Introduction

In recent years, reducing production costs has always been one of the important goals for the survival and development of enterprises. For textile enterprises, the quality of the yarn quality forecast determines the level of production costs. Therefore, yarn quality prediction occupies an extremely important position in textile enterprises, and it is also one of the hot topics that experts and scholars have been paying attention to.

For the evaluation of the effect of yarn quality prediction, many scholars have always used strength as the index for evaluating yarn quality, and have done a lot of research on strength. Among them, Cheng and Adams first tried to use a neural network to solve the problem of inaccurate force prediction in yarn quality prediction. Subsequently, in the period of the rapid development of neural networks, Furferi and Gelli proposed a model based on feedforward back propagation (BP) artificial neural network to predict yarn strength. Mwasiagi proposed a DE-ELM algorithm combining differential evolution (DE) algorithm and Extreme Learning Machine (ELM) to predict the strength of spun yarn; Zhenlong et al. combined convolutional neural network (CNN) and BP neural network, and proposed a CNN-BP neural network to predict yarn strength; Hu et al. considers the influence of time on yarn quality and strength, and proposes a yarn strength and quality prediction model based on artificial recurrent neural network (RNN).

However, the index for judging the yarn quality is not only the strength index, the unevenness of the yarn is also one of the extremely important indexes. The uniformity of the yarn determines the quality of the textile to a large extent, and the influence on the next process and the cloth surface is reflected in the following aspects: the first is to determine the end breakage of the next process, which
affects the production efficiency. The second is to determine the difference in cloth surface, shading, color gradation. The third is to determine the mechanical and physical properties of the cloth surface.

The prediction problem of yarn unevenness is a nonlinear problem. A good solution to nonlinear problems is to use neural network to predict them, and many scholars have studied it. Üreyen and Gürkan proposed the use of artificial neural network (ANN) to predict the unevenness of ring-spun yarn and compared it with the linear regression model. The results showed that: the prediction effect of using ANN to predict the unevenness of ring-spun yarn is better. Üte and Kadoğlu used a multiple linear regression method to establish a model for estimating the uniformity of Siro spin yarn from cotton fiber characteristics. Aiming at the uniformity of blended yarns, Malik et al. also uses ANN to predict the unevenness of yarn and compares it with multiple linear regression. The results show that the use of ANN is effective in predicting the unevenness of yarn. Wu et al. aimed at the difficulty of optimizing weights and thresholds in neural network, and proposed a method of using the Mind Evolutionary Algorithm (MEA) to optimize the weights and thresholds of the BP neural network, and then predict the unevenness of the yarn. Ghammi et al. proposed a fuzzy artificial neural network prediction model based on expert experience on the prediction of yarn quality. This model combines ANN with fuzzy experts. The results show that this method has high accuracy. Singh et al. studied several key production process parameters that may affect the unevenness of the yarn, including three process parameters: spring stiffness, conveying speed, and coil position. The results show that: coil position and stiffness of the spring have a great influence on the unevenness.

With the advancement of technology, people are facing more and more complex problems. The use of neural network alone can’t meet people’s requirements, so deep neural network came into being. However, deep neural network has its limitations: long training time, complex structure, numerous parameters, and it is difficult to find the optimal parameters. Because of the limitations of deep neural network, some scholars have proposed an alternative structure of deep neural network: Broad Learning System (BLS). The predecessor of the Broad Learning System is the functional link artificial neural network (FLANN), which was first proposed by Pao, including a single-layer neural network. Subsequently, Pao and Takefuji established a high-order neural network without hidden units. Without hidden units, there is no need to train and optimize the hidden units, which greatly reduces the running time of the network. Patra et al. proposed the use of FLANN to identify nonlinear dynamic systems for nonlinear dynamic systems. Mahapatra et al. used different polynomial functions to link artificial neural network to predict and classify Indian stocks and proposes four polynomial basis functions in the article: Chebyshev, Laguerre, Legendre, Power, which used different polynomials to enhance input to predict stocks.

Zhang and Suganthan introduced the random vector function link neural network (RVFLNN) into the image tracking field and combined it with the CNN to construct a new network convolutional random vector function link network (CRVFLNN). The new network is easy to train, and the training speed is faster, and there is no need to adjust the hyperparameters, so it can solve the problems in image tracking. Sahoo and Chakraverty proposed the use of FLANN to predict the structural response of high-rise buildings due to seismic loads and compare them with multi-layer neural network. Experimental verification is carried out under two-story, five-story, and ten-story buildings, and the results show that using FLANN not only runs fast but also has high accuracy. Zhang et al. proposed the use of RVFLNN to predict the temperature of subway stations, which met the requirements of the rapid prediction of subway stations. Sheng et al. aimed at the limitations of the previous RVFLNN: only training on a specific training set, and proposed a sliding time window method based on RVFLNN for dynamic prediction of electronic equipment cabinets. Chen and Liu designed BLS based on the idea of RVFLNN for the problem that deep architecture and learning require extremely time-consuming training processes. There is no hidden layer inside the BLS. The feature nodes and enhancement nodes are all obtained by changing the input data through a random function. The weight calculation of the input and output only needs to do an inversion operation. At the same time, increasing the data of the enhanced node can increase the complexity of the model and meet the complexity requirements of the model. Yang et al. used BLS to predict and analyze the throughput of port containers, and the results showed that compared with traditional time series-based models, using BLS has higher accuracy. Based on the previously proposed BLS, Chen et al. carried out a mathematical proof of its universal approximation properties. It can be proved that BLS is a nonlinear function approximator, and several variants of BLS are proposed at the same time.

Based on the research of the above scholars, it can be known that many scholars generally believe that the problem of yarn unevenness prediction is a nonlinear problem, and most of them use neural network to predict it. BLS is also a nonlinear function approximator, which has the advantages of fast speed and high accuracy. Similarly, the current BLS has not proven its universal advantage in all data sets. In response to this problem, this article first uses a multi-layer BP neural network to predict the unevenness of the yarn and then uses BLS to predict the unevenness of the yarn. When it is found that BLS has a poor prediction effect in the current data set, a multilayer neural network based on the idea of BLS is proposed.

The main contributions of this paper can be summarized as:

(i) A feature selection method combining expert experience and Pearson coefficient is proposed;
Data processing and feature selection

Data cleaning

There may be outliers in the data from the factory, and there may be noise between the data. This paper uses the method of expert experience to clean the data. The specific operation process is:

(i) Obtain experimental data from the factory database;
(ii) Eliminate noise data through expert experience. Since the data generated when the device is first started is inaccurate and contains serious noise, it is necessary to delete the noise data through expert experience to ensure the accuracy of the data;
(iii) Supplement or delete missing values in the data. The number of missing parameters in each data is determined in turn. If the number of missing parameters is greater than M, the data is judged to be unsupportable and is deleted. Otherwise, according to the idea of the nearest neighbor, we find out the K nearest neighbors of this data except for the missing parameters, and average them to supplement the missing values in the data;
(iv) Determine abnormal data through expert experience. Obtain the upper and lower limits of each parameter in the data through expert experience, and then judge whether the data is abnormal by judging each parameter to see if it exceeds the limit. If the data is abnormal, delete this data, otherwise, it is normal data;
(v) Normalize the data to eliminate the influence of dimensions.

Data normalization

In order to solve the inconsistency of each parameter unit of the data and to facilitate data processing, the normalization process is performed. The normalization processing method of data is:

\[ x^* = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \]  

(1)

Among them, \( x^* \) represents the normalized data, \( x \) represents the data before normalization, \( x_{\min} \) represents the minimum value of parameter \( x \), and \( x_{\max} \) represents the maximum value of parameter \( x \).

Feature selection based on statistics-Pearson correlation analysis

Pearson’s correlation coefficient is a commonly used method to statistically investigate the degree of correlation between two things, mainly used to explore the linear correlation between two independent continuous numeric variables. The Pearson correlation coefficient between vector \( X \) and \( Y \) can be calculated by the formula (2) to determine the degree of correlation between \( X \) and \( Y \):

\[ r(X,Y) = \frac{\text{cov}(X,Y)}{\sigma(X)\sigma(Y)} \]  

(2)

Judge the positive and negative correlation between \( X \) and \( Y \) according to the sign of \( r \), that is, when \( r > 0 \), it means that \( X \) and \( Y \) are positively correlated; when \( r < 0 \), it means that \( X \) and \( Y \) are negatively correlated. Judge the strength of the relationship between \( X \) and \( Y \) according to the value of \( r \). When \( |r| > 0.8 \), it means that \( X \) and \( Y \) have a strong linear correlation; when \( |r| < 0.2 \), it means that the linear correlation between \( X \) and \( Y \) is relatively weak; when \( |r| = 0 \), it means that there is no linear correlation between \( X \) and \( Y \); when \( |r| = 1 \), it means that \( X \) and \( Y \) are completely linearly correlated.

Feature selection based on expert experience

In the case of a small number of data samples, the expert experience will have a great impact on the result of feature selection.

The weight \( W = (w_1, w_2, ..., w_n) \) is the influence coefficient of the input parameter on the output parameter (unevenness) (\( w_i \) represents the influence coefficient of the first parameter on the unevenness, \( n \) means there are \( n \) total parameters), and the weight is the influence value of the evaluation given by multiple experts. The frequency statistics method is used to determine the weight of the input sample data:

(i) For each input parameter \( i \), find the maximum value \( M_i \) and the minimum value \( m_i \) among the weights \( w_j \) given by \( k \) experts and calculated by formula (3), (4):

\[ M_i = \max\{w_j\} i = 1,2,...,n; j = 1,2,...,k \]  

(3)

\[ m_i = \min\{w_j\} i = 1,2,...,n; j = 1,2,...,k \]  

(4)

Among them, \( w_j \) represents the influence factor of the \( i \)-th parameter on the unevenness given by the \( j \)-th expert.

(ii) For each input parameter \( i \), select an appropriate positive integer \( p \), and divide the weight \( w_j \) into ..
groups from small to large, and the group distance \( S_{j} \) is calculated by formula (5):

\[
S_{j} = \frac{M_{j} - m_{j}}{p}
\]  

(5)

(iii) Calculate the frequencies that fall within each set of weights;
(iv) Take the group median of the group where the maximum frequency is located as the weight of the input parameter \( i \);
(v) According to the above steps, the influence coefficient of each input parameter given by the expert can be obtained, and the weight of each input parameter can be sorted.

**Broad multilayer neural network**

The Broad Multilayer Neural Network (BMNN) introduces the concept of feature nodes and enhanced nodes in BLS based on traditional multi-layer neural network. The feature nodes are used to extract the characteristics of the input data, and the enhancement nodes reflect the state of the input data and collectively serve as the first input layer. The weight between the enhancement nodes and the back propagation algorithm, and then the corresponding relationship is obtained.

**Feature extraction**

The multi-layer neural network based on enhanced nodes does not directly use the original data as the input layer, but first performs some transformations on the data \( X \), which is equivalent to feature extraction, and uses the changed feature \( Z \) as the input layer of the original RVFLNN. As shown in formula (6).

\[
Z = \Phi(XW_{f} + \beta_{f})
\]  

(6)

Among them, \( Z \) represents the feature matrix \((Z = [Z_{1}, Z_{2}, ..., Z_{z}], \text{aisthenumberoffeaturenodes})\), \( W_{f} \) represents the weight matrix of the feature nodes, and \( \beta_{f} \) represents the bias matrix of the feature nodes. \( \Phi \) represents the mapping function.

**Feature enhancement**

Feature enhancement refers to the enhancement matrix transformation of the feature layer. Among them, the enhancement nodes are affected by all the feature nodes, reflects a certain state of the feature nodes, and is obtained by linear mapping and activation function transformation on the feature nodes. As shown in formula (7).

\[
H = \xi([Z_{1}, Z_{2}, ..., Z_{z}]W_{h} + \beta_{h})
\]  

(7)

Among them, \( H \) represents the enhancement matrix \((H = [H_{1}, H_{2}, ..., H_{h}], \text{bisthenumberofenhancementnodes})\), \( W_{h} \) represents the weight matrix of the enhancement nodes, and \( \beta_{h} \) represents the bias matrix of the enhancement nodes. \( \xi \) represents the activation function.

**Input layer**

BMNN is developed based on the Random Vector Function Linked Neural Network (RVFLNN), where RVFLNN is shown in Figure 1.

In RVFLNN, the input nodes are used as a layer, and the enhancement nodes are used as a layer. The structure of Figure 1 is deformed, the input nodes and the enhanced nodes are lined up and regarded as a whole, and a new simplified version of the network structure is obtained, as shown in Figure 2.

The input layer of BMNN is the same as the input layer of RVFLNN’s new simplified version of the network structure. The feature layer and enhancement layer are lined up through structural transformation, and they are regarded as one as the input layer of BMNN. Its specific structure is shown in Figure 3.

As shown in Figure 3, the input layer is a combination of the feature layer and the enhancement layer, and its specific representation is shown in formula (8).

\[
A = [Z \mid H]
\]  

(8)

Among them, \( A \) is the input matrix of the input layer, \( Z \) is the matrix obtained by feature extraction, \( H \) is the matrix of feature enhancement, and \( \mid \) represents the combination of two matrices.

**Hidden layer**

The hidden layer of BMNN performs a weighted summation of the output values of the neurons in the previous layer, performs nonlinear processing with the activation function, and outputs it as the input value of the latter layer. The output values of the \( t \)-th hidden layer are calculated as shown below.

\[
Y_{i} = \begin{bmatrix}
    y_{1}^i \\
    y_{2}^i \\
    \vdots \\
    y_{x}^i
\end{bmatrix} = F(\begin{bmatrix}
    w_{11} & w_{12} & \cdots & w_{1k} \\
    w_{21} & w_{22} & \cdots & w_{2k} \\
    \vdots & \vdots & \ddots & \vdots \\
    w_{x1} & w_{x2} & \cdots & w_{xk}
\end{bmatrix} \begin{bmatrix}
    y_{1}^{i-1} \\
    y_{2}^{i-1} \\
    \vdots \\
    y_{x}^{i-1}
\end{bmatrix} \begin{bmatrix}
    b_{1}^i \\
    b_{2}^i \\
    \vdots \\
    b_{k}^i
\end{bmatrix})
\]  

(9a)

\[
= F(W_{i}Y_{i-1} + B_{i})(t = 1, 2, ..., N)
\]  

(9b)
Among them, $N$ represents that there are $N$ hidden layers, $F$ represents the activation function of the hidden layer, $W_t$ represents the weight of the $t$-th hidden layer, $w_{t}^{g,c}$ represents the value of the weight corresponding to the $g$th output node in the $t$th hidden layer and the $c$th input node in the $(t-1)$th hidden layer, $y_{t}^{g}$ represents the output value of the $t$th hidden layer, $y_{t-1}^{c}$ represents the input value of the $n_{th}$ input node in the $(t-1)$th hidden layer, and $b_{t}^{g}$ represents the bias of the $g$th output node of the $t$th hidden layer. $Y_0$ represents the input layer, that is, $Y_0 = A$.

**Output layer**

When the output of the last hidden layer is obtained, the output value of the output layer is obtained through linear mapping and non-linear transformation. The output value is calculated as shown in formula (10a), (10b).

\[
Y = F\left[w_0 w_1 \ldots w_n\right] \left[y_N^1 \right] + b \quad \text{(10a)}
\]

\[
Y = F(WY + b) \quad \text{(10b)}
\]

Among them, $w_c$ represents the linear mapping weight of the $c$th neuron in the last hidden layer to the output layer, $y_N^c$ represents the value of the $c$th neuron in the last hidden layer, and $b$ represents the bias value of the output layer, $F$ represents the activation function of the output layer, and $Y$ represents the output value of the output layer (in this paper refers to the unevenness).
Parameter update

In the BMNN network, the value of the feature nodes and the enhancement nodes will not change after being calculated and passed into the input layer, and the corresponding random bias will not change. The main changes in the BMNN network are the weight matrix and bias matrix from the input layer to the hidden layer, the hidden layer to the hidden layer, and the hidden layer to the output layer. This article chooses the Adam algorithm to update the weights and bias parameters. The main steps of the update are as follows:

(i) Calculate the loss function at $q-1$:

$$e(\theta_{q-1}) = \hat{Y} - Y$$  \hfill (11)$$

Among them, $e(\theta_{q-1})$ represents the loss value at time $q-1$ when the parameter is $\theta_{q-1}$, $\hat{Y}$ represents the true value at time $q-1$, and $Y$ represents the predicted value at time $q-1$ calculated by formula (10b).

(ii) Calculate the gradient at time $q$. At time $q-1$, the parameters in the loss function are derived to obtain the gradient information at time $q$;

$$g_q = \nabla_{\theta_{q-1}} e(\theta_{q-1})$$  \hfill (12)$$

Among them, $g_q$ represents the gradient at time $q$ , and $\nabla_{\theta_{q-1}}$ represents the derivative with respect to the parameter $\theta_{q-1}$.

(iii) Using the idea of momentum algorithm, comprehensively consider the historical momentum and current gradient, and calculate the gradient of the current step;

$$m_q = \beta_1 m_{q-1} + (1 - \beta_1) g_q$$  \hfill (13)$$

Among them, the $\beta_1$ coefficient is the exponential decay rate, the default is 0.9 , $m_q$ represents the gradient of the current step, when $q = 0 , m_0 = 0$ .

(iv) Calculate the exponential moving average of the square of the gradient;

$$v_q = \beta_2 v_{q-1} + (1 - \beta_2) g_q^2$$  \hfill (14)$$

Among them, the $\beta_2$ coefficient is the exponential decay rate, and the weighted mean of the gradient square is performed, and the default is 0.999 . $v_q$ represents the exponential moving average of the square of the gradient of the current step. When $q = 0 , v_0 = 0$ .

(v) Since $m_q$ and $v_q$ are initialized to 0 , $m_q$ will be biased toward 0 , especially in the initial stage of training. Therefore, it is necessary to correct the deviation of $m_q$ and $v_q$ to solve the problem in the early stage of training;

$$\hat{m}_q = \frac{m_q}{1 - \beta_1^q}$$  \hfill (15)$$

$$\hat{v}_q = \frac{v_q}{1 - \beta_2^q}$$  \hfill (16)$$

Among them, $\hat{m}_q , \hat{v}_q$ are the exponential moving average of the current step gradient and the square of the current step gradient after the deviation correction.
(v) Update the gradient parameters. The update of the gradient can be adaptively adjusted from the two angles of the gradient mean and the gradient square, and at the same time play a certain annealing effect.

\[
\theta_q = \theta_{q-1} - \alpha \cdot \frac{m_q}{\sqrt{v_q} + \epsilon}
\]  

(17)

Among them, \(\alpha\) is the learning rate, the default \(\alpha = 0.001\), \(\epsilon = 10^{-8}\) plays a role in avoiding the divisor being 0. \(\theta_q\) is the updated parameter.

**Experimental preparation**

**Data preparation**

This paper uses the production data of “Combed compact spinning mechanism fine yarn (100 English counts)” from an Anhui company in China to test the forecasting effect of BMNN. In the data, we select the process parameters that influence the yarn: carding linear speed (m/min) \((x_i)\), carding AFIS neps 1 (Neps/g) \((x_i)\) (the neps detected by AFIS before the cotton passes through the carding machine), carding AFIS neps 2 (Neps/g) \((x_i)\) (the neps detected by AFIS after the cotton passes through the carding machine), carding AFIS short fiber percentage 1 (%) \((x_i)\) (the short fiber percentage detected by AFIS before the cotton passes through the carding machine), carding AFIS short fiber percentage 2 (%) \((x_i)\) (the short fiber percentage detected by AFIS after the cotton passes through the carding machine), carding nodding rate (%) \((x_i)\) (the percentage of the quality of cotton lost during the process of cotton passing through the carding machine to the quality of the input cotton), pre-drawing linear speed in the drawing process (m/min) \((x_i)\), combed AFIS neps 1 (Neps/g) \((x_i)\) (the neps detected by AFIS before the cotton passes through the combing machine), combed AFIS neps 2 (Neps/g) \((x_i)\) (the neps detected by AFIS after the cotton passes through the combing machine), combed AFIS short fiber percentage 1 (%) \((x_i)\) (the short fiber percentage detected by AFIS before the cotton passes through the combing machine), combed AFIS short fiber percentage 2 (%) \((x_i)\) (the short fiber percentage detected by AFIS after the cotton passes through the combing machine), combed nodding rate (%) \((x_i)\) (the percentage of the quality of cotton lost during the process of cotton passing through the combing machine to the quality of the input cotton), pre-drawing nipper times (nips/min) \((x_i)\), combed cylinder speed (rpm) \((x_i)\), combed npper times (nips/min) \((x_i)\) (the working speed of combing machine), roving twist rate (%) \((x_i)\), spun yarn twist rate (%) \((x_i)\), average spun yarn speed (rpm) \((x_i)\) (the spindle speed of spinning frame), spun yarn draw ratio \((x_i)\), yarn count (tex) \((x_i)\), raw cotton short fiber percentage (%) \((x_i)\), neps (Neps/g) \((x_i)\), micronaire value \((x_i)\), strength (cN/tex) \((x_i)\), the average length of the right half (mm) \((x_i)\) (also called the quality length, which represents the length index of the fiber), length uniformity (%) \((x_i)\), impurity rate (%) \((x_i)\) as models input factor set, namely \(X_j = x_{i1}, x_{i2}, \ldots, x_{ik} (k = 1, 2, 3, 26)\). Select the unevenness (%) \((y)\) as the model output.

**Data cleaning**

Use the above-mentioned expert experience method to clean up the data. There are a total of 81 pieces of data, select \(K = 5\), \(M = 3\) for data cleaning, and finally get 62 data after data cleaning. Among them, the first 6 data are shown in Tables 1 and 2.

**Feature selection**

In the yarn production process, there are 8 raw cotton parameters and 21 process parameters. Among them, raw cotton parameters are the key parameters that determine the final yarn quality, and the influence of many process parameters on the final yarn quality is not fully known. Therefore, feature selection of many process parameters is a key step for yarn quality prediction.

Considering the limited scale of the data in this paper, feature selection based on statistics alone may misjudge some important features, and subjective opinions in feature selection based on expert experience alone may adversely affect the results. Therefore, this paper uses the combination of statistical-based feature selection and expert experience-based feature selection to perform feature selection together. The specific process is shown in Figure 4.

(i) Calculate the Pearson correlation coefficient of the input data, and obtain the Pearson correlation coefficient of the unevenness and each process parameter, as shown in Table 3;

(ii) Determine whether the Pearson correlation coefficient is greater than 0.2. If it is greater than 0.2, the feature is retained, otherwise, the feature is deleted;

(iii) Seven experts give relevant weights, calculate the final weights of each feature parameter by frequency method, and rank the weights;

(iv) Take the top 10 features with the largest weight to retain, and delete the remaining features. The final features are shown in Table 4;

(v) Combine the features judged by Pearson’s correlation coefficient and the features obtained from expert experience to obtain the result of feature selection, as shown in Table 5.

**Model building**

Due to its outstanding advantages, BP neural network has been widely used in data prediction and other issues in the past 10 years. When predicting the problem through
Table 1. Data after normalization.

| x₁  | x₂  | x₃  | x₄  | x₅  | x₆  | x₇  | x₈  | x₉  | x₁₀ | x₁₁ | x₁₂ | x₁₃ | x₁₄ | x₁₅ |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 320.67 | 49.67 | 8.23 | 7.73 | 8.5 | 420 | 39.75 | 8.1 | 7.6 | 3.7 | 19  | 418.82 | 352 |

Table 2. Data after normalization.

| x₁₅ | x₁₆ | x₁₇ | x₁₈ | x₁₉ | x₂₀ | x₂₁ | x₂₂ | x₂₃ | x₂₄ | x₂₅ | x₂₆ | y   |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 95.43 | 384.02 | 16000 | 73.5 | 5.6 | 159 | 4.56 | 39.2 | 34.53 | 85.5 | 2.14 | 5.9 | 13.88 |
| 95.43 | 384.02 | 16000 | 73.5 | 5.6 | 159 | 4.56 | 39.2 | 34.53 | 85.5 | 2.14 | 5.9 | 14.03 |
| 95.43 | 384.02 | 16000 | 73.5 | 5.6 | 159 | 4.57 | 39.1 | 34.53 | 85.4 | 2.14 | 5.9 | 14.24 |
| 95.43 | 384.02 | 16000 | 73.5 | 5.6 | 159 | 4.57 | 39.1 | 34.53 | 85.4 | 2.14 | 5.9 | 13.99 |
| 95.43 | 384.02 | 16000 | 73.5 | 5.6 | 161 | 4.58 | 38.8 | 34.39 | 85.3 | 2.15 | 5.9 | 14.08 |

Figure 4. Feature selection flowchart.
Table 3. Pearson coefficient of unevenness and various process parameters.

| Parameter                                      | Pearson correlation coefficient |
|------------------------------------------------|---------------------------------|
| Carding linear speed                          | 0.49                            |
| Carding AFIS neps 1                           | 0.44                            |
| Carding AFIS neps 2                           | 0.51                            |
| Carding AFIS short fiber percentage 1         | 0.36                            |
| Carding AFIS short fiber percentage 2         | 0.35                            |
| Carding nodding rate                          | 0.05                            |
| Pre-drawing linear speed in the drawing process | 0.21                           |
| Combed AFIS neps 1                            | 0.30                            |
| Combed AFIS neps 2                            | 0.27                            |
| Combed AFIS short fiber percentage 1          | 0.30                            |
| Combed AFIS short fiber percentage 2          | 0.35                            |
| Combed nodding rate                           | 0.44                            |
| Combed cylinder speed                         | 0.40                            |
| Combed nipper times                           | 0.01                            |
| Roving twist rate                             | 0.19                            |
| Yarn count                                    | 0.00                            |
| Spun yarn twist rate                          | 0.23                            |
| Average spun yarn speed                       | 0.17                            |
| Spun yarn draw ratio                          | 0.39                            |

modeling, the number of hidden layers in the BP neural network needs to be considered. In recent years, in the small sample prediction model, the four-layer BP neural network has become the mainstream due to its fast running speed and high accuracy.

Considering that the scale of the data in this article is only 62. Therefore, this article uses a BMNN that combines BLS and a four-layer BP neural network. The specific construction process of the network is shown in Figure 5 below.

Normalization processing

Because the input data units are not uniform, some parameters have a relatively large value range, such as process parameters: the average spindle speed has a value range of $17500 - 20000$, and some parameters have a relatively small value range, such as process parameters: the range of linter ratio 2 of combing AFIS is $2.24 - 7.16$. If the data is not normalized, it may cause the BP neural network to converge slowly during the training process and the training time is long. At the same time, parameters with a large value range may have a greater impact on the overall data, resulting in a larger bias; a parameter with a small value range may cause the weight of this part to be small, resulting in the inaccurate training model. Therefore, when data is input into the network, it is very important to normalize the network to eliminate the influence of different dimensions on the training results. Among them, the data normalization processing formula is shown in formula (1), and the processed data is shown in Table 6.

Table 4. The top 10 features with the highest weights given by expert experience.

| Top 10 features                                      | Weight |
|------------------------------------------------------|--------|
| Carding linear speed                                 | 0.89   |
| Carding nodding rate                                 | 0.82   |
| Average spun yarn speed                              | 0.78   |
| Combed AFIS neps 2                                   | 0.71   |
| Spun yarn draw ratio                                 | 0.68   |
| Combed cylinder speed                                | 0.66   |
| Carding AFIS short fiber percentage 1                 | 0.65   |
| Combed nipper times                                  | 0.61   |
| Roving twist rate                                    | 0.57   |
| Combed nodding rate                                  | 0.53   |

Table 5. Results of feature selection.

| Results of feature selection                        |
|-----------------------------------------------------|
| Carding linear speed                                 |
| Carding AFIS neps 1                                 |
| Carding AFIS short fiber percentage 1                |
| Carding AFIS short fiber percentage 2                |
| Carding nodding rate                                 |
| Pre-drawing linear speed in the drawing process      |
| Combed AFIS neps 1                                  |
| Combed AFIS short fiber percentage 1                 |
| Combed AFIS short fiber percentage 2                 |
| Combed nodding rate                                  |
| Combed cylinder speed                                |
| Average spun yarn speed                              |
| Combed nipper times                                 |
| Roving twist rate                                    |
| Spun yarn twist rate                                 |
| Spun yarn draw ratio                                 |

Network initialization

BMNN’s network initialization includes three parts: initialization of weight matrix and bias, initialization of the input layer, initialization of the hidden layer and the output layer:

(i) Initialize the weight matrix and bias. Random initialization is a commonly used initialization method in current neural network research. This article uses random initialization to initialize weight matrix and bias;

(ii) Initialize the input layer. The initialization input layer includes: considering the data scale of this article, there are 100 initialization feature nodes and 200 initialization enhancement nodes. Among them, the feature extraction function is the linear mapping function, and the feature enhancement activation function is the relu function;

(iii) Initialize the hidden layer and output layer. Initializing the hidden layer and output layer includes two parts: initializing nodes and initializing activation functions. Initialize nodes: initialize the number of nodes in the two hidden layers to 32, and the number of nodes in the output layer to be 1. The initialization
activation function includes: initializing the input layer to the first hidden layer, the first hidden layer to the second hidden layer as the relu activation function, and the output from the second hidden layer to the output layer as the sigmoid function.

**Deviation calculation**

(i) Calculate the input layer matrix $A$. Input the training data, calculate the corresponding feature nodes and enhancement nodes according to the randomly generated matrix and merge them to obtain the input layer matrix $A$;

(ii) Calculate the output matrix $Y$. According to the corresponding weight matrix and bias matrix, the output matrix $Y$ corresponding to this piece of training data is calculated;

(iii) Calculate the training deviation $e$. Calculate the deviation $e$ between the output matrix $Y$ and the actual output matrix $\hat{Y}$ corresponding to the training data.

**Algorithm termination conditions**

After the network has calculated the training deviation, it needs to determine whether the algorithm stops or continues iteration according to the algorithm termination condition. The termination condition of the algorithm in this paper is to stop when one of the following two is met:

(i) Reach the maximum number of iterations;

(ii) Algorithm convergence. The training bias of the algorithm for 20 consecutive times does not change.

**Update weights and biases**

If the algorithm continues to iterate, the weights and offsets in the algorithm are updated according to the parameter update process.

**Output model**

If the algorithm meets the termination conditions, it exits the loop, outputs the generated model, and the algorithm stops.

**Experimental verification**

Use the BMNN algorithm to test the seven pieces of data in the test set. To prevent over-fitting and obtain a stable and reliable model, this paper uses cross-validation to select the test set and compare and verify it. In this paper, we choose random seeds from 0 to 9 and compare the predicted value with the real value by using MATLAB. The results obtained are shown in Figure 6 below.

It can be seen from the above Figure 6 that when using BMNN to predict, the prediction accuracy of the model changes with the selection of data. There are some as shown in Figure 6(a) to (e), (g), (i) and (j), the overall accuracy is higher, although there may be some data deviations, it does not affect the main performance of the model; some as shown in Figure 6(f) and (h), there are some data deviations.

To demonstrate the reliability and superiority of BMNN for yarn unevenness prediction, this paper conducts controlled experiments on BMNN using four-layer neural network and BLS.
First, we analyze the complexity and running time of the three algorithms, and the analysis results are shown in the Table 7. Where $A$ denotes the number of iterations of the neural network, $T$ denotes the size of the data, $I$ denotes the number of neurons in the input layer, $J$ denotes the number of neurons in the first hidden layer, $G$ denotes the number of neurons in the second hidden layer, $L$ denotes the number of neurons in the output layer, $Q$ denotes the number of feature nodes, and $R$ denotes the number of enhancement nodes.

Next, we analyzed the experimental results of the three algorithms. We compared the experimental results of the three algorithms using the method of selecting random seeds 0–9. The specific comparison results are shown in Figure 7 below.

Since BLS is too far apart from the evaluation indicators of BMNN and four-layer neural network, each evaluation indicator is represented by two graphs. Taking Figure 7(a) and (b) as examples, Figure 7(a) shows that the overall performance of BLS has a large gap with BMNN and four-layer neural network; when Random-seed=2 in Figure 7(b), BLS has extremely good performance, which shows that in some accidental situations the BLS algorithm has extremely good performance, but it is not stable. At the same time, the overall effect of BMNN in Figure 7(b) is superior to the performance of the four-layer neural network, which illustrates the reliability and superiority of the BMNN algorithm. Summarizing the images under the three evaluation indicators in Figure 7 shows that the BMNN algorithm has extremely good performance, but it is not stable. At the same time, the overall effect of BMNN in Figure 7(b) is superior to the performance of the four-layer neural network, which illustrates the reliability and superiority of the BMNN algorithm. Among them, $actual_i$ represents the $i_{th}$ true value, $predicted_i$ represents the $i_{th}$ predicted value, and $N$ represents a total of $N$ data in the test set.

Finally, we performed an $F$-test on the experimental results in order to justify the experimental results, and the results obtained are shown in Table 8. In order to better summarize the effectiveness of the algorithm, we analyzed the above results. Comparing the complexity of the three algorithms shows that: the complexity of the neural network algorithm is higher because the neural network algorithm requires iteration, and the BLS algorithm does not require iteration. The complexity of the BMNN algorithm is mainly related to the complexity of the neural network. Comparing the running time of the three algorithms, it can be seen that the running time of the BMNN algorithm is approximately equal to that of the four-layer neural network, and the running time of the BLS algorithm is shorter. Comparing the results of the three algorithms, it can be seen that the BLS operating effect is unstable, and it can achieve excellent results in individual cases, and the operating effect is particularly poor in individual cases; the four-layer neural network has stable operating results and can achieve good results; BMNN is a combination of two algorithms, which not only maintains the stability of the algorithm, but also ensures the further improvement of its accuracy.

Next, we correlated and analyzed the operational effects of these three algorithms with the structure of the model. BLS obtains the mapping matrix between feature nodes and enhancement nodes and the output, and then predicts through the mapping matrix. This method is convenient for explaining the rationality of the results. However, in the real data set, even if the same batch of yarn is produced, the quality of the final yarn will be different; at the same time, when the data scale does not meet the requirements, each piece of forecast data may be new data that has not appeared in the database. Therefore, if the prediction data exists in the

### Table 6. Data after normalization.

| Index | Carding linear speed | Carding AFIS neps 1 | Carding AFIS neps 2 | Carding AFIS short fiber percentage 1 | Carding AFIS short fiber percentage 2 | Trash content | Unevenness |
|-------|----------------------|---------------------|---------------------|---------------------------------------|---------------------------------------|----------------|------------|
| 1     | 1                    | 0                   | 0.0987              | 1                                     | 1                                     | 0.5094         | 0.58       |
| 2     | 1                    | 0.7410              | 0.4689              | 0.6937                                | 0.5984                                | 0.5660         | 0.35       |
| 3     | 1                    | 0.7410              | 0.4689              | 0.6937                                | 0.5984                                | 0.6604         | 0.23       |
| 4     | 1                    | 0.7410              | 0.4689              | 0.6937                                | 0.5984                                | 0.7359         | 0.69       |
| 5     | 1                    | 0.7410              | 0.4689              | 0.6937                                | 0.5984                                | 0.3962         | 0.21       |
| 6     | 1                    | 0.3429              | 0.0869              | 0.2703                                | 0.4139                                | 0.6981         | 0.13       |
| ...   | ...                  | ...                 | ...                 | ...                                   | ...                                   | ...            | ...        |
| 53    | 1                    | 0.5539              | 0.2843              | 0.3694                                | 0.4426                                | 0.4906         | 0.24       |
| 54    | 1                    | 0.4964              | 0.3337              | 0.0496                                | 0.0                                     | 0.6604         | 0.40       |

![Image of Table 6](image)

\[
MAE = \frac{1}{N} \sum_{i=1}^{N} | actual_i - predicted_i | \tag{18}
\]

\[
MSE = \frac{1}{N} \sum_{i=1}^{N} (actual_i - predicted_i)^2 \tag{19}
\]

\[
RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (actual_i - predicted_i)^2} \tag{20}
\]

Among them, $actual_i$ represents the $i_{th}$ true value, $predicted_i$ represents the $i_{th}$ predicted value, and $N$ represents a total of $N$ data in the test set.

![Image of Table 6](image)

\[
MAE = \frac{1}{N} \sum_{i=1}^{N} | actual_i - predicted_i | \tag{18}
\]

\[
MSE = \frac{1}{N} \sum_{i=1}^{N} (actual_i - predicted_i)^2 \tag{19}
\]

\[
RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (actual_i - predicted_i)^2} \tag{20}
\]

Among them, $actual_i$ represents the $i_{th}$ true value, $predicted_i$ represents the $i_{th}$ predicted value, and $N$ represents a total of $N$ data in the test set.

Finally, we performed an $F$-test on the experimental results in order to justify the experimental results, and the results obtained are shown in Table 8. In order to better summarize the effectiveness of the algorithm, we analyzed the above results. Comparing the complexity of the three algorithms shows that: the complexity of the neural network algorithm is higher because the neural network algorithm requires iteration, and the BLS algorithm does not require iteration. The complexity of the BMNN algorithm is mainly related to the complexity of the neural network. Comparing the running time of the three algorithms, it can be seen that the running time of the BMNN algorithm is approximately equal to that of the four-layer neural network, and the running time of the BLS algorithm is shorter. Comparing the results of the three algorithms, it can be seen that the BLS operating effect is unstable, and it can achieve excellent results in individual cases, and the operating effect is particularly poor in individual cases; the four-layer neural network has stable operating results and can achieve good results; BMNN is a combination of two algorithms, which not only maintains the stability of the algorithm, but also ensures the further improvement of its accuracy.

Next, we correlated and analyzed the operational effects of these three algorithms with the structure of the model. BLS obtains the mapping matrix between feature nodes and enhancement nodes and the output, and then predicts through the mapping matrix. This method is convenient for explaining the rationality of the results. However, in the real data set, even if the same batch of yarn is produced, the quality of the final yarn will be different; at the same time, when the data scale does not meet the requirements, each piece of forecast data may be new data that has not appeared in the database. Therefore, if the prediction data exists in the
Figure 6. Comparison chart of predicted value and true value: (a) random seed = 0, (b) random seed = 1, (c) random seed = 2, (d) random seed = 3, (e) random seed = 4, (f) random seed = 5, (g) random seed = 6, (h) random seed = 7, (i) random seed = 8, and (j) random seed = 9.
database and the relevant features are extracted by BLS will produce excellent results; otherwise, it will not work. The four-layer neural network is a black box model. There are a large number of linear and nonlinear crossovers inside. Although the effect obtained cannot be explained, it generally produces a relatively stable effect, that is, the accuracy of the model is not high, but the error is not large too. The BMNN adds a hidden layer training process after extracting the features, and its purpose is to stabilize the final result and make up for the shortcomings of instability when using BLS prediction. The effect of BMNN under the real data set is also consistent with the network structure, not only the

### Table 7. The complexity and running time of the three algorithms.

|                       | Four-layer neural network | BLS                                      | BMNN                                      |
|-----------------------|---------------------------|------------------------------------------|-------------------------------------------|
| Complexity            | $O(A \times T \times (I \times J + G + G \times L))$ | $O(T \times (I \times Q + Q \times R + R \times L + L \times Q))$ | $O(T \times (I \times Q + Q \times R + R \times L + L \times Q + A \times (I \times J + G + G \times L)))$ |
| Running time (s)      | 3.4348                    | 1.7199                                   | 4.0027                                    |

### Figure 7. Model evaluation: (a) MSE evaluation index-large, (b) MSE evaluation index-small, (c) RMSE evaluation index-large, (d) RMSE evaluation index-small, (e) MAE evaluation index-large, and (f) MAE evaluation index-small.

### Table 8. Results of the $F$-test.

|                      | Four-layer neural network | BLS      | BMNN           |
|----------------------|---------------------------|----------|----------------|
| $P$                  | 2.1827e-13                | 0.0032   | 1.3367e-12     |
accuracy is improved, but the stability is consistent with the four-layer neural network.

Finally, we analyzed the limitations of the method.

(i) Although BMNN is an improvement on BLS and four-layer neural network, it is still a black-box model overall, which cannot explain the relationship between the extracted feature information and the final result;
(ii) In general, the complexity of BMNN is higher compared to BLS as well as neural network;
(iii) BMNN increases the generalizability of the model, but in some special cases, the accuracy is not guaranteed to be better than BLS;
(iv) If the data size is small, BMNN cannot guarantee the accuracy of the model.

In summary, we can conclude that the BMNN algorithm is suitable for parameter prediction under complex process situations with a certain size of data (related to the parameters of the data, if there are more parameters, more data is required).

Conclusion
This paper studies a multi-layer neural network based on the BLS. The model retains the advantages of the traditional BLS (proposes the concept of feature extraction and feature enhancement) and uses a multi-layer neural network to improve the BLS in some practical the effect on the data set. This paper proposes a method of combining expert experience and Pearson coefficients to perform feature selection on factory data after data cleaning. And compares the performance of the four-layer neural network, BLS and BMNN in the prediction of yarn unevenness in the case of small samples. After many different cross-validation comparisons, it is concluded that the prediction effect of BLS in this sample set is unstable, and it has an extremely good prediction effect in some occasional cases, but the effect is poor in most cases; the four-layer neural network has good results under general conditions; and BMNN not only retains the advantages of the stable prediction effect of the four-layer neural network, but also improves the accuracy compared to the four-layer neural network algorithm.

Acknowledgements
Thanks are due to Tingting Fan for assistance with the experiments.

Declaration of conflicting interests
The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding
The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported in part by the Science and Technology Research Program of Henan Province, China (202102210181).

ORCID iD
Jiuxiang Song https://orcid.org/0000-0002-0173-6824

References
1. Cheng L and Adams DL. Yarn strength prediction using neural networks: Part I: fiber properties and yarn strength relationship. Text Res J 1995; 65(9): 495–500.
2. Furferi R and Gelli M. Yarn strength prediction: a practical model based on artificial neural networks. Adv Mech Eng 2010; 2(2): 640103–641660.
3. Mwasagi JI. The use of extreme learning machines (ELM) algorithms to prediction strength for cotton ring spun yarn. Fashion Text 2016; 3(1): 23.
4. Zhenlong H, Qiang Z and Jun W. The prediction model of cotton yarn intensity based on the CNN-BP neural network. Wirel Pers Commun 2018; 102(2): 1905–1916.
5. Hu Z, Zhao Q and Wang J. The prediction model of cotton yarn quality based on artificial recurrent neural network. In: International conference on applications and techniques in cyber security and intelligence. Cham: Springer, 2019, pp.857–866.
6. Ureyen ME and Gürkan P. Comparison of artificial neural network and linear regression models for prediction of ring spun yarn properties. II. Prediction of yarn hairiness and unevenness. Fibers Polym 2008; 9(1): 92–96.
7. Ute TB and Kadoglu HÜS. A statistical model for predicting yarn evenness of cotton sirospun yarns. Eur J Eng Nat Sci 2016; 1(1): 57.
8. Malik SA, Farooq A, Gereke T, et al. Prediction of blended yarn evenness and tensile properties by using artificial neural network and multiple linear regression. Austex Res J 2016; 16(2): 43–50.
9. Wu ZG, Zhang KJ and Yu CW. Cotton yarn quality prediction based on mind evolutionary neural network. DEStech Transactions on Engineering and Technology Research, (MCEE), DEStech Publications, Inc., 2017.
10. Ghanmi H, Ghith A and Benameur T. Ring yarn quality prediction using hybrid artificial neural network. Int J Clothing Sci Technol 2015; 27(6): 940–956.
11. Singh S, Bhowmick N and Vaz A. Influence of can-spring stiffness, delivery speed and sliver coils position on unevenness. J Text Appar Technol Manag 2019; 11(1): 1–12.
12. Pao Y. Adaptive pattern recognition and neural networks. Reading, MA: Addison-Wesley, 1989.
13. Pao YH and Takefuji Y. Functional-link net computing: theory, system architecture, and functionalities. Computer 1992; 25(5): 76–79.
14. Patra JC, Pal RN, Chatterji BN, et al. Identification of non-linear dynamic systems using functional link artificial neural networks. IEEE Trans Syst Man Cybern B 1999; 29(2): 254–262.
15. Mahapatra R, Majhi B and Rout M. Development and performance evaluation of improved classifiers of microarray data. In: International conference on advances in engineering, Nagapattinam, Tamil Nadu, India, 30–31 March 2012.
16. Zhang L and Suganthan PN. Visual tracking with convolutional random vector functional link network. IEEE Trans Cybern 2017; 47: 3243–3253.
17. Sahoo DM and Chakraverty S. Functional link neural network learning for response prediction of tall shear buildings
with respect to earthquake data. *IEEE Trans Syst Man Cybern Syst* 2018; 48: 1–10.

18. Zhang Y, Li B, Wang Y, et al. The forecast of the temperature in subway station based on RVFL neural network. In: 2018 11th International congress on image and signal processing, biomedical engineering and informatics (CISP-BMEI), Beijing, China, 13–15 October 2018, pp.1–5. IEEE.

19. Sheng Z, Zeng Z, Tian Q, et al. Dynamic thermal prediction model for the electronic equipment cabin based on RVFL network. In: 2018 11th International Congress on Image and Signal Processing, biomedical Engineering and Informatics (CISP-BMEI), Beijing, China, 13–15 October 2018.

20. Chen CLP and Liu Z. Broad learning system: an effective and efficient incremental learning system without the need for deep architecture. *IEEE Trans Neural Netw Learn Syst* 2018; 29(1): 10–24.

21. Yang J, Li T, Zuo Y, et al. Forecast application of time series model based on BLS in port cargo throughput. In: 2018 International conference on Security, Pattern Analysis, and Cybernetics (SPAC), Jinan, China, 14–17 December 2018.

22. Chen CLP, Liu Z and Feng S. Universal approximation capability of broad learning system and its structural variations. *IEEE Trans Neural Netw Learn Syst* 2019; 30(4): 1191–1204.