method based on hybrid time and frequency analysis and fully-connected neural network," *IEEE Trans. Ind. Informat.*, vol. 15, no. 2, pp. 6210–6219, Dec. 2019.
[30] H. Wang *et al.*, "A high-precision arrhythmia classification method based on dual fully connected neural network," *Biomed. Signal Process. Control*, vol. 58, 2020, Art. no. 101874.
[31] V. Rodriguez-Fernandez, A. Gonzalez-Pardo, and D. Camacho, "Automatic procedure following evaluation using petri net-based workflows," *IEEE Trans. Ind. Informat.*, vol. 14, no. 6, pp. 2748–2759, Jun. 2018.
[32] M. Wang *et al.*, "IUP: An intelligent utility prediction scheme for solid-state fermentation in 5G IoT," 2021, *arXiv:2103.15073*.
[33] W. Q. Xiong, J. D. Wang, and K. Chen, "Multivariate alarm systems for time-varying processes using Bayesian filters with applications to electrical pumps," *IEEE Trans. Ind. Informat.*, vol. 14, no. 2, pp. 504–513, Feb. 2018.

Min Wang received the master's degree in software engineering in 2019 from the China University of Petroleum, Qingdao, China, where she is currently working toward the Ph.D degree in control science and engineering.

Her research interests include theory and application of Petri net, trusted computing, and deep learning.

Shanchen Pang received the graduation degree from the Tongji University of Computer Software and Theory, Shanghai, China, in 2008 and the Ph.D. degree in computer software and theory from the Tongji University, Shanghai, in 2008.

He is currently a Professor with the China University of Petroleum, Qingdao, China. His research interests include theory and application of Petri net, service computing, trusted computing, and artificial intelligence.

Tong Ding received the master's degree in software engineering from the China University of Petroleum, QingDao, China, in 2020. He is currently working toward the Ph.D. degree in artificial intelligence with the School of Software, Shandong University, Shandong, China.

He has authored or coauthored three papers with Sci Retrieval, which include the *Journal of Information Science*. His research interests include reinforcement learning, federated learning, and unmanned vehicles path processing.

Sibo Qiao received the master's degree in software engineering in 2020 from the China University of Petroleum, Qingdao, China, where he is currently working toward the Ph.D. degree in computer technology and resource information engineering.

His research interests include data mining, deep learning, and machine learning.

Xue Zhai received the B.S. degree in network engineering from the Qingdao University of Technology, Qingdao, China, in 2018. She is currently working toward the Ph.D. degree in computer technology and resource information engineering with the China University of Petroleum, Qingdao.

Her research interests include Petri net, blockchain, and microservices.

Shuo Wang received the B.S. degree from the Shandong University of Science and Technology, Taian, China, in 2017, and the M.S. degree in software engineering from the China University of Petroleum, Qingdao, China, in 2020. He is currently working toward the Ph.D. degree in computer science and technology with the Tongji University, Shanghai, China.

His research interests include formal verification and microservices.

Neal N. Xiong (Senior Member, IEEE) received the Ph.D. degrees in sensor system engineering and dependable communication networks from Wuhan University, Wuhan, China and Japan Advanced Institute of Science and Technology, Nomi, Japan, in 2007 and 2008, respectively.

He is currently an Associate Professor with the Department of Computer Science and Mathematics, Sul Ross State University, Alpine, TX, USA. He was with Sul Ross State University, Georgia State University, Northeastern State University, and a Full Professor with Colorado Technical University. His research interests include cloud computing, security and dependability, parallel and distributed computing, networks, and optimization theory.

Zhengwen Huang received the B.Sc. degree from the University of Science and Technology, Hefei, China, the M.Sc. degree from King's College London, London, U.K., and the Ph.D. degree from the Department of Electronic and Computer Engineering, Brunel University London, Uxbridge, U.K. in 2014.

He is currently a Senior Research Fellow with Systems Engineering Research Group, Brunel University. He is the Leader of Artificial Intelligence and System Optimization Group, BUL-CQUPT Innovation Centre, Chongqing University of Posts and Telecommunications, Chongqing, China. His research interests include evolutionary algorithms (gene expression programming, genetic programming) and data engineering.