Drexel University

CS499I

Advanced Neural Networks

# Facial Recognition With Artificial Neural Networks

*Author:*
Alexander Marion
Matthew D'Amore

*Supervisor:*
Dr. Matthew Burlick

May 16, 2017

## 1 Abstract

Using a shallow artificial neural network and the Yale Faces Database we trained artificial neural networks in an attempt to find the optimal parameters to minimize error. Using data manipulation techniques such as standardization and principal component analysis (PCA) we were able to increase the accuracy of a shallow neural network while decreasing runtime and number of training iterations. We were able to provide reasonable bounds for the number of training iterations, hidden layer size, and image size using empirical data.

## 2 Introduction

This paper explores artificial neural networks for identifying black and white images of frontal faces with different expressions. A shallow neural network comprised of three layers (input, hidden, and output) was trained varying multiple parameters to test and the optimal parameters for the highest accuracy. The Yale Face Database which contains 165 grayscale frontal face images was used as input for training, testing, and validation.

## 3 Dataset

**Yale Faces Database** This dataset contains 165 frontal grayscale images in GIF format of 15 individuals with 11 images per person. There is one image per each of the following configurations: center-light, with glasses, happy, left-light, without glasses, normal, right-light, sad, sleepy, surprised, and winking.

## 4 Training Methods

The artificial neural network was trained using forward-backward propagation which is comprised of two main steps: prediction and error propagation. Forward-backward propagation uses logistic regression to perform gradient descent to minimize the error in the output layer. The weights for each node are updated proportionally to the error at that node.

During forward propagation (or prediction) the input layer (in this case, image features) is used to calculate the values of the hidden layer by multiplying the input values by a matrix of weights which are initialized randomly to values between -1 and 1. As in logistic regression, these values are input to the sigmoid function which outputs a value between 0 and 1. The hidden layer is then multiplied by another matrix of weights to produce the output layer (the prediction) which are also input to the sigmoid function. The sigmoid function returns the output layer which is a collection of probabilities between 0 and 1, the highest of which is chosen as the predicted class.

The forward propagation process is as follows:
For every hidden node $j$ compute the output value using activation function $g$, the input value $x$, and the weights from the input layer to the hidden layer denoted $\beta$. For our purposes let the activation function $g$ denote the sigmoid function $g(x) = \frac{1}{1+e^{-x}}$. The following formula gives the value at hidden node $j$:

$$h_j = g(x\beta_{:,j})$$

This process is repeated for each output layer node $k$. The value of the output node is computed using the output of the hidden layer $h$, the output node's vector of weights connecting to the hidden layer $\theta_{:,k}$, and the activation function $g$. The following is the formula for computing the value at output node $o$:

$$o_k = g(h\theta_{:,k})$$

In matrix form the forward propagation rules can simply be written as:

$$h = g(x\beta)$$
$$o = g(h\theta)$$

After the prediction stage has concluded the backward propagation step begins. The difference between a predicted value and the actual value is calculated as the loss at that output node. The weights between the hidden nodes and the output node are updated based on the error at that output node. The loss at each hidden node is then calculated based on the error of all output nodes scaled by the weights of the connection to the hidden node. The weights between the input layer and the hidden layer are then updated proportionally to the error traveling on the connected edges.
The backward propagation process is as follows:
The error at the each output node is calculated as

$$\delta_k = (y_k - o_k)$$

The weights between the output nodes and the hidden layer ($\theta$) are then updated proportionally to the error at the output nodes they connect to and the data traveling along that edge. This is described by the following equation:

$$\frac{\partial E}{\partial \theta_{j,k}} = \delta_k h_j$$
$$\theta_{j,k} = \theta_{j,k} + \frac{\eta \partial E}{\partial \theta_{i,j}} = \theta_{j,k} + \eta \delta_k h_j$$

The loss is then computed at each hidden node based on the loss of every connection to that node scaled by the weights of those connections and the data which travels across those edges. The error formula is shown below:

$$\delta_j = (\sum_{k=1}^{K} (\delta_k \theta_{j,k})) h_j (1 - h_j)$$

The weights between the hidden layer and input layer are then updated based on the error of their connected hidden node and the data traveling along that edge. The update rule is as follows:

$$\beta_{i,j} = \beta_{i,j} + \eta \frac{\partial E}{\partial \beta_{i,j}} = \beta_{i,j} + \eta \delta_j x_i$$

This process was repeated over multiple training iterations and used to perform gradient descent to minimize the training error on the output nodes. The learning rate was dynamically increased and decreased based on the performance during the current training iteration as compared to past iterations. The learning rate was incremented in small amounts when the change in accuracy rose and decreased drastically when the change in accuracy fell. [2]

Before training the artificial neural network four different data manipulation techniques could be applied.

1. Adding a bias node to the input layer
2. Adding a bias node to the hidden layer
3. Standardizing the data features
4. Performing PCA to retain a percentage of the feature information

Each of the 16 variants ($2^4$ possible combinations of these manipulations) were tested.

The input parameters for the artificial neural network were as follows:

1. Number of training iterations
2. Hidden layer size
3. Image size
4. PCA field retention

These parameters were empirically tested as shown further below.

## 5 Testing Methods

Every time the artificial neural network was trained the data was split across nine folds for cross validation. Each fold had 90% training data and 10% testing data. The training data was then split into 20% for a validation set and 80% for training [1]. The final split across every fold was 72% training samples, 18% validation samples, and 10% testing samples. The training, validation, and testing accuracies were averaged over the nine folds. This cross validation was done to avoid bias based on the selected training set and gives a better statistical model for how this testing would generalize over any training and testing set.

## 6 Baseline Accuracy

To create a baseline accuracy for the shallow artificial neural network on the Yale Faces Database the network was trained with the following parameters:

| | |
|---|---|
| Training iterations | 1000 |
| Hidden nodes | 20 |
| Image size | 40 |
| PCA field retention | 0.95 |

The data was standardized before training, no bias nodes were added, and PCA was not performed. The following is the testing and validation accuracy for the baseline training parameters and a plot of training accuracy vs. the current training iteration.

BASELINE PLOT AND TABLE HERE

Across nine folds the baseline training parameters trained an artificial neural network which achieved 73.3918% average testing accuracy and an average validation accuracy of 70.8812%.

## 7 Variation Testing

The following 15 pairs of tables and figures show the remaining exhaustive set of variations for data manipulation combinations before the training of the artificial neural network. Each network was trained with the following training parameters:

| | |
|---|---|
| Training iterations | 1000 |
| Hidden nodes | 20 |
| Image size | 40 |
| PCA field retention | 0.95 |

Each variation was trained and cross validated on nine folds with a training, validation, and testing set as described above.

| | |
|---|---|
| Input layer bias node | N |
| Hidden layer bias node | N |
| Standardization of features | N |
| PCA applied | N |
| Avg. Testing Accuracy | 0.254061 |
| Avg. Validation Accuracy | 0.214559 |

Table 1: NNNN Testing Accuracy

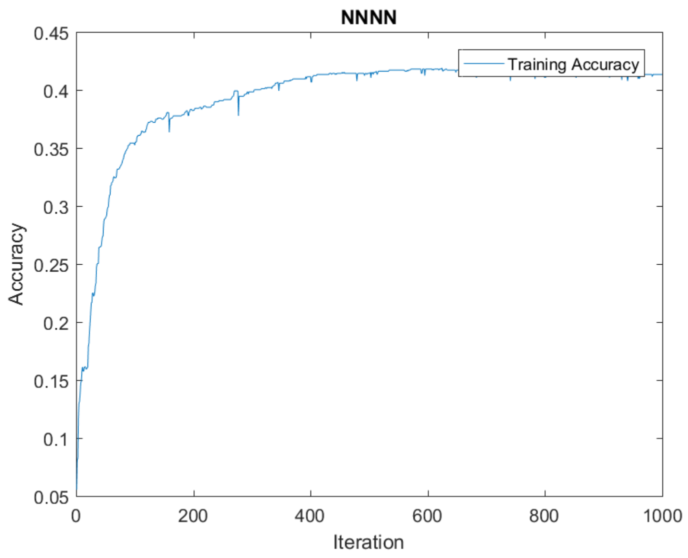

Figure 1: NNNN Training Accuracy

| | |
|---|---|
| Input layer bias node | Y |
| Hidden layer bias node | N |
| Standardization of features | N |
| PCA applied | N |
| Avg. Testing Accuracy | 0.162443 |
| Avg. Validation Accuracy | 0.157088 |

Table 2: YNNN Testing Accuracy

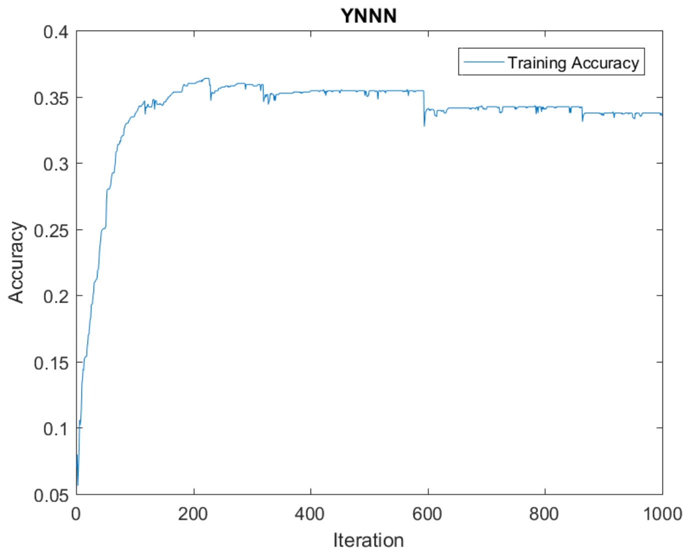

Figure 2: YNNN Training Accuracy

| | |
|---|---|
| Input layer bias node | N |
| Hidden layer bias node | Y |
| Standardization of features | N |
| PCA applied | N |
| Avg. Testing Accuracy | 0.265757 |
| Avg. Validation Accuracy | 0.249042 |

Table 3: NYNN Testing Accuracy

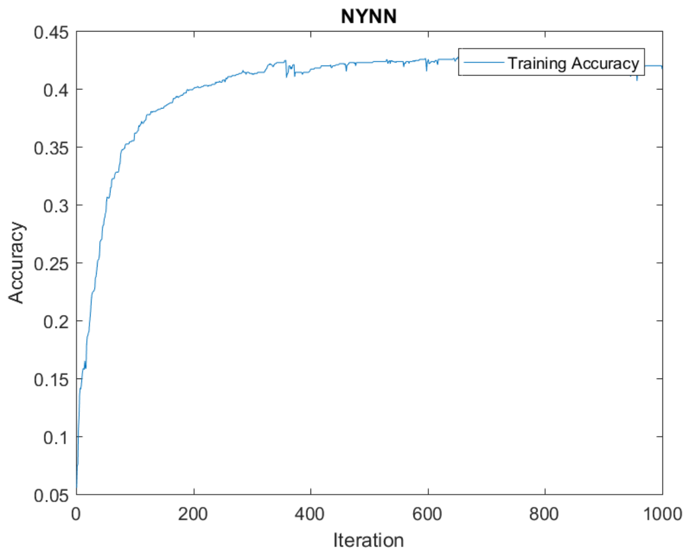

Figure 3: NYNN Training Accuracy

| | |
|---|---|
| Input layer bias node | N |
| Hidden layer bias node | N |
| Standardization of features | N |
| PCA applied | Y |
| Avg. Testing Accuracy | 0.339181 |
| Avg. Validation Accuracy | 0.302682 |

Table 4: NNNY Testing Accuracy

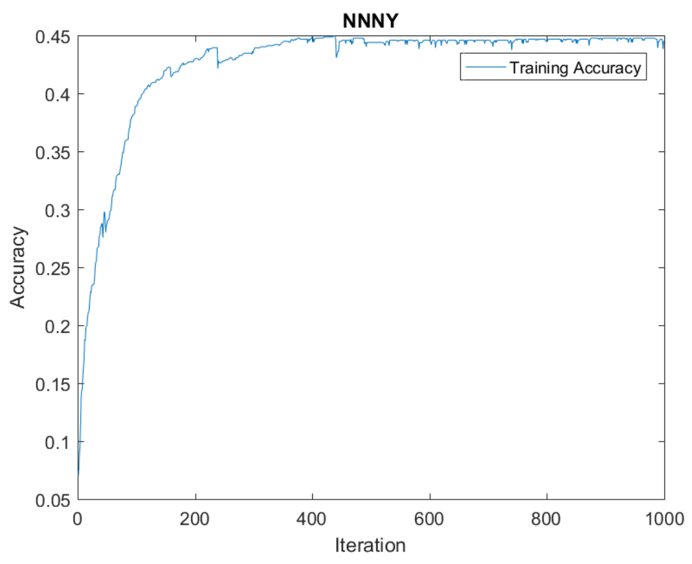

Figure 4: NNNY Training Accuracy

| | |
|---|---|
| Input layer bias node | Y |
| Hidden layer bias node | Y |
| Standardization of features | N |
| PCA applied | N |
| Avg. Testing Accuracy | 0.192658 |
| Avg. Validation Accuracy | 0.187739 |

Table 5: YYNN Testing Accuracy

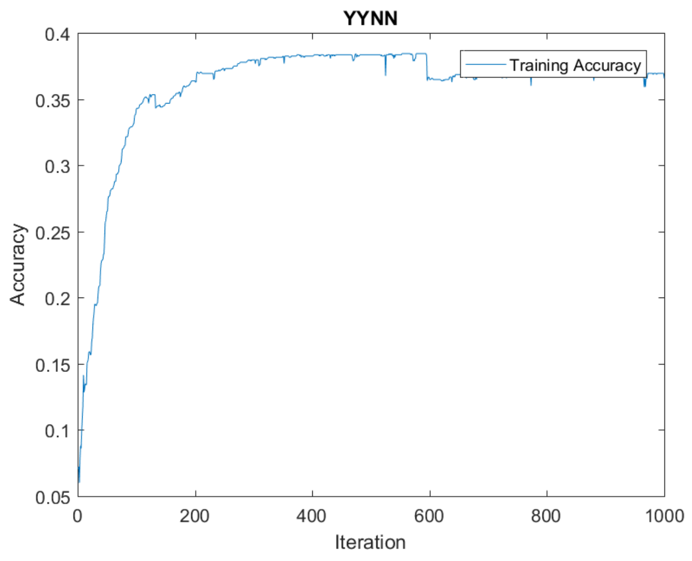

Figure 5: YYNN Training Accuracy

| | |
|---|---|
| Input layer bias node | Y |
| Hidden layer bias node | N |
| Standardization of features | Y |
| PCA applied | N |
| Avg. Testing Accuracy | 0.721897 |
| Avg. Validation Accuracy | 0.724138 |

Table 6: YNYN Testing Accuracy

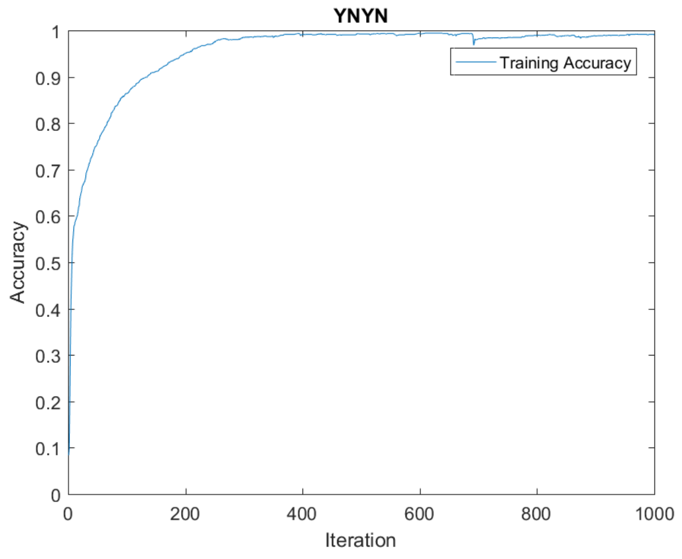

Figure 6: YNYN Training Accuracy

| | |
|---|---|
| Input layer bias node | Y |
| Hidden layer bias node | N |
| Standardization of features | N |
| PCA applied | Y |
| Avg. Testing Accuracy | 0.282651 |
| Avg. Validation Accuracy | 0.302682 |

Table 7: YNNY Testing Accuracy

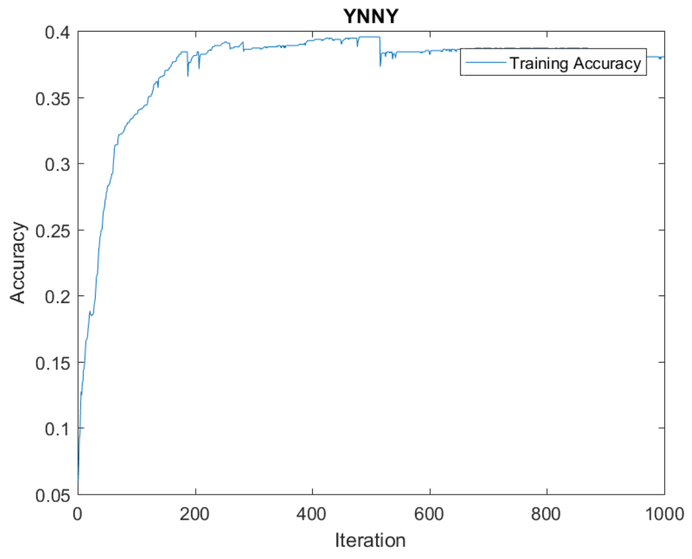

Figure 7: YNNY Training Accuracy

| | |
|---|---|
| Input layer bias node | N |
| Hidden layer bias node | Y |
| Standardization of features | Y |
| PCA applied | N |
| Avg. Testing Accuracy | 0.709552 |
| Avg. Validation Accuracy | 0.697318 |

Table 8: NYYN Testing Accuracy

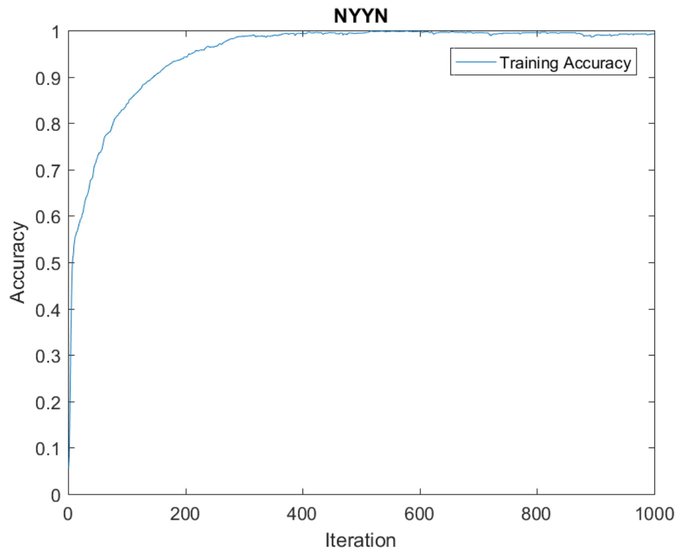

Figure 8: NYYN Training Accuracy

| | |
|---|---|
| Input layer bias node | N |
| Hidden layer bias node | Y |
| Standardization of features | N |
| PCA applied | Y |
| Avg. Testing Accuracy | 0.405133 |
| Avg. Validation Accuracy | 0.318008 |

Table 9: NYNY Testing Accuracy

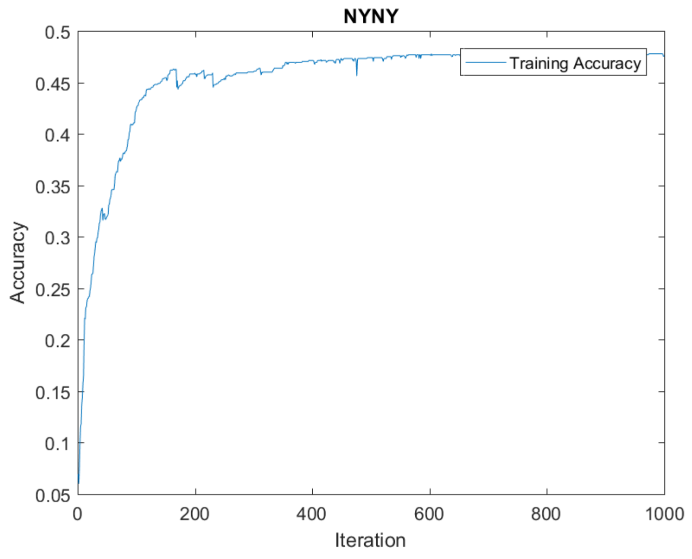

Figure 9: NYNY Training Accuracy

| | |
|---|---|
| Input layer bias node | N |
| Hidden layer bias node | N |
| Standardization of features | Y |
| PCA applied | Y |
| Avg. Testing Accuracy | 0.946069 |
| Avg. Validation Accuracy | 0.950192 |

Table 10: NNYY Testing Accuracy

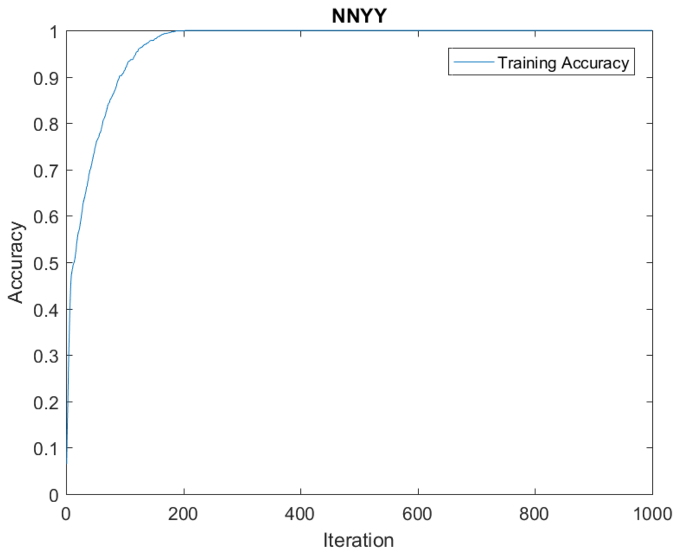

Figure 10: NNYY Training Accuracy

| | |
|---|---|
| Input layer bias node | Y |
| Hidden layer bias node | Y |
| Standardization of features | Y |
| PCA applied | N |
| Avg. Testing Accuracy | 0.692658 |
| Avg. Validation Accuracy | 0.724138 |

Table 11: YYYN Testing Accuracy

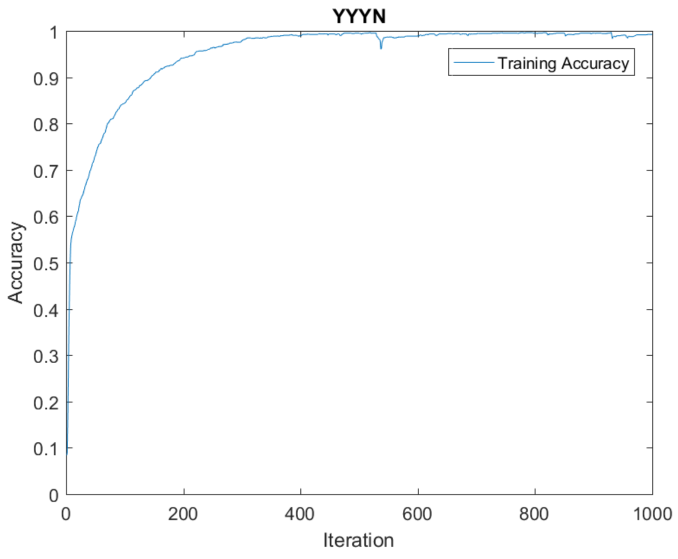

Figure 11: YYYN Training Accuracy

| | |
|---|---|
| Input layer bias node | Y |
| Hidden layer bias node | Y |
| Standardization of features | N |
| PCA applied | Y |
| Avg. Testing Accuracy | 0.318389 |
| Avg. Validation Accuracy | 0.314176 |

Table 12: YYNY Testing Accuracy

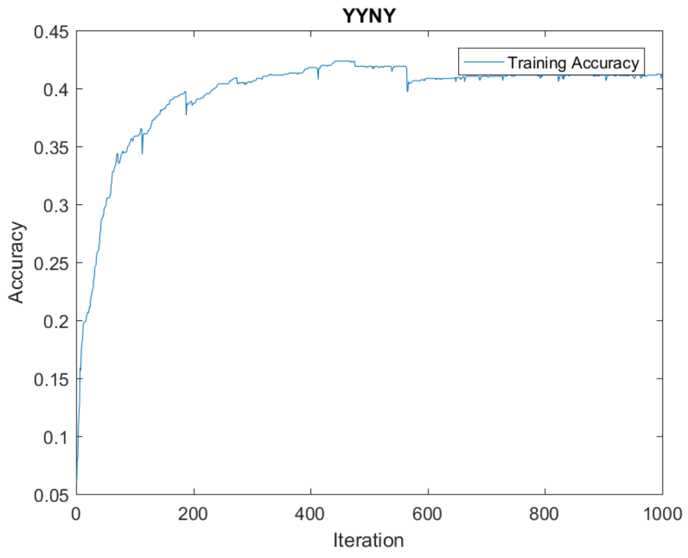

Figure 12: YYNY Training Accuracy

| | |
|---|---|
| Input layer bias node | Y |
| Hidden layer bias node | N |
| Standardization of features | Y |
| PCA applied | Y |
| Avg. Testing Accuracy | 0.951917 |
| Avg. Validation Accuracy | 0.954023 |

Table 13: YNYY Testing Accuracy

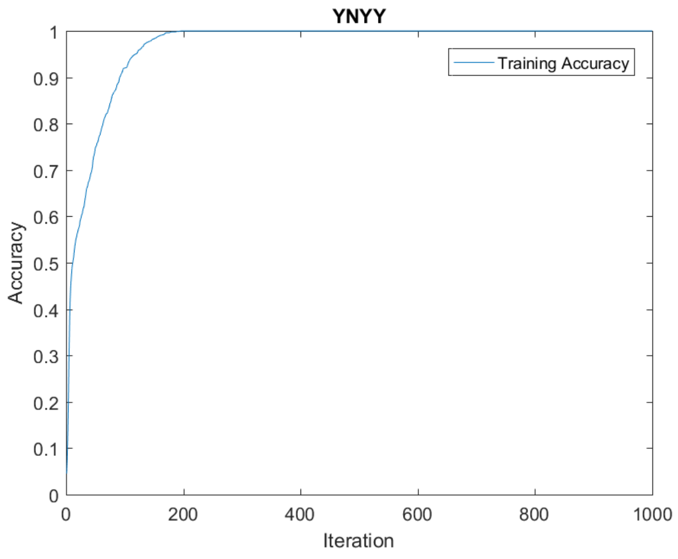

Figure 13: YNYY Training Accuracy

| | |
|---|---|
| Input layer bias node | N |
| Hidden layer bias node | Y |
| Standardization of features | Y |
| PCA applied | Y |
| Avg. Testing Accuracy | 0.940221 |
| Avg. Validation Accuracy | 0.919540 |

Table 14: NYYY Testing Accuracy

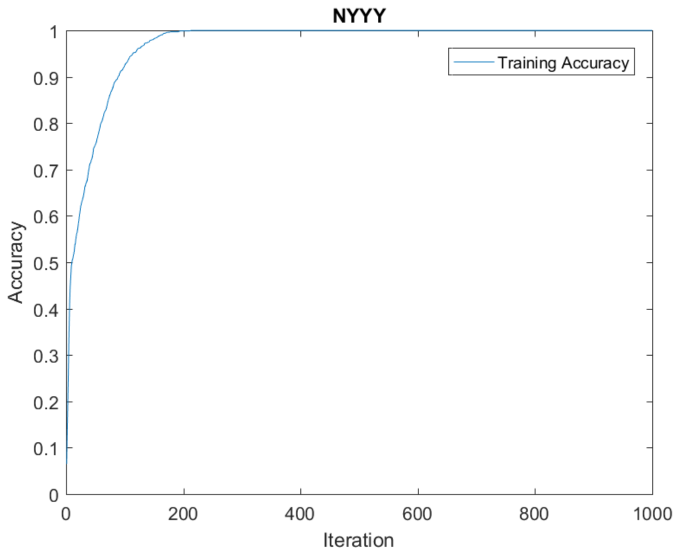

Figure 14: NYYY Training Accuracy

| | |
|---|---|
| Input layer bias node | Y |
| Hidden layer bias node | Y |
| Standardization of features | Y |
| PCA applied | Y |
| Avg. Testing Accuracy | 0.940221 |
| Avg. Validation Accuracy | 0.923372 |

Table 15: YYYY Testing Accuracy

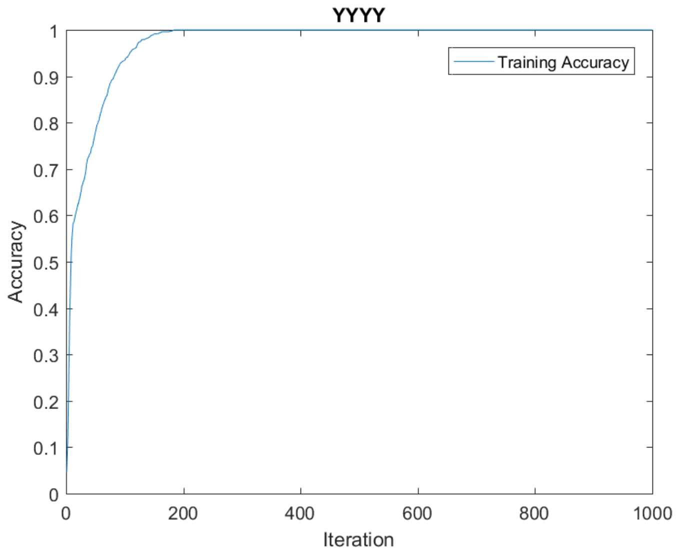

Figure 15: YYYY Training Accuracy

The variant with the highest accuracy was trained by adding a bias node to the input, standardizing the data, and performing PCA with a retention rate of 95%.

## 8 Empirical Parameter Testing

A shallow artificial neural network was trained varying the input parameters to empirically test the effect that each parameter had on the training, testing, and validation accuracy. The four parameters

which were tested were number of training iterations, image size, PCA percent field retention, and number of hidden nodes.
For each test the highest performing variant of the variant testing set was used. Each artificial neural network was trained with a bias node at the input layer, standardized data, and PCA applied before training.

1. **Number of Training Iterations**
   The number of training iterations was varied from 1 to 10,000 in increments of 100. The artificial neural network was trained using each number of training iterations in this set and was cross validated over nine folds. The accuracy plot show below reflects the average accuracies across the nine folds. The artificial neural network was trained with an image size of 40 by 40 and 20 hidden nodes.

   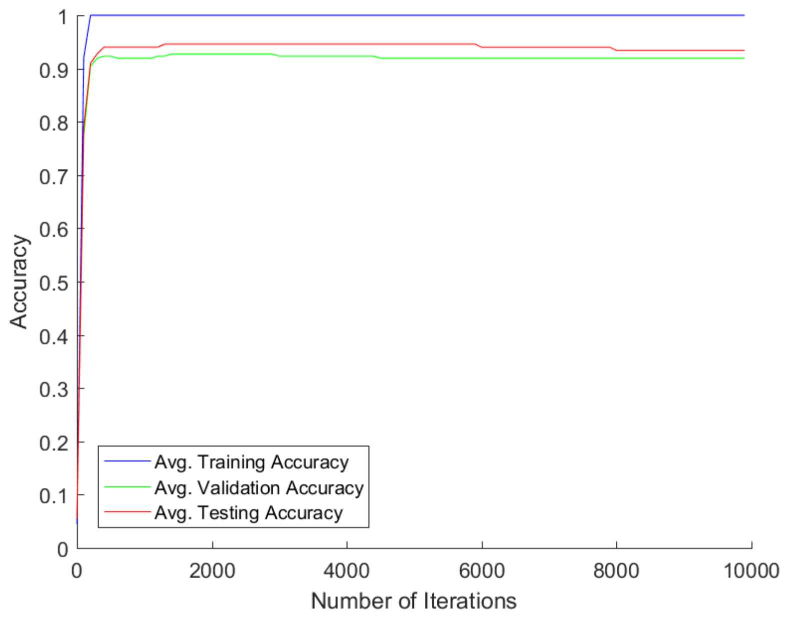

   Figure 16: Plot of accuracy as the number of training iterations increases

   As shown above, the average accuracy of the artificial neural network was not dependent on the number of iterations past 300 or 400 iterations. For future testing we opted to keep our standard 1000 training iterations and adjust if we saw drastic over or underfitting.

2. **Image Size and PCA Percent Field Retention**
   The image size and percent field retention for PCA were varied together exhaustively. Both of these variables determine the number of input nodes and are not independent of one another. The image size was varied from 10 to 100 in increments of five and the retention rate was varied from 0.10 to 1 in increments of 0.05. The artificial neural network was trained with 20 hidden nodes and 1000 training iterations.

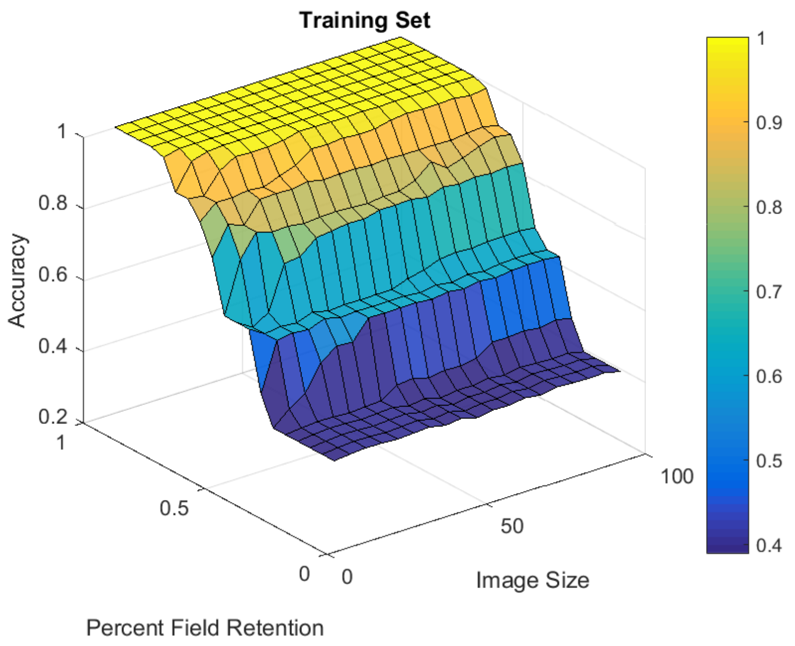

Figure 17: Plot of training accuracy as the image size and percent field retention increases

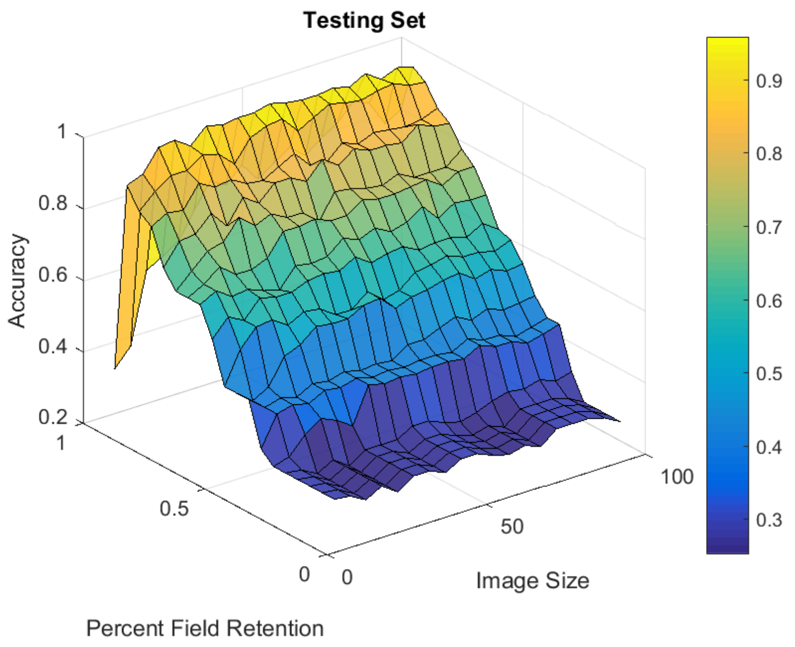

Figure 18: Plot of testing accuracy as the image size and percent field retention increases

The maximum mean testing accuracy was 95.81% and occurred at an image size of 55 by 55 and a PCA retention rate of 0.95.

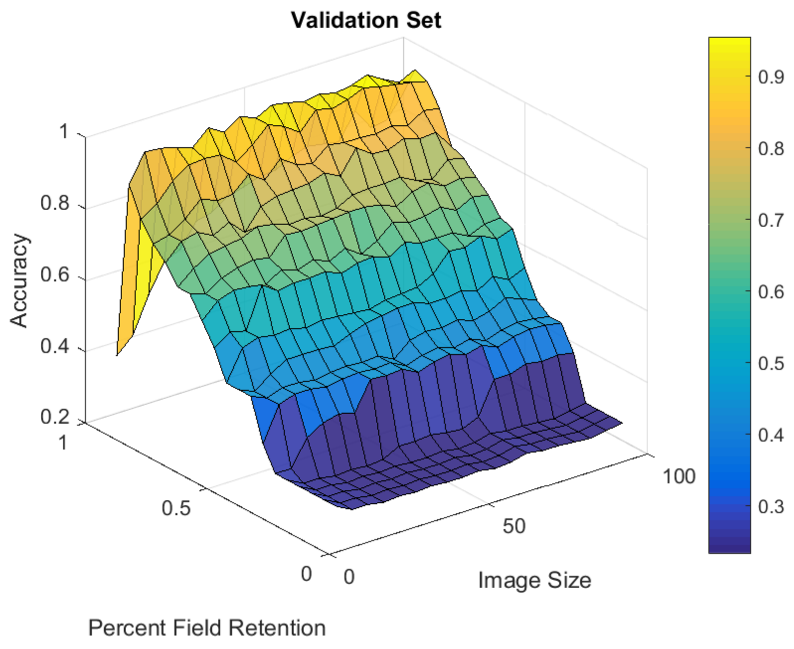

Figure 19: Plot of validation accuracy as the image size and percent field retention increases

The maximum mean validation accuracy was 95.40% and occurred at an image size of 55 by 55 and a PCA retention rate of 0.95.

The accuracy of the artificial neural network was heavily dependent upon the image size paired with the percent field retention. The image size had less of an effect than the percent field retention which can be seen by the close to horizontal line parallel to the image size axis.

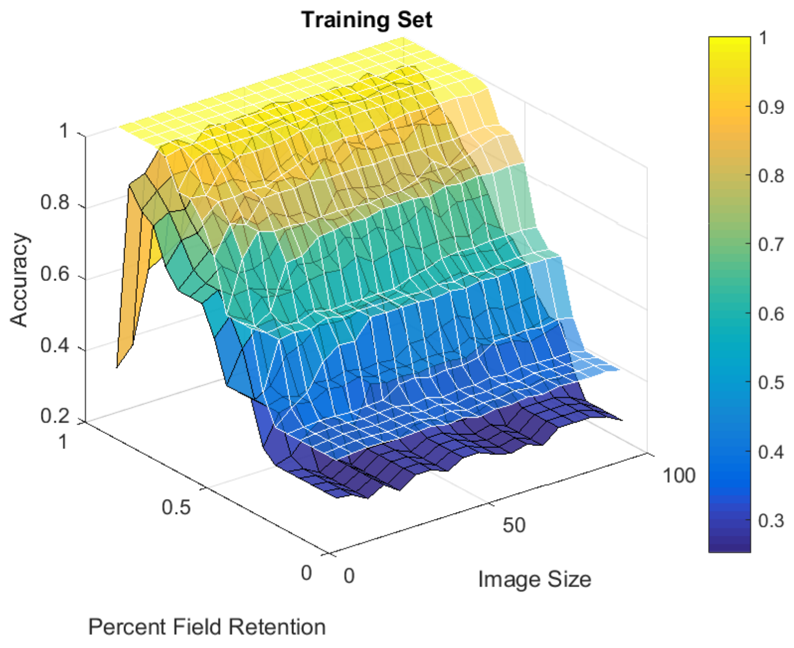

Figure 20: Plot of training and testing accuracy as the image size and percent field retention increases

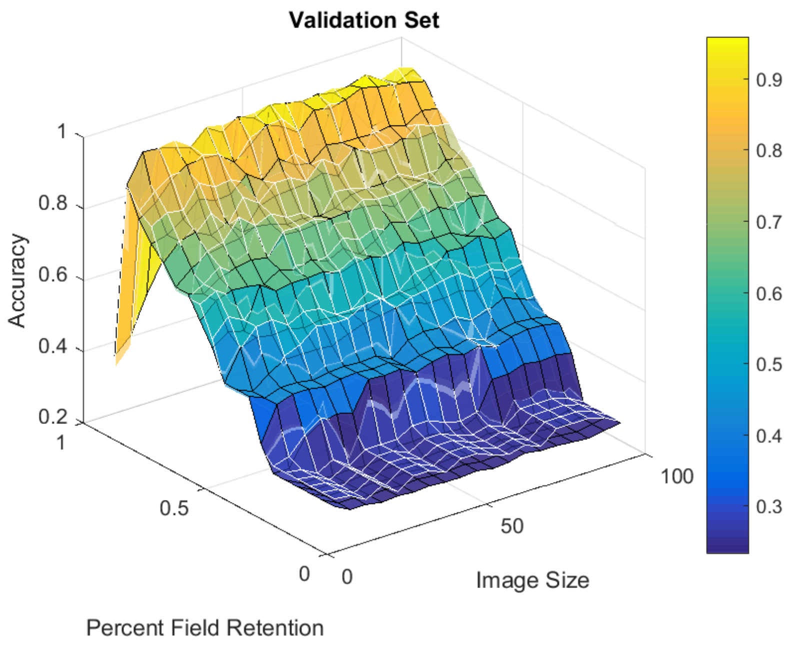

Figure 21: Plot of validation and testing accuracy as the image size and percent field retention increases

3. **Number of Hidden Nodes** The accuracy of an artificial neural network was tested with PCA applied as well as without PCA applied. The number of hidden nodes should be between the number of input nodes and the number of output nodes [5]. The number of output nodes was 15 (one for every classifier) and the number of input nodes with PCA applied was 33 and without PCA applied was the image size squared. The artificial neural network was trained using the previously investigated optimal values for image size and percent field retention of 55 by 55 and 0.95 respectively.

   Without PCA the number of hidden nodes was varied from 15 to 3025 by 50. With PCA the number of hidden nodes was varied from 15 to 38 by 1.

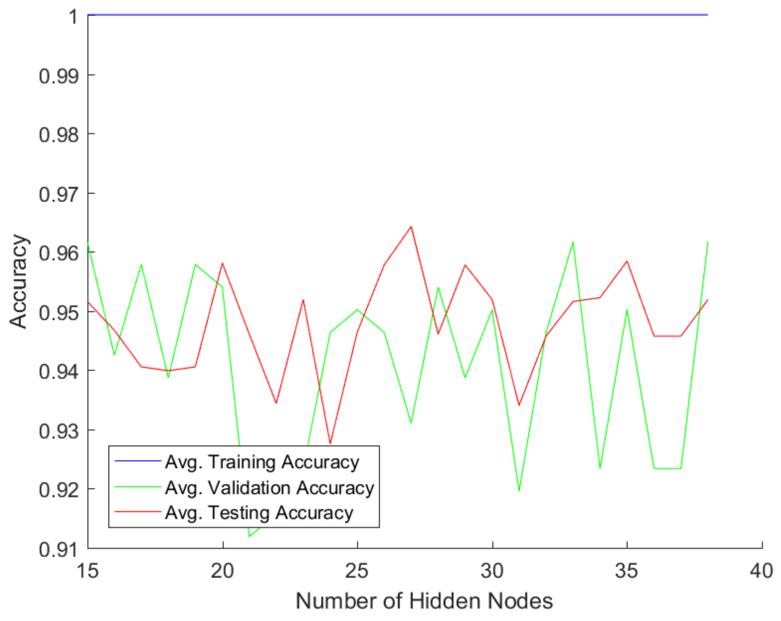

Figure 22: Plot of accuracy as the number of hidden nodes increases

The accuracy of the artificial neural network did not seem to be directly correlated to the number of hidden nodes in the hidden layer. With PCA applied, the mean testing accuracies varied from 0.9643 to 0.9276 and the mean validation accuracies varied between 0.9617 and 0.9119. Without PCA applied, the mean testing accuracy varied from 0.7950 and 0.6686 and the mean validation accuracy varied from 0.8102 and 0.6893.

## 9 Conclusions

A shallow neural network can be used for facial recognition and classification reasonably well given some data manipulation beforehand. The most important data manipulations to increase accuracy are data standardization and PCA. Every variation that included data standardization performed markedly better than those that did not. The highest valued variation that did not standardize data (NYNY) only reached a training accuracy of 40.5133% and a validation accuracy of 31.8008%. The parallel variation which did standardize the data (NYYY) reached a training accuracy of 94.0221% and a validation accuracy of 91.9540%. Similarly the highest valued variation that did not use PCA (YNYN) only reached a testing accuracy of 72.1897% and a validation accuracy of 72.4138%. The similar variation which used PCA (YNYY) was the highest performing variation and reached a testing accuracy of 95.1917% and a validation accuracy of 95.4023%.

After the importance of PCA and data standardization was determined the number of training iterations was empirically tested. This had very little effect on the accuracy of the artificial neural network past 300 to 400 iterations. The percent field retention and image size had the largest effect on the accuracy. Depending on the image size and percent field retention the mean testing accuracy ranged from 95.81% to 25.31% while the mean validation accuracy ranged from 95.40% to 23.37%.

The number of hidden nodes in the hidden node layer had very little effect on the accuracy of the artificial neural network. With PCA applied the range of the mean testing accuracies was between 96.43% to 92.76% and the mean validation accuracies varied between 96.17% and 91.19%.

By combining data manipulation techniques such as data standardization and principal component analysis a shallow neural network can be trained efficiently and to a high degree of accuracy. It is important to reduce the number of features to lessen the computational complexity and exclude features which have little to no information gain.

## References

[1] Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. *Deep learning*. Cambridge, MA: The MIT Press, 2017. Print.

[2] Ho, K.l., Y.-Y. Hsu, and C.-C. Yang. "Short Term Load Forecasting Using a Multilayer Neural Network with an Adaptive Learning Algorithm." IEEE Transactions on Power Systems 7.1 (1992): 141-49. Web.

[3] Shivdas, Ashvini E. "Face Recognition Using Artificial Neural Network." International Journal of Research in Management, Science & Technology (2014): n. pag. Web.

[4] Rowley, Henry, Shumeet Baluja, and Takeo Kanade. "Neural Network-Based Face Detection". IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE 20.1 (1998): 23-38. Print.

[5] Heaton, Jeff. *Introduction to Neural Networks for Java*. 2nd ed. N.p.: Heaton Research, n.d. Print.