Journal of Systems Engineering and Electronics
Vol. PP, No. 99, April 2023, pp.1 – 13

Product quality prediction based on RBF optimized by firefly algorithm

HAN Huihui*, WANG Jian, CHEN Sen, and YAN Manting

College of Electronics and Information Engineering, Tongji University, Shanghai 201804, China

Abstract: With the development of information technology, a large number of product quality data in the entire manufacturing process is accumulated, but it is not explored and used effectively. The traditional product quality prediction models have many disadvantages, such as high complexity and low accuracy. To overcome the above problems, we propose an optimized data equalization method to pre-process dataset and design a simple but effective product quality prediction model: radial basis function model optimized by the firefly algorithm with Levy flight mechanism (RBFFALM). First, the new data equalization method is introduced to pre-process the dataset, which reduces the dimension of the data, removes redundant features, and improves the data distribution. Then the RBFFALFM is used to predict product quality. Comprehensive expe riments conducted on real-world product quality datasets validate that the new model RBFFALFM combining with the new data pre-processing method outperforms other previous me thods on predicting product quality.

Keywords: product quality prediction, data pre-processing, radial basis function, swarm intelligence optimization algorithm.

DOI: 10.23919/JSEE.2023.000061

1. Introduction

Proposals of some great strategies, such as 'Industry 4.0' [1], 'Industrial Internet', and 'Made in China 2025' have promoted the traditional manufacturing industry to move towards intelligent production, and thus the manufacturing industry has entered the era of big data. The core content of an intelligent manufacturing execution system is quality management, which is crucial for enterprises to obtain profits and long-term competitive advantages. Product quality problems often stem from the manufacturing process, which has the characteristics of complex technology, multi-flow, and long cycle. With the continuous development of information construction in manufacturing enterprises, massive data from the entire product production process is accumulated in the enterprise database. However, the potential knowledge behind data is not mined effectively, resulting in that the product quality data is rich, but the knowledge behind it is poor. Knowledge discovery in database (KDD) [2,3], combining probability theory, statistics, and other technologies, can efficiently discover the potential knowledge behind the data, help enterprises dig for useful knowledge, and assist technicians in making decisions. The mined rule knowledge by KDD, the association rule between parts processing data and product quality, can be taken as the basis for quality prediction. There are many classic algorithms to extract association rules such as Apriori [4], Eclat [5], FP-Growth [6], and so on.

In the complex multi-stage manufacturing process of products, it is difficult to trace the quality of products due to the influence of various factors. In recent years, research on product quality prediction has been conducted widely. The fuzzy analytical hierarchy process is applied to figure out the crucial factor affecting product quality in [7], and the state space equation is used to build a mechanical assembly accuracy prediction model, aiming to improve the quality stability of machining products. A multi-task learning model in encoder-decoder architecture was put forward in [8], applying matrix factorization technique and attention mechanism to improve the accuracy of predicting product quality. A hierarchical quality monitoring algorithm based on the distributed parallel semi-supervised Gaussian mixture model is proposed to monitor product quality in [9]. To remove the bad influence of unknown disturbance on the accuracy of product quality prediction, Yang et al. [10] proposed a novel hybrid denoising-linear-nonlinear quality prediction model, which also introduces the support vector machine (SVM) to process nonlinear data. Ren et al. [11] introduced the semisupervised learning mechanism and

Manuscript received June 21, 2021.
*Corresponding author.
This work was supported by the National Science and Technology Innovation 2030 Next-Generation Artifical Intelligence Major Project (2018AAA0101801) and the National Natural Science Foundation of China (72271188).

the manifold regularization to construct a new product quality prediction model, which can deal with the unlabeled samples issue. To overcome the problem that dataset from the semiconductor manufacturing process is of high dimension and nonlinear, Hu et al. [12] put forward a modified SVM model on the basis of feature selection.

Although a lot of research works have improved the accuracy of predicting product quality, challenges still exist. First, most product quality prediction models do not take data pre-processing seriously. Product quality data, from complex manufacturing environments, such as steel production lines, water heater production, rocket engine manufacturing, etc, is characterized by a large amount, high dimension, mutual coupling, imbalance, and so on. The quality of data has a great influence on training product quality prediction models. The data equalization operation, as one type of data pre-processing method, contains undersampling, oversampling, and hybrid sampling. However, in current studies, the undersampling methods often have the problems of mistakenly deleting key samples and destroying the overall distribution of samples. The oversampling methods also have the problems of destroying the overall distribution of samples and generating noise. These problems lead to that the information mined from data is not accurate. Therefore, this paper introduces an optimized data equalization method (ODEM) to pre-process the data and reduce the bad influence of unbalanced data on training the new product quality prediction model. Second, the common product quality prediction models have some limitations, such as high complexity, low accuracy, and slow convergence speed. In contrast, the radial basis function (RBF) [13] neural network prediction model, where the Gaussian kernel function is selected as the kernel function, has a better approximation ability and global optimal performance. Besides, RBF can minimize training error and confidence interval. The learning speed and training speed are much faster than the back propagation (BP) neural network, and there is no local minimum problem. Therefore, this paper proposes a novel product quality prediction model: RBF optimized by firefly algorithm (FA) [14] with Levy flight mechanism (LFM), abbreviated as RBFFALFM.

In summary, the main contributions of this paper can be summarized as follows:

(i) An optimized data equalization method (ODEM) is designed to pre-process the dataset, which greatly reduces the bad influence of the unevenly distributed data on the training product quality prediction model.

(ii) To improve prediction accuracy and reduce model complexity, we propose a new product quality prediction model: RBFFALFM, where the FA with LFM is introduced to optimize the parameters in RBF.

(iii) Extensive experiments conducted on the real-world dataset from a hot strip rolling line indicate that the new product quality prediction model achieves more impressive performance than other the state-of-the-art methods.

2. Related work

2.1 Data pre-processing

The dataset in complex industrial process is characterized by missing, wrong, unbalanced, duplicated, etc. These features often have a negative influence on model training and model performance. To solve the above problems, a lot of research work has been carried out on data pre-processing. Combining the self-organizing map clustering mechanism, Tang et al. [15] put forward a novel data pre-processing approach with high feasibility and effectiveness in the research of intelligent root selection of hanger and support. To improve the recognition rate of the speaker identification systems, two pre-processing approaches, adaptive noise canceller and Savitzky-Golay filter, were applied in [16] to reduce the influence of noise in a complex environment. Yan et al. [17] presented a smoothing pre-processing approach to effectively enhance the identification performance of mean spectral radius-based detectors. Wang et al. [18] took the overall process of the gas reaction as a feature map in data pre-processing to further improve the gas classification of electronic nose algorithms. To improve the classification accuracy, Nickolas et al. [19] proposed a data pre-processing method, which combines the correlation-based variable selection technique, synthetic minority oversampling technique, and the imputation technique. Bezin et al. [20] proposed an automatic pre-processing method for measured rail data to overcome the challenge of correcting vertical and lateral misalignments of the profile cross-sections.

2.2 Clustering algorithm

As one of the most fundamental tasks in data mining, clustering aims to divide unlabeled data into groups based on similarity without supervision, instead of offering a precise characterization of unobserved samples [21]. The classic clustering algorithms include graph theory-based clustering, combinatorial search techniques-based clustering, fuzzy clustering, neural networks-based clustering, and kernels techniques-based clustering. It is widely applied in various fields, such as pattern recognition, energy system, image segmentation, and so on. The clustering algorithm is applied in wireless sensor networks in

[22] to reduce the loss of node energy and expend network life, which further efficiently saves energy. Miao et al. [23] proposed a dynamic cluster head adaptive clustering algorithm in wireless sensor network target tracking to improve the tracking accuracy and save sensor energy. To improve the detection rate of the 5G-enabled industrial Internet intrusion detection algorithm, density-awared fuzzy clustering was introduced in [24]. Hooda et al. [25] introduced the fuzzy-gravitational search algorithm to segment the brain tissues in magnetic resonance imaging brain image, which reduces the error of manual segmentation and greatly promotes efficiency.

2.3 Swarm intelligence optimization algorithm

Nature-derived algorithms are the most powerful algorithms for optimization. Recently, swarm intelligence [26], the biologically inspired algorithms, have been studied dramatically, such as firefly swarm optimization algorithm [14], ant colony optimization technique [27], particle swarm optimization (PSO) algorithm [28], artificial fish swarm optimization algorithm [29], and swallow swarm optimization (SSO) [30]. They have been applied to solve various problems, such as healthcare, finance, energy, image thresholding, and others. Rahul et al. [31] applied the FA to optimize data analysis in healthcare industries, and further offer optimized solutions. Wu et al. [32] proposed a novel discrete PSO algorithm on the basis of the 0-1 knapsack mechanism to deal with the complicated matching problems between enterprises' financial products. Combining the beetle swarm optimization and the PSO, Bhagavathi et al. [33] designed an improved beetle swarm optimization algorithm, which can effectively reduce the energy consumption in the process of virtual machine consolidation. Dutta et al. [34] proposed the qubit fractional order PSO and qutrit fractional order PSO for hyperspectral image thresholding.

3. Methods

3.1 Data pre-processing

Generally, the product quality dataset is multi-dimension, which includes crucial variables and irrelevant ones. This kind of data easily leads to overfitting problems during the training process. It is necessary to reduce the dimension of the data, and remove redundant features and irrelevant features. Also, these data are unevenly distributed, thus it is crucial to modify the distribution of an unbalanced dataset. An ODEM is developed. The key rule undersampling and clustering in generative adversarial networks (ClusterGAN) oversampling [35] are introduced to ODEM, which can effectively improve both the calculation efficiency and the prediction accuracy of the new quality prediction model RBFFALFM.

3.1.1 Key rule undersampling method

The association rule algorithm FP-Growth is first conducted on the dataset D to generate the rule set R_{all}. Then the ordered rule set is obtained by sorting the degree of sort and confidence. The key rule is the one at the top of the sorting list. Then, the k-means clustering algorithm is executed on the samples with a most class to obtain clusters, and the samples conforming to the key rule are kept in each cluster. Next, the number of samples that need to be kept after sampling can be calculated by

$$N_i = \frac{M_i}{M} \times N \tag{1}$$

where N_i denotes the number of samples that need to be kept after sampling in the ith cluster. M and M_i represent the number of samples with a most class before sampling, and the total number of samples in the ith cluster respectively.

Key rules are used to ensure that the key samples are not deleted by mistake. Using clustering, the individual sampling within each cluster is guaranteed to be consistent with the overall distribution of samples. The pseudocode can be seen in Algorithm 1.

Algorithm 1 Key rule undersampling method

Input: dataset D, the number of samples N that needs to be kept after sampling, the number of clusters K in k-means cluster

Output: dataset after undersampling

1: data standardization

2: data discretization

3: data standardization in a one-hot way

4: generating association rule by FP- Growth

5: obtaining the ordered rule set R_{all}

6: getting K clusters by conducting k-means clustering on samples with most classes

7: initializing $i = 1$, $j = 1$, $S = \phi$, S_{ij} contains all samples in the ith cluster

8: **while** $i \leqslant k$ **do**

9: calculating N_i by (1)

10: flag = 0

11: **while** $j \leqslant |\text{R}_{\text{all}}|$ **do**

12: selecting samples meeting the jth rule in R_{all} to form S_{ij}

13: **if** $|\text{S}_{ij}| \leqslant N_i$ **do**

14: selecting $N_i - |S_{ij}|$ samples and adding S_{ij} to form S_i

15: adding S_i to S

16: flag = 1
17: break
18: **else**
19: $j++$
20: **end if**
21: **end while**
22: **if** flag = 0 **do**
23: selecting N_i samples from S_{ij} to form S_i
24: adding S_i to S
25: **end if**
26: $i++$
27: **end while**

3.1.2 ClusterGAN oversampling

The oversampling method eliminates the imbalance problem by generating a few classes. The existing oversampling methods often destroy the population distribution of samples and fail to consider that the newly generated samples may be noise points. To overcome the above disadvantages, an improved oversampling method based on ClusterGAN is constructed. First, a few class samples are clustered, which ensures the distribution consistency between individual sampling in each cluster and the overall samples. Then the ClusterGAN method is used to generate new samples in each cluster. The number of new samples in the ith cluster that should be generated can be calculated by

$$n_i = \frac{m_i}{m+n} \times n \tag{2}$$

where n_i represents the number of samples with a few classes that need to be generated in the ith cluster. m and m_i represent the total number of samples with a few classes before ClusterGAN, and the initial number of samples in the ith cluster respectively.

Next, the k-nearest neighbour method is applied to detect the outliers in the new samples and delete these outliers to ensure that no noise points are generated. The pseudo-code of the ClusterGAN oversampling can be seen in Algorithm 2.

Algorithm 2 ClusterGAN algorithm

Input: dataset D; before sampling, the sample numbers of majority class and minority class are m and M respectively; the number of samples that most classes need to keep is n; the number that a few classes need to generate is N; the proportion between before sampling and after sampling is r_1; after sampling, the proportion between qualified products and unqualified products is r_2; s is the original sample set with a few class.

Output: dataset after ClusterGAN

1: obtaining K clusters by conducting k-means clustering on samples with a few classes
2: initializing $i = 1$,
3: **while** $i \leqslant k$ **do**
4: calculating n_i by (2)
5: utilizing ClusterGAN to produce new samples for each sample in the ith cluster to form s_i
6: updating the discriminator parameters by $\max\limits_{D} \sum\limits_{i=1}^{n} \lg D(C(\boldsymbol{x}_i), \boldsymbol{x}_i) + \sum\limits_{j=1}^{n} \lg(1 - D(\boldsymbol{z}_j, G(\boldsymbol{z}_j)))$
7: updating the generator parameters by $\min\limits_{G} \sum\limits_{j=1}^{n} \lg(1 - D(\boldsymbol{z}_j, G(\boldsymbol{z}_j)))$
8: updating the cluster parameters by $\min\limits_{C} \sum\limits_{i=1}^{n} \lg D(C(\boldsymbol{x}_i), \boldsymbol{x}_i) + v_i l_i + \|C(\boldsymbol{x}_i) - C(\tilde{\boldsymbol{x}}_i)\|^2$
9: updating the self-paced learning parameters by $\min\limits_{v} \sum\limits_{i=1}^{n} v_i l_i - \lambda_v \|v\|_1 + \gamma \|v\|_{\varepsilon}$ s.t. $v \in [0,1]^n$
10: repeat
11: utilizing the k-nearest neighbour method to detect noises
12: deleting noises
13: generating $n_i - |s_i|$ samples and adding to s_i
14: until $n_i = |s_i|$
15: adding s_i to s
16: $i++$
17: **end while**

3.1.3 Autoencoder

Autoencoder, a popular neural network, is widely applied to the unsupervised data compression and feature extraction. An autoencoder is composed of an encoder, which transforms the input vector into the latent representation $\boldsymbol{\Psi} = E(\boldsymbol{X}) \in \boldsymbol{R}_d$, and a decoder, aiming to reconstruct the latent representation back to the reconstructed vector.

In the encoder stage, the input vector $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n)$ is transformed to $\boldsymbol{\Psi} = (\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \cdots, \boldsymbol{\xi}_d)$ by the encoder, which can be described by

$$\boldsymbol{\Psi} = L\_\mathrm{ReLU}(\boldsymbol{W}_e \boldsymbol{X} + \boldsymbol{b}_e) \tag{3}$$

where $\boldsymbol{W}_e$ and $\boldsymbol{b}_e$ represent the weights and bias in terms of the encoder, respectively. The decoder retrieves $\boldsymbol{X}'$ from $\boldsymbol{\Psi}$ by

$$\boldsymbol{X}' = L\_\mathrm{ReLU}(\boldsymbol{W}_d \boldsymbol{\Psi} + \boldsymbol{b}_d) \tag{4}$$

where $\boldsymbol{W}_d$ and $\boldsymbol{b}_d$ denote the weights and bias in terms of the decoder, respectively.

The gradient descent is utilized into the pretraining stage, focusing on minimizing the loss, which can be seen in (5). The loss measures the deviation between $\boldsymbol{X}$ and $\boldsymbol{X}'$.

The parameters of the network are updated via backpropagation with the target of minimizing loss, and the quality information of the input vector is recorded into the latent feature.

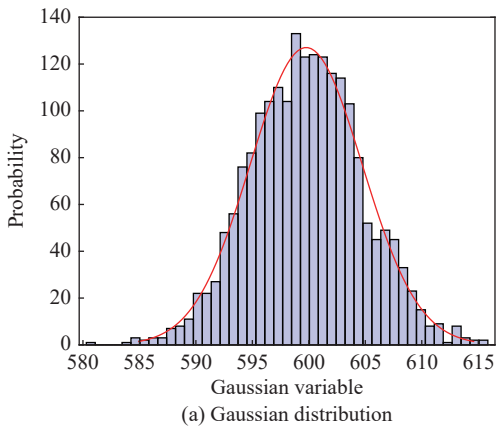
$$\text{Loss} = \|\boldsymbol{X} - \boldsymbol{X}'\|^2 = \|\boldsymbol{X} - f[g(\boldsymbol{x})]\|^2 = \|\boldsymbol{X} - L\_\text{ReLU}(\boldsymbol{W}_d(L\_\text{ReLU}(\boldsymbol{W}_e\boldsymbol{X} + \boldsymbol{b}_e)) + \boldsymbol{b}_d)\|^2 \quad (5)$$

By restricting the latent space to a lower dimensionality than the input space ($d < n$), the autoencoder parametrizes a low-dimensional non-linear manifold that captures the data structure. After pretraining, the encoded feature $\boldsymbol{\Psi}$ is fed into the RBFFAFLM model.

3.2 RBFFAFLM network

3.2.1 Original RBF

In this work, we obtain the dataset from the hot strip rolling line of a steel company, a typical complex industrial production environment. There are two types of distributions among these data, i.e., Gaussian distribution and non-Gaussian distribution, as shown in Fig. 1. Data imbalance results from data transformations, data missing, noises, etc. Most importantly, every record of data has too many features, difficult to represent by simple linear algorithms, which motivates us to make use of nonlinear machine learning algorithms, especially the RBF prediction model. RBF is a popular forward network, which has the following advantages. (i) The Gaussian kernel function is selected as the kernel function, which makes the model solve the problem of local optimal in the BP network and approximate any continuous function with arbitrary precision; (ii) RBF can minimize training error and confidence interval, and simplify the structure; (iii) RBF has concise training process and fast convergence. The learning speed and training speed are much faster than BP neural network. Therefore, we apply RBF to construct the new product quality prediction model.



(a) Gaussian distribution

(b) Non-Gaussian distribution

Fig. 1 Distributions of the dataset

As shown in Fig. 2, RBF consists of an input layer, a hidden layer, and an output layer. The output of RBF responds to the input information by a linear combination of nonlinear hidden functions.

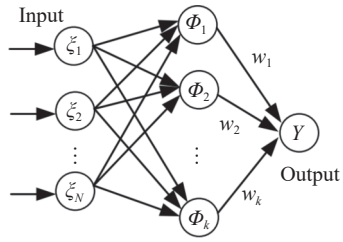


Fig. 2 Structure of RBF neural network

Its output $Y(t)$ can be calculated by

$$Y(t) = \sum_{q=1}^{Q} \boldsymbol{w}_q(t) Z_q(\boldsymbol{\Psi}(t)) \quad (6)$$

where $\boldsymbol{w}_q$ denotes the connection weight between the kth hidden neuron and the output neuron. Z_q is the output of the qth hidden neuron. $\boldsymbol{\Psi}=(\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \cdots, \boldsymbol{\xi}_{Ma})$ is the input vector with Ma input neurons. Q is the number of hidden neurons. The main relationship between input neurons and hidden neurons is often set as a Gaussian radial function.

$$Z_q(\boldsymbol{\Psi}(t)) = \mathrm{e}^{-(\|\boldsymbol{\Psi}(t) - \boldsymbol{o}_q\|^2 / \eta_q^2(t))} \quad (7)$$

where $\boldsymbol{o}_q$ and η_q represent the center vector and the radius of the qth hidden neuron respectively. $\|\boldsymbol{\Psi}(t) - \boldsymbol{o}_q\|$ is the Euclidean distance between $\boldsymbol{\Psi}(t)$ and $\boldsymbol{o}_q$.

3.2.2 Optimized RBFNN

Generally, the approximation ability of RBF is determined by the neuron centers, neuron widths, neuron number, and connection weights. Therefore, we will give a detailed description about how to determine the above parameters.

The center of each cluster is taken as the corresponding center c_i in neurons. The base width of the basis function is calculated by

$$\sigma_i = \alpha \min_j \|c_i - c_j\| \tag{8}$$

where α denotes the overlap factor.

The connection weights $\boldsymbol{W}$ can be calculated by

$$\boldsymbol{W} = (\boldsymbol{U}_R^{\mathrm{T}}\boldsymbol{U}_R)^{-1}\boldsymbol{U}_R Y, \tag{9}$$

$$\boldsymbol{U}_R = \begin{bmatrix} 1 & \varphi_{11} & \cdots & \varphi_{1p} \\ 1 & \varphi_{21} & \cdots & \varphi_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \varphi_{n1} & \cdots & \varphi_{np} \end{bmatrix}, \tag{10}$$

where Y represents the output of the model, and $\boldsymbol{U}_R$ denotes the matrix $n_s \times p$ of the basis function related to samples.

The number of neurons in the hidden layer can be determined by the empirical method, but the final calculation is very complicated when the input dataset is large. This is because the reasonable initial value K is first assumed to train the model, then the value of K will be regulated based on the training effect. Considering the calculation complexity and the execution efficiency, the fuzzy c-means (FCM) algorithm is adopted in this paper to optimize and determine the overlap factor α and the value of the number of neurons K, that is, $Q = K$.

(i) FCM

$\boldsymbol{G}=\{\boldsymbol{g}_1,\boldsymbol{g}_2,\cdots,\boldsymbol{g}_n\}$ denotes the dataset with n objects, and each object has d properties. FCM aims to divide the n objects into $c(2 \leqslant c \leqslant n)$ clusters, and the centers are defined as $V = \{v_1, v_2, \cdots, v_c\}$. To obtain the optimal clustering result, the objective function J_{mk} of FCM should be the smallest after continuous iteration.

$$J_{mk} = \sum_{i=1}^{n} u_{i,j}^{mk} \|g_i - v_j\|^2 \tag{11}$$

where mk is the fuzzy weighted index controlling the classification degree of the fuzzy matrix. $\boldsymbol{U} = [u_{ij}]_{c\times n}$ is the membership matrix, $u_{ij} \in [0,1]$ presents the degree to which the jth object belongs to the ith cluster. $\|g_i - v_j\|^2$ represents the Euclidean distance between the jth object and the ith cluster center.

To obtain the smallest objective function J_{mk}, the cluster centers V and membership matrix $\boldsymbol{U}$ should be calculated repeatedly. When the distance between the updated cluster center and the original cluster center is less than the given threshold ε, the iteration stops and we obtain cluster center v_j and membership matrix $\boldsymbol{U}$.

$$u_{ij} = \frac{1}{\sum_{k=1}^{c}\left(\dfrac{\|g_i - v_j\|}{\|g_i - v_k\|}\right)^{\frac{2}{mk-1}}}, \quad 1 \leqslant i \leqslant c; 1 \leqslant j \leqslant n \tag{12}$$

$$v_j = \frac{\sum_{i=1}^{n} u_{ij}^{mk} x_i}{\sum_{i=1}^{n} u_{ij}^{mk}}, \quad 1 \leqslant j \leqslant c \tag{13}$$

The FCM clustering algorithm obtains the optimal clustering result by iterative calculation of the clustering center and membership matrix. It is a search method based on gradient descent with fast convergence speed and strong local search ability.

(ii) FA

The basic idea of FA is that each firefly presents a solution to the objective function, and the brightness of the fireflies presents how well of the objective function solution. Fireflies with low brightness move to ones with high brightness and update their own brightness. The process of moving is the process of optimizing objective function. By continuous iteration of location and brightness, the optimal value of the objective function is obtained.

FA has two major advantages over other algorithms: automatical subdivision and the ability to deal with multimodality. The light intensity I varies according to the inverse-square law,

$$I_{\mathrm{i}} \propto \frac{1}{J(\mathrm{po}_i)}, \ 1 \leqslant i \leqslant n \tag{14}$$

where I_{i} and po_i represent the brightness and position of the ith firefly respectively. $J(\mathrm{po}_i)$ denotes the objective function of the ith firefly.

The attractiveness β of a firefly:

$$\beta = \beta_0 \mathrm{e}^{-\gamma r_{ij}^2} \tag{15}$$

where β_0 denotes the attractiveness at zero distance $r = 0$. γ represents light absorption coefficient. β and r_{ij} indicate the relative attractiveness and distance between the ith firefly and the jth firefly respectively.

The distance between two fireflies i and j at po_i and po_j respectively is defined by the Cartesian distance

$$r_{ij} = \|\mathrm{po}_i - \mathrm{po}_j\| = \sqrt{\sum_{k=1}^{d} (\mathrm{po}_{ik} - \mathrm{po}_{jk})^2} \tag{16}$$

where po_{ik} indicates the kth component of the spatial coordinate po_i of the ith firefly, and d denotes the dimension of input.

The movement of a firefly i attracted to another more attractive (brighter) firefly j is determined by

$$\mathrm{po}_i^{t+1} = \mathrm{po}_i^t + \beta_0 \mathrm{e}^{-\gamma r_{ij}^2}(\mathrm{po}_j^t - \mathrm{po}_i^t) + \alpha \boldsymbol{\varsigma}_i^t \quad (17)$$

where α is the scaling factor and $\boldsymbol{\varsigma}_i$ represents a vector of the random number drawn from a Gaussian distribution or uniform distribution.

(iii) FCM based on FA with LFM

In traditional FA, a random perturbation method is adopted in the process of fireflies' position update, which easily leads to local optimum. Thus the FCM based on FA with LFM is introduced, which can change the random mobile strategy in FA, balance the local and global search ability, and make up for the disadvantage of FCM. Also, it can reduce the iteration times greatly. In the process of a firefly location update, the LFM is introduced, which can improve the global optimization ability of the algorithm. The dynamic scale coefficient is utilized to limit the search range of LFM in different periods, which greatly accelerates the convergence speed of the algorithm. The location update formula of FA is redefined as follows:

$$\mathrm{po}_i^{k+1} = \mathrm{po}_i + \beta(\mathrm{po}_j - \mathrm{po}_i) + \alpha(t) \cdot \mathrm{sign}(\mathrm{rand}) \cdot \mathrm{levy}(\lambda), \quad (18)$$

$$\mathrm{levy}(\lambda) \sim |s|^{-\lambda}, \quad 1<\lambda<3, \quad (19)$$

$$s = \frac{\mu}{|v|^{1/\lambda}}, \quad 0<\lambda<2, \quad (20)$$

where levy(λ) and sign(rand) represent the stride of LFM and search direction respectively. s and λ denote the random stride and the index parameters. u and v are random parameters in the normal distribution. rand represents a random number uniformly distributed on $[0, 1]$:

$$\begin{cases} u \sim N(0, \delta_u^2) \\ v \sim N(0, 1) \end{cases}, \quad (21)$$

$$\mathrm{sign}(\mathrm{rand}) = \begin{cases} 1, & 1 \geqslant \mathrm{rand} \geqslant 0.5 \\ -1, & 0 \leqslant \mathrm{rand} < 0.5 \end{cases} \quad (22)$$

To decrease the iteration times, the scaling parameter $\alpha(t)$, which controls the search range of the Levy flight mechanism, is introduced.

$$\alpha(t) = \alpha_{\mathrm{init}} \cdot \exp\left(-\frac{\mathrm{it}}{I_{\mathrm{gbest}} - I_{\mathrm{present}}}\right) \quad (23)$$

where it represents iterations and α_{init} denotes initial scaling factor. I_{gbest} and I_{present} represent the brightness of the brightest firefly and the present firefly respectively.

In the early stage of the algorithm, Levy's flying range is relatively large, which is conducive to expanding the search range and finding the approximate global optimal solution. In the later stage, when the population tends to be stable, the searching range of Levy's flying is reduced to prevent the algorithm from oscillating around the optimal value, aiming to approach the optimal solution as soon as possible. Formally, Algorithm 3 summarizes the approach.

Algorithm 3 FCM optimized by FA with Levy

Input: the attractiveness β_0, light absorption coefficient γ, initial scaling factor α_{init}, the maximum iterations maxIter, current iteration k; the threshold value limi, the population size of fireflies N, fuzzy weighted index m

Output: optimal firefly position V

1: Initialize the position $V = (v_{i1}, v_{i2}, \cdots, v_{ic})$ of each firefly $\boldsymbol{G} = (\mathrm{g}_{i1}, \mathrm{g}_{i2}, \cdots, \mathrm{g}_{ic})$
2: Calculate the light intensity I of each firefly by (14)
3: Range the firefly clusters by light intensity and record the I_{best} and x_{best} of the brightest firefly
4: Update scaling factor α_{init} of each firefly by (23), and update the position of each firefly by (18)
5: Update the light intensity I of each firefly, and update the I_{best} and g_{best} of the brightest firefly
6: Take the position g_{best} of the brightest firefly as the initial clustering center
7: Update the membership matrix $\boldsymbol{U}$ by (12)
8: Update cluster center and update g_{best}
9: Calculate the objective function J_{mk} of FCM
10: **If** the difference between J_{mk} and initial cluster center is greater than the given threshold limi **then**
11: Return Step 1
12: **if** $k <$ maxIter **then**
13: $k = k+1$
14: Return Step 4
15: **else** stop training
16: Return g_{best}
17: **end if**

3.2.3 Whole model

In brief, the main modeling flowchart of RBFFALFM is illustrated in Fig. 3. First, the data pre-processing is conducted on dataset. According to the ordered association rules obtained by FP-Growth, the ODEM is applied to eliminate and supplement certain data in certain clusters, which contributes to solving the problem of data imbalance and model overfitting. Then, the dataset is divided into the training set and test set. The training set is used to train the new prediction model RBFFALFM. During the training process, the FA with LFM is used to determine the optimal number of clusters k and the overlap coefficient α in the RBF network. The number of clusters and the overlap coefficient determine the independent learning ability and anti-interference ability of the model. After obtaining the optimized RBF model, the testing set is applied to evaluate the effectiveness of the whole model.

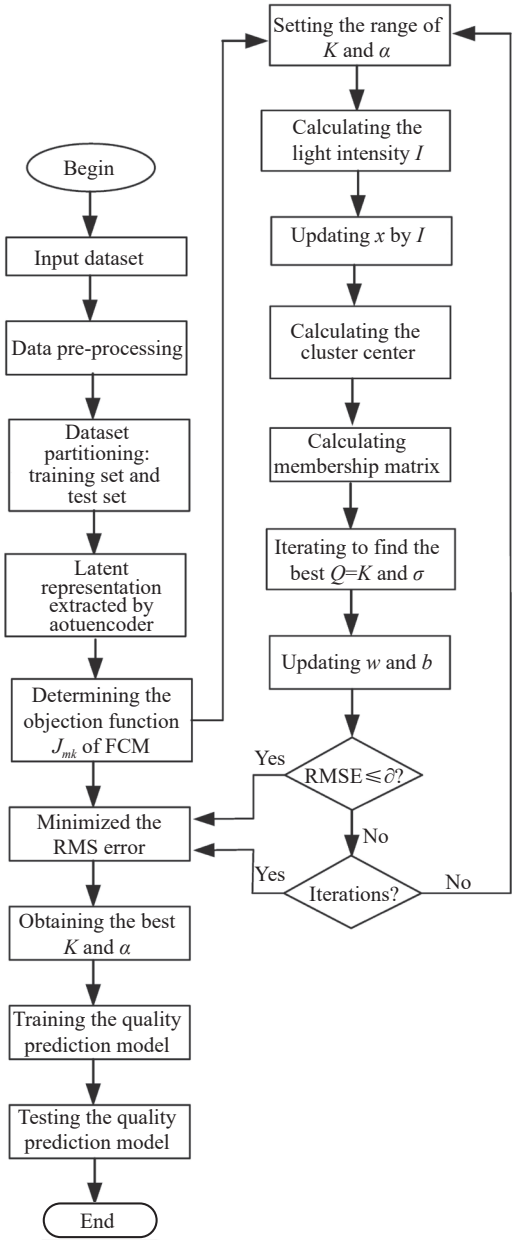


Fig. 3 Framework of the new product quality prediction model

4. Experiments and analysis

In this section, to verify the effectiveness and superiority of the new proposed method RBFFALFM, extensive experiments and analysis are conducted in detail.

4.1 Dataset and settings

We obtain the dataset from the hot strip rolling line of a steel company, which contains 2 000 labeled data and 3 580 unlabeled data. All experiments are conducted on TensorFlow, and the models are trained on an Nvidia 1 080 Ti GPU with Windows 10. Data equalization operation and normalization are implemented first. The root mean square error (RMSE) loss and the confusion matrix are taken as the performance evaluation indices to reflect model performance. Besides, the confusion matrix is an effective performance metric to evaluate the binary classification problem, where four basic metrics true positive (TP), false positive (FP), true negative (TN), and false negative (FN) are used to measure the output. Based on the above basic metrics, some other evaluation indices, Accuracy, Precision, Recall, F1-score, and ROC_AUC Score, are calculated by (24)−(30).

$$\text{Accuracy} = \frac{\text{TP}+\text{TN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}} \tag{24}$$

$$\text{Percision} = \frac{\text{TP}}{\text{TP}+\text{FP}} \tag{25}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP}+\text{FN}} \tag{26}$$

$$\text{F1-score} = \frac{2\times\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}} \tag{27}$$

$$\text{ROC\_AUC} = \frac{1}{2}\sum_{i=1}^{m-1}(\text{FPR}_{i+1}-\text{FPR}_i)(\text{TPR}_{i+1}-\text{TPR}_i) \tag{28}$$

where

$$\text{TPR}=\frac{\text{TP}}{\text{TP}+\text{FN}}, \tag{29}$$

$$\text{FPR}=\frac{\text{FP}}{\text{TN}+\text{FP}}. \tag{30}$$

4.2 Contrast experiments

In this section, to prove the effectiveness of the new model RBFFALFM, some experiments compared with several classic models such as BP, SVM, LeNet, the benchmark RBF, and random forest (RF), are conducted. The evaluation results are summarized in Table 1. Obviously, the new model obtains higher accuracy than other methods. Besides, it is clear that the training time of RBFFALFM is much shorter than other models, which reveals that the simple RBF benchmark does make the product quality prediction task more efficient.

Table 1 Comparison of RBFFALFM with other models

Method	Accuracy/ %	Precision/ %	Recall/ %	F1-score/ %	ROC_ AUC/ %	Time/ s
BP	74.65	89.10	73.20	80.37	86.32	13
SVM	79.36	88.32	74.19	80.64	85.71	10
LeNet	85.31	90.04	81.74	85.69	89.68	18
RBF	84.65	90.19	82.65	86.25	89.32	6
RF	88.91	92.34	90.54	91.43	93.44	7
RBFFALFM	91.89	96.12	92.30	94.17	96.23	8

To show that the performance of this model is better than others, a new performance index is defined by

$$\text{Improvement}=\frac{\text{RBFFALFM}_{\text{metric}}-\text{other model}_{\text{metric}}}{\text{RBFFALFM}_{\text{metric}}}\times 100\%. \tag{31}$$

The comparison results with other methods are illustrated in Fig. 4. It is clear that the improvements across all metrics are positive. In addition, the contrast results between the real values and the predicted values are illustrated in Fig. 5, where the blue line and red line denote real value and predicted value respectively. From Fig. 5, we can see that the fitting effect of RBFFALFM is better than other models. Empirically, RBFFALFM has substantial strengths over other models.

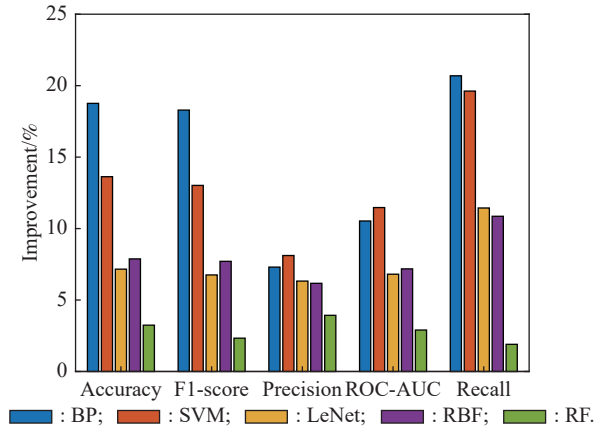


Fig. 4 Improvements of RBFFALFM compared with other product quality prediction models

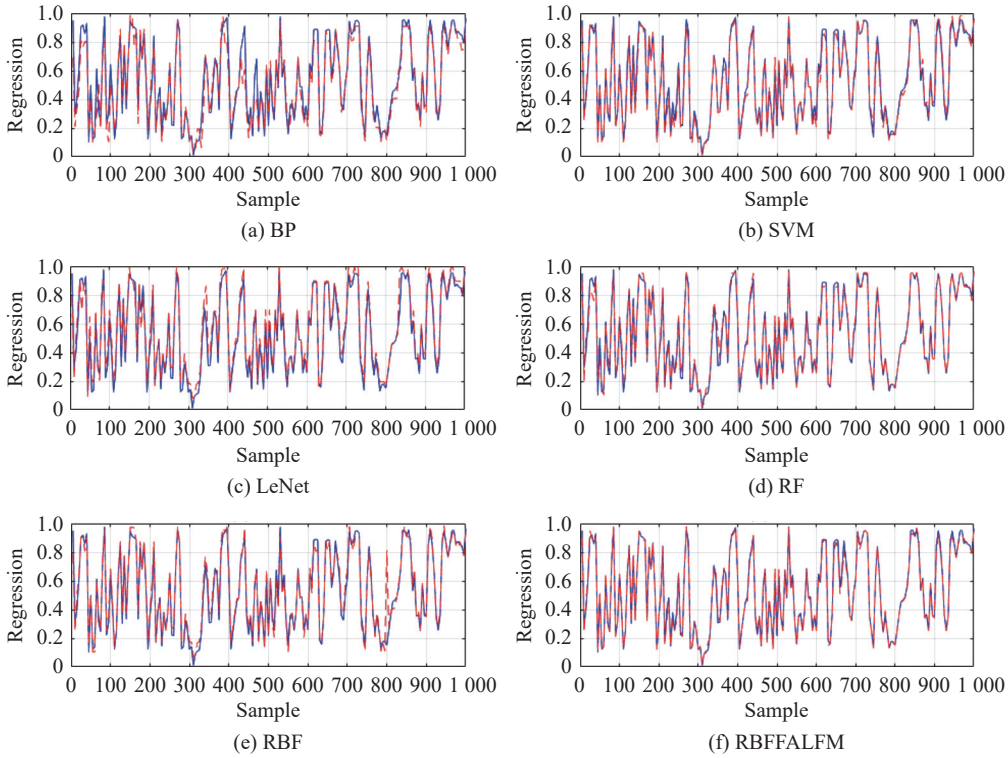


Fig. 5 Comparison results of various product quality prediction models

4.3 Ablation studies

4.3.1 Ablation studies on different data pre-processing methods

There are many classic methods for dealing with the problem that raw data from the production process is badly shaped, such as linear discriminant analysis (LDA), multi-objective genetic algorithm (MOGAImp) [36], the deterministic version of synthetic minority over-sampling technique (SMOTE-D) [37], multi-layer perceptron-based undersampling technique (MLPUS) [38]. The comparison experimental results of the new model using different data pre-processing methods are summarized in Table 2.

Table 2 Comparison of RBFFALFM with different data pre-processing methods %

Method	Accuracy	Precision	Recall	F1-score	ROC_AUC
LDA + RBFFALFM	86.48	90.16	87.90	89.01	91.97

Continued

Method	Accuracy	Precision	Recall	F1-score	ROC_AUC
MOGAImp + RBFFALFM	87.21	91.62	88.59	90.08	92.71
SMOTE-D + RBFFALFM	88.41	93.08	89.35	91.38	92.79
MLPUS + RBFFALFM	89.45	93.17	90.82	91.98	94.47
ODEM + RBFFALFM	91.89	96.12	92.30	94.17	96.23

LDA + RBFFALFM: The LDA is used to preprocess the raw dataset before applying the RBFFALFM to predict product quality.

MOGAImp+RBFFALFM: The MOGAImp is used to preprocess the raw dataset before applying the RBFFALFM to predict product quality.

SMOTE-D+RBFFALFM: The SMOTE-D is used to preprocess the raw dataset before utilizing the RBFFALFM to predict product quality.

MLPUS+RBFFALFM: The MLPUS is applied to preprocess raw dataset before using the RBFFALFM to predict product quality.

The prediction accuracy of the RBFBALFM with the new data pre-processing method ODEM is higher than other combinations, this is because other data pre-processing methods do not take into consideration the distribution of the unbalanced dataset. Besides, the RMSE loss of the RBFBALFM with different data pre-processing methods is shown in Fig. 6. It is obvious that along with the increase of iteration, the RMSE loss gradually decreases. Especially, the RMSE loss of the RBFBALFM with ODEM is lower than the model with other data pre-processing methods.

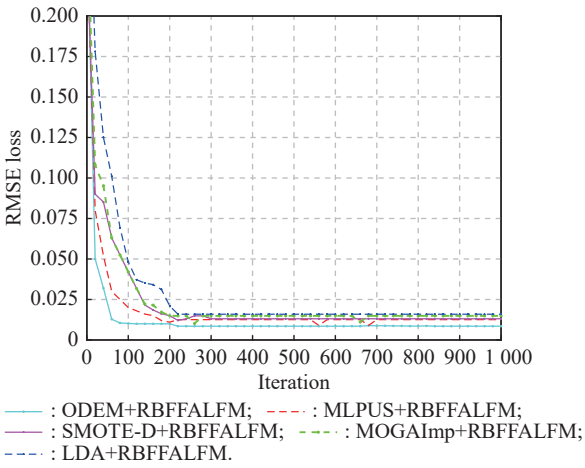


Fig. 6 RMSE loss of RBFFALFM with different data pre-processing methods

In addition, the improvement comparison of different data pre-processing methods used in the RBFFALFM is illustrated in Fig. 7. By comparison, it can be concluded that the new established data pre-processing method ODEM successfully deals with the problem of uneven data distribution and improves the forecasting accuracy to some extent.

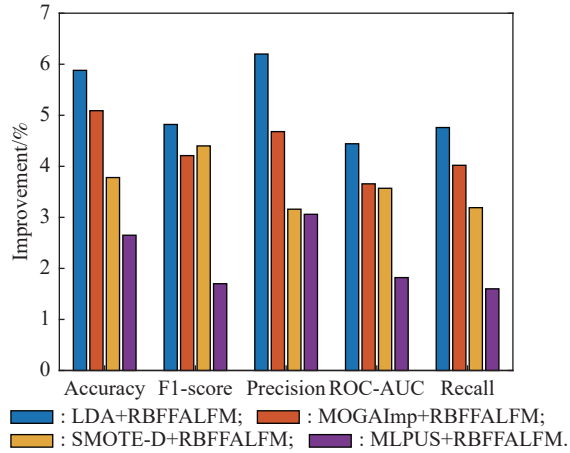


Fig. 7 Improvements of RBFFALFM compared with RBF neural network combining with other data pre-processing methods

4.3.2 Ablation studies on different swarm intelligent optimization methods

In this section, to validate the effectiveness of the FA with LFM in optimizing the RBF model, we conduct several comparison experiments. In these experiments, the RBF model is optimized by different swarm intelligence optimization methods. The results are presented in Table 3.

Table 3 Comparison of RBF combined with different optimized methods

%

Method	Accuracy	Precision	Recall	F1-score	ROC_AUC
RBF	84.65	89.19	83.32	86.15	88.99
RBF + SSO	87.63	92.15	86.14	89.04	94.71
RBF + GA	88.31	94.04	88.26	91.05	92.41
RBF + PSO	88.96	92.19	89.34	90.74	94.14
RBF + FA	89.65	95.04	90.10	92.50	94.62
RBFFALFM	91.89	96.12	92.30	94.17	96.23

RBF+FA: The FA [14] is used to optimize the RBF network.

RBF+GA: The genetic algorithm (GA) [39] is applied to optimize the RBF network.

RBF+PSO: The PSO [40] is utilized to optimize the RBF network.

RBF+SSO: The SSO [30] is applied to optimize the RBF network.

It is evident that the RBF model optimized by FA with LFM has relatively high prediction accuracy. Also, we present the improvements comparison results in Fig. 8, and the RMSE loss in Fig. 9. From the comparison results, it is easy to conclude that FA with LFM contributes more to improving prediction performance.

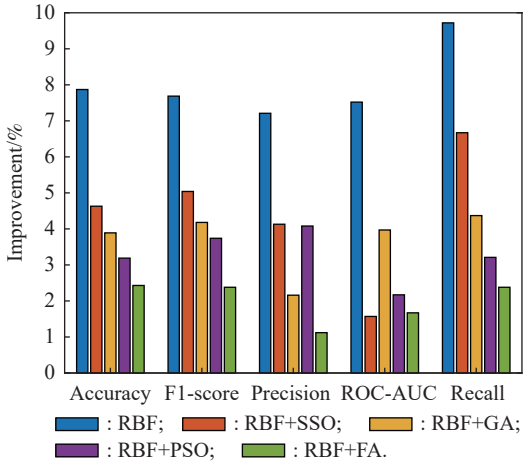


Fig. 8 Improvements of RBFFALFM compared with different optimized methods

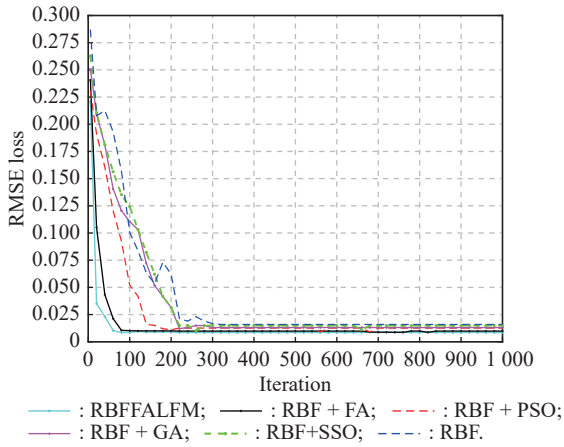


Fig. 9 RMSE loss of RBF neural network with different optimized methods on the test set

5. Conclusions

This paper proposes a novel model for product quality prediction, which can be applied into complex industrial production process. First, an optimized data equalization method ODEM is used to pre-process the dataset, which removes the bad influence of unevenly distributed data on the training product quality prediction model. This operation also ensures the correctness of the association rules, further providing the basis for product quality prediction tasks. To improve the accuracy of product quality prediction and reduce model complexity, a novel product quality prediction model RBFFALFM is proposed, and the FA with LFM is introduced to optimize the parameters in RBF. The new model can extract useful information under complicated production conditions. Extensive experimental results indicate that the new product quality prediction model based on RBF optimized by FA with LFM has higher accuracy and better robustness than other traditional models. In a word, this new method not only can predict product quality in advance, but also can help technicians to choose more reasonable process parameters, further reducing the trial production times and enterprise cost.

Although this new product quality prediction model RBFFALFM achieves excellent performance in a complex industrial production environment, it does not take into consideration a small sample environment, especially, multivariate time series data. This disadvantage limits its application in specific industrial scenarios. In future work, we will make greater efforts to further overcome the above problems by introducing reinforcement learning mechanism and semi-supervised learning mechanism.

References

[1] MAJERNIK M, DANESHJO N, MALEGA P, et al. Sustainable development of the intelligent industry from industry 4.0 to industry 5.0. Advances in Science and Technology-Research Journal, 2022, 16(2): 12–18.

[2] CHAUHAN R, YAFI E. Applicability of classifier to discovery knowledge for future prediction modelling. Journal of Ambient Intelligence and Humanized Computing, 2022. DOI: 10.1007/s12652-022-03694-3.

[3] ZHANG Q C, YANG L T, CHEN Z K, et al. PPHOPCM: privacy-preserving high-order possibilistic c-means algorithm for big data clustering with cloud computing. IEEE Trans. on Big Data, 2022, 8(1): 25–34.

[4] DU X P, MA X L, TANG S W, et al. A fast algorithm for mining of association rules. Computer Engineering and Application, 2002, 38(11): 1–4.

[5] PACHECO C, ERNST M D. Eclat: automatic generation and classification of test inputs. Lecture Notes in Computer Science, 2005, 3586: 504–527.

[6] HAN J W, PEI J, YIN Y W. Mining frequent patterns without candidate generation. ACM Sigmod Record, 2000, 29(2): 1–12.

[7] MAO J, CHEN D J, ZHANG L Q. Mechanical assembly quality prediction method based on state space model. International Journal of Advanced Manufacturing Technology, 2016, 86(1/4): 107–116.

[8] YEH C H, FAN Y C, PENG W C. Interpretable multi-task learning for product quality prediction with attention mechanism. Proc. of the IEEE 35th International Conference on

Data Engineering, 2019: 1910−1921.
[9] YAO L, SHAO W M, GE Z Q. Hierarchical quality monitoring for large-scale industrial plants with big process data. IEEE Trans. on Neural Networks and Learning Systems, 2021, 32(8): 3330–3341.
[10] YANG F, LI X, BAI J J, et al. Nonlinear process quality prediction using wavelet denoising OSC-SVM-PLS. Industrial & Engineering Chemistry Research, 2020, 59(13): 6021–6032.
[11] REN L, MENG Z H, WANG X K, et al. A data-driven approach of product quality prediction for complex production systems. IEEE Trans. on Industrial Informatics, 2021, 17(9): 6457–6465.
[12] HU J Y, QIAN X F, PEI J, et al. A novel quality prediction method based on feature selection considering high dimensional product quality data. Journal of Industrial and Management Optimization, 2021. DOI: 10.3934/jimo.2021099.
[13] THOMSPSON R. Radical basis function modelling and prediction of time series. ACS Nano, 1996, 6(10): 8857–8867.
[14] YANG X S. Firefly algorithms for multimodal optimization. Lecture Notes in Computer Science, 2009, 5792: 169−178.
[15] TANG Y T, DUAN Y Q, WEN J. Study on nuclear class piping hanger and support root intelligent selection based on SOM clustering algorithm. Nuclear Power Engineering, 2020, 41(3): 193–196.
[16] ABD EL-MONEIM S, EL-RABAIE ESM, NASSAR M A, et al. Speaker recognition based on pre-processing approaches. International Journal of Speech Technology, 2020, 23(2): 435−442.
[17] YAN Y J, WU G X, DONG Y, et al. An improved MSR-based data-driven detection method using smoothing pre-processing. IEEE Signal Processing Letters, 2021, 28: 444–448.
[18] WANG S H, CHOU T I, CHIU S W, et al. Using a hybrid deep neural network for gas classification. IEEE Sensors Journal, 2021, 21(5): 6401–6407.
[19] NICKOLAS S, SHOBHA K. Efficient pre-processing techniques for improving classifiers performance. Journal of Web Engineering, 2022, 21(2): 203–228.
[20] BEZIN Y, SAMBO B, COSTA J N. Challenges and methodology for pre-processing measured and new rail profiles to efficiently simulate wheel-rail interaction in switches and crossings. Vehicle System Dynamics, 2022. DOI: 10.1080/00423114.2021.2014897.
[21] BARALDI A, ALPAYDIN E. Constructive feedforward ART clustering networks - Part II. IEEE Trans. on Neural Networks, 2002, 13(3): 662−677.
[22] HOU J, QIAO J H, HAN X L. Energy-saving clustering routing protocol for wireless sensor networks using fuzzy inference. IEEE Sensors Journal, 2022, 22(3): 2845–2857.
[23] MZAO L, YUAN C A, QIN X. Wireless sensor target tracking algorithm based on improved adaptive clustering. Fire Control & Command Control, 2022, 47(1): 57–64. (in Chinese)
[24] KOU L, DING S S, RAO Y, et al. A lightweight intrusion detection model for 5G-enabled industrial internet. Mobile Networks & Applications, 2022. DOI: 10.1007/s11036-021-01891-6.
[25] HOODA H, VERMA O P. Fuzzy clustering using gravitational search algorithm for brain image segmentation. Multimedia Tools and Applications, 2022. DOI: 10.1007/s11042-022-12336-x.
[26] SHAIKH P W, EL-ABD M, GAO K Z. A review on swarm intelligence and evolutionary algorithms for solving the traffic signal control problem. IEEE Trans. on Intelligent Transportation Systems, 2022, 23(1): 48–63.
[27] KUMAZAWA T, TAKIMOTO M, KAMBAYASHI Y. Exploration strategies for balancing efficiency and comprehensibility in model checking with ant colony optimization. Journal of Information and Telecommunication, 2022. DOI: 10.1080/24751839.2022.2047470.
[28] XU X L, YAN F. Random walk autonomous groups of particles for particle swarm optimization. Journal of Intelligent & Fuzzy Systems, 2022, 42(3): 1519–1545.
[29] YUAN M F, KAN X, CHI C H, et al. An adaptive simulated annealing and artificial fish swarm algorithm for the optimization of multi-depot express delivery vehicle routing. Intelligent Data Analysis, 2022, 26(1): 239–256.
[30] NESHAT M, SEPIDNAM G, SARGOLZAEI M. Swallow swarm optimization algorithm: a new method to optimization. Neural Computing & Applications, 2013, 23(2): 429–454.
[31] RAHUL K, BANYAL R K. Firefly algorithm: an optimization solution in big data processing for the healthcare and engineering sector. International Journal of Speech Technology, 2021, 24(3): 581–592.
[32] WU Z B, WU L J, XU L. Bilateral matching optimization of industrial chain financial products based on 0-1 knapsack strategy and improved discrete particle swarm optimization algorithm. Computer Integrated Manufacturing Systems, 2019, 25(12): 3279–3288.
[33] BHAGAVATHI H, RATHINAVELAYATHAM S, SHANMUGAIAH K, et al. Improved beetle swarm optimization algorithm for energy efficient virtual machine consolidation on cloud environment. Concurrency and Computation-Practice & Experience, 2022. DOI: 10.1002/cpe.6828.
[34] DUTTA T, DEY S, BHATTACHARYYA S, et al. Quantum fractional order Darwinian particle swarm optimization for hyperspectral multi-level image thresholding. Applied Soft Computing, 2021. DOI: 10.1016/j.asoc.2021.107976.
[35] MUKHERJEE S, ASNANI H, LIN E, et al. ClusterGAN: latent space clustering in generative adversarial networks. Proc. of the AAAI Conference on Artificial Intelligence, 2019, 33: 4610–4617.
[36] LOBATO F, SALES C, ARAUJO I, et al. Multi-objective genetic algorithm for missing data imputation. Pattern Recognition Letters, 2015, 68: 126–131.
[37] TORRES F R, CARRASCO-OCHOA J A, MARTINEZ-TRINIDAD J F, et al. SMOTE-D a deterministic version of SMOTE. Lecture Notes in Computer Science, 2016, 9703: 177–188.
[38] BABAR V, ADE R. MLP-based undersampling technique for imbalanced learning. Proc. of the International Conference on Automatic Control and Dynamic Optimization Techniques, 2016: 142−147.
[39] WHITLEY D. A genetic algorithm tutorial. Statistics and Computing, 1994, 4(2): 65–85.
[40] VENTER G, JAROSLAW S S. Particle swarm optimization. AIAA Journal, 2003, 41(8): 129–132.

Biographies

HAN Huihui was born in 1993. She received her M.S. degree from Hefei University of Technology in 2020. She is currently pursuing her Ph.D. degree in control science and engineering at Tongji University, Shanghai, China. From 2020 until now, she is working with the team of the CIMS Research Center at Tongji University, and her research interests are deep learning, data mining, and knowledge graph.
E-mail: hanhuihui@tongji,edu.cn

WANG Jian was born in 1962. He is the director of the CIMS Research Center of Tongji University, and has been long engaged in research and development in the field of automatic control. In recent years, he has been mainly engaged in enterprise informatization, CIMS, business process management, workflow technology, energy and transportation systems, networked manufacturing and system integration. His research interests are deep learning, data mining, and knowledge graph.
E-mail: jwang@tongji.edu.cn

CHEN Sen was born in 1978. He received his M.S. degree from Donghua University, Shanghai. He is currently pursuing his Ph.D. degree in control science and engineering at Tongji University, Shanghai, China. From 2019 until now, he is working with the team of the CIMS Research Center at Tongji University. His research interests are deep learning, data mining, and knowledge graph.
E-mail: 1910067@tongji.edu.cn

YAN Manting was born in 1995. She received her M.S. degree from Shenyang University of Technology, Shenyang, in 2020. She is currently pursuing her Ph.D. degree in electronic information at Tongji University, Shanghai, China. Her research interests are deep learning, data mining, and knowledge graph.
E-mail: 2011618@tongji.edu.cn