Comparison of RBFNN, FBNN and KNN Algorithms for Face Recognition using PCA and Rectangular Feature

Sanjay Pagare
Computer Science and Engineering Department
Shri Dadaji Institute of Technology and Science, Khandwa

Surendra Gupta
Computer Engineering Department
Shri G S Institute of Technology and Science, Indore

ABSTRACT
Face Recognition is a popular application of computer vision in recent years. Not only the computer science researchers, but also the psychologists and neuroscientists are involved in this area. Appearance-based face recognition system includes linear Analysis like PCA, ICA, LDA, and Non-linear analysis is Model-based face recognition like, Elastic Bunch Graph Matching, 2D Morphable Model, 3D Morphable Model etc. In linear Analysis, matching score between the test face image and training images can be achieved by calculating the differences between their projection vectors determined by PCA, ICA or LDA. In nonlinear analysis, a mapping function is required to be applied on data space and then the linear analysis on the mapped data is applied. In this paper, RBFNN, FBNN and KNN classifier algorithms for face recognition using PCA and Rectangular feature have been implemented for the purpose of their comparison. The number of PCA features on different image size has been used along with four rectangular features. The recognition rate of different classifiers has been recorded in different conditions for the comparison.

General Terms
Rectangular Feature, Neural Network Classifiers, Face Recognition.

Keywords
Face Detection, Face Recognition, Image Processing, Principal Component Analysis, Levenberg-Marquardt Algorithm.

1. INTRODUCTION
Face recognition is the crucial technique to recognize the face in the field of Image processing. Now-a-days high detection rate achieved in rectangular based feature extraction system [2]. In recent years, face recognition has attracted much attention and its research has rapidly expanded by not only engineers but also neuroscientists. There have many potential applications in computer vision communication and automatic access control system. They depending on the application, in face recognition system can be working either on identification or verification mode. Despite the fact that there are more reliable biometric recognition techniques such as fingerprint and iris recognition the human face is undoubtedly the most common characteristic used by humans to recognize other people and this is why personal identification based on facial images is considered the friendliest among all biometrics.

We have to use face Recognition technique for identification and verification of person. The key to back propagation is a method for calculating the gradient of the error with respect to the weights for a given input by propagating error backwards through the network. Three classifier used for classifying the results they are KNNR, RBFNN, and FBNN. So we have to classify the best results and compare with corresponding classifier results that which is the best result.

In this paper, we use various classification techniques, namely, KNNR, FBNN and RBFNN [8]. PCA is a powerful technique, for dimensionality reduction which can predict not only for the seen data, but also for the unseen data. It works well for both linear and non-linearly separable datasets.

KNNR is a simple classification model that exploits lazy learning. Since test image will compare against all training images, KNNR encounters high response time. Furthermore, KNNR does not work well when data is high dimensional. For the dimensionality reduction, we exploit PCA [11] and select important features in the projected space.

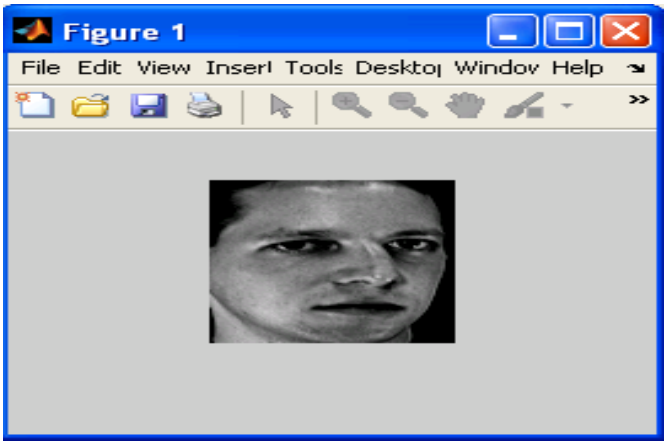

Fig 1: Face image for testing

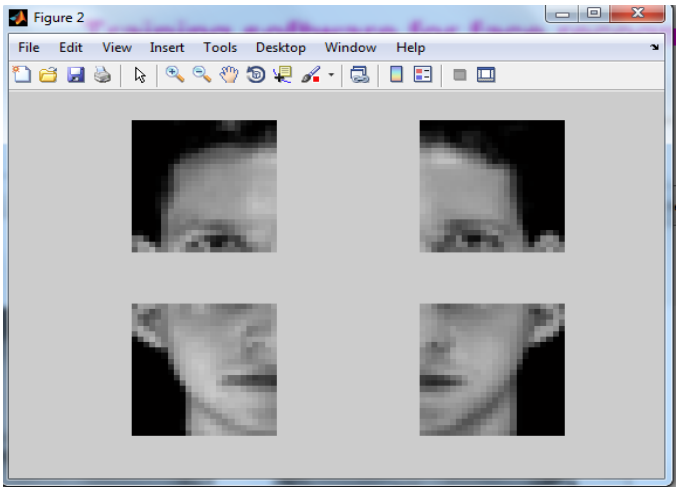

Fig 2: Rectangular features

2. FEATURE EXTRACTION ALGORITHM

2.1 Principal Component Analysis

Principal component analysis (PCA) is a mathematical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of uncorrelated variables called principal components. The number of principal components is less than or equal to the number of original variables. This transformation is defined in such a way that the first principal component has as high a variance as possible (that is, accounts for as much of the variability in the data as possible), and each succeeding component in turn has the highest variance possible under the constraint that it be orthogonal to (uncorrelated with) the preceding components. Principal components are guaranteed to be independent only if the data set is jointly normally distributed. PCA is sensitive to the relative scaling of the original variables.

3. FACE RECOGNITION

3.1 Face Recognition

Face recognition techniques should be roughly divided into two main phases: global approaches and feature based techniques. In global approaches the whole image serves as a feature vector, while in local feature approaches a number of control points are extracted and used for classification.

3.2 Global Approaches for Face Recognition

Global approaches model the variability of the face by analysing its statistical properties based on a large set of training images. Representative global techniques are Eigen faces, linear discriminant analysis (LDA), Support Vector Machines (SVM) and neural networks [13]. The first really successful face recognition method (and a reference point in face recognition literature) is a holistic approach based on principal component analysis (PCA). It is being applied to set of images in order to extract a set of Eigen-images, known as Eigen faces. Every face is modelled as a linear combination of a small subset of these Eigen faces and the weights of this representation are used for recognition.

3.3 Feature based Face Recognition Techniques

The main goal of this feature-based technique [13] is to discriminate among different faces based on measurements of structural attributes of the face and classifies the faces. Most recently used approaches are the Embedded Hidden Markov Models (EHMMs), the Elastic Graph Matching and Dynamic Link Architecture. For frontal views the significant facial features appear in a natural order and the degree of movement of face is very less. For frontal face Top to bottom (forehead, eyes, nose, and mouth) and from left to right (e.g. left eye, right eye). EHMMs model the face as a sequence of states roughly corresponding to facial features regions. The probability distribution functions of EHMM states are approximated using observations extracted by scanning training images from left-to-right and top-to-bottom. To verify a face, first the observations are extracted from the input image and then their probability given the stored EHM model is calculated.

4. TRAINING ALGORITHM

In this section we describe about the training algorithm and FBNN (feed forward back propagation neural network) it is our neural network classifier.

4.1 Levenberg Marquardt Algorithm

The Levenberg-Marquardt (LM) algorithm[14] is an iterative technique that locates the minimum of a multivariate function that is expressed as the sum of squares of non-linear real-valued functions. It has become a standard technique for non-linear least-squares problems, widely adopted in a broad spectrum of disciplines. LM can be thought of as a combination of steepest descent and the Gauss-Newton method. When the current solution is far from the correct one, the algorithm behaves like a steepest descent method: slow, but guaranteed to. Converge. When the current solution is close to the correct solution, it becomes a Gauss-Newton method. Next, a short description of the LM algorithm based on the material in [4] is supplied.

4.2 Training of Classifiers

The back-propagation algorithm is easiest to understand if all the units in the network are linear. The algorithm computes each EW by first computing the EA, the rate at which the error changes as the activity level of a unit is changed. For output units, the EA is simply the difference between the actual and the desired output. To compute the EA for a hidden unit in the layer just before the output layer, we first identify all the weights between that hidden unit and the output units to which it is connected. We then multiply those weights by the EAs of those output units and add the products. This sum equals the EA for the chosen hidden unit. After calculating all the EAs in the hidden layer just before the output layer, we can compute the EAs for other layers, moving from layer to layer in a direction opposite to the way activities propagate through the network. This is what gives back propagation its name. Once the EA has been computed for a unit, it is straight forward to compute the EW for each incoming connection of the unit. The EW is the product of the EA and the activity through the incoming connection. Before back propagating, the EA must be converted into the EI, the rate at which the error changes as the total input received by a unit is changed.

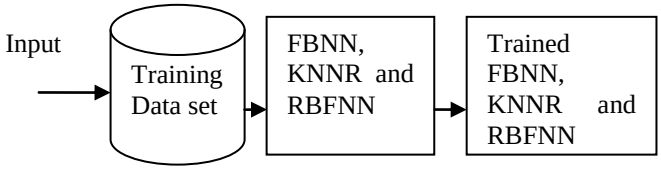

Fig 3: Training of KNN, FBNN and RFBNN algorithm

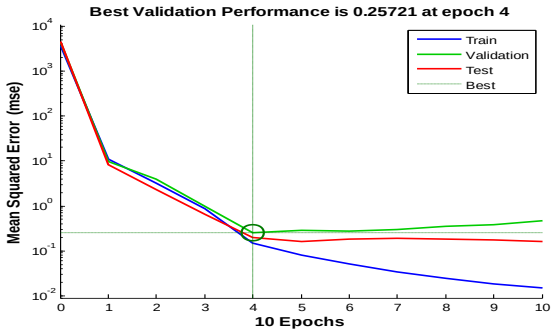

Fig 4: Performance of the system FBNN

It is well-known that the feed forward back propagation neural network (FBNN) overcomes some of the MLP problems by relying on a rapid training phase, and presenting systematic low responses to input patterns that have fallen into regions of the input space where there are no training samples.

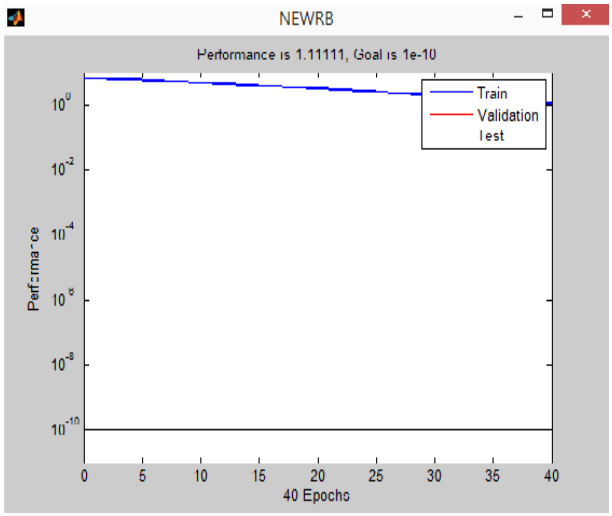

Fig 5: Performance of the system RBFNN

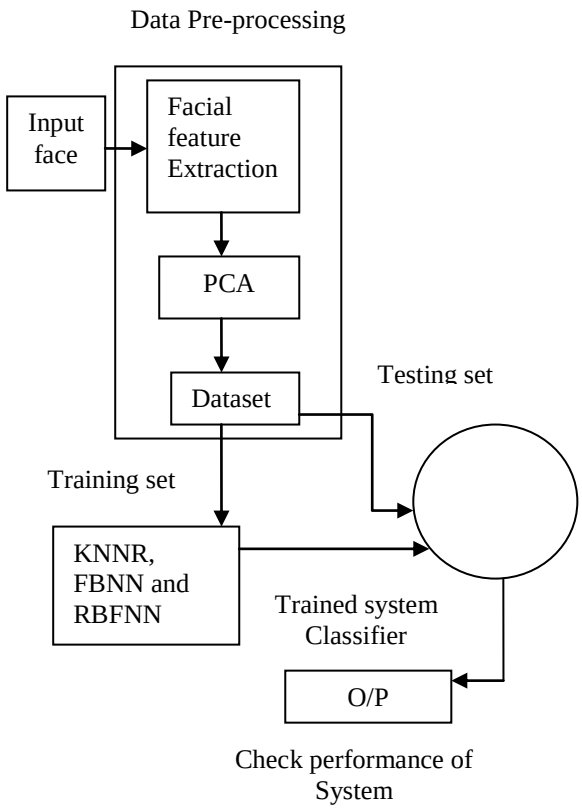

Fig 6: Block diagram of face recognition system

5. ALGORITHM DISCRIPTION

The number of neurons in the hidden layer, the coordinates of the centre of each hidden-layer RBF function, the radius (spread) of each RBF function in each dimension, the weights applied to the RBF function outputs as they are passed to the summation layer. Various methods have been used to train RBF networks. One approach first uses K-means clustering to find cluster centres which are then used as the centres for the RBF functions. However, K-means clustering is a computationally intensive procedure, and it often does not generate the optimal number of centres. Another approach is to use a random subset of the training points as the centres.

5.1 RBFNN Algorithm

Although the implementation is very different, RBF neural networks [15] are conceptually similar to K-Nearest Neighbor (K-NN) models. The basic idea is that a predicted target value of an item is likely to be about the same as other items that have close values of the predictor variable. Assume that each case in the training set has two predictor variables, x and y. The cases are plotted using their x,y coordinates as shown in the figure. Also assume that the target variable has two categories, positive which is denoted by a square and negative which is denoted by a dash. Now, suppose we are trying to predict the value of a new case represented by the triangle with predictor values x=6, y=5.1. Radial basis functional networks consist of three layers an input layer, a hidden layer and an output layer. The hidden unit provide a set of functions that constitute an arbitrary basis for the input patterns. Hidden units are known as radial centers and represented by the vectors c_1, c_2,.....,c_h. Transformation from input space to hidden unit space is nonlinear whereas transformation from hidden unit space to output space is linear. The radial basis function in the hidden layer produces a significant non-zero response only when the input falls within a small localized region of the input space. Each hidden unit has its own receptive field in input space. An input vector x_i which lies in the receptive field for center c_j, would activate c_j and by proper choice of weights the target output is obtained. The output as given as

$$y = \sum_{j=1}^{h} \Phi_j w_j$$

$$\Phi_j = \Phi(\|x - c_j\|)$$

Where w_j is weight of j^{th} center and ϕ is some radial function

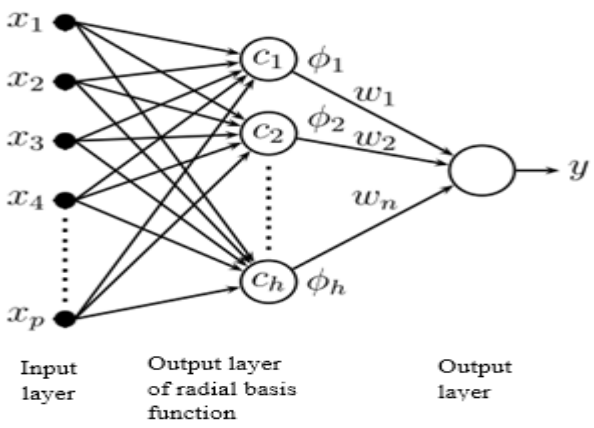

Fig 7: Architecture of radial basis function neural network

5.2 Back Propagation Neural Network

A typical back propagation network with Multi-layer, feed-forward supervised learning[4] is as shown in the figure 4.7[16]. Here learning process in Back propagation requires pairs of input and target vectors. The output vector ‘o ‘is compared with target vector’t ‘. In case of difference of ‘o’ and‘t‘vectors, the weights are adjusted to minimize the

difference. Initially random weights and thresholds are assigned to the network. These weights are updated every iteration in order to minimize the mean square error between the output vector and the target vector.

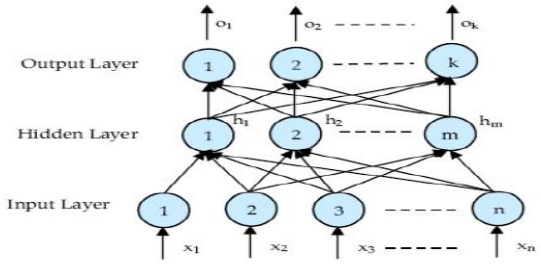

Fig 8: Architecture of back propagation neural networks

Input for hidden layer is given by

$$net_m = \sum_{z-1}^{n} x_z w_z$$

The units of output vector of hidden layer after passing through the activation function are given by

$$h_m = \frac{1}{1+\exp(-net_m)}$$

In same manner, input for output layer is given by

$$net_k = \sum_{z=1}^{m} h_z w_{kz}$$

For updating the weights, we need to calculate the error. This can be done by

$$E = \frac{1}{2}\sum_{i=1}^{k}(o_i - t_i)^2$$

o_i and t_i represents the real output and target output at neuron i in the output layer respectively. If the error is minimum than a predefined limit, training process will stop; otherwise weights need to be updated.

5.3 KNN Algorithm

K-Nearest Neighbors is a memory-based model defined by a set of objects known as examples (also known as instances) for which the outcome are known (i.e., the examples are labeled). Each example consist of a data case having a set of independent values labeled by a set of dependent outcomes. The independent and dependent variables can be either continuous or categorical. For continuous dependent variables, the task is regression; otherwise it is a classification. Thus, KNN can handle both regression and classification tasks. Given a new case of dependent values (query point), we would like to estimate the outcome based on the KNN examples. KNN achieves this by finding k examples that are closest in distance to the query point, hence, the name k-Nearest Neighbors. For regression problems, KNN predictions are based on averaging the outcomes of the k nearest neighbors; for classification problems, a majority of voting is used.

5.3.1 Distance Metric

As mentioned before, given a query point, KNN makes predictions based on the outcome of the k neighbours[17] closest to that point. Therefore, to make predictions with KNN, we need to define a metric for measuring the distance between the query point and cases from the examples sample. One of the most popular choices to measure this distance is known as Euclidean. Other measures include Euclidean squared

$$D(x,p) = \begin{pmatrix} \sqrt{(x-p)^2}, & Euclidean \\ (x-p)^2, & Euclidean\ \ squared \end{pmatrix}$$

Where x and p are the query point and a case from the examples sample, respectively.

5.3.2 K-Nearest Neighbor Predictions

After selecting the value of k, you can make predictions based on the KNN examples. For regression, KNN predictions are the average of the k-nearest neighbor's outcome.

$$y = \frac{1}{k}\sum_{i=1}^{k} y_i$$

Where y_i is the i^{th} case of the examples sample and y is the prediction (outcome) of the query point. In contrast to regression, in classification problems, KNN predictions are based on a voting scheme in which the winner is used to label the query.

5.3.3 Distance Weighting

Since KNN predictions are based on the intuitive assumption that objects close in distance are potentially similar, it makes good sense to discriminate between the K nearest neighbors when making predictions, i.e., let the closest points among the K nearest neighbors have more say in affecting the outcome of the query point. This can be achieved by introducing a set of weights W, one for each nearest neighbor, defined by the relative closeness of each neighbor with respect to the query point. Thus:

$$w(x,p) = \frac{e^{(-D(x_i y_i))}}{\sum_{i=1}^{k} e^{(-D(x_i y_i))}}$$

Where $D(x, p_i)$ is the distance between the query point x and the i^{th} case p_i of the example sample. For classification problems, the maximum of the above equation is taken for each class variable. It clear from the above discussion that when k>1, one can naturally define the standard deviation for predictions in regression tasks using:

$$\text{Error bar} = \pm \sqrt{\frac{1}{k-1}\sum_{i=1}^{k}(y - y_i)^2}$$

6. EXPERIMENTAL RESULTS

The face recognition system uses the ORL database for training and testing that available on internet. We can also get database from webcam. For all experiments it uses the MATLAB for code running on a pc with Intel Pentium 4 dual core, 2.0-GHZ CPU and 1024-Mb RAM. Before doing any experiment, we cropped the input images to reduce their size to 24×24. Our selected dataset contains grayscale images of 10 subjects in BMP format. In these experiments, we considered 9 images per each subject (total 90 images) containing different illumination and different poses, which 9 images of each used for training and also 9 images used for testing the method.

The neural network classifiers used for getting results these are KNNR, FBNN and RBFNN. We train the feature vectors using training algorithm Levenberg Marquardt (LM). It is associative mapping in which the network learns to produce a particular pattern on the set of input units whenever another particular pattern is applied on the set of input units. The

FBNN contain two hidden layer each layer having 20 neurons. RBFNN having maximum 40 neurons on hidden layer. We normalize the images on different equalizer which is 100x100, 150x150 and 200x200. In our face recognition system comparison is done between three algorithms which is KNN, RBFNN and FBNN.

Output of the neural network classifier is used for comparing the results for best recognition rate. Based on the results it is found that table 2 shows the best recognition rate of FBNN algorithm compared to RBFNN and KNN algorithms. In this paper we considered feature vector from PCA algorithm and rectangular features. Figure one is the face image for testing, figure two shows the rectangular features, and figure three shows the block diagram of training classifiers.

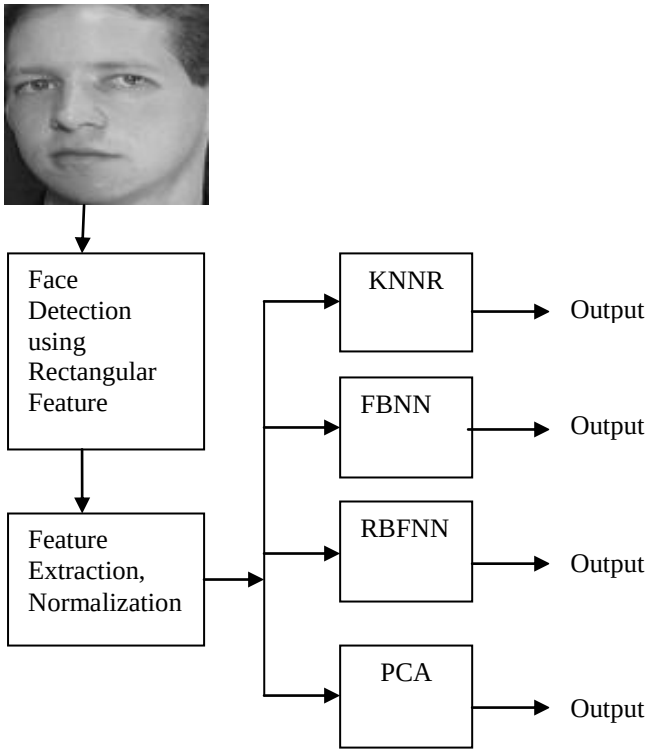

Fig 9: Output of different classifiers

Fig 10: Training set

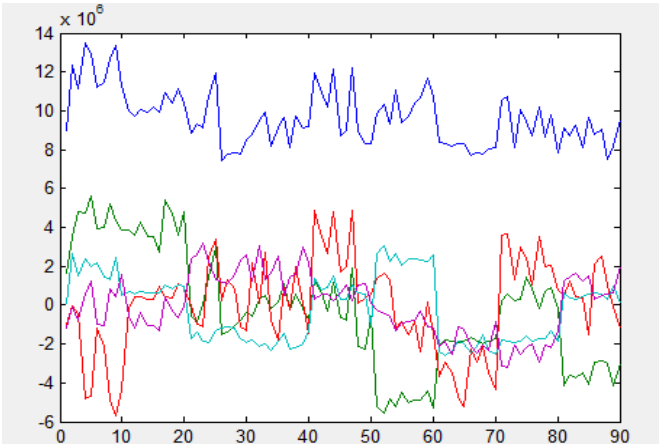

Fig 11: Training of KNNR

In our paper totally research based improvement of face recognition method. Figure four shows the performance of FBNN system. Training set gives training and compared with testing set that identifies the correct face in the form of recognition rate. Figure fifth shows the performance of RBFNN system. Figure sixth shows the overall working of face recognition system. Figure seventh shows the architecture of RBFNN. Figure number eight shows the architecture of FBNN. Figure nine is the output of different classifier. Figure ten is training set. Figure eleven shows the training of KNN with energy pulses. Figure twelve shows the best performance of FBNN algorithm on feature number ten in terms of recognition rate. Figure thirteen shows the overall comparison of three algorithms and it found that the FBNN algorithm gives best recognition rate when number of feature is ten on 200x200 equalizer.

Table 1. Recognition rate for FBNN

No of Features	Normalize Size		
	100x100	150x150	200x200
5	58.889	80	74.444
10	81.111	80	87.778
15	64.444	68.889	73.333
20	86.667	83.333	68.889
25	84.444	82.333	82.222
30	71.111	73.333	78.889

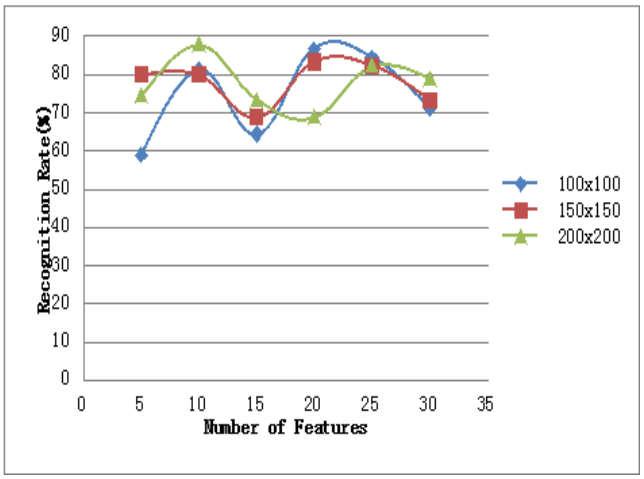

Fig 13: Graph for FBNN

Table 2. Comparison of recognition rate of KNNR, FBNN, and RBFNN

No of Features	Normalize Size		
	200x200	200x200	200x200
	KNNR	RBFNN	FBNN
5	66.667	55.556	74.444
10	86.667	55.556	87.778
15	86.667	55.556	73.333
20	86.667	61.111	68.889
25	84.444	61.111	82.222
30	85.889	61.111	78.889

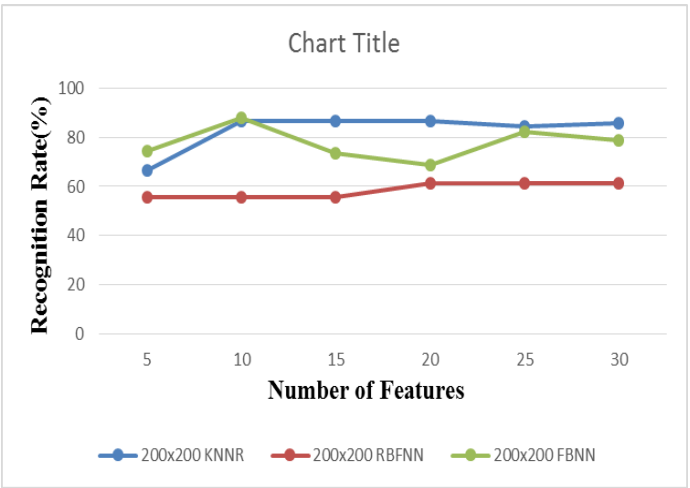

Fig 12: Graph for KNNR, FBNN, and RBFNN

7. CONCLUSION

The face recognition system using PCA and rectangular feature has been implemented successfully. The results are analysed in terms of recognition rate. It is found that the increase in number of features for the classifier will help to improve the classification up to certain extent. The increase in the number of feature does not significantly help above 20 features. The comparison of KNN, RBFNN and FBNN classifier for recognition rate is done on the same dataset and same number of features. The tuning of the neural network classifier has been performed. It was found that FBNN has significant improvement in recognition rate over RBFNN and result is comparable with the KNN. The advantage of FBNN over KNN is that, it does not require to store the feature database for classification. The exercise has been repeated for different image size like 100x100, 150x150, and 200x200. The result pattern is same but produces higher recognition rate which is also expected.

7.1 Future Scope

Some problems that remain unaddressed and can be addressed in future are as follows:

i. Investigations can be made on the behavior of RBFNN and KNNR.

ii. Other intricate classifier and algorithms can be proposed to make more intense study for the effect of face recognition system.

iii. The face recognition classifier showed some unexpected results for some input parameters, this input domain can be extended.

8. ACKNOWLEDGMENTS

Our thanks to Surendra Gupta assistant professor sgsits indore for contributed towards development of this research paper. I am thankful to Professor shrikant tare principal of SDITS khandwa for extending all help and support. I wish to acknowledge my deep sense of gratitude for Assistant professor Vijay Mandle, Head of Computer science Engineering Department for providing all the necessary resources and lab facilities.

9. REFFERENCES

[1] Pallabi Parveen and Bhavani Thuraisingham" Face Recognition using Multiple Classifiers" Proceedings of the 18th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'06)0-7695-2728-0/06.

[2] Dong-Liang Lee and Jen-Sheng Liang, "A Face Detection and Recognition System based on Rectangular Feature Orientation" 2010 International Conference on System Science and Engineering.

[3] R. Setiono and L. C. K. Hui, "Use of a quasi-Newton method in a feed forward neural network construction algorithm," IEEE Trans. Neural Networks, vol. 6, pp. 273–277, Nov. 1995.

[4] Robert Hecht-Nielsen, "Theory of the Back propagation Neural Network," Department of Electrical and Computer Engineering University of Caliiomia at San Diego La Jolla, CA 92139.

[5] K. Madsen, H.B. Nielsen, and O. Tingleff. Methods for Non-Linear Least Squares Problems. Technical University of Denmark, 2004. Lecture notes, available at http://www.imm.dtu.dk/courses/02611/nllsq.pdf.

[6] T. Mitchell, Machine Learning, McGraw Hill, 1997.

[7] DF. Zuo and P.H.N.de With, Two-stage face recognition incorporating individual-class discriminant criteria, Proc. WIC 2004, pp. 137–144, 2004.

[8] F.Zuo and P.H.N.de With, Two-stage face recognition incorporating individual-class discriminant criteria, Proc. WIC 2004, pp. 137–144, 2004.

[9] Khairul Azha A. Aziz, Ridza Azri Ramlee, Shahrum Shah Abdullah and Ahmad Nizam Jahari" Face Detection Using Radial Basis Function Neural Networks with Variance Spread Valu**e"** 2009 International Conference of Soft Computing and Pattern Recognition.

[10] O.Veksler, Pattern Recognition Lecture http:// www.csd.uwo.ca/ faculty/olga/ Courses/ CS434a_541a/

[11] R.A. Fisher. The use of multiple measurements in taxonomic problems. Annals of Eugenics, 7:179-188, 193

[12] http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html

[13] http://encyclopedia.jrank.org/articles/pages/6741/Face-Recognition.html.

[14] Manolis I. A. Lourakis "A Breif Dicription of the Levenberg-Marquardt Algorithm Implemened by levmar" Institute of Computer Science Foundation for Research and Technology-Hellas(FOURTH).

[15] http://en.wikipedia.org/wiki/Types_of_artificial_neural_networks.

[16] Alaa Eleyan and Hasan Demiral "PCA and LDA based Neural Networks for Human Face Recognition" Eastern Mediterranean University Northern Cyprus.

[17] http://www.statsoft.com/textbook/k-nearest-neighbors.