STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **6**, December 2018, pp 588–599.
Published online in International Academic Press (www.IAPress.org)

Applying Bayesian Decision Theory in RBF Neural Network to Improve Network precision in Data Classification

Nader Rezazadeh *

Department of Computer / Science and Research Branch of, Islamic Azad University, Tehran, Iran

Abstract One of the common tools used for classification of data is RBF neural network. The lack of connectivity of features in each layer in the structure of neural networks such as the RBF neural network causes the values of the features to not be multiplied, and the action and dependency of the values of a feature on other features not to be considered in the classification or regression process. The most important reason for the lack of connectivity among the features can be considered as the problem of learning weights. This research tries to use the multiplication of values of event probability of features to improve the efficiency of data classification in the RBF neural network based on the reasons mentioned above through classification style in the Bayesian decision theory. Moreover, the linear weight coefficients at the final layer of the RBF neural network are used to determine the importance of the feature event in the final decision. This research tries to use the capabilities of the RBF neural network in assigning event probabilities to the values of input features based on data centers. The presence of linear weight in the final layer makes learning weights improve the classification. Empirical experiments show good results.

Keywords Bayesian decision theory, Neural network, RBF neural network

AMS 2010 subject classifications 68T05

DOI: 10.19139/soic.v6i4.498

1. Introduction

Neural networks are one of the common tools in regression and data classification [1]. The neural network has the capability to create required nonlinear relationship between the inputs and outputs. This relationship can be achieved through learning weights. As features of neural networks are not connected at each layer, they cause that the direct relationship among the features values not to be considered in determining the final output. Therefore, the dependency among the features is not considered in determining the output. The most important reason for the lack of connection among the input features is the difficulty of learning weights [2]. Therefore, connections among the features are deleted. One of the important examples in this regard is the reduction of connections among the features in the Boltzmann machine and its reduction to the restricted Boltzmann machine [4]. One of the methods, which consider the values of the features and their dependencies in the classification, is the Bayesian decision theory [5]. Bayesian decision theory is an appropriate method for decision making based on the dependency among the features and considering the interaction of features in the final output. However, modeling and calculating the degree of dependency among all features is difficult when the number of features is very high .On the other hand, the features take different values, making it difficult use Bayesian decision theory. This algorithm can consider all dependencies. However, by increasing the number of features, creating a Bayesian network becomes impossible to model the dependency relationship among all the features. Thus, in addition to modeling the relationship among

*Correspondence to: Department of Computer / Science and Research Branch of, Islamic Azad University, Tehran, Iran (Email: Naderrezazadeh1984@gmail.com)

ISSN 2310-5070 (online) ISSN 2311-004X (print)

the features and multiplying the values of features, this research tries to model the dependency among them using the learnable coefficients. The linear weights determined based on desired output in each class minimize the error caused by the dependency among the features as much as possible .Since the coefficients are linear, they dot have the complexities of multilayer neural networks for learning weight. In this research, back propagation algorithm was changed in such a way that it can be easily used to learn weights in the proposed neural network structure. The proposed method shows good results for a particular data set. Section 2 reviews the literature of the today's common neural networks used in classification. Section 3 presents the methodology and Section 4 presents the results of the experiments

2. Literature Review

Neural networks are a popular tool in classification of data due to their efficiency and relatively easy learning. The need to establish a relationship between desired input and output has caused relatively diverse structures to be defined for it. In this section, some types of neural networks used in research experiments are briefly reviewed.

2.1. Backpropagation neural network

In Backpropagation neural network [6], the error rate is propagated from output side to lower layers to calculate the output error in each layer. Then, learning weight is performed based on parameters such as learning error, desired output, real output, activation function derivative, and input value. Thus, it is appropriate for learning with supervisor. Backpropagation neural networks can be implemented on flat neural networks. Backpropagation calculates a gradient that is needed in the calculation of the weights to be used in the network [7]. In the learning process, backpropagation is commonly used by the gradient descent optimization algorithm. backpropagation adjusts the weight of neurons by calculating the gradient of the loss function [8]. Given the reasons mentioned, this algorithm was used to learn linear weight coefficients in this paper.

2.2. RBF neural networks

The RBF neural network[9] is one of the common tools in data classification[10]. In RBF neural networks, the classification is performed with the help of the value of membership function set on each feature. Data centers play an important role in the mentioned network. Given the use of membership functions and data centers, the statistical data center plays an important role in determining the output in the RBF neural network.

2.3. Naive Bayesian Classifier

The main purpose of the bayesian classifiers is to minimize the probability of miss classification[11]. naive Bayes classifiers are a family of probabilistic classifiers. The Naive Bayesian classifier is suitable for problems with high and independent input variables[12] So if the input variables are not independent, it's best to review the classification approach.

2.4. Deep Belief Network

Deep Belief Networks(DBN)[13] are new generation of neural networks, performing classification and regression with regard to higher level features [14]. DBN extract the higher level features using the statistical approach. These networks are developed by putting restricted Boltzmann machines together [15]. The restricted Boltzmann machines are a kind of Boltzmann machine, in which the relationship between the features of the same level has been deleted. Figure 1 shows the structure of the Boltzmann machine and the restricted Boltzmann machine for the visible input vector v and the hidden layer vector h [16]. In Figure (1), L is the symmetric weight between the visible layer and J is the symmetric weight between hidden layer features. W is the symmetric weight between the visible and hidden layer .The problem of Boltzmann machine weights has caused the connection among the features of the same level to be deleted [16]. Despite the deletion of connectivity among the same level features in the hidden and visible layer, the restricted Boltzmann machine is considered as an efficient tool in feature engineering.

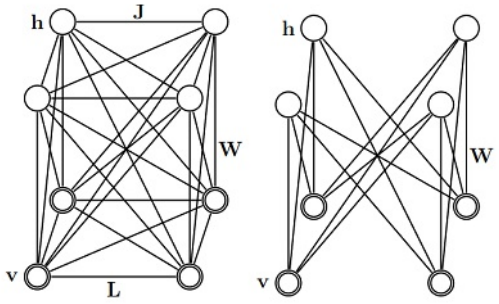

Figure 1. Boltzmann machine structure (left side), restricted Boltzmann machine structure (right side)

3. Methodology

In the methodology section, Bayesian decision theory and multiplication of the values of the features are used to determine the class of data using direct multiplication of data features values. Using this approach, the relationship among the features and their interactions in the final decision can be considered. On the other hand, given the different values received by the feature, we try to make decision to determine the probability of the event of a feature, based on the center of data set assigned to one feature. Thus, RBF neural network was used due to its ability to use radial functions. The combination of these two approaches was used in the methodology of this paper. The reason to present this method is the lack of relationship among the features in the input layer. In order to show the lack of relationship among the features in the neural network, the m_{th} output in the n_{th} layer is shown by equation (1) [17].

$$Y_m^n = f(W_{1,m}^{(n)} X_1^{(n-1)} + W_{2,m}^{(n)} X_2^{(n-1)} + \cdots + W_{N_{(n-1)},m}^{(n)} X_{N_(n-1)}^{(n-1)}), \tag{1}$$

In equation (1), $N_{(n-1)}$is equal to the number of nodes in the layer $(n-1)$.$w_{N_{(n-1)},m}^{(n)}$ is equal to the connection weight of nude m to nude N in the n layer $(n-1)$.$X_{(n-1)}^{(n-1)}$is equal to value of nude N in the layer $(n-1)$. Equation (2) can be written in terms of the network input in the layer as in equation (2).

$$\begin{aligned} Y_m^n \quad &= f(W_{1,m}^{(n)} f(W_{1,1}^{(n-1)} X_1^{(n-2)} + \ldots + W_{N_{(n-2)},1}^{(n-1)} X_{N_{(n-2)}}^{(n-2)})) + W_{N_{n-1},m}^{(n)} f(W_{1,N_{(n-1)}}^{(n-1)} X_1^{(n-2)} \\ &\quad + \ldots + W_{N_{(n-2)},N_{(n-1)}}^{(n-1)} X_{N(n-2)}^{(n-2)}). \end{aligned} \tag{2}$$

Given the max and min terms in the equation above, it can be rewritten as Equation (3).

$$Y_m^n = f(\sum_{i=1}^{N_{(n-1)}} \prod_{j=1}^{N_{(n-2)}} W_{i,j}^{(N_{(n-1)})} f(\sum_{i=1}^{N(n-2)} \prod_{j=1}^{N_{(n-3)}} W_{i,j}^{(N_{(n-2)})} \ldots f(\sum_{i=1}^{N_2} \prod_{j=1}^{N_{(1)}} W_{i,j}^{(2)} X_j^{(1)}))) \tag{3}$$

$$\rightarrow Y_m^n = f(\sum_{i=1}^{N_{(n-1)}} \prod_{j=1}^{N_{(n-2)}} W_{i,j}^{(N_{(n-1)})} X_j^{(N_{(n-2)})}).$$

In fact, equation(2) can be written as a set of $N_{(n-1)} \times N_{(n-2)}$ max term. Each max term includes multiplication of a set of weights in each feature. Accordingly, in total, no two features are multiplied. As seen, features are not multiplied and the direct effect of the values of the features is not considered in the calculation of the output. Even in the learning process, the value of gradient is calculated separately based on the ratio of changes in weight

coefficients to each attribute. Its reason is the lack of relationship among the features of each layer in the structure of the neural network. The problem of learning weights for network is the most important reason for deleting the relationship among the features of each layer. As in Boltzmann machine, there is a problem of learning [16]. This paper presents a method for learning the weights of the network and calculating the output based on the direct multiplication of the features. In the proposed method, network weights are set using parameters and statistical computations.

3.1. The implementation of the Bayesian decision theory

Before introducing the proposed method, the way of using the membership functions and weight coefficients in Bayesian classification is described in this paper. In a Bayesian classifier, assuming the presence of input set $X = \{X_1, \ldots, X_n\}$, the probability value of the class S^i event is equal to Equation (4)[18]

$$P(S^i|X_1, \ldots, X_n) = \frac{P(S^i)P(X_1, \ldots, X_n|S^i)}{P(X_1, \ldots, X_n)}. \tag{4}$$

Equation (5)can be written for Conditional probability of S^i given multiple independent events $X_1, \ldots, Xn$[19].

$$P(S^i| \bigcap_{j=1,\ldots,n} X_j) = \frac{P(S^i)P(\bigcap_{j=1,\ldots,n} X_j|S^i)}{P(\bigcap_{j=1,\ldots,n} X_j)} \tag{5}$$

$$= \frac{P(S^i)P(\bigcap_{j=1,\ldots,n} X_j|S^i)}{\prod_{j=1}^{n} P(X_j)}.$$

Assuming the independence of events $X_1, \ldots, X_n$,the expression $P(\bigcap_{j=1,\ldots,n} Xj|S^i)$ can be written as follows[20]

$$P(\bigcap_{j=1,\ldots,n} Xj|S^i) = \prod_{j=1}^{n} P(X_j|S^i). \tag{6}$$

Therefore, If the set of observations $X = \{X_1, \ldots, X_n\}$, is independent in pair, equation (7) will be applied

$$P(S^i|X_1, \ldots, X_n) = \frac{P(S^i)P(X_1|S^i) \times \ldots \times P(X_n|S^i)}{\prod_{j=1}^{n} P(X_j)}. \tag{7}$$

According to Equation (7), the most probable condition, in the case of independency of features and the absence of dependency among the features, can be achieved using Equation (8) [18]

$$i^* = Argmax_i P(S^i)P(X_1|S^i) \times \ldots \times P(X_n|S^i). \tag{8}$$

Given the different values received by each feature X_j , $1 \leq j \leq n$, Gaussian membership function can be used to implement the value of $P(X_j|S_i)$. Gaussian membership function helps model the event probability value of the feature X_j by assuming the event Si. For this purpose, base point set is required set to examine the probability of the event X_j in which the input is compared to it. The important point is that which point should be found, which always occur for a class or at least has the maximum event. Such a point can be used as the center of Gaussian membership function. The solution proposed in this paper is the data centers .With an initial preprocessing without supervisor, data centers are calculated for the class S^i and the center $C_i = \{C_1^i, , C_n^i\}$is extracted. To measure the input distance $X_j$from the center C_j^i and assigning the appropriate probability value to the input X_j, the Gaussian membership function, [21] shown in Equation(9), is used

$$\varphi(x_j, c_j^i) = \frac{1}{\sigma\sqrt{2\Pi}} e^{\frac{-[(x_j - c_j^i)^2]}{2\sigma^2}}. \tag{9}$$

In simple words, the data center of a feature, as the point that has the maximum event, takes the maximum probability value of an event. The probability of other values of a feature is determined by the proximity to the data

center. Given what was stated above, the Bayesian equation for classification can be rewritten as Equation (10)

$$P(S^i|X_1,\ldots,X_n) = \frac{P(S^i)\times\varphi(X_1,C_1^i)\times\ldots\times\varphi(X_n,C_n^i)}{\Pi_{j=1}^n\varphi(X_j)} \tag{10}$$

$$\rightarrow i^* = Argmax_i P(S^i)\times\varphi(X_1,C_1^i)\times\ldots\times\varphi(X_n,C_n^i),$$

where

$$\varphi(X_j) = \Sigma_{i=1}^m\varphi(X_j,C_j^i).$$

Parameter m is the number of classes. The value of$P(S^i)$ can be easily calculated by dividing the number of learning vectors having the class i (S^i) by total number of learning vectors. The above equation would be true if the event probability for the observations $\{X_1,\ldots,X_2\}$ is independent in pair. Thus, in the case of dependencies between observations, the above equation would have an error, depending on the degree of dependency. In the implementation, there is assumed that there is no information on the level of dependency between observations. To achieve correct probability equation, set of weight coefficients $W_i = \{W_1^i,\ldots,W_n^i\}$, due to increase the precision of the calculation $P(S^i|X_1,\ldots,X_n)$, is used, as you can see in equation (11)

$$P(S^i|X_1,\ldots,X_n) = \frac{P(S^i)\times W_1^i\varphi(X_1,c_1^i)\times\ldots\times W_n^i\varphi(X_n,C_n^i)}{\Pi_{j=1}^n\varphi(X_j)} \tag{11}$$

$$\rightarrow i^* = Argmax_i p(s^i)\times w_1^i\varphi(x_1,c_1^i)\times\ldots\times w_n^i\varphi(x_n,c_n^i).$$

The weight set $W_i = \{W_1^i,,W_n^i\}$is exclusively determined for s^i . If the value of these weight coefficients is correctly determined, computational error caused by non-determining the relationship between the event of features of each data set is reduced. As extracting the graph of the relationship between the probabilities of the event of features and increasing the number of features is very difficult, the proposed method can be useful. These linear coefficients can be learnt in the network. As only these weights are present in the classification layer, extracting the network error gradient is feasible based on weights. Thus, there are no problems of learning weight of connected features, as mentioned in the previous section.

3.2. The proposed RBF neural network structure

In the proposed method, the RBF neural network structure is changed in a way that Equation (11) can be implemented. To this purpose, the activation function was deleted. In addition, AND gate was used to multiply the event probability values of features and weight coefficients. However, to implement these changes, it is necessary to change the learning linear weights method, which is explained below. Figure (2) shows the changed RBF in order to implement Bayesian decision rules. The linear weight vector W^i is learned based on the decision of the i class, S^i. The vector of weights V_i is set so that it can guide input X_j to only $\varphi(X_j,C_j^i)$. so that its membership value is determined only based on the data center C_j^i . Thus, weights are firstly set as in Equation (12).

$$\left\{\begin{array}{ll} if\ (i==j) & V_{jk}^i = 1 \\ if\ (i\neq j) & V_{jk}^i = 0 \end{array}\right\},\ \ where\ \ 1\leq j,k\leq n. \tag{12}$$

3.3. Learning weighs algorithm

The learning algorithm for the vector set $W^i = \{W_1^i,\ldots,W_n^i\}$ has been set based on the back propagation algorithm. Due to the modification of the network structure, the most important part of the weight learning is the extraction of the output gradient of the network. In the following, how to establish compatibility between the network output and the backpropagation algorithm is explained. Learning weight in backpropagation algorithm is done through following equation [22].

$$\begin{aligned} W_j(t) &= W_j(t-1)+\Delta W_j \\ \Delta W_j &= \eta\delta_j Xj,\ \ 1\leq j\leq n \end{aligned} \tag{13}$$

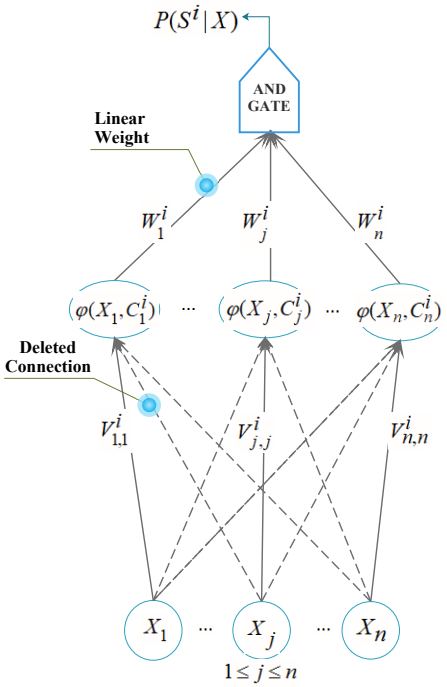

Figure 2. The changed structure of the RBF network for Bayesian classification on class S^i

Parameter n is the number of features. In Equation (13), The parameter δ_j is equal to the error of neuron j. the parameter η is equal to the learning rate .Given the assumptions of the problem, equation (13) can be written as follows

$$W_j^i(t) = W_j^i(t-1) + \eta \Delta W_j^i \phi(X_j, C_j^i), \tag{14}$$

where

$$\delta_j^i = \frac{\partial E}{\partial W_j^i}, \quad E = \frac{1}{2}(t-y)^2, \quad 1 \le j \le n, \quad 1 \le i \le m.$$

Parameter m is the number of classes. As the weights are linear per output, the learning weight process is simply performed simply as shown Equation (15). In equation (14), the gradient $\frac{\partial E}{\partial W_j^i}$ can be rewritten as

$$\frac{\partial E}{\partial W_j^i} = \frac{\partial E}{\partial y}\frac{\partial y}{\partial net}\frac{\partial net}{\partial W_j^i}. \tag{15}$$

To calculate the Bayesian probability, there is no need to use the activation function. Thus, the output of the network and the final output are same. Equation (16) can be written as follows

$$\frac{\partial E}{\partial W_j^i} = \frac{\partial E}{\partial y}\frac{\partial y}{\partial W_j^i}. \tag{16}$$

Set of linear weights should be set in a way that generates output 1 for per class i. In other words, the network should give the highest possible probability to the training data. It means that target value in equation (11) for class S^i would equal to 1. The aim of weight Learning is to achieve output 1 for training data, given that S^i occurred.

regarding to descriptions, the value of $\frac{\partial E}{\partial y}$ is calculated as Equation (17).

$$\begin{aligned}\frac{\partial E}{\partial y} &= \frac{\partial \frac{1}{2}(t-y)^2}{\partial y} \\ &= \frac{\partial \frac{1}{2}(1-y)^2}{\partial y} \\ &= (y-1)\end{aligned} \tag{17}$$

On the other hand, given the linear equation (11), the value of $\frac{\partial y}{\partial W_j^i}$ would be equal to Equation (18).

$$\frac{\partial y}{\partial W_j^i} = P(S^i)\phi(X_j, C_j^i) \prod_{k=1, k\neq j}^{n} W_k^j \phi(X_k, C_k^i) \tag{18}$$

If you do not want to use the prior probability in decision making, the value of $\frac{\partial y}{\partial W_j^i}$ would be equal to Equation (19).

$$\frac{\partial y}{\partial W_j^i} = \phi(X_j, C_j^i) \prod_{k=1, k\neq j}^{n} W_k^j \phi(X_k, C_k^i) \tag{19}$$

Given the results of Equations (16), (17) and (18), Equation(20) can be written as follows.

$$\frac{\partial E}{\partial W_j^i} = \frac{\partial E}{\partial y}\frac{\partial y}{\partial net}\frac{\partial net}{\partial W_j^i} \tag{20}$$

$$\frac{\partial E}{\partial W_j^i} = (y-1)P(S^i)\phi(X_j, C_j^i) \prod_{k=1, k\neq j}^{n} W_k^j \phi(X_k, C_k^i)$$

Thus, the final equation for adjusting weights $W_i(t) = \{W_1^i(t), \ldots, W_n^i(t)\}$ based on the error propagation rule would be equal to Equation (21).

$$W_j^i(t) = W_j^i(t-1) + \eta \left((y-1)P(S^i)\phi(X_j, C_j^i) \prod_{k=1, k\neq j}^{n} W_k^j \phi(X_k, C_k^i) \right) \tag{21}$$

where $1 \leq j \leq n, \;\; 1 \leq i \leq m$.
Similarly, Adjusting the weights without considering the prior probability is as equation (22).

$$W_j^i(t) = W_j^i(t-1) + \eta \left((y-1)\phi(X_j, C_j^i) \prod_{k=1, k\neq j}^{n} W_k^j \phi(X_k, C_k^i) \right) \tag{22}$$

where $1 \leq j \leq n, \;\; 1 \leq i \leq m$.
Adjustment of weights can continue to reach the desired error. Preferably, it is best to determine the number of training epoch. Because achieving the optimal error for the network may not be possible.

4. Experimental results

The data in the experimental results section were selected based on specific goals. The importance of the selected data is related to fact that there is high dependency between the features values and the network output. For example, in the Deal or No Deal [17] data set, if each of features has no acceptable value, no final agreement is achieved. Thus, it is necessary to consider the dependency between the values of the features of each class in the

Table 1. Datasets used in experiments

No	Dataset	Description	H
1	Student Performance[24]	Predict student performance in secondary education (high school)	0.172
2	Deal or no Deal[25]	basic information on "Deal or No Deal" games, including US, German, and Dutch TV shows and two classroom experiments	0.294

final decision, as Bayesian decision theory. The Conditional Entropy [18] criterion was used to measure the degree of dependency in an event of a feature and other features. This criterion can determine the degree of dependency of an event of a feature and other features. For this purpose, the Conditional Entropy criterion is used among the dataset samples as presented in Equation (23)[23].

$$H(X|Y) = -\sum_{x\in X} p(x) \sum_{y\in Y} p(y|x) log(p(y|x)) \tag{23}$$

The $H(Y|X)$ criterion is the amount of information required to estimate the value of Y provided that X has occurred. This can be a good criterion to measure the degree of dependency of an event of a feature and other features in determining the final output. The value of the Conditional Entropy for the two random variables X and Y is defined as Equation (24).

$$H(X_i|X_j) = -\sum_{x_i\in X_i} p(x_i) \sum_{x_j\in X_j} p(x_j|x_i) log(p(x_j|x_i)) \tag{24}$$

The mean of Conditional Entropy among all features can also be defined as Equation (25).

$$\bar{H} = \frac{\sum_{i,j} H(X_i|X_j)}{(N) \times (N-1)} \tag{25}$$

where $1 \leq i, j \leq N, i \neq j$.

Equation (25) simply represents the mean value of the dependencies in the event of the features of a dataset. This criterion is considered in the experimental results section has been considered data set 2.

In the experiments section, the process of classification was performed on each of the data sets using the proposed Bayesian RBF method, the back propagation neural network, the RBF neural network and the deep belief neural network. The number of learning weight courses was 100, and the learning rate and standard deviation for all experiments were 0.12, 0.1, respectively. Figure (3) shows the the MSE learning error during the learning on data set 1.

As shown in Figure (3), the Bayesian RBF showed a lower MSE value during the experiment. Moreover, the convergence rate of the learning error of the mentioned network was higher than that of other methods, especially the back propagation neural network. As only linear coefficients are learned in the proposed method, this convergence rate was predictable. Figure (3) shows the MSE learning error during the learning period on the data set 2.

The need for creating symmetric weight among the values of features in the data set 2 to consider the relationship between the values of the features has caused that learning process to be associated with a higher error rate and lower convergence rate. Due to the higher dependency among the features, the need for a direct multiplication of the values of features would be also greater. By solving the problem mentioned by Bayesian RBF, the network has a higher convergence rate and less learning error compared to other methods. Then, the mean precision of data classification by Bayesian RBF, RBF, back propagation, and deep belief network methods is compared on data set 1 and 2. Its results are shown in Table (2).

The proposed Bayesian RBF method could perform better than other methods used in these tests. This superiority is more evident in the percentage of precision of classification in data set 2. Given the results obtained in Table

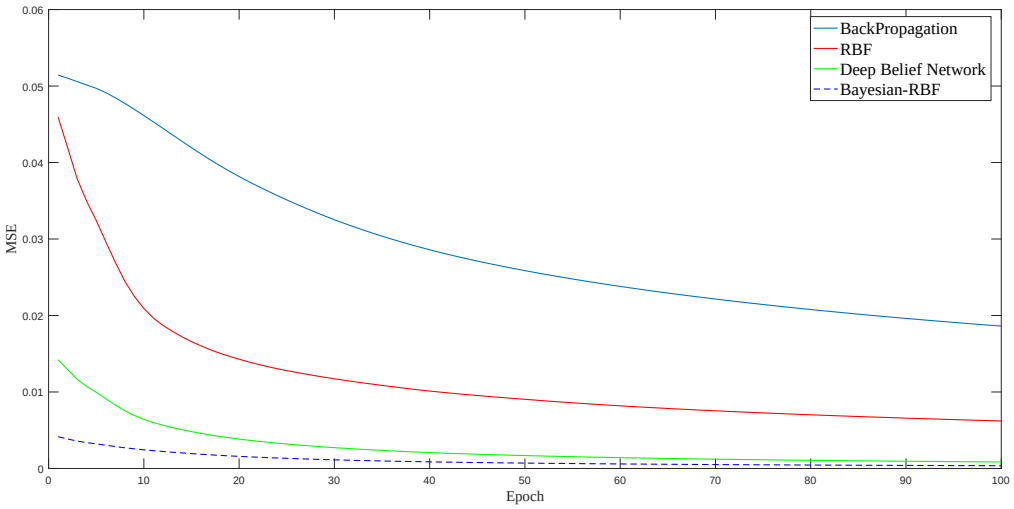

Figure 3. MSE for training Dataset 1

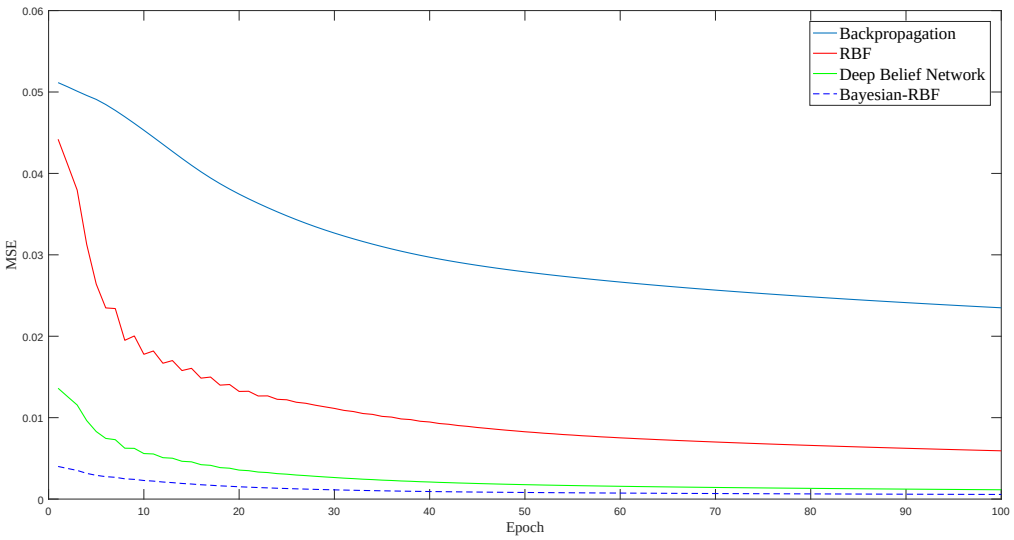

Figure 4. MSE for training Dataset 2

Table 2. Results of algorithm estimation precision

Dataset	H	Algorithm	Precision
Student Performance	0.172	Back Propagation	0.8134
Student Performance	0.172	RBF Neural Network	0.8728
Student Performance	0.172	Deep Belief Network	0.8917
Student Performance	0.172	Bayesian RBF	0.9239
Deal or no Deal	0.294	Back Propagation	0.7251
Deal or no Deal	0.294	RBF Neural Network	0.7629
Deal or no Deal	0.294	Deep Belief Network	0.8327
Deal or no Deal	0.294	Bayesian RBF	0.8724

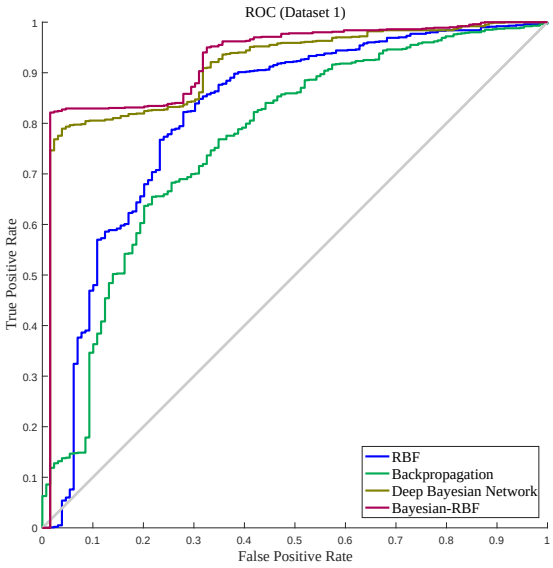

Figure 5. ROC curve of four prediction of four selected classifiers in dataset 1

Table 3. Accuracy of classifiers

Algorithm	Accuracy
Back Propagation	0.7103
RBF Neural Network	0.7361
Deep Belief Network	0.8662
Bayesian RBF	0.8704

(2), the precision of all methods for data set 2 has a significant reduction compared to that for data set 1. However, this reduction in precision in the proposed method was very lower. The most important reason for this reduction in precision was dependency among the values of features in determining the network output and the need for action terms among the features in the Deal or no data se for correct estimation.

In the following, receiver operating characteristic curve, i.e., ROC curve, had been used to, illustrates the diagnostic ability of a classifier system. But the ROC curve for the binary classifier is used. Hence, The target data values of the student's performance are divided into two main classes. Deal or no Deal Dataset has 2 classes, so there is no problem in using the ROC curve. In the following, the diagnostic ability of selected classifiers on the dataset 1 is shown using the curve of the ROC in Fig(5).

To accurately analyze the quadrilateral curves, one can use the Accuracy value corresponding to each ROC curve. The accuracy value corresponding to each curve is obtained based on the area under the curve, which brings in the figure (5)

As shown in figure (5) and table (3), Bayesian-RBF and Deep Belief Network have a higher accuracy than others. In the following, the diagnostic ability of selected classifiers on the dataset 2 is shown using the curve of the ROC in Fig(6). In the same way, The accuracy value for each ROC curve is brought in the Table (4)

In both experiments, the Bayesian-RBF neural network had a higher precision and accuracy compared to other networks. The Bayesian RBF method could perform more successfully in estimating class values by obtaining operating weights of the features. The superiority of the proposed network's accuracy is more evident in Dataset 2. This is despite the fact that the Bayesian-RBF accuracy, like other network, decreased in the second experiment

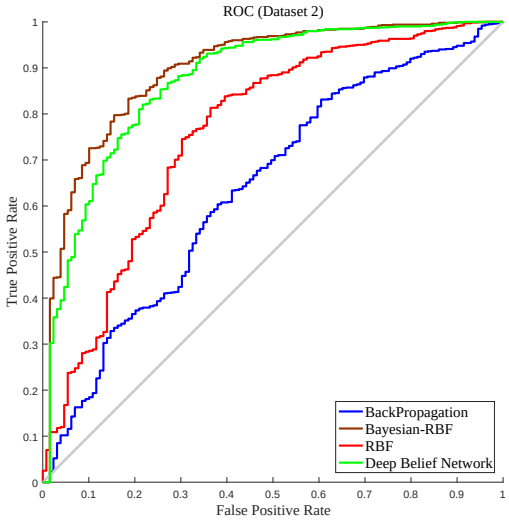

Figure 6. ROC curve of four prediction of four selected classifiers in dataset 2

Table 4. Accuracy of classifiers

Algorithm	Accuracy
Back Propagation	0.6201
RBF Neural Network	0.7112
Deep Belief Network	0.8014
Bayesian RBF	0.8412

The reason for the reduction of the accuracy of all networks in the second experiment is to increase the mean of conditional entropy between features .Increasing conditional entropy means increasing the ambiguity in the conditional probability of the feature event. The ambiguity itself reduces the accuracy of classification in all networks. the proposed method, using embedded operating weights, reduces the effects of increasing the ambiguity in the conditional probability of event. embedded weights reduce the classifier error caused by the dependency between the feature event. with the help of embedded linear weights, the maxterms of the decision-making formulas are more accurately adjusted. This adjustment is achievable by using the target of training dataset and corresponding learning algorithm. Consequently, the proposed classifier has a better accuracy compared to other classifiers selected in the experiments.

5. Conclusion

One of the shortcomings of the neural network is lack of a connection among the features of each layer, which makes the direct relationship of a feature with other features in classification not to be considered. The proposed Bayesian RBF network makes it possible that the effect of the values of one feature on other features in decision making to be considered by multiplying the values of the features and setting the action linear weights among the values of features. In addition, due to linearity of the coefficients, they can be learned by a back propagation learning algorithm. Although the number of connections and linear weights considered among the features is less than the total number of possible edges of a dependency graph, it can greatly reduce the classification error. The low classification error and the high convergence rate in the learning process are features of the Bayesian RBF

method, which is also evident in empirical experiments. The proposed method could show higher precision and accuracy in classification on data set, in which there are influential dependencies among its features.

REFERENCES

1. Feng Jia, Yaguo Lei, Jing Lin, Xin Zhou, Na Lu, *Deep neural networks:A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data*, Mechanical Systems and Signal Processing , vol(72), pp.303–315, 2016.
2. H. J. Kappen and F. B. Rodrłguez, *Efficient Learning in Boltzmann Machines Using Linear Response Theory*, Neural Computation ,vol (10), Issue(5), 1998.
3. George E. Dahl, MarcAurelio Ranzato, Abdel-rahman Mohamed, and Geoffrey Hinton, *Phone Recognition with the Mean-Covariance Restricted Boltzmann Machine* , Advances in Neural Information Processing Systems(NIPS), pp.469-488, 2010.
4. Hugo Larochelle, Michael Mandel, Razvan Pascanu, Yoshua Bengio , *Learning Algorithms for the Classification Restricted Boltzmann Machine* , Journal of Machine Learning Research ,vol(13), pp.643?669, 2012.
5. Jeffrey M. Beck, Wei Ji Ma, Roozbeh Kiani, Tim Hanks, Anne K. Churchland, Jamie Roitman, Michael N. Shadlen, Peter E. Latham, Alexandre Pouget, *Probabilistic Population Codes for Bayesian Decision Making* , Neuron, Vol (60), Issue (6), PP. 946-949, 26 December 2008.
6. Buscema, Massimo, *Substance use & misuse* , Substance use misuse 33, no. 2, pp:233-270, 1998.
7. Bengio, Yoshua, Ian J. Goodfellow, and Aaron Courville, *Deep learning* , Nature 521, no. 7553, pp:436-444, 2015.
8. Mandt, Stephan, Matthew D. Hoffman, and David M. Blei, *Stochastic gradient descent as approximate Bayesian inference.*, The Journal of Machine Learning Research 18, no. 1, pp:4873-4907, 2017.
9. [9] Guang-Bin Huang , P. Saratchandran, N. Sundararajan, *A generalized growing and pruning RBF (GGAPRBF) neural network for function approximation*, IEEE Transactions on Neural Networks, vol (16), Issue(1), 2005.
10. R.N. Mahanty, P.B. Dutta Gupta , *Application of RBF neural network to fault classification and location in transmission lines* , IEE Proceedings - Generation, Transmission and Distribution, vol(151), Issue(2), pp. 201 C 212, 2004.
11. Shree, SR Bhagya, and H. S. Sheshadri, *Diagnosis of Alzheimer's disease using Naive Bayesian Classifier* , Neural Computing and Applications 29, no. 1,pp:123-132 ,2018.
12. Suresh, K and Dillibabu, R, *Designing a Machine Learning Based Software Risk Assessment Model Using Na?ve Bayes Algorithm*, 2018.
13. Goodfellow, Ian, Yoshua Bengio, Aaron Courville, and Yoshua Bengio, *Deep learning*, Vol. 1. Cambridge: MIT press, 2016.
14. Rezazadeh, Nader , *A modification of the initial weights in the restricted Boltzman machine to reduce training error of the deep belief neural network* , vol.15, No.7 , pp.1-6, 2017
15. Rezazadeh, Nader , *Initialization of Weights in Deep Belief Neural Network Based on Standard Deviation of Feature Values in Training Data Vectors* , Vol(6), Issue(6), pp:708-715, 2017.
16. Salakhutdinov, Ruslan, and Geoffrey Hinton, *Deep boltzmann machines*, In Artificial Intelligence and Statistics, pp: 448-455. 2009.
17. Martin T Hagan T, Howard B. Demuth, and Mark H. Beale, *Neural network design* , Boston: Pws Pub, Vol(20), 1996.
18. Glenn Shafer, Jeff Barnet , Wei Ji Ma, Roozbeh Kiani, Tim Hanks, Anne K. Churchland, Jamie Roitman, Michael N. Shadlen, Peter E. Latham, and Alexandre Pouget , *Probabilistic population codes for Bayesian decision making*, Neuron, Vol(60), Issue(6), pp:1142-1152, 2008.
19. Spiegel, Murray R., John J. Schiller, R. Alu Srinivasan, and Mike LeVan, *Probability and statistics.*, Vol. 2. New York: Mcgraw-hill, 2009.
20. Nelson, Randolph. Probability, *stochastic processes, and queueing theory: the mathematics of computer performance modeling.* , Springer Science Business Media, 2013.
21. Nitish Srivastava, Elman Mansimov, Ruslan Salakhudinov, *Unsupervised learning of video representations using lstms* , In International Conference on Machine Learning, pp: 843-852, 2015.
22. Sangjae Lee, Wu Sung Choi. *A multi-industry bankruptcy prediction model using back-propagation neural network and multivariate discriminant analysis* , Expert Systems with Applications Vol(40), Issue(8), pp:2941-2946, 2013.
23. Dai, Jianhua, Wentao Wang, Qing Xu, and Haowei Tian. *Uncertainty measurement for interval-valued decision systems based on extended conditional entropy* , Knowledge-Based Systems, Vol(27),pp: 443-450, 2012.
24. Data set Del or no Deal, https://dataverse.harvard.edu/dataset.xhtml?persistentId=hdl%3A1902.1/13633, [On Line Available]
25. Dataset Student Performance, https://archive.ics.uci.edu/ml/datasets/Student+Performance,[On Line Available]