Unit Code: CIS007-2

Assignment 1: Design of Machine Learning Solution for Biometric Recognition Task

Unit Coordinator: Dr Vitaly Schetinin

Student Name: Santosh Tamang

ID: 2147440

E-mail: Santosh.Tamang@study.beds.ac.uk

# Table of Contents

# Introduction

The biometric facial recognition is a topic discussed for over decades. High performance in the recognition is achieved, but still due to the illumination the performance is affected. Various Machine Learning(ML) (Mitchell, 1997) and Deep Learning(DL) (Bengio, et al., 2015) techniques are used to improve the accuracy of recognition. The recognition algorithms are based on the recognition of shape of eyes, nose, distance between them etc. The factors affecting the performance of the solution developed are picture quality, orientation, illumination, distortion, clarity, sample data property, etc.

Analyzing various journals, research papers, paper from IEEE conferences, etc. The challenge lies in the data provided for the training due to factors mentioned above. To identify and extract the crucial biometric features from face, methods like Linear Discriminant Analysis(LDA) (Moon & P Johnathon, 2001), Locally Linear Embedding(LLE) (Riddler & P.W. Duin, n.d.), Independent Component Analysis(ICA) (Comon, 1992) and Principal Component Analysis(PCA) (Bruno A. Olshausen, 1996) were purposed. The feasibility of using the methods in facial recognition is already explored and it has been found quite useful. The PCA identifies and extracts the crucial features, while LDA maximizes the difference between the classes. ICA identifies and isolates the independent of the input data, LLE models the relationship between data points as a set of linear equations. Researchers uses the various extraction technique according to their dataset and preferences. Various methods have been implemented by many researchers on various datasets such as Yale Face database, AgeDB-30, etc.

Artificial Neural Networks(ANN) (Yegnanarayan, 2006) is an algorithm suggested for facial recognition with 3 layers-structured, i.e., input, hidden and output layers. ANN is a feedforward neural network, where the flow of information is unidirectional, from input to output layer, without any feedback connections. Back-propagation is used to train feedforward neural networks. Back-propagation is a supervised learning that adjust weights to minimize the difference between the expected result and the obtained result. Many researchers have preferred pairwise for the face recognition. Pairwise compare all possible pairs within the given dataset.

Although the employed techniques exhibit promising results in increasing the accuracy in facial recognition, some limitations cannot be ignored due to human ignorance. Some researchers did not calculate the k-fold cross-validation score. The k-fold cross-validation score is a technique for evaluating the performance of a predictive model over the entire data of the dataset. This technique is important for obtaining a comprehensive assessment of classifier accuracy since each classifier is trained on and tested against all the available data points. Not doing so has resulted in decreased information about accuracy that may vary during multiple runs. Utilizing minimum data to obtain optimal accuracy increases the possibility of biases in the results, as practiced by some researchers. For the future classification, the k-fold cross-validation method and large dataset can be used for the real-life experience as data are never small.

## Designing a Solution

The given assignment asks to design a Machine Learning(ML) solution, using an established Artificial Intelligence(AI) technique, to make a biometric recognition task with the possible highest recognition accuracy. The algorithm provided by the University was able to produce the accuracy of around 88%, using Multi-Layer Perceptron(MLP) classifier (Ramchoun, et al., 2017), as this classifier is efficient at processing large data and fast.

The selected architecture for the assignment is a feedforward neural network, i.e., MLP, where the value of the input data is preserved throughout the process of classifying the data. Only weights of the input data are modified to reduce the deviation between the predicted and the actual output, in feedforward neural network. Feedforward neural network architecture is resistant to noise, making it suitable for generalizing unseen data. Feedforward neural network is effective in solving complex problems as they can establish a nonlinear correlation between input and output data.

The dataset provided for the assignment is yaleB dataset that contains 50 pictures of 30 person each. The facial taken from real subjects are in different conditions, so it is common for some pictures to be recognized incorrectly, under high exposure, low exposure, etc. Thus, the possibility of attending ideal 100% recognition accuracy is near impossible. The image is pre-process to gray-scaled and the size of each image is 68x77 pixels. A ML solution is developed with the minimal recognition error. Classifying images into class of 30 is very hard as various features must be classified.

The total size of each image would be 5236(66x77) pixels, which is very large data, so the dimension of the data is reduced to 200 pca component per image using PCA. The new data in passed through the MLP classifier with various hyperparameters. Hyperparameters are the parameters whose values are set before the training process being, and they control the behavior and the performance of the algorithm or neural network during training. Selecting the right values is crucial for obtaining the optimal accuracy. Configurations involves making decisions about the value of hyperparameters, such as activation function, solver, number of hidden neurons etc.

There is a function train_neural in the code that declares the classifier, prediction, and k-fold cross-validation score, with variables in hyperparameters so that only the hyperparameter is modified and the process is same. Only the values for number of hidden neurons, solver and activation function are passed while calling the method. The random state is declared to reproduce the same result.

```
# Train nerual network
# Show the accuracy
def train_nerual(nohn, solver_name, activation_name):
  print("Fitting the classifier to the training set")
  clf = MLPClassifier(hidden_layer_sizes=(nohn,), solver=solver_name,
                      activation=activation_name, batch_size=256, verbose=True, random_state=26,
                     early_stopping=True).fit(X_train_pca, y_train)
  y_pred = clf.predict(X_test_pca) # reoognises the test images

  acc = metrics.accuracy_score(y_test, y_pred)

  clf_cross_val_result = cross_val_score(clf, X_pca, y, cv=5)
  cross_val = np.mean(clf_cross_val_result)

  print(classification_report(y_test, y_pred)) # the recognition accuracy

  print("Cross Validation Scores : ", clf_cross_val_result)
  print("Average Cross Validation Scores: ", cross_val)

  nohns.append(nohn)
  accs.append(acc)
  cross_vals.append(cross_val)
```

Fig.no.1. train_neural function

## Experiments

The pre-processed data is loaded, dimensionally reduced, and split into training and testing data. A solver, an activation function (Elliott, 1993) and number of hidden neurons are selected and passed into the function train_neural then a neural network is trained. The prediction accuracy of the neural network is check with classification report and k-fold mean cross-validation score to find the optimal parameters to obtain highest possible accuracy.

### Test 1

In test 1, ‘sgd’ solver, ‘tanh’ activation function with various number of hidden neurons is passed to the method, .i.e., 200 – 350 with increment of 50 per round. SGD stands for Stochastic Gradient Descent(SGD) is an optimization algorithm widely used for finding the optimal parameters for a model. The SGD randomly sample a subset of training data to compute the gradient of the loss function. The algorithm works by initializing the model parameters with some initial value and then iteratively updating them. Tanh is a non-linear activation functions that maps the input in the range [-1,1].

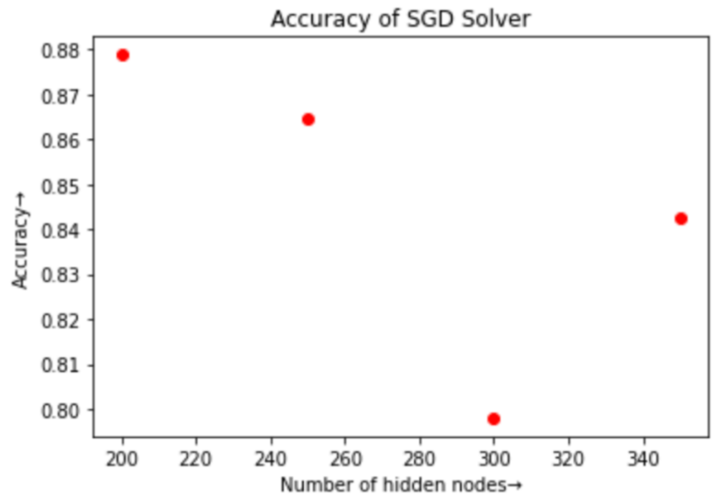


Fig.2.1. SGD tanh Accuracy

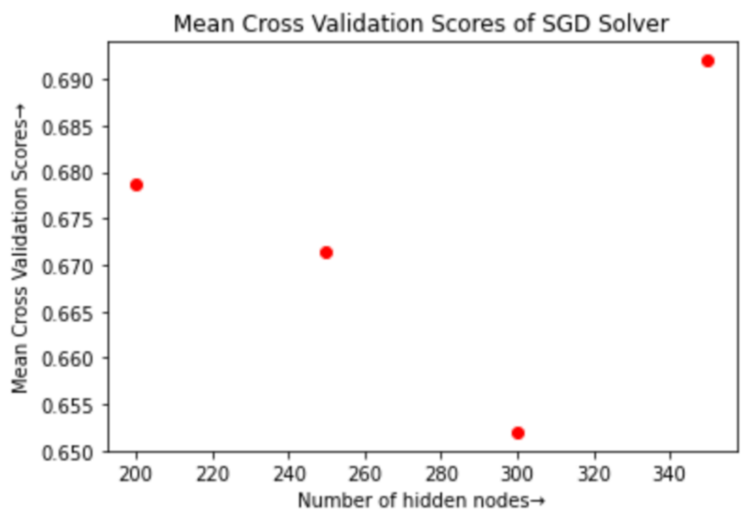


Fig.2.2. SGD tanh Mean Cross-Validation Score

## Test 2

Similarly, in test 2, ‘sgd’ solver, ‘relu’ activation function with various number of hidden neurons are passed to the method, .i.e., 200 – 350 with increment of 50 per round. Relu is a non-linear activation function that maps the input to the range of [0, inf).

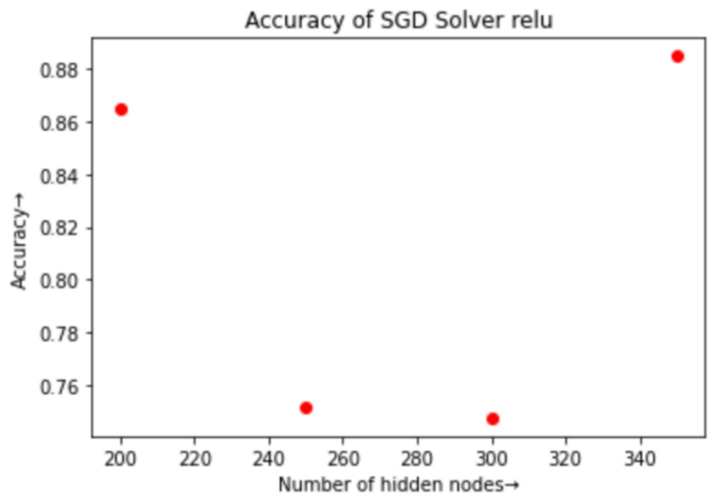


Fig.3.1. SGD relu Accuracy

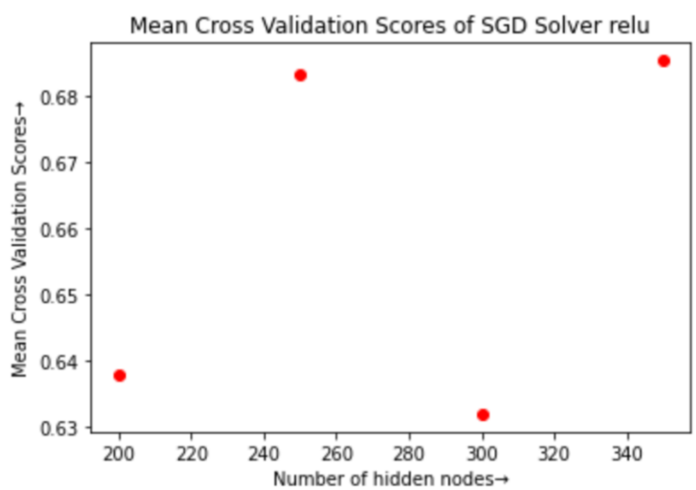


Fig.3.2. SGD relu Mean Cross-Validation Score

## Test 3

In test 3, ‘lbfgs’ solver, ‘tanh’ activation function with various number of hidden neurons is passed to the method, .i.e., 200 – 350 with increment of 50 per round. LBFGS stands for Limited-memory Broyden-Fletcher-Goldfarb-Shanno is an optimization algorithm that is used to find the minimum of a function by iteratively improving an estimate of the gradient. LBFGS is capable of handling high-dimensional optimization with large number of hyperparameters.

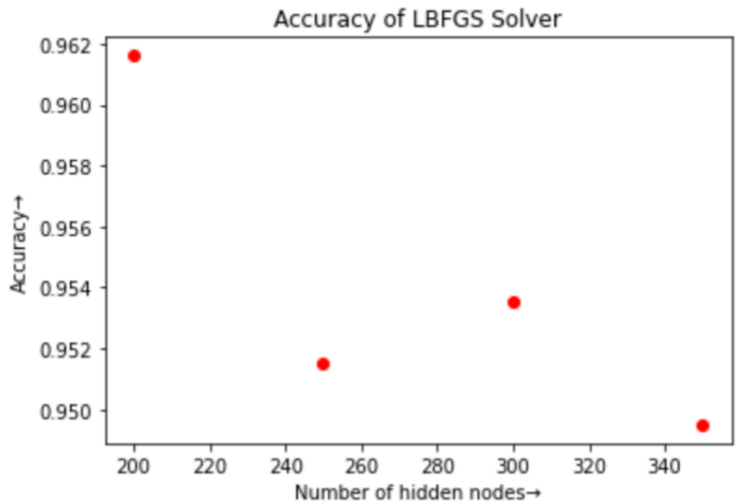


Fig.4.1. LBFGS tanh Accuracy

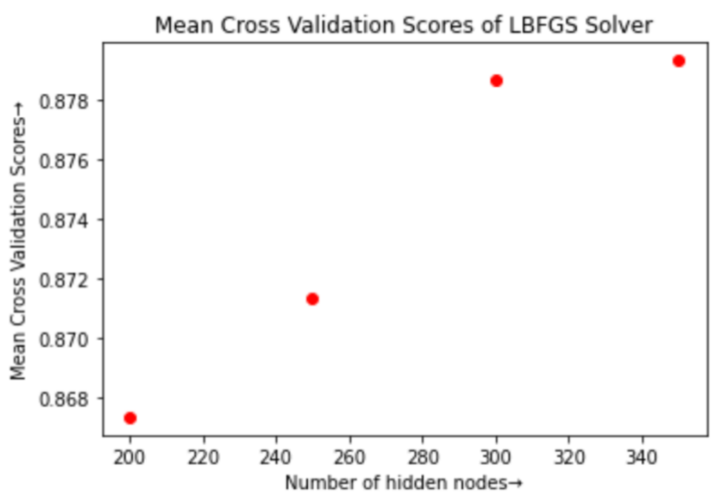


Fig.4.2. LBFGS tanh Mean Cross-Validation Score

## Test 4

In test 4, 'lbfgs' solver, 'relu' activation function with various number of hidden neurons are passed to the method, .i.e., 200 – 350 with increment of 50 per round.

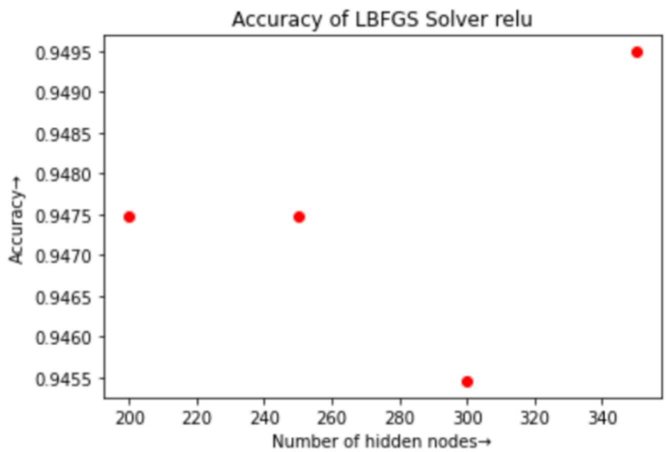


Fig. 5.1. LBFGS relu Accuracy

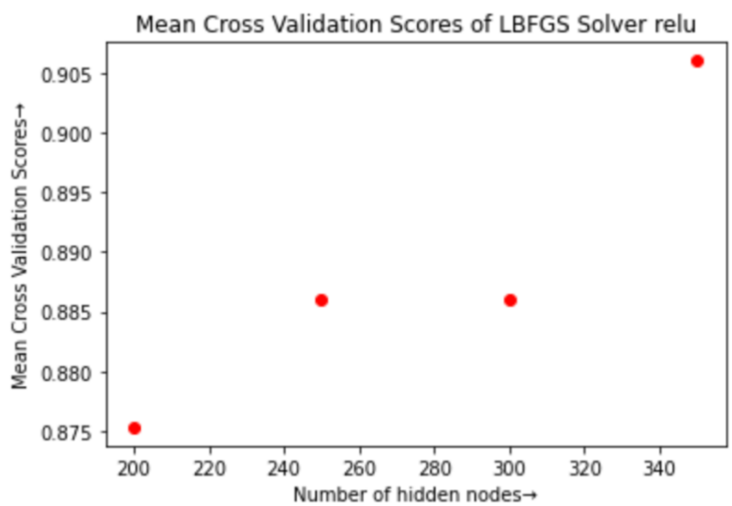


Fig.5.2. LBFGS relu Mean Cross-Validation Score

## Conclusion

In the above figures 3.1 and 3.2, the lowest mean cross-validation score is of ‘sgd’ solver and ‘relu’ activation function. As the ‘sgd’ solver works by iteratively updating the parameters and the learning rate determine how quickly the algorithm converges to the optimal solution; however, if the learning rate is too high the algorithm may overshoot the optimal solution and diverge. As ‘relu’ maps input in the range [0, inf), if the input to the function is negative or gradient becomes zero the neuron is unable to anything new. This may be the cause of low accuracy in the model containing ‘sgd’ solver and ‘relu’ activation function.

From the above figures 5.1 and 5.2, it can be determined that the highest mean cross-validation score is of ‘lbfgs’ solver and ‘relu’ activation function. As ‘lbfgs’ can handle high-dimension optimization problems with large number of parameters, it can handle the optimization of the given data better than ‘sgd’ solver.

## References

Moon, H. & P Johnathon, P., 2001. Computational and performance aspects of PCA-based face-recognition algorithms. *Perception,* Volume 30, pp. 303-321.
Riddler, D. d. & P.W. Duin, R., n.d.
Comon, P., 1992. Independent Component Analysis. In: J. Lacoume, ed. *Higher-Order Statistics.* s.l.:Elsevier, pp. 29-38.
Bruno A. Olshausen, D. J. F., 1996. Principal Component Analysis. *Sparse coding and decorrelation in primary visual cortex during natural vision,* 37(23), pp. 2393-2403.

Mitchell, T., 1997. *Machine Learning.* New Dehli: McGraw Hill Education India.
Bengio, Y., Goodfellow, I. & Courville, A., 2015. *Deep Learning.* s.l.:s.n.
Yegnanarayan, B., 2006. *Artificial Neural Networks.* New Delhi: Prentice-Hall of India Private Limited.
Ramchoun, H., Amine Janati Idrissi, M., Ghanou, Y. & Ettaouil, M., 2017. Multilayer Perceptron: Architecture Optimization and Training. *International Journal of Interactive Multimedia and Artificial Intelligence*, 4(1).
Elliott, D. L., 1993. *A Better Activation Function for Arti cial Neural Networks* , Washington: The National Science Foundation Engineering Research Center Program, The University of Maryland.

# Appendix