Due to the requirement of the trainig a separate folder was created for training and testing data- for using that data for rerunning the code please download the folder from here -

https://drive.google.com/file/d/1MGTN6uhb-ugBO3nofgjGNFxR8_baWAjm/view?usp=sharing

The model being very large have also attached the link for the model and sent a mail with a .rar file including images folder and model to Prof Edward Jones.-

https://drive.google.com/file/d/1q0Pla-hbpUJBXwwk2vDjq0PRtoxzQHR6/view?usp=sharing

Random seed of 10 is taken and the data is split between train and validation, for data preprocessing image augmentation is done along with resizing.

```
rng(10)
DatasetPath = fullfile('C:\Users\CS-Guest-2\Desktop\Nuig\MSc_AI\MSc-AI\Embeded_Image\train_data
imds = imageDatastore(DatasetPath, ...
    'IncludeSubfolders',true,'LabelSource','foldernames');

[imdsTrain,imdsvalidation] = splitEachLabel(imds,0.8,'randomized');

pixelRange = [-30 30];
imageAugmenter = imageDataAugmenter( ...
    'RandXReflection',true, ...
    'RandXTranslation',pixelRange, ...
    'RandYTranslation',pixelRange);
augimdsTrain = augmentedImageDatastore([32 32 3],imdsTrain, ...
    'DataAugmentation',imageAugmenter);

augimdsValidation = augmentedImageDatastore([32 32 3],imdsvalidation);
```

For the training purposes the following layers are set to train the model along with the training parameters and types.

```
layers = [
    imageInputLayer([32 32 3]);

    convolution2dLayer(3,64,'Stride' ,1,'Padding',"same");
    convolution2dLayer(3,64,'Stride' ,1,'Padding','same' );
    batchNormalizationLayer;
    reluLayer;

    maxPooling2dLayer(1, 'Stride', 1);
    convolution2dLayer(3,64,'Stride' ,1,'Padding','same' );
    convolution2dLayer(3,64,'Stride' ,1,'Padding','same' );
    batchNormalizationLayer;
    reluLayer;

    maxPooling2dLayer(1, 'Stride', 1);
    convolution2dLayer(3,128,'Stride' ,1,'Padding','same' );
    convolution2dLayer(3,128,'Stride' ,1,'Padding','same' );
    convolution2dLayer(3,128,'Stride' ,1,'Padding','same' );
    batchNormalizationLayer;
    reluLayer;
```

```
    maxPooling2dLayer(1, 'Stride', 1);
    convolution2dLayer(3,256,'Stride' ,1,'Padding','same' );
    convolution2dLayer(3,256,'Stride' ,1,'Padding','same' );
    convolution2dLayer(3,256,'Stride' ,1,'Padding','same' );
    batchNormalizationLayer;
    reluLayer;

    maxPooling2dLayer(1, 'Stride', 1);
    fullyConnectedLayer(64);
    fullyConnectedLayer(32);
    fullyConnectedLayer(5);
    softmaxLayer;
    classificationLayer];
```

```
opts = trainingOptions('sgdm', ...
    'InitialLearnRate', 0.0001, ...
    'LearnRateSchedule', 'piecewise', ...
    'LearnRateDropFactor', 0.1, ...
    'LearnRateDropPeriod', 8, ...
    'Shuffle','every-epoch', ...
    'ValidationData',augimdsValidation, ...
    'L2Regularization', 0.004, ...
    'MaxEpochs', 10, ...
    'MiniBatchSize', 32, ...
    'Verbose', true);
```

```
net= trainNetwork(augimdsTrain, layers, opts);
```

```
Training on single CPU.
Initializing input data normalization.
|=========================================================================================================================
|  Epoch  |  Iteration  |  Time Elapsed  |  Mini-batch  |  Validation  |  Mini-batch  |  Validation  |  Base Learni
|         |             |   (hh:mm:ss)   |   Accuracy   |   Accuracy   |     Loss     |     Loss     |      Rate
|=========================================================================================================================
|       1 |           1 |       00:00:12 |       21.88% |       24.56% |       2.6970 |       2.2799 |      1.0000e
|       2 |          50 |       00:03:32 |       43.75% |       28.51% |       4.0929 |       5.8275 |      1.0000e
|       4 |         100 |       00:07:20 |       34.38% |       22.81% |       3.2000 |       4.7795 |      1.0000e
|       6 |         150 |       00:11:15 |       31.25% |       28.51% |       3.2321 |       3.3935 |      1.0000e
|       8 |         200 |       00:15:13 |       28.12% |       26.32% |       1.9250 |       3.1777 |      1.0000e
|       9 |         250 |       00:19:17 |       46.88% |       32.46% |       1.2791 |       3.4961 |      1.0000e
|      10 |         280 |       00:21:49 |       53.12% |       29.39% |       1.0940 |       3.3705 |      1.0000e
|=========================================================================================================================
```

Storing the trained model

```
net_model = net
```

```
net_model =
  SeriesNetwork with properties:

         Layers: [28×1 nnet.cnn.layer.Layer]
```

```
     InputNames: {'imageinput'}
    OutputNames: {'classoutput'}
```

```
save('modelfinal.mat','net_model')
```

Loading the model

```
file = load('modelfinal.mat')
```

```
file = struct with fields:
    net_model: [1×1 SeriesNetwork]
```

```
model = file.net_model
```

```
model =
  SeriesNetwork with properties:

         Layers: [28×1 nnet.cnn.layer.Layer]
     InputNames: {'imageinput'}
    OutputNames: {'classoutput'}
```

Model tested for accuracy on the test data

```
DatasetPath = fullfile('C:\Users\CS-Guest-2\Desktop\Nuig\MSc_AI\MSc-AI\Embeded_Image\test_data
imds_test = imageDatastore(DatasetPath, ...
    'IncludeSubfolders',true,'LabelSource','foldernames');

augimdstest = augmentedImageDatastore([32 32 3],imds_test)
```

```
augimdstest =
  augmentedImageDatastore with properties:

           NumObservations: 387
                     Files: {387×1 cell}
   AlternateFileSystemRoots: {}
              MiniBatchSize: 128
          DataAugmentation: 'none'
        ColorPreprocessing: 'none'
                OutputSize: [32 32]
            OutputSizeMode: 'resize'
       DispatchInBackground: 0
```

Model classified with the test data and confusion matrix examined

```
labels = classify(model, augimdstest);
confMat = confusionmat(imds_test.Labels, labels);
confMat = confMat./sum(confMat,2);
mean(diag(confMat))
```

```
ans = 0.5175
```

The overall model accuracy is calculated, the data folder contains all the speed limit folders.(20,30,50,80,100)

```
DatasetPath = fullfile('C:\Users\CS-Guest-2\Desktop\Nuig\MSc_AI\MSc-AI\Embeded_Image\data');
imds_testall = imageDatastore(DatasetPath, ...
```

```
    'IncludeSubfolders',true,'LabelSource','foldernames');

augimdstestall = augmentedImageDatastore([32 32 3],imds_testall)
```

```
augimdstestall =
  augmentedImageDatastore with properties:

            NumObservations: 1528
                      Files: {1528×1 cell}
    AlternateFileSystemRoots: {}
               MiniBatchSize: 128
          DataAugmentation: 'none'
         ColorPreprocessing: 'none'
                 OutputSize: [32 32]
             OutputSizeMode: 'resize'
        DispatchInBackground: 0
```

```
labelsall = classify(model, augimdstestall);

confMat = confusionmat(imds_testall.Labels, labelsall);
confMat = confMat./sum(confMat,2);
mean(diag(confMat))
```

```
ans = 0.5372
```

The accuracy for the stress data is calculated and and to detect the circles imfindcircle is used and a fixed pixel around the detected data is cropeed to pass for classification.

```
labelstress = {'40', '40', '100', '30', '20','20','20','40', '40','40', '40','40', '40','40', '
C = categorical(labelstress)
```

```
C = 1×16 categorical
40          40          100         30          20          20          20 ···
```

```
c = 0;
path = 'C:\Users\CS-Guest-2\Desktop\Nuig\MSc_AI\MSc-AI\Embeded_Image\stress';
imagefiles = dir(fullfile(path,'*.TIF'));
for i=1:length(imagefiles)
    filename = fullfile(path,imagefiles(i).name);
    I = imread(filename);
    [centers,radii] = imfindcircles(I,[9 70],'ObjectPolarity','bright', ...
    'Sensitivity',0.82);
    if centers >0
        x = centers(1) - (1.5 * radii);
        X = 70;
        y = centers(2) - (1.5 * radii);
        Y = 70;
        img = imcrop(I,[x y X Y]);
        img1 = augmentedImageDatastore([32 32 3],img);
        lbl = classify(model,img1);
        if lbl == C(i)
            c = c + 1;
        end
    end
end
```

```
acc = c / numel(imagefiles) * 100;
fprintf('Accuracy for stress dataset is %d ',acc );
```

```
Accuracy for stress dataset is 0
```

Accuracy for all the speed limit categories is individually calculated.

```
c1 = 0;
L20 = {'20'};
L20 = categorical(L20);
pathtwenty = 'C:\Users\CS-Guest-2\Desktop\Nuig\MSc_AI\MSc-AI\Embeded_Image\data\20';
imagefiles20 = dir(fullfile(pathtwenty,'*.jpg'));
for i=1:length(imagefiles20)
    filename20 = fullfile(pathtwenty,imagefiles20(i).name);
    I = imread(filename20);
    img20 = augmentedImageDatastore([32 32 3],I);
    lbl20 = classify(model,img20);
    if lbl20 == L20(1)
        c1 = c1 + 1;
    end
end
acc = c1 / numel(imagefiles20) * 100;
fprintf('Accuracy for 20 dataset is %d ',acc );
```

```
Accuracy for 20 dataset is 4.857143e+01
```

```
c2 = 0;
L30 = {'30'};
L30 = categorical(L30);
paththirty = 'C:\Users\CS-Guest-2\Desktop\Nuig\MSc_AI\MSc-AI\Embeded_Image\data\30';
imagefiles30 = dir(fullfile(paththirty,'*.jpg'));
for i=1:length(imagefiles30)
    filename30 = fullfile(paththirty,imagefiles30(i).name);
    I = imread(filename30);
    img30 = augmentedImageDatastore([32 32 3],I);
    lbl30 = classify(model,img30);
    if lbl30 == L30(1)
        c2 = c2 + 1;
    end
end
acc30 = c2 / numel(imagefiles20) * 100;
fprintf('Accuracy for 30 dataset is %d ',acc30 );
```

```
Accuracy for 30 dataset is 4.809524e+01
```

```
c3 = 0;
L50 = {'50'};
L50 = categorical(L50);
pathfifty = 'C:\Users\CS-Guest-2\Desktop\Nuig\MSc_AI\MSc-AI\Embeded_Image\data\50';
imagefiles50 = dir(fullfile(pathfifty,'*.jpg'));
for i=1:length(imagefiles50)
    filename50 = fullfile(pathfifty,imagefiles50(i).name);
```

```
    I = imread(filename50);
    img50 = augmentedImageDatastore([32 32 3],I);
    lbl50 = classify(model,img50);
    if lbl50 == L50(1)
        c3 = c3 + 1;
    end
end
acc50 = c3 / numel(imagefiles50) * 100;
fprintf('Accuracy for 50 dataset is %d ',acc50 );
```

```
Accuracy for 50 dataset is 7.757576e+01
```

```
c4 = 0;
L80 = {'80'};
L80 = categorical(L80);
patheighty = 'C:\Users\CS-Guest-2\Desktop\Nuig\MSc_AI\MSc-AI\Embeded_Image\data\80';
imagefiles80 = dir(fullfile(patheighty,'*.jpg'));
for i=1:length(imagefiles80)
    filename80 = fullfile(patheighty,imagefiles80(i).name);
    I = imread(filename80);
    img80 = augmentedImageDatastore([32 32 3],I);
    lbl80 = classify(model,img80);
    if lbl80 == L80(1)
        c4 = c4 + 1;
    end
end
acc80 = c4 / numel(imagefiles80) * 100;
fprintf('Accuracy for 80 dataset is %d ',acc80 );
```

```
Accuracy for 80 dataset is 4.498480e+01
```

```
c5 = 0;
L100 = {'100'};
L100 = categorical(L100);
pathhun = 'C:\Users\CS-Guest-2\Desktop\Nuig\MSc_AI\MSc-AI\Embeded_Image\data\100';
imagefiles100 = dir(fullfile(pathhun,'*.jpg'));
for i=1:length(imagefiles100)
    filename100 = fullfile(pathhun,imagefiles100(i).name);
    I = imread(filename100);
    img100 = augmentedImageDatastore([32 32 3],I);
    lbl100 = classify(model,img100);
    if lbl100 == L100(1)
        c5 = c5 + 1;
    end
end
acc100 = c5 / numel(imagefiles100) * 100;
fprintf('Accuracy for 100 dataset is %d ',acc100 );
```

```
Accuracy for 100 dataset is 6.686930e+01
```