CS 4732

MACHINE VISION

PROJECT 4

Deep Learning for Classification

INSTRUCTOR

Dr. Mahmut KARAKAYA

Jose Rodriguez

000976962

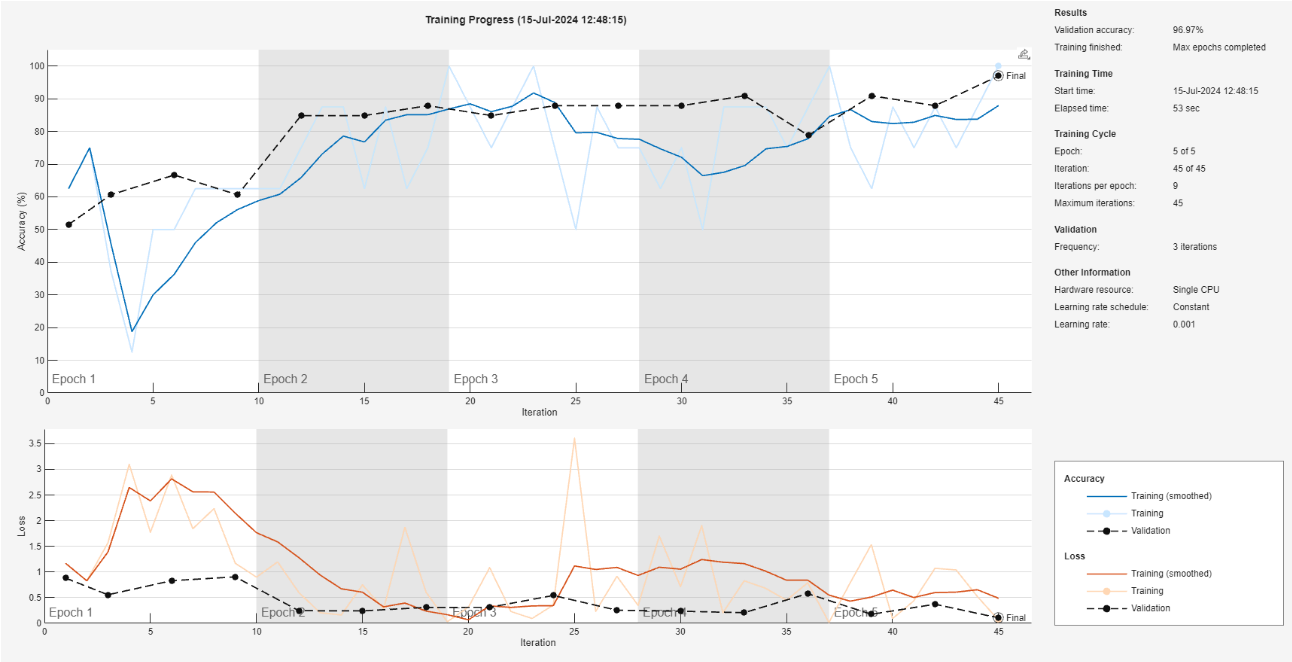
1. Abstract

In our final project, we are task with using deep learning in order to detect Diabetic Retinopathy That can be found in the eyes, In order to do that task we were asked to use a pretrained cnn like AlexNet. Upon working on this project I used many MATLAB pages about AlexNet and deep learning models along with other cnn's to use but ultimately choose to use AlexNet. While I'm tasked with using the data set to help locate diabetic retinopathy, I have found myself asking more questions rather than answers found, but in my search and research to complete the final project I have learned more about deep learning than I ever imaged I would achieve.

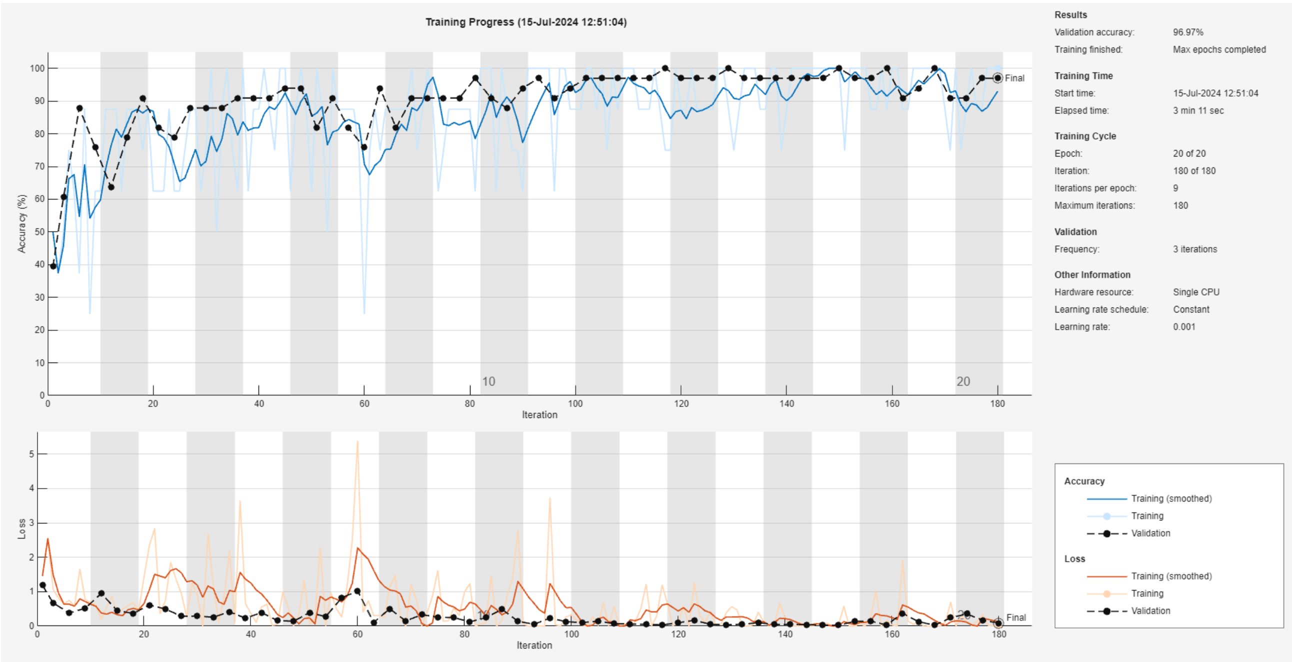
2. Test Results of Images

2.1 first attempt

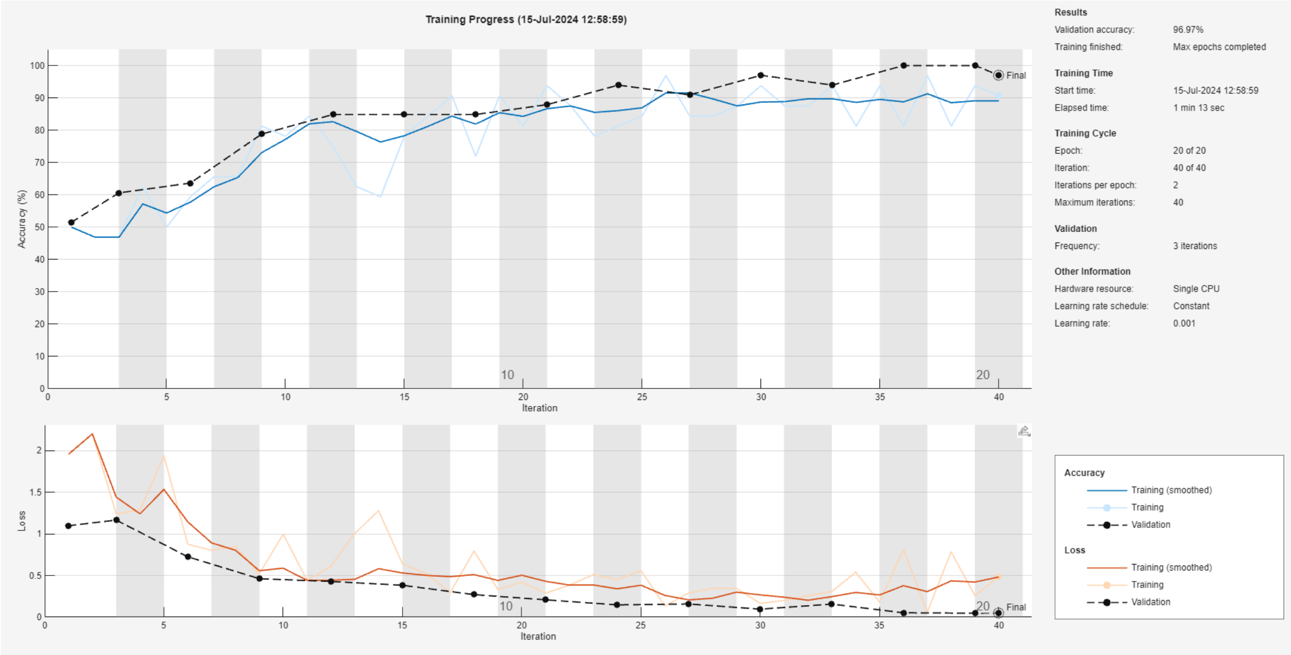
Test: 1, LR: 0.000100, Batch Size: 8, Epochs: 5, Accuracy: 0.878788
Test: 2, LR: 0.000100, Batch Size: 8, Epochs: 10, Accuracy: 0.909091
Test: 3, LR: 0.000100, Batch Size: 8, Epochs: 20, Accuracy: 0.909091
Test: 4, LR: 0.000100, Batch Size: 16, Epochs: 5, Accuracy: 0.787879
Test: 5, LR: 0.000100, Batch Size: 16, Epochs: 10, Accuracy: 0.909091
Test: 6, LR: 0.000100, Batch Size: 16, Epochs: 20, Accuracy: 0.939394
Test: 7, LR: 0.000100, Batch Size: 32, Epochs: 5, Accuracy: 0.545455
Test: 8, LR: 0.000100, Batch Size: 32, Epochs: 10, Accuracy: 0.666667
Test: 9, LR: 0.000100, Batch Size: 32, Epochs: 20, Accuracy: 0.909091
Test: 10, LR: 0.001000, Batch Size: 8, Epochs: 5, Accuracy: 0.969697
Test: 11, LR: 0.001000, Batch Size: 8, Epochs: 10, Accuracy: 0.848485
Test: 12, LR: 0.001000, Batch Size: 8, Epochs: 20, Accuracy: 0.969697
Test: 13, LR: 0.001000, Batch Size: 16, Epochs: 5, Accuracy: 0.818182
Test: 14, LR: 0.001000, Batch Size: 16, Epochs: 10, Accuracy: 0.848485
Test: 15, LR: 0.001000, Batch Size: 16, Epochs: 20, Accuracy: 0.909091
Test: 16, LR: 0.001000, Batch Size: 32, Epochs: 5, Accuracy: 0.909091
Test: 17, LR: 0.001000, Batch Size: 32, Epochs: 10, Accuracy: 0.909091
Test: 18, LR: 0.001000, Batch Size: 32, Epochs: 20, Accuracy: 0.969697
Test: 19, LR: 0.010000, Batch Size: 8, Epochs: 5, Accuracy: 0.878788
Test: 20, LR: 0.010000, Batch Size: 8, Epochs: 10, Accuracy: 0.969697
Test: 21, LR: 0.010000, Batch Size: 8, Epochs: 20, Accuracy: 0.818182
Test: 22, LR: 0.010000, Batch Size: 16, Epochs: 5, Accuracy: 0.848485
Test: 23, LR: 0.010000, Batch Size: 16, Epochs: 10, Accuracy: 0.636364
Test: 24, LR: 0.010000, Batch Size: 16, Epochs: 20, Accuracy: 0.909091
Test: 25, LR: 0.010000, Batch Size: 32, Epochs: 5, Accuracy: 0.696970
Test: 26, LR: 0.010000, Batch Size: 32, Epochs: 10, Accuracy: 0.939394
Test: 27, LR: 0.010000, Batch Size: 32, Epochs: 20, Accuracy: 0.909091



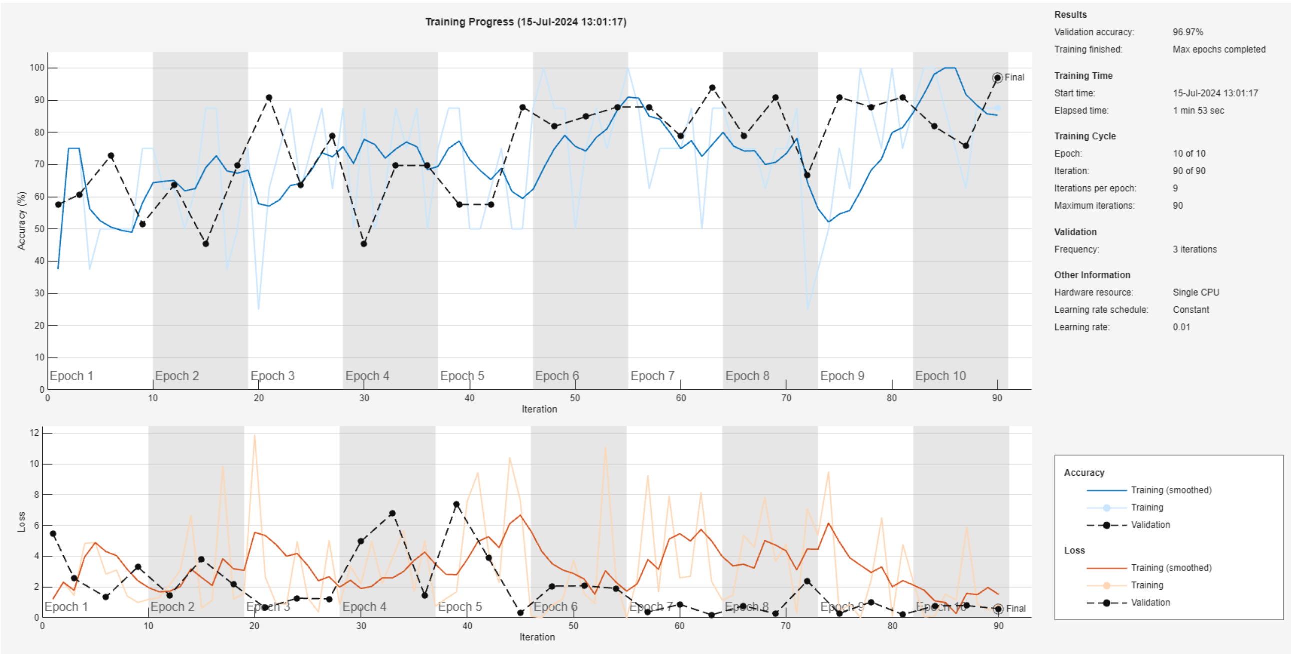
Test 10^



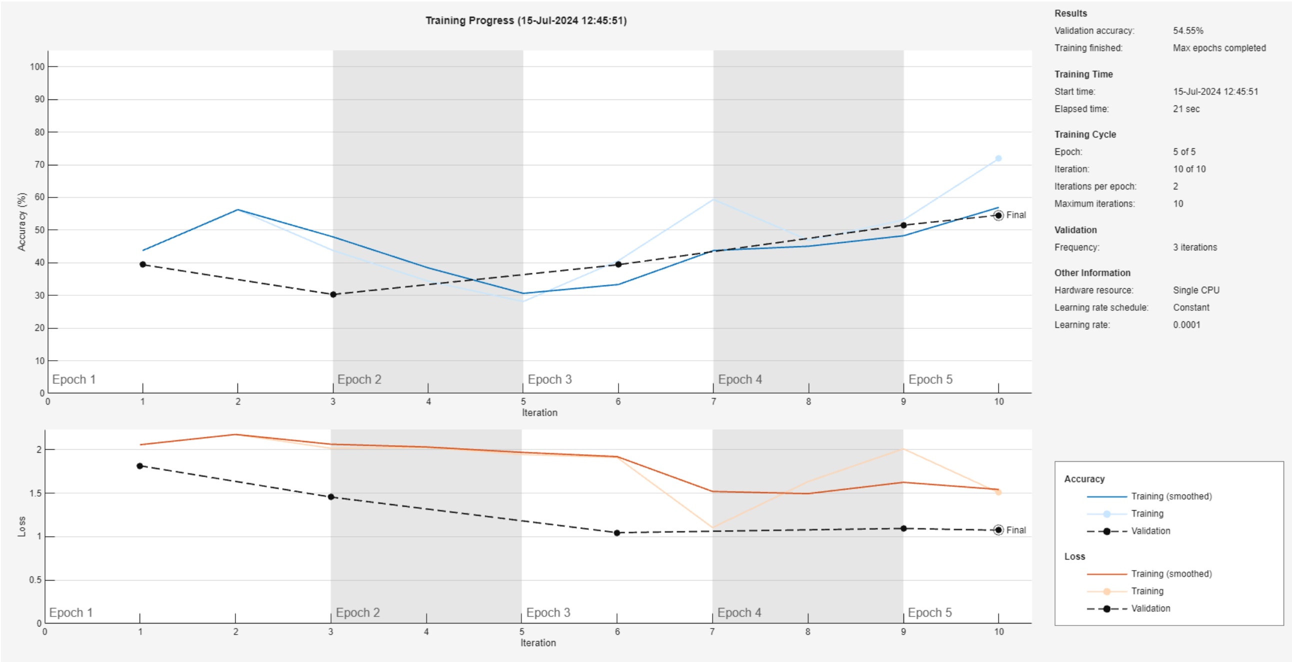
Test 12^



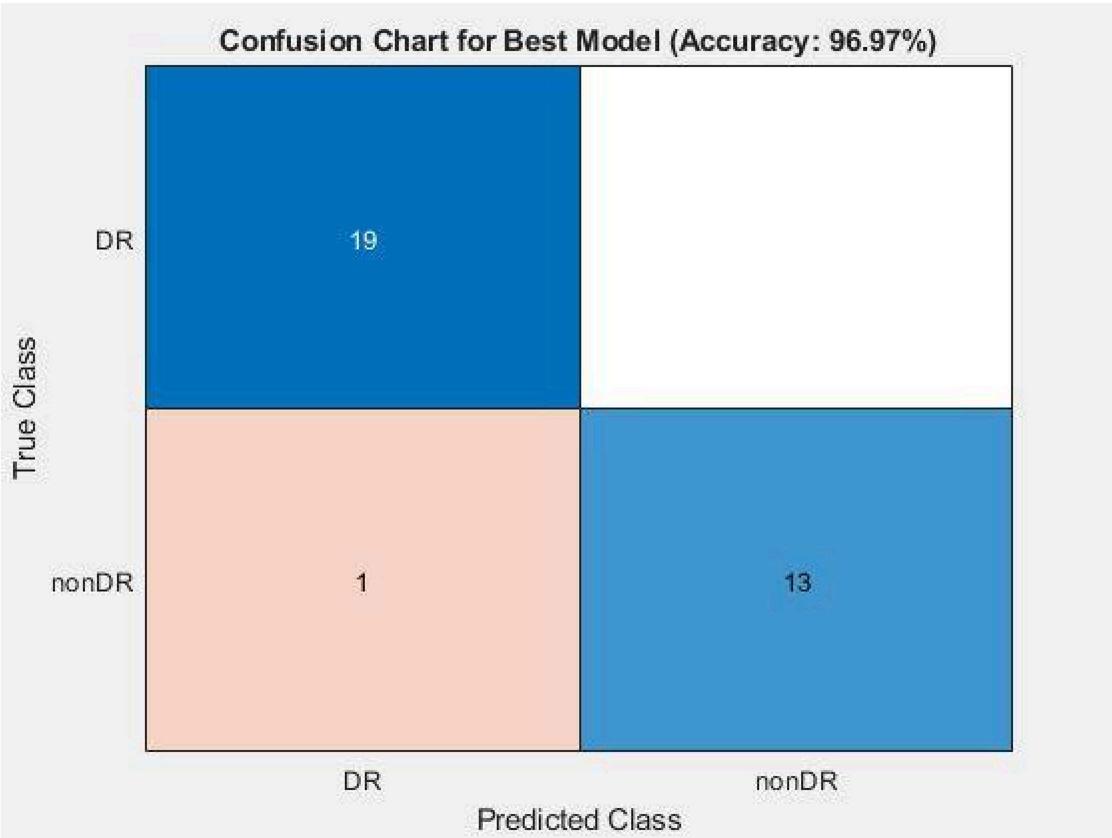
Test 18^



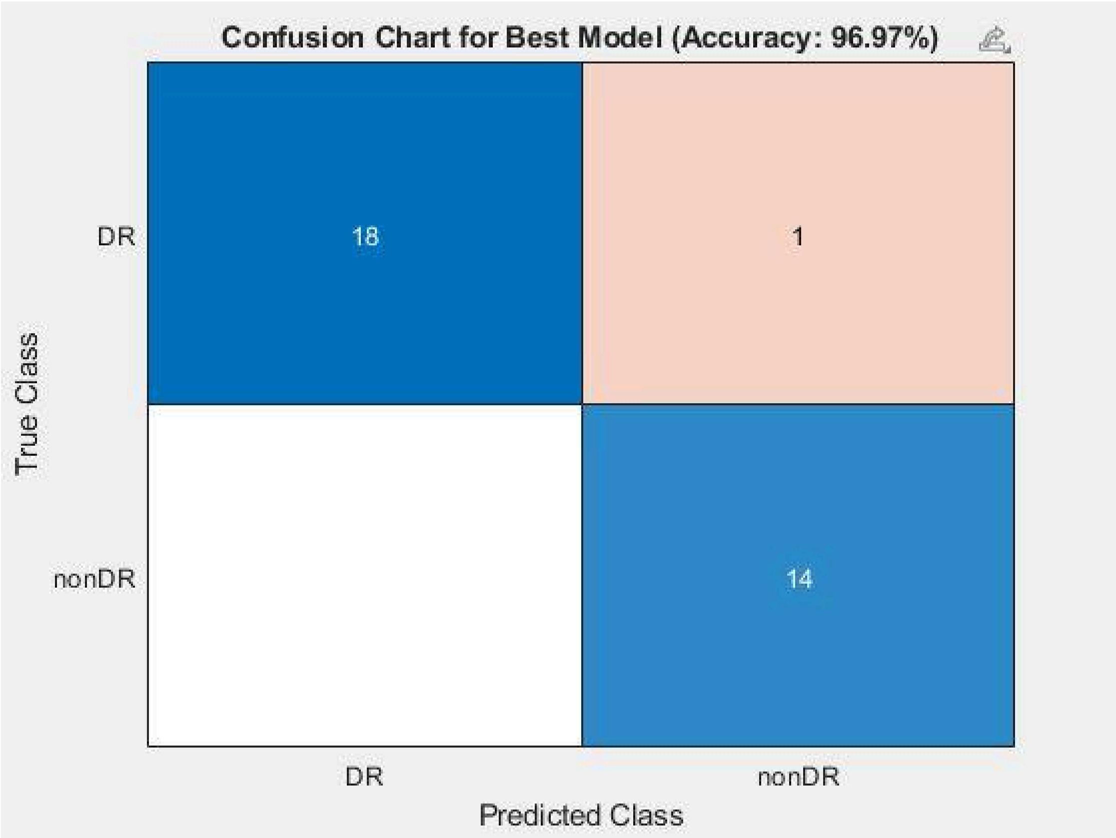
Test 20^



Worst Test 7 ^



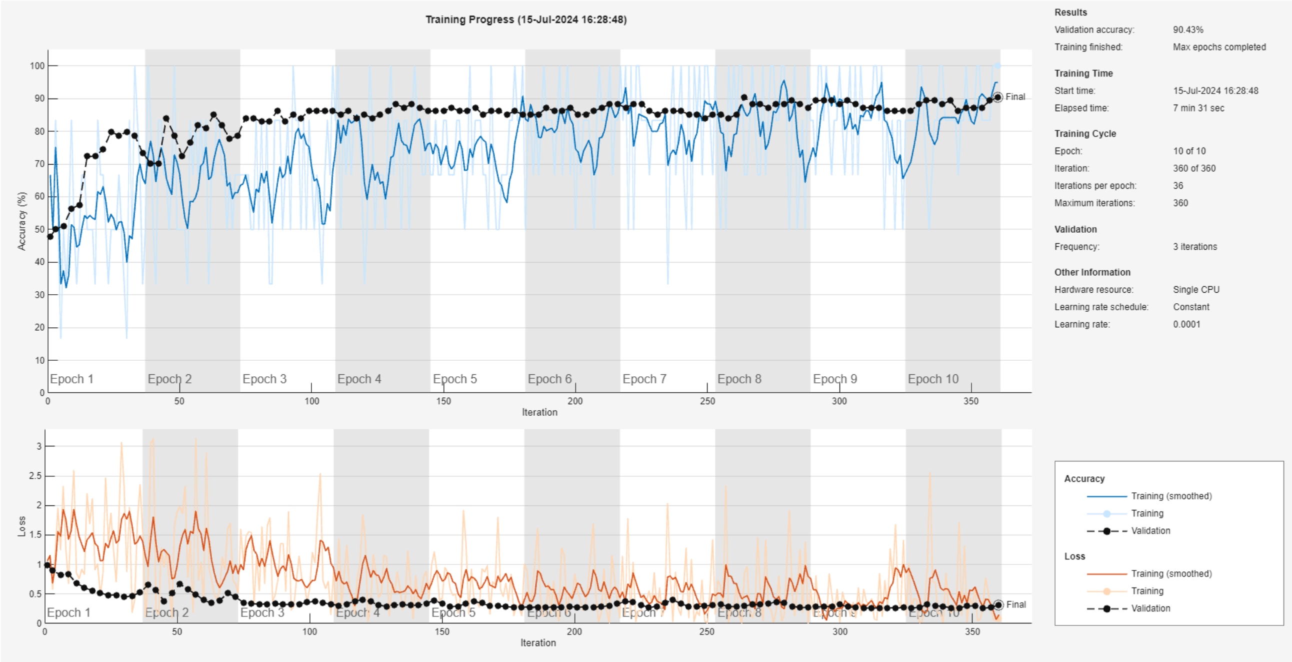
Test 10^



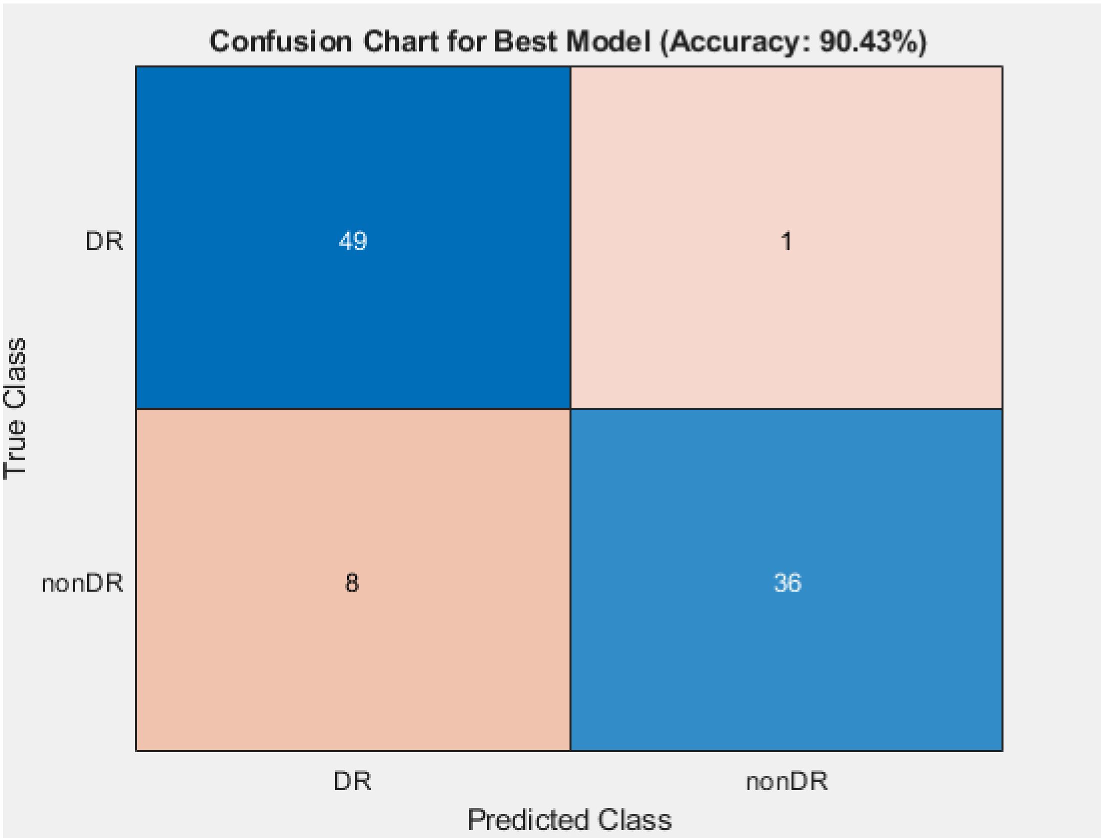
Test 12^

2.2 Final attempt

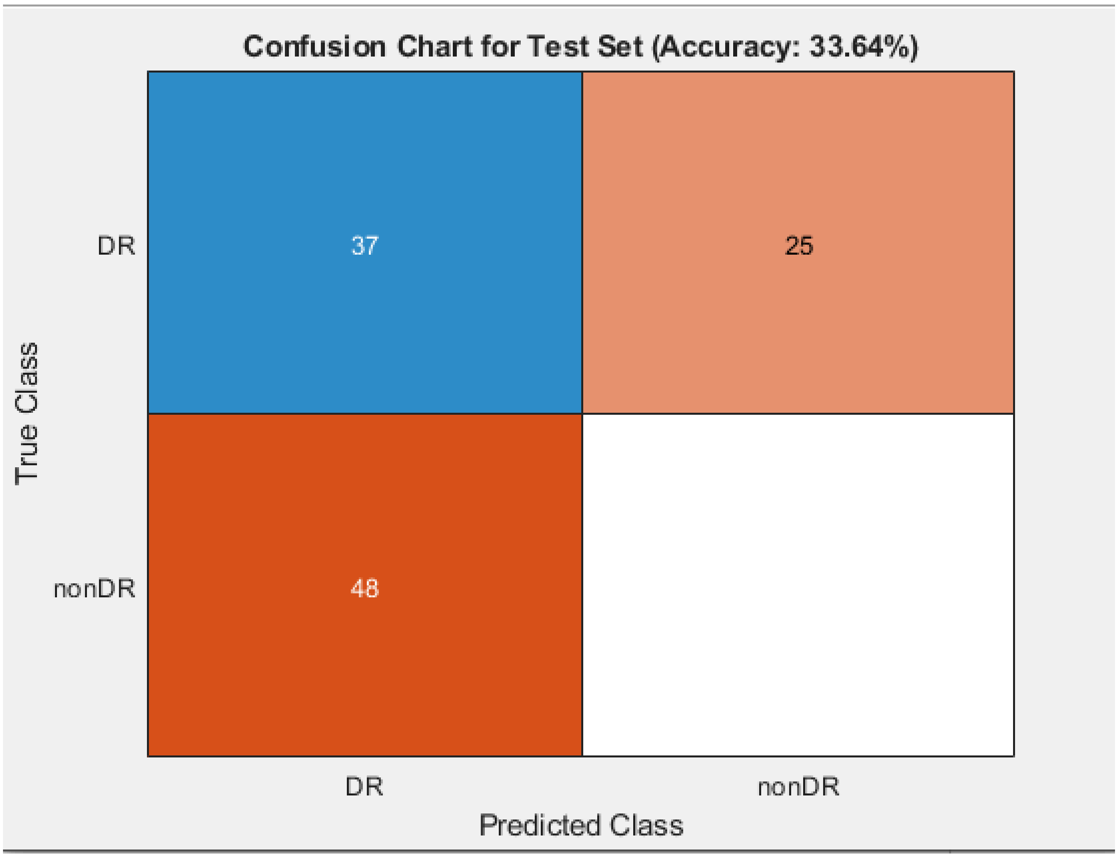
Test: 1, LR: 0.000100, Batch Size: 6, Epochs: 5, Accuracy: 0.86
Test: 2, LR: 0.000100, Batch Size: 6, Epochs: 10, Accuracy: 0.90
Test: 3, LR: 0.000100, Batch Size: 8, Epochs: 5, Accuracy: 0.84
Test: 4, LR: 0.000100, Batch Size: 8, Epochs: 10, Accuracy: 0.85
Test: 5, LR: 0.001000, Batch Size: 6, Epochs: 5, Accuracy: 0.88
Test: 6, LR: 0.001000, Batch Size: 6, Epochs: 10, Accuracy: 0.78
Test: 7, LR: 0.001000, Batch Size: 8, Epochs: 5, Accuracy: 0.86
Test: 8, LR: 0.001000, Batch Size: 8, Epochs: 10, Accuracy: 0.83
Test Accuracy: 33.64%



Test 2^ best result



Confusion chart from best training dataset ^



Confusion chart for Test dataset ^

4 Discussion

In working on this project, I have learned a lot about the AlexNet cnn and how to use it properly. While I will admit I could not accomplish all the tasks or achieve a higher accuracy result when testing the test dataset, I did learn that many of the different testing parameters led to various training progress times and results and that certain combinations will return the same accuracy. While my test data was extremely low when working with the training dataset, I did manage to achieve a 96% accuracy rate as my best model, which only had 1 one wrong prediction when looking at the matrix confusion chart. Overall, this project tested my patience and ability to find and search for ways to properly test and train the AlexNet cnn. Some of the most useful tools ended up being the slides in which our professor went over the AlexNet cnn.

5 CODES

```
% sets the path to get the data set
datasetPath = 'C:\Users\genjo\Documents\MATLAB\project_4\Train';
% grabs the data and stores it in a variable
imds = imageDatastore(datasetPath, 'IncludeSubfolders', true, 'LabelSource', 'foldernames');
The function below will resize all the images in the data set to 227x227
imds.ReadFcn = @(filename)imresize(imread(filename),[227 227]);
%splits the network into training and validation sets
[imdsTrain, imdsValidation] = splitEachLabel(imds, 0.7, 'randomized');
Loads the pretrained alexnet cnn
net = alexnet;
% Replaces the last few layers of the pretrained cnn
layersTransfer = net.Layers(1:end-3);
numClasses = numel(unique(imds.Labels)); %
layers = [
layersTransfer
fullyConnectedLayer(numClasses, 'WeightLearnRateFactor', 20, 'BiasLearnRateFactor', 20)
softmaxLayer
classificationLayer];
%gets the input size from the network
inputSize = net.Layers(1).InputSize;
```

```matlab
%sets up the augmentation parameters
pixelRange = [-30 30];
imageAugmenter = imageDataAugmenter( ...
'RandXReflection',true, ...
'RandXTranslation',pixelRange, ...
'RandYTranslation',pixelRange);
%stores the augmented data to be used later
augimdsTrain = augmentedImageDatastore(inputSize(1:2),imdsTrain, ...
'DataAugmentation',imageAugmenter);
augimdsValidation = augmentedImageDatastore(inputSize(1:2),imdsValidation);
%define different parameter for testing
learning_Rates = [1e-4, 1e-3];
Mini_Batch_Size = [6,8];
max_epochs = [5,10];
%setup variables for the most accurate set of test completed
counter = 0;
best_accuracy = 0;
best_learning_Rates = 0;
best_batchSize = 0;
best_epochs = 0;
%the nested loops below will run through all the different parameters
for i = learning_Rates
  for k = Mini_Batch_Size
     for l = max_epochs
        counter = counter + 1;
        %defines training options
        options = trainingOptions('sgdm', ...
           'MiniBatchSize', k, ...
           'MaxEpochs', l, ...
           'InitialLearnRate', i, ...
           'Shuffle', 'every-epoch', ...
           'ValidationData', augimdsValidation, ...
           'ValidationFrequency', 3, ...
           'Verbose', false, ...
           'Plots', 'training-progress', ...
           'GradientThreshold', 1, ...
           'ExecutionEnvironment', 'auto');
        %trains the network
        netTransfer = trainNetwork(augimdsTrain,layers,options);
        % updates the network on validation set
        [YPred,scores] = classify(netTransfer,augimdsValidation);
        YValidation = imdsValidation.Labels;
        accuracy = mean(YPred == YValidation);
        %prints out each results of each parameter associated
        fprintf('Test: %d, LR: %f, Batch Size: %d, Epochs: %d, Accuracy: %.2f\n',counter, i, k, l, accuracy);
        %keeps up with the most accuracte model
        if(accuracy > best_accuracy)
           best_accuracy = accuracy;
           best_learning_Rates = i;
           best_batchSize = k;
           best_epochs = l;
           best_net = netTransfer;
        end
     end
  end
end
%retest the best model and checks it
[YPred,scores] = classify(best_net,augimdsValidation);
```

```
YValidation = imdsValidation.Labels;
%creates the confusion matrix
figure;
confusionchart(YValidation, YPred);
title(sprintf('Confusion Chart for Best Model (Accuracy: %.2f%%)', best_accuracy * 100));
%the code below will use the test data set to see the results of our
%trained data
testPath = 'C:\Users\genjo\Documents\MATLAB\project_4\Test';
imdsTest = imageDatastore(testPath, 'IncludeSubfolders', true, 'LabelSource', 'foldernames');
imdsTest.ReadFcn = @(filename)imresize(imread(filename),[227 227]);
augimdsTest = augmentedImageDatastore(inputSize(1:2),imdsTest);
%uses the best model on test data
[YPredTest, scoresTest] = classify(best_net, augimdsTest);
YTest = imdsTest.Labels;
test_accuracy = mean(YPredTest == YTest);
%displays the result
fprintf('Test Accuracy: %.2f%%\n', test_accuracy * 100);
% creates the confusion matrix
figure;
confusionchart(YTest, YPredTest);
title(sprintf('Confusion Chart for Test Set (Accuracy: %.2f%%)', test_accuracy * 100));
```