Hindawi
Complexity
Volume 2019, Article ID 8682124, 21 pages
https://doi.org/10.1155/2019/8682124

Research Article

A Differential Evolution-Oriented Pruning Neural Network Model for Bankruptcy Prediction

Yajiao Tang,[1,2] Junkai Ji,[3] Yulin Zhu,[1] Shangce Gao,[2] Zheng Tang,[2] and Yuki Todo[4]

[1] *College of Economics, Central South University of Forestry and Technology, Changsha 410004, China*
[2] *Faculty of Engineering, University of Toyama, Toyama 930-8555, Japan*
[3] *College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China*
[4] *School of Electrical and Computer Engineering, Kanazawa University, Kanazawa-shi 920-1192, Japan*

Correspondence should be addressed to Shangce Gao; gaosc@eng.u-toyama.ac.jp and Yuki Todo; yktodo@ec.t.kanazawa-u.ac.jp

Received 7 June 2019; Accepted 14 July 2019; Published 4 August 2019

Guest Editor: Thiago C. Silva

Financial bankruptcy prediction is crucial for financial institutions in assessing the financial health of companies and individuals. Such work is necessary for financial institutions to establish effective prediction models to make appropriate lending decisions. In recent decades, various bankruptcy prediction models have been developed for academics and practitioners to predict the likelihood that a loan customer will go bankrupt. Among them, Artificial Neural Networks (ANNs) have been widely and effectively applied in bankruptcy prediction. Inspired by the mechanism of biological neurons, we propose an evolutionary pruning neural network (EPNN) model to conduct financial bankruptcy analysis. The EPNN possesses a dynamic dendritic structure that is trained by a global optimization learning algorithm: the Adaptive Differential Evolution algorithm with Optional External Archive (JADE). The EPNN can reduce the computational complexity by removing the superfluous and ineffective synapses and dendrites in the structure and is simultaneously able to achieve a competitive classification accuracy. After simplifying the structure, the EPNN can be entirely replaced by a logic circuit containing the comparators and the logic NOT, AND, and OR gates. This mechanism makes it feasible to apply the EPNN to bankruptcy analysis in hardware implementations. To verify the effectiveness of the EPNN, we adopt two benchmark datasets in our experiments. The experimental results reveal that the EPNN outperforms the Multilayer Perceptron (MLP) model and our previously developed preliminary pruning neural network (PNN) model in terms of accuracy, convergence speed, and Area Under the Receiver Operating Characteristics (ROC) curve (AUC). In addition, the EPNN also provides competitive and satisfactory classification performances in contrast with other commonly used classification methods.

1. Introduction

The overwhelming 2007/2008 financial crisis led to the bankruptcy of many large-scale financial institutions and made some subject to takeover by their government. Thus, bankruptcy risk management has become an important field of study worldwide. Bankruptcy by a company denotes a situation in which the operating cash flow of the company and its negative net assets cannot be balanced. This always results in the practical weakening of the profitability of a company. The purpose of bankruptcy prediction is to evaluate the present and future financial status of a company from the perspective of its long-term operation in the market.

Various quantitative statistical approaches have been adopted to improve bankruptcy forecasting models. Discriminant analysis is adopted to classify observations between good and bad payers [1], and logistic regression is adapted to determine the default probability of the borrowers [2]. However, it is argued that these popular models are inaccurate [3]. Hence, several machine learning tools are explored to assess bankruptcy risk using computer technology. Because bankruptcy risk analysis is similar to pattern recognition tasks, most methods can be adapted to classify the creditworthiness of potential clients of financial institutions [4, 5]. Among them, ANNs achieve outstanding performances in applications such as predicting financial crises [6], scoring credit [7], and building up credit analysis models [8]. The

adoption of ANNs in bankruptcy prediction has been studied since the 1990s [9, 10]. Prior studies revealed that ANNs are powerful for use in pattern recognition because of their nonlinear and nonparametric adaptive-learning properties [11]. This imbues ANNs with obvious advantages over conventional statistical algorithms and inductive learning methods, especially in comparison with discriminant analysis and logistic regression [12]. Hence, researchers have put a major emphasis on the application of ANNs in finance and accounting.

ANNs are flexible and nonparametric modelling tools capable of performing any complicated function mapping with arbitrarily required accuracies [13–15]. Among the diverse types of ANNs, MLP is one of the simplest and most widely applied models, in which the hidden layer determines the mapping relationships between input and output layers and the relationships between neurons stored as the weights of the connecting links [16]. The MLP's learning algorithm implements a gradient search to minimize the squared error between the realized and desired outputs. This type of three-layer MLP is a commonly adopted ANN structure for binary classification problems such as bankruptcy prediction [11]. Although the characteristics of ANN ensembles, such as efficiency, robustness, and adaptability, make them a valuable tool for classification, decision support, financial analysis, and credit scoring, it should be noted that some researchers have shown that the ensembles of multiple neural network classifiers are not always superior to a single best neural network classifier [17]. Hence, we focus on applying a single neural network model to bankruptcy prediction.

In biological neuron models, a dendritic computation mechanism can provide a concrete explanation concerning the positioning of the synaptic inputs at the proper connections. This means that redundant synapses and dendrites are left in the neural network initially, while the useless ones are quickly deleted, with the remaining being strengthened. Ultimately, this process creates an enhanced neural network function form. Inspired by these histological theories, Koch et al. notes that interactions between excitatory and inhibitory inputs have apparent nonlinearity. Once inhibitory inputs and excitatory inputs are located on the same path to the soma, the inhibitory inputs can specifically eliminate the excitatory inputs. However, issues, such as whether the excitatory or inhibitory synapse should be kept, where it should locate, and which dendritic branch should be strengthened, are unaddressed in this model [18]. Later, Koch et al. noted that the interactions among synapses and the responses at the connection nodes could be regarded as logic operations [19], and a specialized learning algorithm based on the plasticity in dendrites was required to train the model [20].

In our previous research, a PNN model, in which the particular locations and types of synapses on the dendrite branches are formulated via learning, is proposed, and useless and superfluous synaptic and dendritic connections are eliminated. Thus, the efficiency of the model is enhanced [21, 22]. Similar to most other ANNs, PNN adopts the backpropagation (BP) algorithm as its learning method. However, learning algorithms are widely considered to have significant influences on the performances of ANNs [23, 24]. The BP algorithm and its variations [23, 25] are considered rather inefficient because of their obvious drawbacks such as their slow convergence [26], sensitivity to initialization [27], and a tendency to become trapped in local minima [28, 29]. Specifically, first, during the learning process, the error often remains large because the learning algorithm leads the ANNs to local minima instead of the global minimum. This problem is quite common in gradient-based learning approaches. Second, the convergence of the BP algorithm is strongly dependent on the initial values of the learning rate and momentum. The unsuitable values for these variables may even lead to divergence. Third, the learning time increases substantially when the dataset becomes larger [30]. Many researchers have focused on making improvements to resolve these shortcomings of BP, but each method has its disadvantages [31, 32]. These disadvantages make them unreliable for risk classification applications and inspire us to adopt other algorithms to train the neural model to avoid the computational inefficiency and local minimum problems.

In this study, we propose an EPNN model with a dendritic structure as a global optimization algorithm called the JADE algorithm [33]. With respect to the EPNN, the axons of the other neurons transmit the input signals to the synaptic layer; then, the interaction of the synaptic signals transfers to every branch of the dendrites. Next, the interactions are collected and sent to the membrane layer and then transformed to the soma body. In addition, the neuronal pruning function can remove extra synapses and dendrites and simultaneously achieve high accuracy. Specifically, during the training process, the superfluous inputs and dendrites are eliminated, while the useful and necessary ones are retained. Then, the neuronal pruning function can produce a simplified dendritic morphology without a loss of classification accuracy. Furthermore, the simplified topological morphology can operate similarly to a logic circuit composed merely of comparators and logic NOT, AND, and OR gates. Thus, applying EPNN to bankruptcy analysis can easily be implemented in hardware. To the best of our knowledge, we note that if achieved through hardware implementation, this technique will achieve the highest computation speed when compared with other methods. This demonstrates an excellent adoption possibility for financial institutions. JADE is a state-of-the-art variant of the differential evolution algorithm and uses a self-adaptive mechanism to select suitable parameters for each optimization problem. This imbues JADE with a better balance between exploration and exploitation compared to other heuristic algorithms [34]. In training the EPNN, JADE can avoid local minima and speed up the training process during the optimization process. Thus, JADE allows the EPNN to obtain satisfactory results and produce an effective logic circuit for each bankruptcy prediction problem.

In addition to avoiding misleading and contradictory conclusions, four key components are carefully defined to allow one to draw well-founded conclusions from the experimental results. First, the research has adopted both benchmark and application-oriented databases, namely, a Qualitative Bankruptcy dataset from the UCI Machine Learning Database Repository and a Distress dataset from the Kaggle dataset. Second, in the simulation, the two datasets are

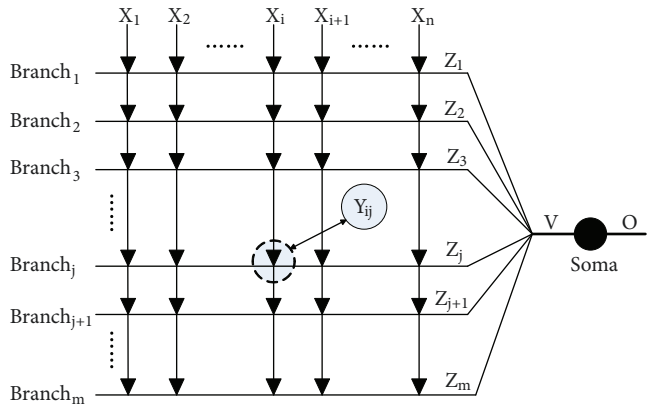

Figure 1: The morphological architecture of the PNN.

separated into a training set and a testing set at proportions of 50% each. Third, the average accuracy, sensitivity, specificity, convergence speed, and AUC are used as the evaluation metric framework; such metrics can be used to effectively and efficiently analyse the possibility of bankruptcy. Fourth, a nonparametric test called the Wilcoxon rank-sum test has been adopted to allow us to claim that the observed result differences in performance are statistically significant and not simply caused by random splitting effects.

To conclude, our main contributions are clarified as follows: first, a novel EPNN model is proposed in this paper which can adopt synaptic and dendritic pruning to simplify its neuron morphology during the training process. Second, the simplified model of EPNN can be completely replaced by logic circuits which be easily implemented on hardware. The logic circuits can maintain high classification accuracy and obtain extremely high computation speed, simultaneously. Last but not least, comprehensive comparison experiments have been implemented to demonstrate that the EPNN outperforms the MLP, PNN, and other commonly used classifiers on the bankruptcy prediction problems.

The remainder of this paper is constructed as follows. Section 2 presents an overview of the related theories in bankruptcy analysis. Section 3 introduces the proposed EPNN model in detail. Moreover, the EPNN's learning algorithm JADE is described. Section 4 presents the experimental results obtained using the EPNN and makes a comparison with other algorithms by adopting the Qualitative Bankruptcy dataset and Distress dataset. Section 5 concludes this paper.

2. Proposed Model

We build up the EPNN, which has a dendritic structure and which is trained by JADE, to achieve a high bankruptcy classification accuracy. The morphological architecture of the EPNN is shown in Figure 1. The network has four layers, namely, a synaptic layer, a dendritic layer, a membrane layer, and a soma layer. The inputs $x_{1\sim n}$ from the axons of the prior neurons enter the synaptic layer; then, the interactions of the synaptic signals occur on each branch of dendrites. After that, the interactions are collected and sent to the membrane layer; finally, they are sent to the soma body. During the training process, the necessary inputs and useful dendrites are held, whereas the unnecessary ones are filtered out. The cell would be motivated and would then send an output signal to other neurons through the axon terminal when the input of the soma exceeds its threshold. The morphological architecture of the EPNN model is presented below in detail.

2.1. Synaptic Layer. The synaptic layer of a neuron represents the specific area at which nerve impulses are transmitted among neurons, thereby passing through the axon terminal of a neuron where neurotransmitters are released in response to an impulse [35]. The impulse is implemented using a certain pattern of a specific ion. When an ion transmits to the receptor, the potential of the receptor is changed and determines the excitatory or inhibitory characteristic of a synapse [36]. The flow direction of the synaptic layer is feed-forward, which conventionally starts from a presynaptic neuron and transmits to a postsynaptic neuron. In the EPNN, these connections are formulated by a sigmoid function with a single input and a single output. The equation of the j^{th} ($j = 1, 2, 3, \ldots, J$) synaptic layer receiving the i^{th} ($i = 1, 2, 3, \ldots, I$) input is expressed as follows:

$$Y_{ij} = \frac{1}{1 + e^{-k(w_{ij}x_i - q_{ij})}}, \tag{1}$$

where k is a positive constant, w_{ij} and q_{ij} are synaptic parameters that need to be optimized by the learning algorithm, and x_i is the input of the synapse, with a range of $[0, 1]$. There are four types of connection states corresponding to different values of w_{ij} and q_{ij}: a direct connection, a reverse connection, a constant-1 connection, and a constant-0 connection, as shown in Figure 2. θ_{ij} represents the threshold of a synaptic layer; this threshold can be defined by the following equation,

$$\theta_{ij} = \frac{q_{ij}}{w_{ij}}. \tag{2}$$

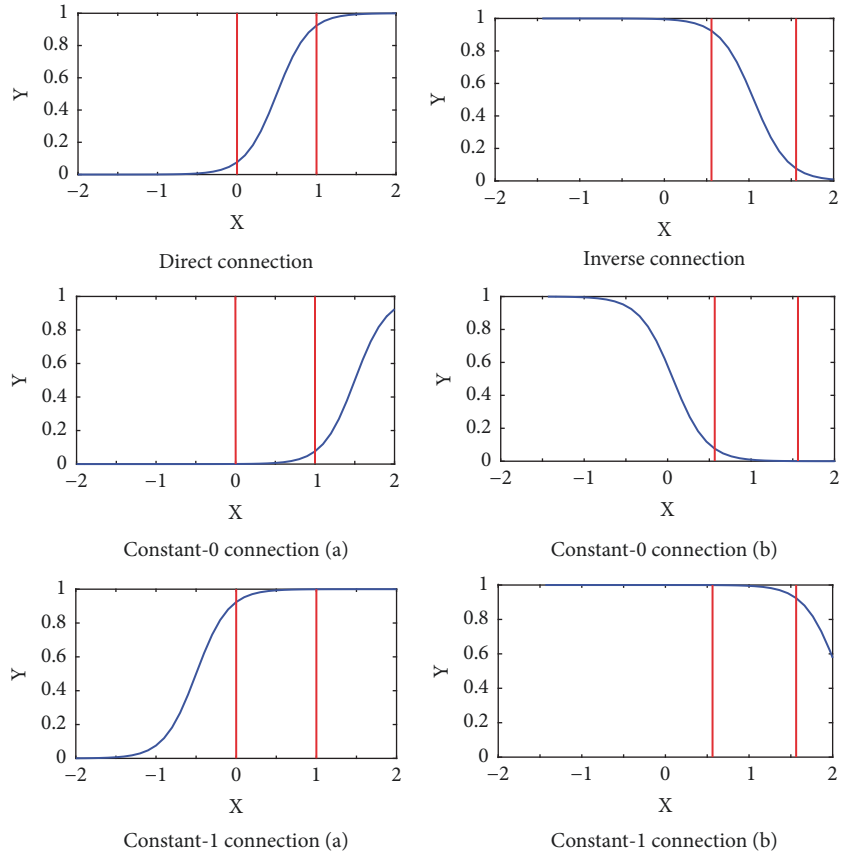

Figure 2: Six connection cases of the synaptic layer.

2.1.1. Direct Connection. $0 < q_{ij} < w_{ij}$, e.g., $q_{ij} = 0.5$ and $w_{ij} = 1.0$, corresponds to a direct connection. Once $x_i > \theta_{ij}$, the output Y_{ij} approximates to 1, the synapse becomes excitatory, and it depolarizes the soma layer. When $x_i \leq \theta_{ij}$, the corresponding output tends to be 0, the synapse becomes inhibitory, and it hyperpolarizes the soma layer in a transient manner. In general, regardless of the input values, the outputs always approximate the inputs.

2.1.2. Inverse Connection. $w_{ij} < q_{ij} < 0$, e.g., $q_{ij} = -0.5$ and $w_{ij} = -1.0$, leads to an inverse connection. Once $x_i > \theta_{ij}$, the output Y_{ij} is approximately 0, and the synapse becomes inhibitory. In addition, it will hyperpolarize the soma layer in a transient manner. In contrast, when $x_i \leq \theta_{ij}$, the output Y_{ij} is approximately 1, the synapse will become excitatory, and it depolarizes the soma layer. Briefly, regardless of the values of the inputs in $[0, 1]$, the output will receive an inverse signal triggered by the input. This can be regarded as a logic NOT operation.

2.1.3. Constant-0 Connection. There are two states in the constant-0 connection: $w_{ij} < 0 < q_{ij}$, e.g., $q_{ij} = 0.5$ and $w_{ij} = -1.0$, and $0 < w_{ij} < q_{ij}$, e.g., $q_{ij} = 1.5$ and $w_{ij} = 1.0$. Regardless of the value of the input, the outputs are always approximately 0.

2.1.4. Constant-1 Connection. There are two states in the constant-1 connection: $q_{ij} < 0 < w_{ij}$, e.g., $q_{ij} = -0.5$ and $w_{ij} = 1.0$, and $q_{ij} < w_{ij} < 0$, e.g., $q_{ij} = -1.5$ and $w_{ij} = -1.0$. The corresponding output tends to be 1 all the time regardless of whether the input signal x_i exceeds the threshold θ_{ij}. This means that the signals of the synaptic layer have minimal impact on the dendritic layer. Whenever the excitatory input signals transport in, depolarization occurs in the next membrane layer.

The values of w_{ij} and q_{ij} are initialized randomly between -1.5 and 1.5. This represents that the inputs connect to each dendritic branch in one of the four synaptic connection statuses randomly. When the values of w_{ij} and q_{ij} change, the connection states of the synaptic layer vary. In Figure 3, four marks are adopted to represent the four connection states: a direct connection (•), an inverse connection (■), a constant-1 connection (①), and a constant-0 connection (⓪).

2.2. Dendrite Layer. A dendrite layer denotes a typical nonlinear interaction of the synaptic signals on each branch of dendrites. The multiplication operation plays a vital role in the process of transporting and disposing the neural information [37, 38]. Thus, the nonlinearity calculation of the synaptic layer can be implemented by a typical multiplication operation instead of summation. The interaction of a dendritic branch is equivalent to a logic AND operation.

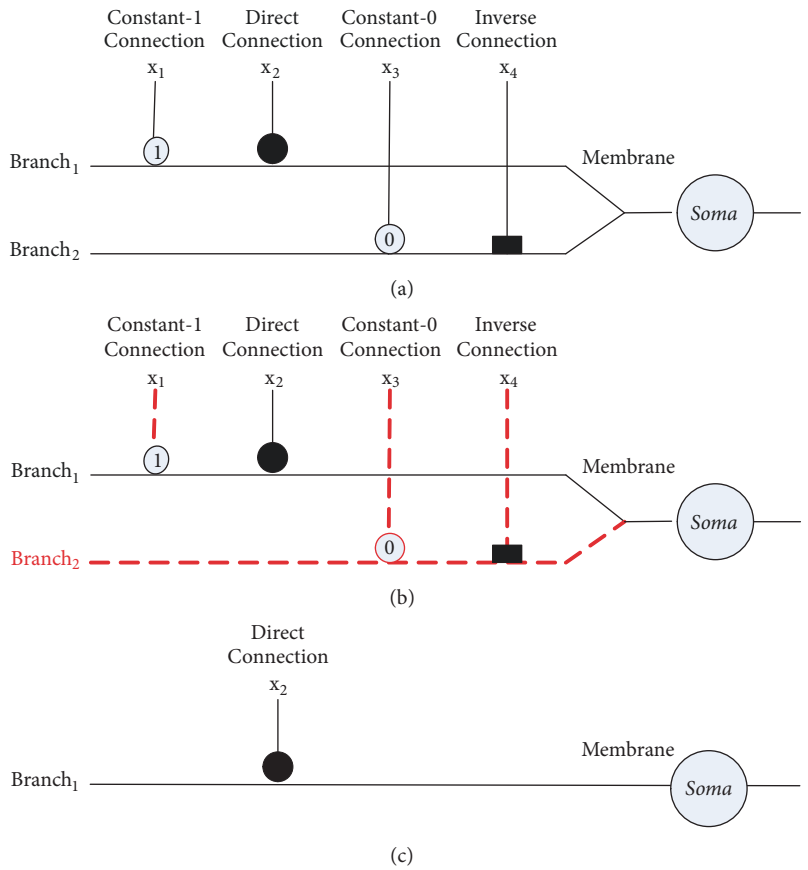

Figure 3: An example of a synaptic and dendritic pruning procedure.

The operation of the input variables will generate a 1 when and only when all input variables equal 1 simultaneously. The corresponding equation of the dendrite layer is defined as follows:

$$Z_j = \prod_{i=1}^{I} Y_{ij}. \tag{3}$$

2.3. Membrane Layer. A membrane layer accumulates the linear summation of the dendritic signals from the upper dendrite layer. It is similar to the logic OR operation in the binary cases. This logic OR operation generates a 1 when at least one of the variables is 1. Its equation is given below:

$$V = \sum_{j=1}^{J} Z_j. \tag{4}$$

2.4. Soma Layer. The output of the membrane layer is transmitted to the soma layer. Once the membrane potential exceeds the threshold, the neuron fires. A sigmoid operation is used to describe the function of the soma layer:

$$O = \frac{1}{1 + e^{-k_{soma}(V - \theta_{soma})}}. \tag{5}$$

2.5. Neural Pruning Function. The EPNN possesses the ability to perform a neural pruning function to simplify its topological morphology. The neural pruning technique represents omitting the extra nodes and weights by learning and training the neural network [39]. In the EPNN, the pruning function can eliminate unnecessary synapses and dendrites and then form a unique neural structure for a given problem. The function contains two parts: synaptic pruning and dendritic pruning.

Synaptic pruning: when the synaptic layer that receives the input from the axon is in the constant-1 connection case, the synaptic output is always 1. Because of the multiplication operation, the result of any arbitrary value multiplying 1 will equal itself in the dendrite layer. It is evident that the synaptic input in the constant-1 connection has minimal impact on the output of the dendrite layer. Therefore, this type of synaptic input can be neglected entirely.

Dendritic pruning: when the synaptic layer that receives the input signal is in the constant-0 connection case, the output is always 0. Consequently, the output of the adjacent dendrite layer becomes 0 because the result of any value multiplying 0 equals 0. This implies that this entire dendrite layer should be omitted because it has minimal influence on the result of the soma layer.

An example of a synaptic and dendritic pruning procedure is presented in Figure 3. The neural structure is

composed of four synaptic inputs, two dendrite branches, one membrane layer, and one soma layer as shown in Figure 3(a). The connection case of input x_1 is ① in Branch 1; this synaptic layer can be omitted according to the mechanism of synaptic pruning. In addition, the connection case of input x_3 is ⓪ in Branch 2; Branch 2 can be completely deleted based on the dendritic pruning mechanism. The unnecessary synaptic inputs and dendritic branches, which are shown with dotted lines in Figure 3(b), should be removed. Finally, the simplified dendritic morphology can be obtained, as in Figure 3(c). Only the synaptic layer on Branch 1 remains in the structure because only the input x_2 can affect the final output of the soma.

3. Learning Algorithm

Actually, PNN suffers from the curse of dimensionality. When the dimension increases largely, any small change of the weights on one dendritic branch will produce a great disparity of its final results because of the multiplication operation. This is the main limitation of EPNN. Thus it needs us to propose more powerful optimization algorithms to figure it out. Conventional classifiers use BP to adjust the weights and threshold. However, BP suffers an inherent local minimum trapping problem and has difficulties in achieving the globally best values of its weights and thresholds. This disadvantage of BP has largely limited the computational capabilities of our neural mode. To improve the performance of the EPNN, we adopt JADE to train the model.

JADE has been regarded as one of a few "important variants of Differential Evolution (DE)" in a major DE review published in 2011 [34]. The vast popularity of DE algorithms has led to an increasing interest in developing their variants [46–48]. It is well known that the performances of many metaheuristic methods are influenced by the choice of their control parameters [49, 50]. JADE can use a self-adaptive mechanism to select suitable parameters F and CR for different optimization problems and implement a "DE/current-to-$p$best" mutation strategy with an optional external archive. The experimental results verified that JADE obtains a better balance between exploration and exploitation during the evolutionary process and is superior to other optimization algorithms in terms of solution quality and convergence rate [33]. JADE follows the general procedure of an evolutionary algorithm. After initialization, DE executes a loop of evolutionary operations: mutation, crossover, and selection. In addition, JADE dynamically updates control parameters as the evolutionary search proceeds.

Initialization: each agent in the initial population $x_{i,0} = x_{1,i,0}, x_{2,i,0}, \ldots, x_{D,i,0}$, $i = 1, 2, \ldots, NP$ is generated randomly according to a uniform distribution.

$$x_j^{low} \leqslant x_{j,i,0} \leqslant x_j^{up}, \quad j = 1, 2, \ldots, D, \tag{6}$$

where D is the dimensionality of the problem and NP is the population size.

Mutation: At each generation g, the mutation vector $v_{i,g}$ is created based on the current parent population.

$$v_{i,g} = x_{i,g} + F_i * \left(x_{best,g}^p - x_{i,g}\right) + F_i * \left(x_{r_1,g} - x_{r_2,g}\right), \tag{7}$$

where the indices r_0, r_1, and r_2 are distinct integers uniformly chosen from the set $\{1, 2, \ldots, NP\}$; $x_{best,g}^p$ is chosen randomly as one of the top $100p\%$ individuals in the current population, with the probability $p \in (0, 1]$; and F_i is the mutation factor of the individual x_i that will be regenerated at each generation by the adaptation mechanism.

Crossover: a binomial crossover operation is adopted to generate the final offspring vector $u_{i,g} = u_{1,i,g}, u_{2,i,g}, \ldots, u_{D,i,g}$.

$$u_{j,i,g} = \begin{cases} v_{j,i,g}, & if \ \text{rand}(0,1) \leqslant CR_i \ or \ j = j_{rand}, \\ x_{j,i,g}, & otherwise, \end{cases} \tag{8}$$

where rand$(0, 1)$ is a uniform random number on the interval $[0, 1]$. $j_{rand} = \text{rand int}(1, D)$ is an integer randomly extracted from the set $[1, 2, \ldots, D-1, D]$, where each individual i has its own crossover probability CR_i. The crossover probability $CR_i \in [0, 1]$ approximately corresponds to the fraction of vector components inherited from the mutation vector.

Selection: the selection operation compares the parent vector $x_{i,g}$ with the trial vector $u_{i,g}$ according to their fitness values $f(\cdot)$, and it chooses the better vector for the next generation. For example, if given a minimization problem, the selected vector is generated by the following equation:

$$x_{i,g+1} = \begin{cases} u_{i,g}, & if \ f\left(u_{i,g}\right) < f\left(x_{i,g}\right), \\ x_{i,g}, & otherwise. \end{cases} \tag{9}$$

In addition, if the trial vector $u_{i,g}$ is better than the parent vector $x_{i,g}$, the control parameters F_i and CR_i of the individual are called a successful mutation factor and a successful crossover probability, respectively.

Parameter adaptation: Better controlling the parameter values can result in individuals that have greater possibility to survive, and hence, these values should be retained in the next generation. At each generation g, the crossover rate CR_i is formed independently according to a normal distribution of mean μ_{CR} and standard deviation 0.1 and then normalized to the range $[0, 1]$, which can be described as follows:

$$CR_i = \text{rand}\, n_i\left(\mu_{CR}, 0.1\right), \tag{10}$$

where S_{CR} is the set that records all successful crossover rates CR_i at generation g. The initial value of μ_{CR} is set as 0.5; then, it is updated by the following equation at the end of each generation:

$$\mu_{CR} = (1 - c) * \mu_{CR} + c * mean_A\left(S_{CR}\right), \tag{11}$$

where c is a positive constant in the interval $[0, 1]$ and $mean_A(S_{CR})$ represents the arithmetic mean of the agents in S_{CR}.

Similarly, the mutation factor F_i is also independently generated according to a Cauchy distribution with location

parameter μ_F and scale parameter 0.1, subsequently being normalized to [0, 1]. This can be expressed as follows:

$$F_i = \text{rand}\, c_i \left(\mu_F, 0.1\right). \tag{12}$$

Furthermore, the set S_F contains all the successful mutation factors in generation g. The initial value of μ_F of the Cauchy distribution is set to 0.5, and then, they are updated at the end of each generation by the following equation:

$$\mu_F = (1 - c) * \mu_F + c * \frac{\sum_{F \in S_F} F^2}{\sum_{F \in S_F} F}. \tag{13}$$

4. Application to Bankruptcy Classification

4.1. Bankruptcy Dataset Description. To evaluate the performance of the EPNN, both benchmark and application-oriented databases are adopted in our experiments. Each option has its advantages and disadvantages. The benchmark database allows future experiments to make extensive comparisons among different prediction models, but it cannot represent current socioeconomic statuses. Thus, the experiments may be out of date and lead to meaningless conclusions. In contrast, the application-oriented database is capable of addressing real-world problems, but it is difficult to employ for further comparison. Therefore, it is generally better to employ a mixture of both benchmark and application-oriented databases [51]. This study adopts a Qualitative Bankruptcy dataset from the UCI Machine Learning Database Repository and a Distress dataset from the Kaggle dataset to draw a significant and meaningful conclusion. In this paper, it is assumed that the state of a company's financial situation is expressed through a qualitative variable, such as the binary variable, where "1" represents a financially sound company and "0" denotes a company falling into bankruptcy.

The Qualitative Bankruptcy dataset is from the UCI repository, which has been applied successfully for bankruptcy classification in several previous works in the literature. The dataset is composed of 250 instances based on 6 attributes, with each corresponding to qualitative parameters concerning bankruptcy, namely, industrial risk, management risk, financial flexibility, credibility, competitiveness, and operating risk. The output has two classes of nominal types, which describe the instances as "Bankruptcy" (107 cases) or "Non-bankruptcy" (143 cases). The Distress dataset is from the Kaggle dataset and can be found in https://www.kaggle.com/shebrahimi/financial-distress. This dataset addresses financial distress prediction for a sampling of companies. The first column represents the sample companies, which include 422 companies. The second column shows different time periods to which the data belong. The time series length varies between 1 and 14 for each company. The third column, named the target variable, is the "Financial Distress". If this value is higher than -0.50, the company should be considered as healthy; otherwise, it is regarded as financially distressed. The fourth-to-last column denotes the features, which are denoted x_1 to x_{83}; they represent some financial and nonfinancial characteristics of the sample companies. These features belong to the previous period, which should be used to predict whether the company will be financially distressed (classification). Until now, there has been no relevant literature adopting these datasets to solve the problem of bankruptcy prediction.

4.2. Data Preprocessing. Generally, data preprocessing is an initial and basic step of data analysis. Because artificial neural networks require that every data sample be expressed as a real number vector, we need to change the nominal attributes of the data samples into numerical values before inputting them into the classifier [52].

There are no missing values in the Qualitative Bankruptcy dataset, but all the attributes are nominal. This dataset includes 250 samples, and each sample possesses 6 features. The 6 features are all represented by 3 labels: "P", "A", and "N". We convert the qualitative features into the values 1, 2, and 3, respectively.

The original Distress dataset is an extensive dataset; it includes 422 companies, and each company behaves differently in different time series. Moreover, this dataset is imbalanced and skewed; there are 136 financially distressed companies against 286 healthy ones, 136 firm-year observations are financially distressed, while 3546 firm-year observations are healthy. To make the structure of the distress dataset under observation be similar to the Qualitative Bankruptcy dataset, we perform some preprocessing. First, all the distressed companies are chosen from time series period 1 to period 14, and the total number of distressed companies is 126. In each time series period, 15 healthy companies are selected randomly, and the number of healthy companies is 210. Thus, there are 336 samples remaining in the newly generated dataset. Because each company presents 83 features, this dataset remains relatively large. We have adopted the minimal-redundancy-maximal-relevance (mRMR) criterion to generate the feature selection. The mRMR criterion offers an excellent way to maximize the dependence of the results on the input features by combining the max-relevance criterion with the min-redundancy criterion. Moreover, mRMR can not only enhance the appropriate feature selection but also achieve high classification accuracy and high computation speeds [53]. The max-relevance mechanism of mRMR is defined as follows:

$$\max D\left(S, c\right), D = \frac{1}{|S|} \sum_{x_i, x_j \in S} I\left(x_i, c\right), \tag{14}$$

where S is the set that both contains m individual features x_i and has the most considerable dependency on the target class c. If the features selected according to the max-relevance criterion are of high redundancy, there exists a large dependency among these features. To increase the respective class-discriminative power, the minimal redundancy (min-redundancy) condition is added to select noninteracting features [54],

$$\min D\left(S\right), R = \frac{1}{|S|^2} \sum_{x_i, x_j \in S} I\left(x_i, x_j\right). \tag{15}$$

Table 1: Parameter levels in EPNN of the Qualitative Bankruptcy data set.

D	K	θ_{soma}
8, 10, 12, 14	3, 5, 7, 10	0.1, 0.3, 0.5, 0.7

Table 2: $L_{16}(4^3)$ orthogonal array and factor assignment of the Qualitative Bankruptcy data set.

Expe. NO./Parameter	D	K	θ_{soma}	Testing accuracy
1	8	3	0.1	99.52 ± 1.00
2	8	5	0.3	99.01 ± 1.61
3	8	7	0.5	99.15 ± 1.26
4	8	10	0.7	99.01 ± 1.24
5	**10**	**3**	**0.3**	**99.57 ± 0.55**
6	10	5	0.1	99.23 ± 1.02
7	10	7	0.9	99.12 ± 1.32
8	10	10	0.7	99.12 ± 1.06
9	12	3	0.5	99.15 ± 1.28
10	12	5	0.7	99.23 ± 1.27
11	12	7	0.1	98.88 ± 1.83
12	12	10	0.3	99.12 ± 1.41
13	14	3	0.7	98.64 ± 2.10
14	14	5	0.5	99.15 ± 1.15
15	14	7	0.3	99.07 ± 1.56
16	14	10	0.1	98.93 ± 1.43

mRMR combines the two constraints through the operator $\Phi(D, R)$ by adopting a simple form to optimize D and R simultaneously.

$$\max \Phi(D, R), \Phi = D - R. \tag{16}$$

Using mRMR, we sort the features of this dataset. x_{43}, x_{75}, x_{52}, x_{45}, x_{80}, x_{64}, x_{60}, x_{48}, x_{34}, and x_{76} are the first ten max-dependent and min-redundant features and are used in our experiments. The updated Distress dataset includes 336 samples, where each sample has 10 features.

4.3. Optimal Parameter Setting. To realize a specific accuracy rate and achieve fast convergence in the training dataset, an optimal set of parameters must be selected. The Taguchi method is employed to decrease the number of experimental runs using orthogonal arrays [55]. Under this method, the time cost, human effort required, and material requirements can also be effectively controlled in our simulation. Selecting the orthogonal arrays that are proper for the simulation is a vital step. First, three parameters, D, K, θ_{soma}, are considered to be important in the EPNN. D denotes the branch number of the dendritic layer, K represents a parameter of the sigmoid function in the synaptic layer, and θ_{soma} denotes the threshold of the soma. Tables 1 and 3 show the ranges of parameter values of the two datasets. There are 3 parameter trials, and each parameter contains 4 values. The $L_{16}(4^3)$ orthogonal arrays of the two datasets are presented in Tables 2 and 4. To obtain a reliable average testing accuracy, each experiment is repeated 30 times. The population sizes are set to 50, and the maximum number of generations is set to 1000. The accuracy rate results of the Qualitative Bankruptcy dataset and Distress dataset are shown in Tables 2 and 4.

Table 3: Parameter levels in EPNN of the Distress credit data set.

D	K	θ_{soma}
12, 15, 18, 21	3, 5, 7, 10	0.3, 0.5, 0.7, 0.9

From Table 2, it is obvious that the highest classification accuracy of the Qualitative Bankruptcy dataset is achieved by the combination of the parameters $D = 10, K = 3$, and $\theta_{soma} = 0.3$. In addition, from Table 4, the best performance of the Distress dataset is $D = 18, K = 10$, and $\theta_{soma} = 0.5$. These parameter sets are reasonable for obtaining acceptable performance, and they are optimal for further comparisons with other algorithms.

In addition, because both MLP and PNN are adopted as the competitors in our experiment, several other parameters for these algorithms are considered cautiously. Table 5 lists the relevant parameters.

Next, to make a fair comparison, the performances of the EPNN, MLP, and PNN should be compared with an approximately equal number of thresholds and weights. In addition, the learning rate of the PNN and MLP trained by using the back-error propagation algorithm is set to 0.01. The number of dendrites in the PNN should be defined in this simulation, as should the number of hidden layers and neurons of the MLP. The MLP's parameter number depends mainly on the number of neurons in the hidden layer. Thus, we denote MLP as a D-dimensional vector, which consists

Table 4: $L_{16}(4^3)$ orthogonal array and factor assignment of the Distress data set.

Expe. NO./Parameter	D	K	θ_{soma}	Testing accuracy
1	12	3	0.3	76.25 ± 4.04
2	12	5	0.5	75.73 ± 2.64
3	12	7	0.7	75.69 ± 3.14
4	12	10	0.9	74.78 ± 2.77
5	15	3	0.5	76.15 ± 3.24
6	15	5	0.3	76.31 ± 3.66
7	15	7	0.9	76.11 ± 3.81
8	15	10	0.7	75.83 ± 2.27
9	18	3	0.7	75.06 ± 2.93
10	18	5	0.9	75.10 ± 3.60
11	18	7	0.3	75.48 ± 3.02
12	**18**	**10**	**0.5**	**76.41 ± 3.23**
13	21	3	0.9	75.20 ± 2.42
14	21	5	0.7	75.08 ± 3.60
15	21	7	0.5	75.42 ± 3.53
16	21	10	0.3	76.27 ± 2.83

Table 5: Initial Parameters of the algorithms in comparison.

Methods	Relative Parameters	Parameters' values
EPNN	popSize	50
	Max-gen	1000
	D	10(Bankruptcy),18(Distress)
	K	3(Bankruptcy),10(Distress)
	θ_{soma}	0.3(Bankruptcy),0.5(Distress)
PNN	Max-gen	1000
	lr	0.01
	D	10(Bankruptcy),18(Distress)
	K	3(Bankruptcy),10(Distress)
	θ_{soma}	0.3(Bankruptcy),0.5(Distress)
MLP	epoch	1000
	lr	0.01

of weights and biases. The dimension number D can be calculated as follows:

$$D = (Input \times Hidden) + (Hidden \times Output) + Hidden_{bias} + Output_{bias}, \quad (17)$$

where $Input$, $Hidden$, and $Output$ denote the neuron numbers in the input, hidden, and output layers of the MLP, respectively. $Hidden_{bias}$ and $Output_{bias}$ represent the bias in the hidden and output layer [56].

Meanwhile, in the PNN and EPNN, the synaptic input number is $Input$, and the dendritic branch number is $Branch$. The dimension number D of the PNN and EPNN can be calculated in the following equation:

$$D = 2 * Input \times Branch. \quad (18)$$

The structural description of the MLP and EPNN is shown in Table 6. Both models have nearly equal parameter numbers for the two datasets.

To evaluate the performance of the classification methods, each dataset is separated randomly into two subsets: a training set and a testing set. The training subset is used to set up the classification model, and the testing dataset is adopted to test the model's accuracy. The splitting strategy is significantly relevant to achieve reliable model evaluation because the case data are usually very scarce. According to prior experiments, a larger training set results in a better classifier [57]. In this simulation, 50% of the samples are chosen randomly for training, while the remaining 50% is for testing to guarantee high test accuracy. The average value of the classification accuracy rate over 30 runs is regarded as the overall classification performance.

4.4. Performance Measures. In general, most performance evaluation metrics attempt to estimate how well the learned model predicts the correct class of new input samples; however, different metrics yield different orderings of model performances [58]. The classification accuracy has been by far

Table 6: Structures of EPNN and MLP for the Qualitative Bankruptcy and Distress data set.

Dataset	Model	No. of input	No. of Branch Hidden node	No. of output	No. of adjusted parameters
Qualitative Bankruptcy	EPNN	6	10	1	120
	MLP	6	15	1	121
Distress	EPNN	10	18	1	360
	MLP	10	30	1	361

Table 7: Contingency matrix of prediction results.

Hypothesis class	Real class	
	Positive	Negative
Positive	True Positive (TP)	False Positive (FP)
Negative	False Negative (FN)	True Negative (TN)

the most frequently adopted indicator of performance in the literature [51]. Sensitivity and specificity can be highlighted as straightforward indices. The AUC does not implicitly assume equal misclassification costs, and it corresponds to the most preferred score calculated as the empirical probability that a randomly chosen positive observation ranks above a randomly chosen negative sample [59]. Hence, the overall accuracy rate, sensitivity, specificity, and AUC are used to construct the performance evaluation system.

Table 7 demonstrates that the result of a classifier can be measured by a 2-dimensional contingency matrix. The accuracy rate is the critical indicator in evaluating the classification algorithms; another indicator of accuracy analysis is related to the possibility of misclassifying bankruptcies. The classification accuracy rate is measured by the following equation:

$$Accuracy\ rate = \frac{TP + TN}{TP + TN + FP + FN}(\%), \quad (19)$$

where TP, TN, FP, and FN represent true positive, true negative, false positive, and false negative, respectively. True positive (TP) means that the company is detected as healthy, and the teacher target label is healthy as well. True negative (TN) represents that the input and the teacher target label are detected as unhealthy simultaneously. False positive (FP) shows that the input is detected as healthy, whereas the teacher target label is unhealthy. False negative (FN) denotes that input is detected unhealthy, and the teacher target label shows the opposite,

$$Sensitivity = \frac{TP}{TP + FN}(\%), \quad (20)$$

$$Specificity = \frac{TN}{TN + FP}(\%), \quad (21)$$

$$AUC\,(\%) = \frac{1}{2}\left(\frac{TP}{TP + FN} + \frac{TN}{TN + FP}\right) \times 100\,(\%), \quad (22)$$

where the sensitivity and specificity are called the true positive rate and true negative rate, respectively. Sensitivity measures the percentage of real positives that are identified correctly. This metric shows how successfully a classifier can identify regular records, which means that the companies are healthy in terms of bankruptcy analysis. Therefore, financial institutions can achieve correct and efficient analysis by adopting a classifier with a higher sensitivity. Specificity represents how successfully a classifier can distinguish abnormal records; i.e., it is the proportion of true negatives. Hence, a higher specificity can help financial institutions reduce the possibility of misclassifying healthy companies. AUC represents the ratio of companies that are not in danger of bankruptcy. In other words, a score of 100% indicates that two classes can be correctly discriminated by the classifier, whereas a score of 50% indicates that the classifier has an insignificant ability to classify companies correctly.

In addition to comparing different classification algorithms, the convergence performances of the two models, EPNN and MLP, are compared. When the mean squared error (MSE) achieves a predetermined minimum value, the learning tends to be completed. The training error is calculated as shown in (21),

$$MSE = \frac{1}{R}\sum_{a=1}^{R}\left[\frac{1}{S}\sum_{b=1}^{S}\left(E_{ab} - O_{ab}\right)^2\right], \quad (23)$$

where E_{ab} and O_{ab} are the desired output and the actual output, respectively; S represents the number of instances applied for training; and R denotes the number of simulation runs.

4.5. Performance Comparison. For a fair comparison, EPNN and PNN are equipped with the same parameters, and the learning rates of the PNN and MLP are the same. All three algorithms are run 30 times independently. Tables 8 and 13 show the classification performances obtained by these algorithms. In addition, to detect the significant differences among the results, a nonparametric test called the Wilcoxon rank-sum test [60] is adopted in this study. A review in the literature has summarized that it is preferable to use a nonparametric test instead of a parametric test to achieve high

Table 8: Experimental results for Qualitative Bankruptcy data set.

Method	Average accuracy (AVE±STD)	Wilcoxon rank-sum test (p-value)	Sensitivity	Specificity	AUC (AVE±STD)	Wilcoxon rank-sum test (p-value)
EPNN	**99.57±0.5452**	N/A	**0.9976**	**0.9775**	**0.9984±0.0055**	N/A
PNN	98.11±1.2690	$2.82E^{-05}$	0.9852	0.9185	0.9871±0.0195	$1.20E^{-03}$
MLP	94.59±3.3330	$8.76E^{-07}$	0.9393	0.9552	0.9868±0.0147	$1.50E^{-05}$

Table 9: Initialization parameters of other classification methods.

Method	Parameter	Value
KNN	k	5
RBF	Spread of radial basis function	1.0
RF	Number of trees	50
DT	Children	1
SVM	Type of kernel function	Radial basis function
	Degree	3
DA	Prior probabilities	uniform

statistical accuracy, especially when the sample size is small [61]. Thus, the calculated p-values of the Wilcoxon rank-sum test are presented in the tables as well. In the following comparison tables, N/A represents "Not Applicable", which indicates that the relevant algorithm cannot be compared with itself in the test. In our experiments, the significance level is set to 5%. As a matter of routine, there is substantial evidence to reject the null hypothesis when p-values are less than 0.05. In order to further verify the superiority of EPNN, we compare it with other popularly applied classification methods, such as K-nearest neighbor algorithm (KNN) [62], radical basis function (RBF) [63], random forest (RF) [64], decision tree (DT) [65], support vector machine (SVM) [66], and discriminant analysis (DA) [67]. Each method runs 30 times independently. The initial parameters of each method are summarized in Table 9.

4.5.1. Qualitative Bankruptcy Dataset. For the Qualitative Bankruptcy dataset, as shown in Table 8, the proposed EPNN obtains an average testing accuracy of 99.57%, which is higher than the 98.11% obtained by the PNN and the 94.59% obtained by MLP. In addition, the statistical results also show that the EPNN achieves significantly better performances than the PNN and MLP. Moreover, the EPNN also performs better than the PNN and MLP in terms of sensitivity and specificity. A comparatively higher sensitivity value indicates the powerful capability of the EPNN in identifying the companies that are healthy. Higher specificity values represent the EPNN's ability to not misclassify an unhealthy company. Furthermore, the convergence rate of the three models, EPNN, PNN, and MLP, are also compared in our experiments in Figure 4. As observed, the EPNN achieves the highest convergence rate compared to the PNN and MLP. Moreover, Figure 5 shows the ROC of the EPNN, the PNN, and MLP. The corresponding AUC value of the EPNN is larger than that of the PNN and of MLP. It is emphasized that the EPNN is superior to the PNN and MLP in solving the Qualitative Bankruptcy dataset problem.

Table 10: Classification performance of EPNN in comparison with other classification methods on Qualitative Bankruptcy dataset.

Method	Average accuracy MEAN±STD
KNN	99.12±0.6759
RBF	96.75±2.2425
RF	98.99±1.2238
DT	97.92±1.1810
SVM	99.39±0.5009
DA	99.52±0.4506
EPNN	**99.57±0.5452**

In addition, the performances' comparisons between the EPNN and other commonly used classifiers are presented in Table 10. It is clear that the EPNN also shows its superiority in the average accuracy rate on Qualitative Bankruptcy dataset.

Since there are many proposed methods which are adopted to classify Qualitative Bankruptcy dataset in the relative literatures, we summarized the classification performances and compared them with that of the EPNN. Specifically, Table 11 presents some single classification methods and Table 12 demonstrates some hybrid classification methods, respectively. From Table 11, it can be observed that the accuracy rate of the EPNN is only slightly less than RBF-based SVM, Ant-miner, and Random Forest. As Table 12 shows, compared with other hybrid classification methods, the average accuracy rate of the EPNN is only slightly lower than the hybrid logistic regression-naive bayes. Thus, it can be concluded that although the EPNN adopted 50%-50% train-to-test ratio, it has still presented a competitive performance on the Qualitative Bankruptcy dataset. And it is worth mentioning that hybrid classification methods are not always superior to single classification methods based on the above experimental results.

Table 11: Classification accuracies for Qualitative Bankruptcy dataset obtained by other single classification methods in relative literatures.

Author (year)	Method (train-to-test ratios)	Average accuracy (%)
Kalyan Nagaraj, Amulyashree Sridhar (2015) [40]	Logistic Regression (2/3-1/3)	97.2
Kalyan Nagaraj, Amulyashree Sridhar (2015) [40]	Rotation Forest (2/3-1/3)	97.4
Kalyan Nagaraj, Amulyashree Sridhar (2015) [40]	Naive Bayes (2/3-1/3)	98.3
Kalyan Nagaraj, Amulyashree Sridhar (2015) [40]	RBF-based SVM (2/3-1/3)	99.6
E. K. Kornoushenko (2017) [41]	Nearest Neighborhood (50%-50%)	97.6
J.Uthayakumar et. al. (2017) [42]	Ant-Miner (10 fold cross validation)	100
J.Uthayakumar et. al. (2017) [42]	Logistic Regression (10 fold cross validation)	99.2
J.Uthayakumar et. al. (2017) [42]	MLP (10 fold cross validation)	99.2
J.Uthayakumar et. al. (2017) [42]	Random Forest (10 fold cross validation)	100
J.Uthayakumar et. al. (2017) [42]	Radical Basis Function (10 fold cross validation)	99.2
J.Uthayakumar et. al. (2018) [43]	Genetic Algorithm (Not Mentioned)	71.48
J.Uthayakumar et. al. (2018) [43]	Ant Colony Algorithm (Not Mentioned)	83.05
Our Method (2019)	EPNN (50%-50%)	99.57
Our Method (2019)	EPNN(10 fold cross validation)	99.68

Table 12: Classification accuracies for Qualitative Bankruptcy dataset obtained by other hybrid classification methods in relative literatures.

Author (year)	Method (train-to-test ratios)	Average accuracy (%)
Yi Tan et. al. (2016) [44]	Hybrid logistic regression-naive bayes (90%-10%))	99.64
Nanxi Wang (2017) [45]	Neural network model with robust logistic regression (50%-50%)	69.44
Nanxi Wang (2017) [45]	Neural network model with inductive learning algorithm (50%-50%)	89.7
Nanxi Wang (2017) [45]	Neural network model with genetic algorithm (50%-50%)	94
Nanxi Wang (2017) [45]	Neural network model with neural networks without dropout (50%-50%)	90.3
Nanxi Wang (2017) [45]	Neural network model with SVM (50%-50%)	98.67
Nanxi Wang (2017) [45]	Neural network model with decision tree (50%-50%)	99.33
J.Uthayakumar et. al. (2018) [43]	Genetic ant colony algorithm (Not mentioned)	91.32
J.Uthayakumar et. al. (2018) [43]	Fitness-scaling chaotic Genetic ant colony algorithm (Not mentioned)	92.14
J.Uthayakumar et. al. (2018) [43]	Improved K-means clustering and fitness-scaling chaotic genetic ant colony algorithm (Not mentioned)	97.93
Our Method (2019)	EPNN (50%-50%)	99.57
Our Method (2019)	EPNN (10 fold cross validation)	99.68

Table 13: Experimental results for Distress data set.

Method	Average accuracy (AVE±STD)	Wilcoxon rank-sum test (p-value)	Sensitivity	Specificity	AUC (AVE±STD)	Wilcoxon rank-sum test (p-value)
EPNN	**76.41±3.2320**	N/A	0.8108	0.6546	**0.7762±0.0478**	N/A
PNN	54.03±19.636	$2.43E^{-05}$	0.3166	**0.8353**	0.6338±0.1848	$6.67E^{-04}$
MLP	66.15±5.4324	$9.09E^{-07}$	**0.8475**	0.3575	0.6814±0.0906	$2.13E^{-05}$

4.5.2. Distress Dataset. Concerning the Distress dataset, as shown in Table 13, the EPNN acquires an average testing accuracy of 76.41%, which is higher than the 54.03% obtained by the PNN and the 66.15% accuracy obtained by MLP. In addition, the p-values of the Wilcoxon test show there are significance differences between EPNN and the other two methods. Although not all the sensitivity and specificity values of the EPNN are larger than those of the PNN and MLP, the PNN performs worse on sensitivity, and MLP is the worst on specificity. The EPNN achieves better performances on both sensitivity and specificity. The convergence curves of the EPNN, the PNN, and MLP are compared in Figure 6. This figure shows that the EPNN achieves the highest convergence rate compared to the PNN and MLP. In addition, Figure 7 presents the ROC of the EPNN, PNN, and MLP. In addition, the corresponding AUC value of the EPNN is larger than that of the PNN and MLP. This implies that, compared with the PNN and MLP, the EPNN is a more effective classifier on the Distress dataset. Besides, the classification performance of the EPNN is compared with KNN, RBF, RF, DT, SVM, and DA, and the corresponding results are presented in Table 14. As Table 14 illustrated, EPNN performs better than all the other

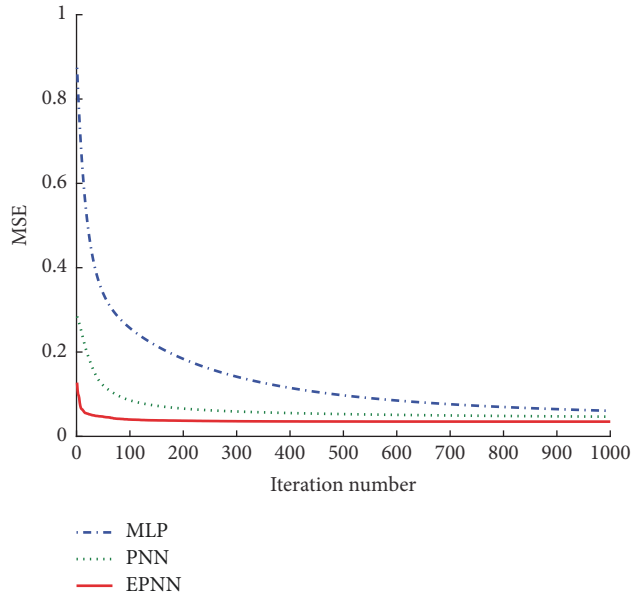

Figure 4: The convergency curve of the Qualitative Bankruptcy data set.

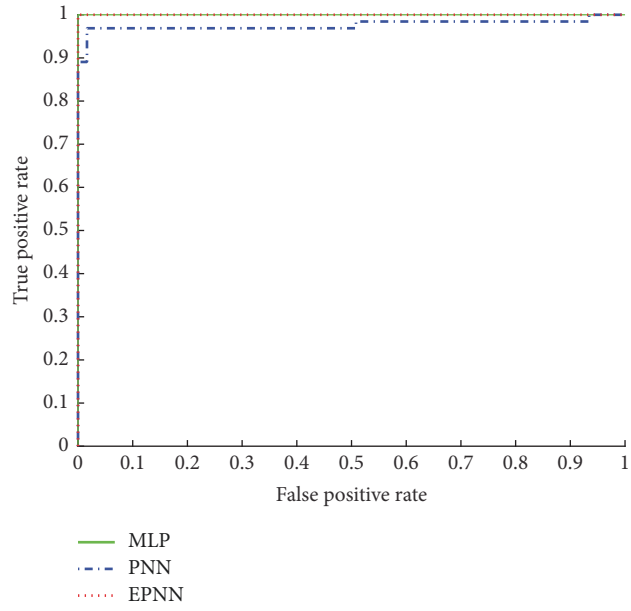

Figure 5: The ROC of the Qualitative Bankruptcy data set.

classification methods except RF. Since there are no other classification methods applied to classify Distress dataset in the literature, horizontal comparison can not be fulfilled for this dataset.

4.6. Dendrite Morphology Reconstruction

4.6.1. The Ultimate Synaptic and Dendritic Morphology. As mentioned above, the EPNN can implement synaptic pruning and dendritic pruning during the training process. Thus, superfluous synapses and dendrites can be removed, and then, a simplified and distinct topological morphology is formed for each problem. Figure 8 shows the particular dendritic structure of the EPNN on the Qualitative Bankruptcy dataset after learning. The unnecessary dendrites (Branch 2, Branch 4 and Branch 9) of the PNN are presented in Figure 9, and superfluous synaptic layers are provided in Figure 10. Finally, the simplified structural morphology is described in Figure 11. It can be observed that 7 dendritic branches and 4 features are reserved in the structure. This means that the features x_1 and x_2 are not crucial for the EPNN and

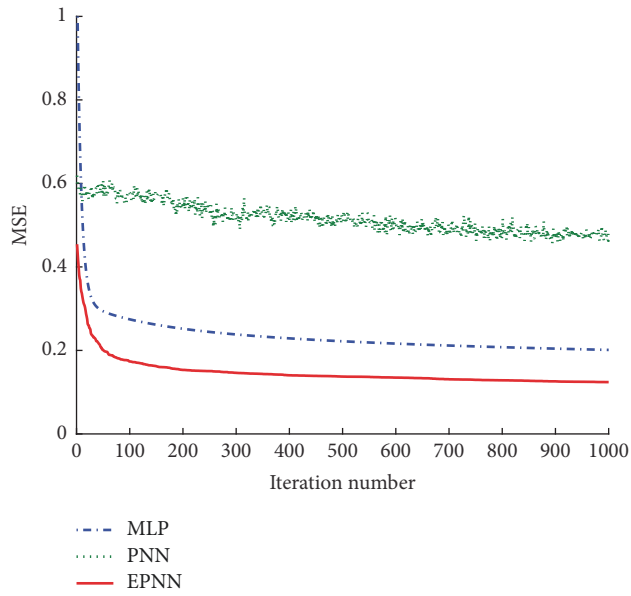

Figure 6: The convergency curve of the Distress data set.

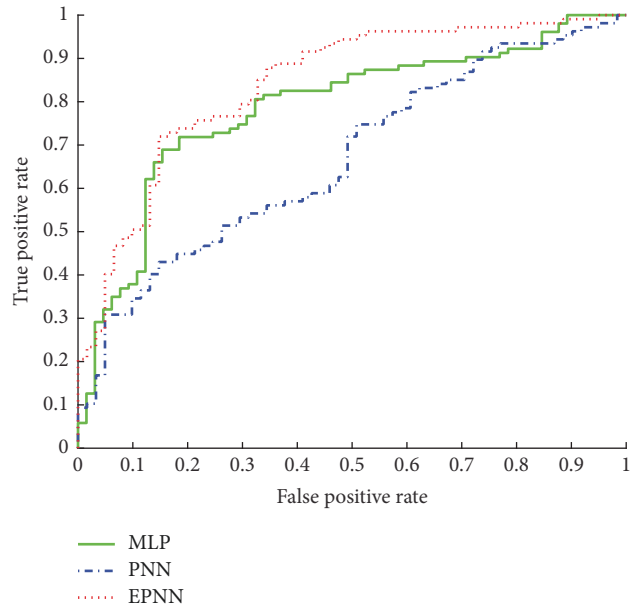

Figure 7: The ROC of the Distress data set.

have no contribution to solving the Qualitative Bankruptcy dataset problem. In addition, Figure 12 illustrates the unique dendritic morphology of the EPNN on the Distress dataset after learning. Figure 13 shows that all ineffective dendrites are removed, and Figure 14 rules out all ineffective synaptic layers. Thus, the final structural morphology is presented in Figure 15. Only four branches of the dendrites are remaining, and the feature x_6 is removed. As summarized in Table 15, it can be observed that synaptic pruning and dendritic pruning mechanism can largely simplify the structure of the EPNN. Thus, it is able to speed up the bankruptcy prediction analysis by the simplified EPNN obviously.

4.6.2. The Simplified Logic Circuit of the After-Learning Morphology. In addition to the neural pruning function, the other function worth emphasizing is that the simplified structure of the EPNN can form an approximate logic circuit applicable to hardware implementations. Figures 16 and 17 present further simplified logic circuits of the structural morphologies. We use an analog-to-digital converter, which can be regarded as a "comparator", to compare the input with the threshold θ. Once the input x_i is less than the threshold θ, the "comparator" will output 0; otherwise, it will output 1. Using the logic circuits, we can classify the companies into bankrupt and not-bankrupt on both the Qualitative

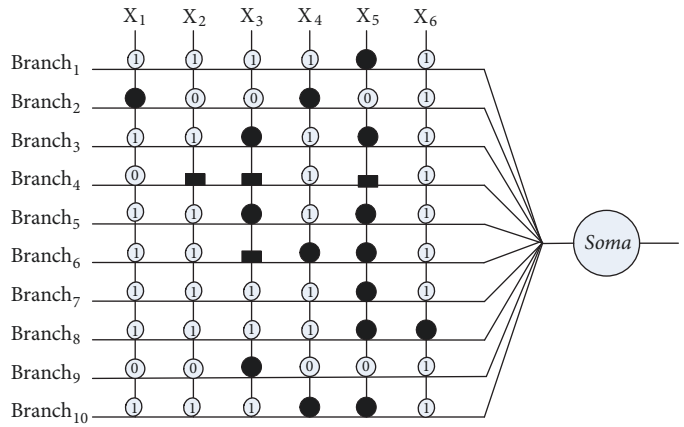

Figure 8: The dendritic morphology of the Qualitative Bankruptcy data set after learning.

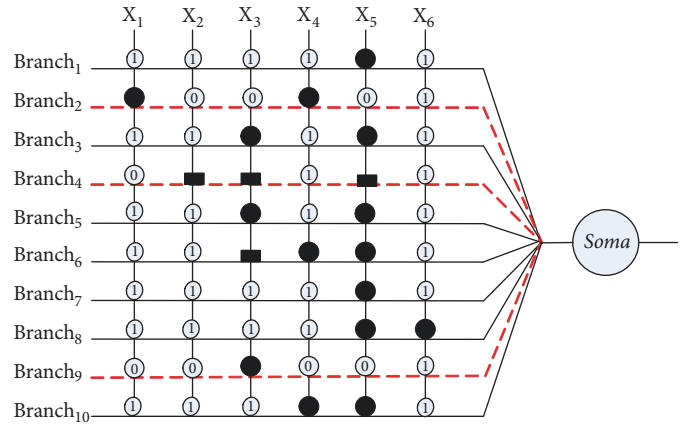

Figure 9: The dendritic morphology of the Qualitative Bankruptcy data set after dendritic pruning.

Table 14: Classification performance of EPNN in comparison with other classification methods on Distress dataset.

Method	Average accuracy MEAN±STD
KNN	76.35±3.1065
RBF	49.15±14.4445
RF	**81.25±2.2907**
DT	76.11±2.8599
SVM	62.42±2.5047
DA	74.74±3.4414
EPNN	76.41±3.2320

Bankruptcy dataset and Distress dataset. The accuracies of the logic circuits are shown in Table 16. Clearly, the test accuracies of the logic circuits do not decrease and are higher than those of the EPNN. Note that the logic circuits in Figures 16 and 17 are selected randomly from an arbitrarily chosen experiment, and they are not unique to each problem. Forming a logic circuit can further increase the classification speed of the EPNN, thereby creating a more powerful method for the prediction of financial bankruptcy.

5. Conclusion

Artificial intelligence algorithms, such as neural network methods, are being widely applied in bankruptcy analysis. In this paper, we introduce a more realistic neural model called the EPNN to facilitate bankruptcy analysis. This technique adopts the JADE algorithm to train the model to obtain satisfactory classification performances. In contrast with the PNN and MLP, the proposed EPNN performs the best in terms of the average accuracy and AUC on both benchmark and application-oriented datasets, namely, the Qualitative Bankruptcy dataset and the Distress dataset. In addition, compared with other classification methods such as KNN, RBF, RF, DT, SVM, and DA, the EPNN also provides competitive and satisfactory classification performances. Note that the neuronal pruning mechanism is an important aspect of the EPNN. After synaptic pruning and dendritic pruning, the number of input features in both datasets is reduced, and

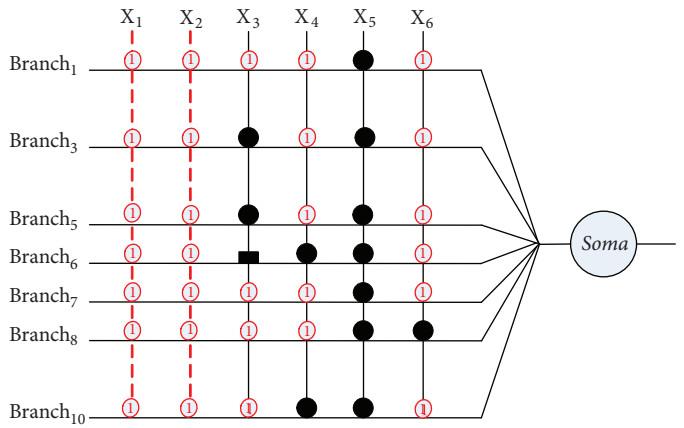

Figure 10: The dendritic morphology of the Qualitative Bankruptcy data set after synaptic pruning.

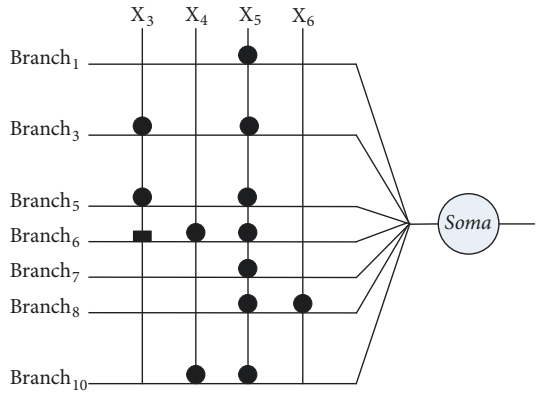

Figure 11: The final structure of the Qualitative Bankruptcy data set.

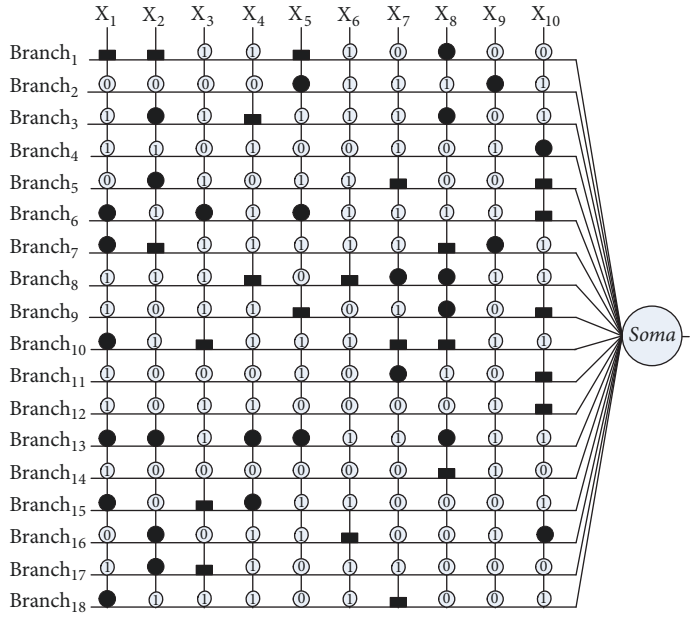

Figure 12: The dendritic morphology of the Distress data set after learning.

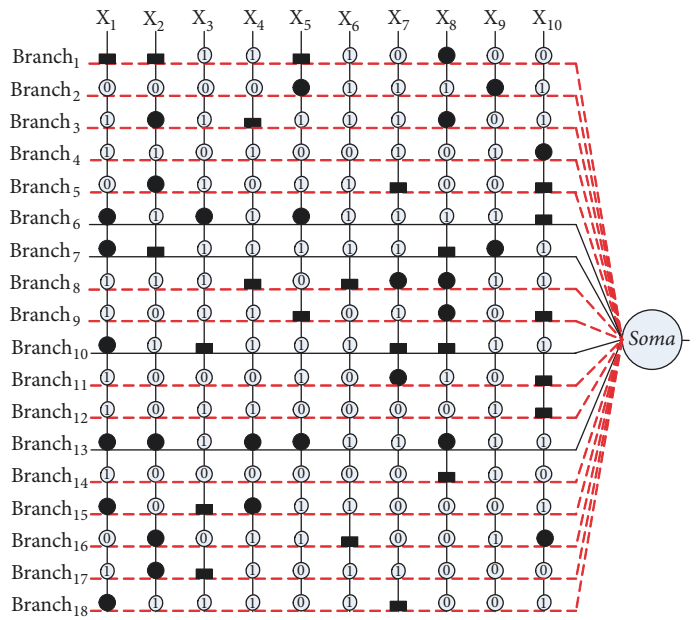

Figure 13: The dendritic morphology of the Distress data set after dendritic pruning.

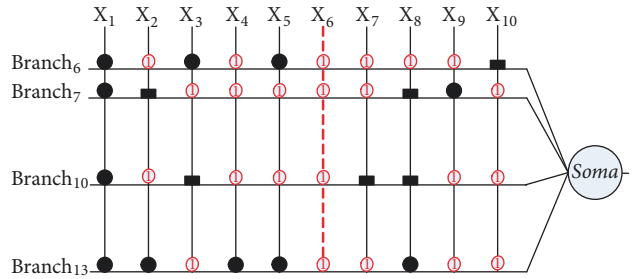

Figure 14: The dendritic morphology of the Distress data set after synaptic pruning.

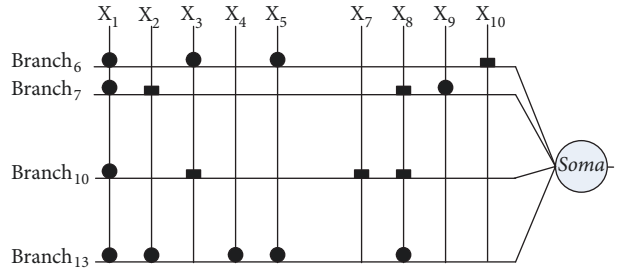

Figure 15: The final structure of the Distress data set.

Table 15: Model structure comparison between Qualitative Bankruptcy and Distress data sets.

Dataset	feature input		Dendritic layer		Adjusted weight	
	initial value	selected value	initial value	selected value	initial value	selected value
Qualitative Bankruptcy	6	4	10	7	120	56
Distress	10	9	18	4	360	72

Table 16: Verification of the logic circuits.

Dataset	Accuracy of the EPNN	Accuracy of the logic circuit
Qualitative Bankruptcy	99.57	100
Distress	76.41	79.17

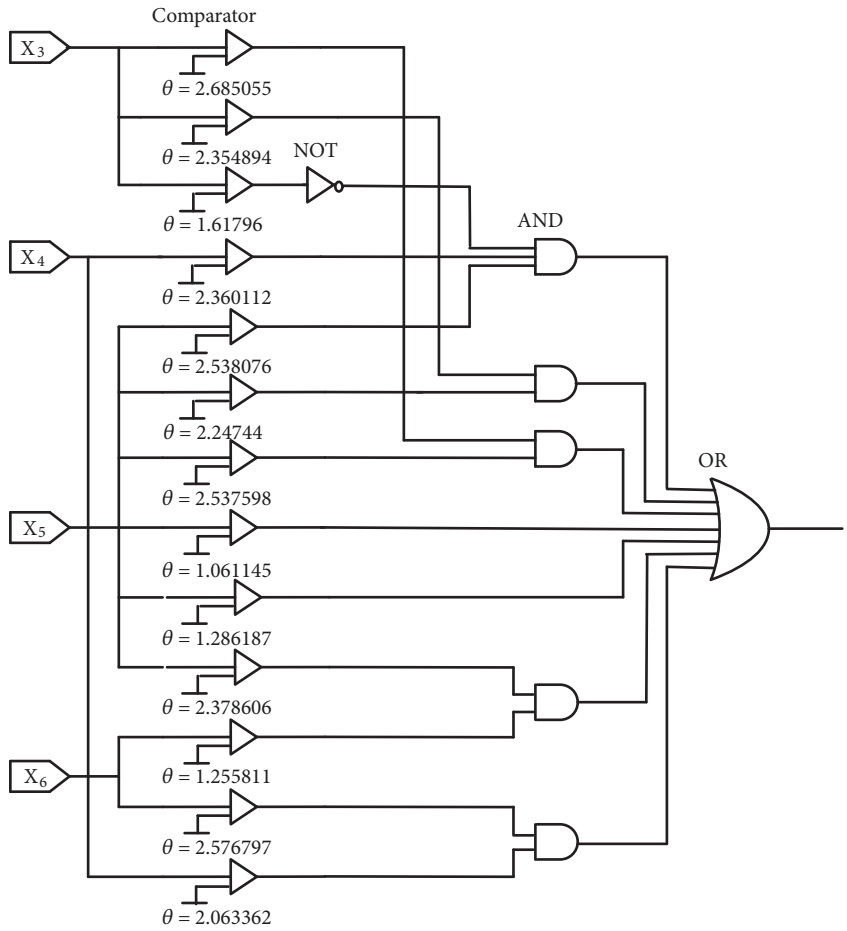

Figure 16: The logic circuit of the Qualitative Bankruptcy data set.

the structure of the neural network is simplified. Moreover, the simplified structural morphology can form a logic circuit which can also be employed as a powerful tool to solve bankruptcy prediction problems. Thus, the contribution of this paper can be summarized from three aspects: First, we provide a comprehensive study by comparing different classification models in terms of bankruptcy prediction problems. Although many novel algorithms are continually emerging, a large proportion of approaches still only focus on the bankruptcy prediction model's ability to improve the classification accuracy. Compared with some other models, the EPNN possesses a certain advantage with respect to average accuracy and AUC. Second, the EPNN can implement synaptic and dendritic pruning to realize pattern extraction and reconstruct a more compact neuronal morphology. The EPNN has a large initial neuronal topology, which makes it not very sensitive to its initial conditions, but it can utilize neuronal pruning after learning, which increases the efficiency of the neural network, speeds up the convergence, avoids becoming trapped in local minima, and reduces the operation time and computational cost. Third, the simplified models can be replaced by logic circuits, which can increase the classification accuracy and be easily implemented on the hardware. Therefore, these findings provide details and offer insight into technical development for understanding and tracing the operating mechanisms and construction of single neurons. In addition, the results also imply that the proposed EPNN classifier possesses an excellent potential to be applied in other binary classification problems. The EPNN makes it possible to draw standard profiles of the failing companies and provide a theoretical contribution to the phenomenon of bankruptcy. It is believed that the EPNN will be suitable for not only bankruptcy prediction but also other fields of application within the scope of financial analysis such as performance analysis.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

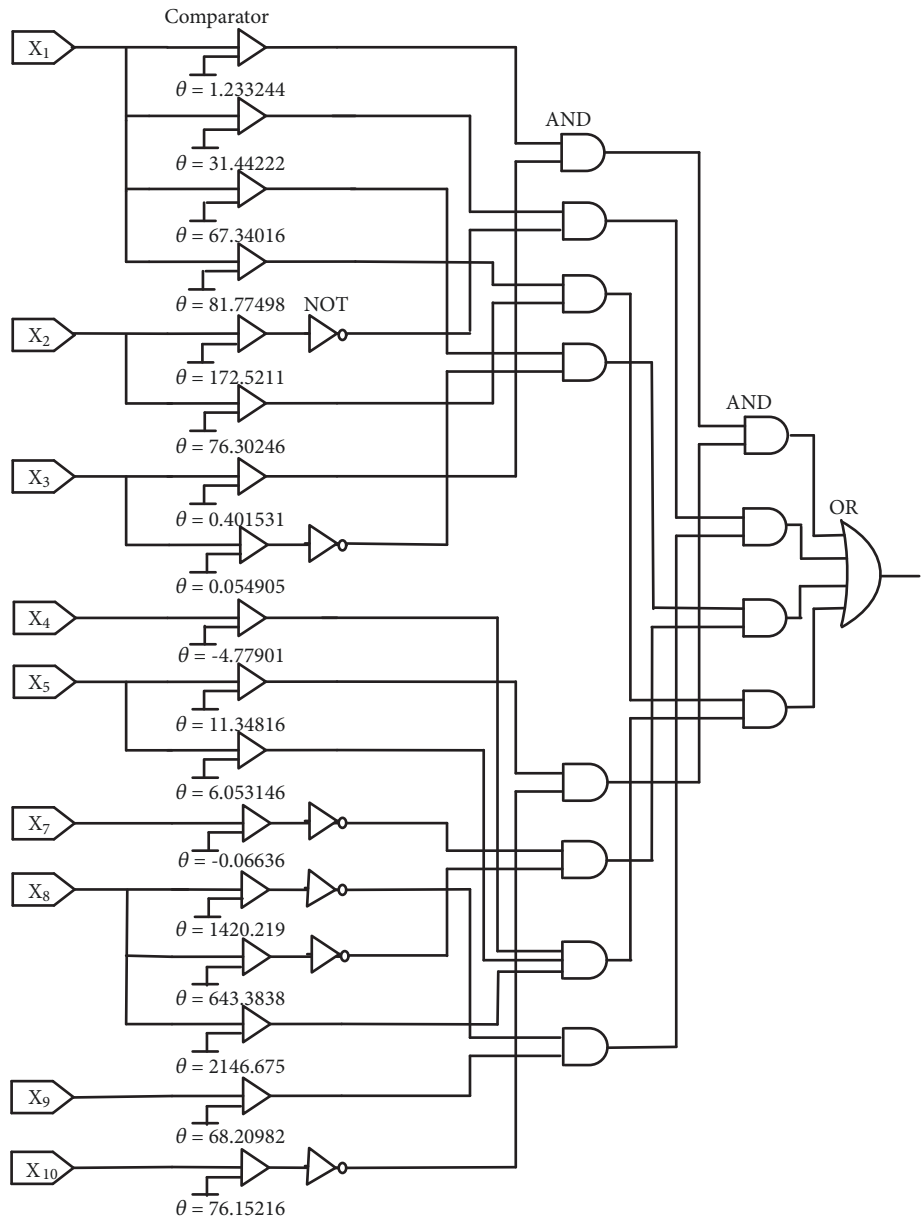

Figure 17: The logic circuit of the Distress data set.

References

[1] E. I. Altman, "Financial ratios, discriminant analysis and the prediction of corporate bankruptcy," *The Journal of Finance*, vol. 23, no. 4, pp. 589–609, 1968.

[2] J. A. Ohlson, "Financial ratios and the probabilistic prediction of bankruptcy," *Journal of Accounting Research*, vol. 18, no. 1, pp. 109–131, 1980.

[3] J. Begley, J. Ming, and S. Watts, "Bankruptcy classification errors in the 1980s: an empirical analysis of altman's and ohlson's models," *Review of Accounting Studies*, vol. 1, no. 4, pp. 267–284, 1996.

[4] J. Kruppa, A. Schwarz, G. Arminger, and A. Ziegler, "Consumer credit risk: Individual probability estimates using machine learning," *Expert Systems with Applications*, vol. 40, no. 13, pp. 5125–5131, 2013.

[5] R. Pal, K. Kupka, A. P. Aneja, and J. Militky, "Business health characterization: A hybrid regression and support vector machine analysis," *Expert Systems with Applications*, vol. 49, pp. 48–59, 2016.

[6] A. E. Celik and Y. Karatepe, "Evaluating and forecasting banking crises through neural network models: An application for Turkish banking sector," *Expert Systems with Applications*, vol. 33, no. 4, pp. 809–815, 2007.

[7] M. Perez, "Artificial neural networks and bankruptcy forecasting: A state of the art," *Neural Computing and Applications*, vol. 15, no. 2, pp. 154–163, 2006.

[8] H. A. Abdou and J. Pointon, "Credit scoring, statistical techniques and evaluation criteria: a review of the literature," *Intelligent Systems in Accounting, Finance and Management*, vol. 18, no. 2-3, pp. 59–88, 2011.

[9] M. D. Odom and R. Sharda, "A neural network model for bankruptcy prediction," in *Proceedings of the International Joint Conference on Neural Prediction Networks (IJCNN '90)*, pp. 163–168, IEEE, 1990.

[10] T. E. McKee and M. Greenstein, "Predicting bankruptcy using recursive partitioning and a realistically proportioned data set," *Journal of Forecasting*, vol. 19, no. 3, pp. 219–230, 2000.

[11] G. Zhang, M. Y. Hu, B. E. Patuwo, and D. C. Indro, "Artificial neural networks in bankruptcy prediction: general framework and cross-validation analysis," *European Journal of Operational Research*, vol. 116, no. 1, pp. 16–32, 1999.

[12] T. Bellotti and J. Crook, "Support vector machines for credit scoring and discovery of significant features," *Expert Systems with Applications*, vol. 36, no. 2, pp. 3302–3308, 2009.

[13] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, no. 2, pp. 251–257, 1991.

[14] K. Hornik, "Some new results on neural network approximation," *Neural Networks*, vol. 6, no. 8, pp. 1069–1072, 1993.

[15] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[16] W. Chen and Y. Du, "Using neural networks and data mining techniques for the financial distress prediction model," *Expert Systems with Applications*, vol. 36, no. 2, pp. 4075–4086, 2009.

[17] H. Wang, Q. Xu, and L. Zhou, "Large unbalanced credit scoring using lasso-logistic regression ensemble," *PLoS ONE*, vol. 10, no. 2, Article ID e0117844, 2015.

[18] C. Koch, T. Poggio, and V. Torre, "Nonlinear interactions in a dendritic tree: Localization, timing, and role in information processing," *Proceedings of the National Acadamy of Sciences of the United States of America*, vol. 80, no. 9 I, pp. 2799–2802, 1983.

[19] C. Koch, T. Poggio, and V. Torre, "Retinal ganglion cells: a functional interpretation of dendritic morphology," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 298, no. 1090, pp. 227–263, 1982.

[20] C. Koch, "Computation and the single neuron," *Nature*, vol. 385, no. 6613, pp. 207–210, 1997.

[21] J. Ji, S. Gao, J. Cheng, Z. Tang, and Y. Todo, "An approximate logic neuron model with a dendritic structure," *Neurocomputing*, vol. 173, pp. 1775–1783, 2016.

[22] Y. Tang, J. Ji, S. Gao et al., "A pruning neural network model in credit classification analysis," *Computational Intelligence and Neuroscience*, vol. 2018, Article ID 9390410, 22 pages, 2018.

[23] S. Piramuthu, M. J. Shaw, and J. A. Gentry, "A classification approach using multi-layered neural networks," *Decision Support Systems*, vol. 11, no. 5, pp. 509–525, 1994.

[24] S. Gao, M. Zhou, Y. Wang, J. Cheng, H. Yachi, and J. Wang, "Dendritic neuron model with effective learning algorithms for classification, approximation, and prediction," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 2, pp. 601–614, 2019.

[25] R. R. Trippi and E. Turban, *Neural Networks in Finance and Investing: Using Artificial Intelligence to Improve Real World Performance*, McGraw-Hill, Inc, 1992.

[26] T. P. Vogl, J. K. Mangis, A. K. Rigler, W. T. Zink, and D. L. Alkon, "Accelerating the convergence of the back-propagation method," *Biological Cybernetics*, vol. 59, no. 4-5, pp. 257–263, 1988.

[27] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Let a biogeography-based optimizer train your multi-layer perceptron," *Information Sciences*, vol. 269, pp. 188–209, 2014.

[28] M. Gori and A. Tesi, "On the problem of local minima in backpropagation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 1, pp. 76–86, 1992.

[29] Y. Lee, S. Oh, and M. W. Kim, "An analysis of premature saturation in back propagation learning," *Neural Networks*, vol. 6, no. 5, pp. 719–728, 1993.

[30] M. S. Hung and J. W. Denton, "Training neural networks with the GRG2 nonlinear optimizer," *European Journal of Operational Research*, vol. 69, no. 1, pp. 83–91, 1993.

[31] M. K. Weir, "A method for self-determination of adaptive learning rates in back propagation," *Neural Networks*, vol. 4, no. 3, pp. 371–379, 1991.

[32] A. Van Ooyen and B. Nienhuis, "Improving the convergence of the back-propagation algorithm," *Neural Networks*, vol. 5, no. 3, pp. 465–471, 1992.

[33] J. Q. Zhang and A. C. Sanderson, "JADE: adaptive differential evolution with optional external archive," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 945–958, 2009.

[34] S. Das and P. N. Suganthan, "Differential evolution: a survey of the state-of-the-art," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 4–31, 2011.

[35] M. Morey, S. K. Yee, T. Herman, A. Nern, E. Blanco, and S. L. Zipursky, "Coordinate control of synaptic-layer specificity and rhodopsins in photoreceptor neurons," *Nature*, vol. 456, no. 7223, pp. 795–799, 2008.

[36] C. Koch, *Biophysics of Computation: Information Processing in Single Neurons*, Oxford University Press, 2004.

[37] E. Salinas and L. F. Abbott, "A model of multiplicative neural responses in parietal cortex," *Proceedings of the National Acadamy of Sciences of the United States of America*, vol. 93, no. 21, pp. 11956–11961, 1996.

[38] F. Gabbiani, H. G. Krapp, C. Koch, and G. Laurent, "Multiplicative computation in a visual neuron sensitive to looming," *Nature*, vol. 420, no. 6913, pp. 320–324, 2002.

[39] J. Sietsma and R. J. Dow, "Neural net pruning-why and how," in *Proceedings of 1993 IEEE International Conference on Neural Networks (ICNN '93)*, vol. 1, pp. 325–333, IEEE, San Diego, CA, USA, 1988.

[40] K. Nagaraj and A. Sridhar, "A predictive system for detection of bankruptcy using machine learning techniques," https://arxiv.org/abs/1502.03601.

[41] E. K. Kornoushenko, "Classification algorithm based on pairwise comparison of features," *Automation and Remote Control*, vol. 78, no. 11, pp. 2062–2074, 2017.

[42] J. Uthayakumar, T. Vengattaraman, and P. Dhavachelvan, "Swarm intelligence based classification rule induction (cri) framework for qualitative and quantitative approach: An application of bankruptcy prediction and credit risk analysis," *Journal of King Saud University-Computer and Information Sciences*, 2017.

[43] J. Uthayakumar, N. Metawa, K. Shankar, and S. K. Lakshmanaprabu, "Intelligent hybrid model for financial crisis prediction using machine learning techniques," *Journal of Information Systems and e-Business Management*, pp. 1–29, 2018.

[44] Y. Tan, P. P. Shenoy, M. W. Chan, and P. M. Romberg, "On construction of hybrid logistic regression-nave bayes model for classification," in *Proceedings of the Conference on Probabilistic Graphical Models*, pp. 523–534, 2016.

[45] N. Wang, "Bankruptcy prediction using machine learning," *Journal of Mathematical Finance*, vol. 7, no. 04, p. 908, 2017.

[46] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[47] Q. Lin, S. Liu, Q. Zhu et al., "Particle swarm optimization with a balanceable fitness estimation for many-objective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 32–46, 2018.

[48] Q. Lin, S. Liu, K. Wong et al., "A clustering-based evolutionary algorithm for many-objective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 3, pp. 391–405, 2019.

[49] A. E. Eiben and S. K. Smit, "Parameter tuning for configuring and analyzing evolutionary algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 19–31, 2011.

[50] A. Zamuda and J. Brest, "Self-adaptive control parameters? randomization frequency and propagations in differential evolution," *Swarm and Evolutionary Computation*, vol. 25, pp. 72–99, 2015.

[51] V. García, A. I. Marqués, and J. S. Sánchez, "An insight into the experimental design for credit risk and corporate bankruptcy prediction systems," *Journal of Intelligent Information Systems*, vol. 44, no. 1, pp. 159–189, 2015.

[52] C.-M. Wang and Y.-F. Huang, "Evolutionary-based feature selection approaches with new criteria for data mining: A case study of credit approval data," *Expert Systems with Applications*, vol. 36, no. 3, pp. 5900–5908, 2009.

[53] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005.

[54] C. Ding and H. Peng, "Minimum redundancy feature selection from microarray gene expression data," *Journal of Bioinformatics and Computational Biology*, vol. 3, no. 2, pp. 185–205, 2005.

[55] R. Jugulum, S. Taguchi et al., *Computer-Based Robust Engineering: Essentials for DFSS*, ASQ Quality Press, 2004.

[56] Z. Beheshti, S. M. H. Shamsuddin, E. Beheshti, and S. S. Yuhaniz, "Enhancement of artificial neural network learning using centripetal accelerated particle swarm optimization for medical diseases diagnosis," *Soft Computing*, vol. 18, no. 11, pp. 2253–2270, 2013.

[57] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, *Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann, 2016.

[58] D. J. Hand, "Assessing the performance of classification methods," *International Statistical Review*, vol. 80, no. 3, pp. 400–414, 2012.

[59] N. M. Kiefer, "Default estimation for low-default portfolios," *Journal of Empirical Finance*, vol. 16, no. 1, pp. 164–173, 2009.

[60] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945.

[61] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[62] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[63] Y. Lu, N. Sundararajan, and P. Saratchandran, "Performance evaluation of a sequential minimal Radial Basis Function (RBF) neural network learning algorithm," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 9, no. 2, pp. 308–318, 1998.

[64] V. Svetnik, A. Liaw, C. Tong, J. Christopher Culberson, R. P. Sheridan, and B. P. Feuston, "Random forest: a classification and regression tool for compound classification and QSAR modeling," *Journal of Chemical Information and Computer Sciences*, vol. 43, no. 6, pp. 1947–1958, 2003.

[65] S. R. Safavian and D. Landgrebe, "A survey of decision tree classifier methodology," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 21, no. 3, pp. 660–674, 1991.

[66] M. M. Adankon and M. Cheriet, "Support vector machine," *Encyclopedia of Biometrics*, pp. 1303–1308, 2009.

[67] S. Mika, G. Ratsch, J. Weston, B. Scholkopf, and K.-R. Muller, "Fisher discriminant analysis with kernels," in *Proceedings of the 9th IEEE Signal Processing Society Workshop on Neural Networks for Signal Processing (NNSP '99)*, pp. 41–48, Madison, Wis, USA, August 1999.