Exp. No. 1 **Date:**

Vision Based Defect Detection System

Objective:

To develop an Artificial Intelligence (AI) system for detecting defects in automotive parts using vision system and train the dataset using AlexNet.

Software required:

- MATLAB software R2024a

Steps to be followed:

- **Data Collection**
  - The dataset consists of over all 800 images in which 80% taken for training and 20% taken for testing
    - Training images: 740 images
    - Testing images : 260 images

Defective sample image

Non defective sample image

- **Data pre-processing**
  - All the training and testing images were resized into 227 x 227 size.

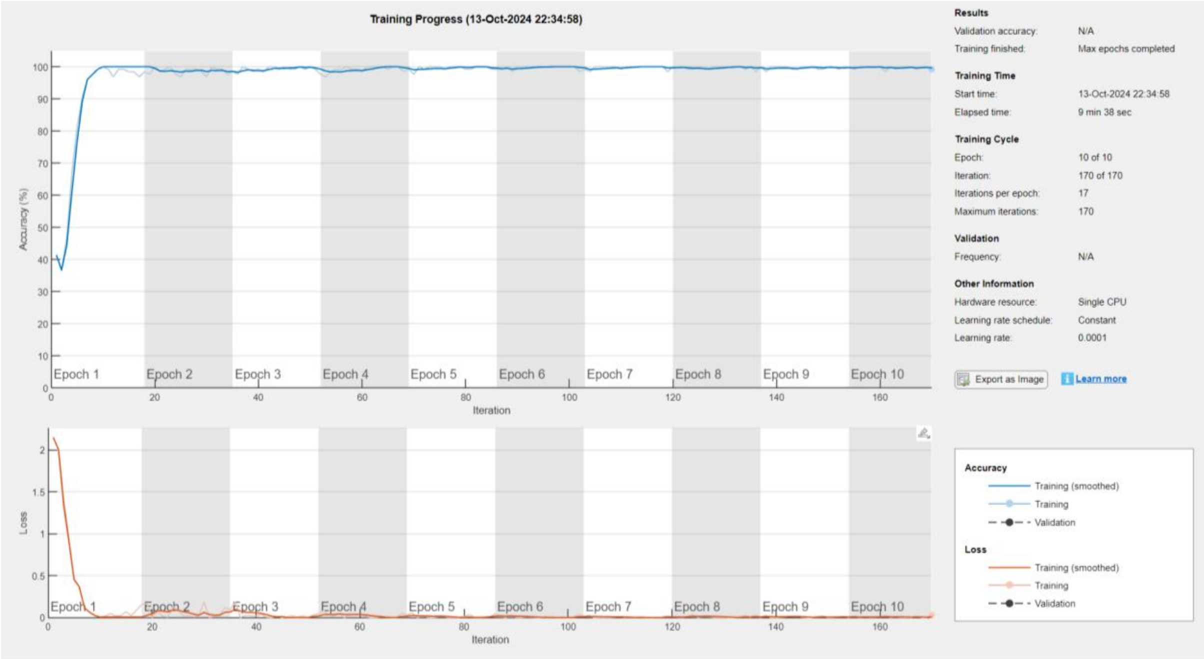
- **Data Augmentation**
  - NIL.

- **Training**
  - Network used for training is Alexnet.
  - It has 25 layers including 5 convolutional layers.
  - The size of the input image must be in 227*227.
  - The max epochs used is 10 and done 170 iteration.
  - Training and testing split up is done on 80% and 20%

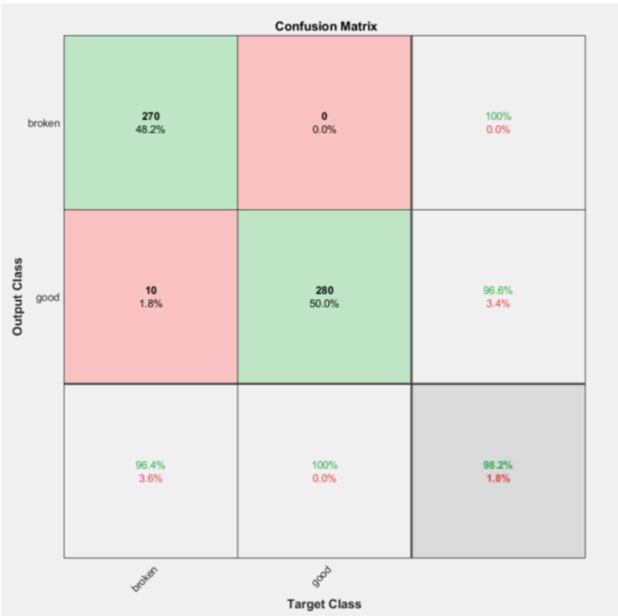
MATLAB code:

```
clc
clear all
close all
%loading the pre-trained network
net = alexnet
%obtain the layer details in Alexnet
layers = net.Layers
%obtain the first layer complete details
inputLayer = layers(1)
%obtain the last layer complete details
outputLayer = layers(end)
%check the nature of classes for which alexnet is pre-trained
classes = outputLayer.Classes
%load the defect detection dataset
imds=imageDatastore('D:\matlab\harish\ring-Defective-Detection-and-Classification-master\ring-
Defective-Detection-and-Classification-
master\data\train','IncludeSubfolders',true,'LabelSource','foldernames')
%count the images in each lebel of subfolders
tbl = countEachLabel(imds);
%split the images into training 80% and testing 20%
[trainIm,testIm] = splitEachLabel(imds,0.8)
%resize the images according the Alexnet input layer size
imageSize = net.Layers(1).InputSize;
trainData=augmentedImageDatastore(imageSize,trainIm,"ColorPreprocessing","gray2rgb")
testData=augmentedImageDatastore(imageSize,testIm,"ColorPreprocessing","gray2rgb")
%change the output classes to the defect detection dataset classes
layers(23) = fullyConnectedLayer(2)
layers(end) = classificationLayer()
%mention the training options
options = trainingOptions("sgdm","Plots","training-progress","InitialLearnRate",1e-
4,"MaxEpochs",10);
%train the Alexnet for the defect detection dataset
defectnet = trainNetwork(trainData,layers,options);
%validate the trained network using test data and plot the confusion matrix
testPred = classify(defectnet,testData)
testAcc = nnz(testPred == testIm.Labels)/numel(testPred)
plotconfusion(testIm.Labels,testPred)
```

Training progress plot:



- **Testing and performance evaluation**
  - Test image accuracy : 97.21



Inference:

- Machine Learning helps computers learn from data without needing specific instructions.
- We use this method to teach computers how to recognize patterns in data.
- The computer learned to find mistakes very well

Exp. No. 2 **Date:**

Vision Based Model Identification System using VGG16

Objective:

To develop an Artificial Intelligence (AI) system for identifying the models in automotive parts using vision system and train the dataset using VGG16

Software required:

MATLAB software

Steps to be followed:

Data Collection

The dataset consists of over all 2934 images in which 80% taken for training and 20% taken for testing

Training images:947
Testing images:187

Sample Model 1(Radiator)

Sample model 2 (Radiator Fan)

Data pre-processing

All the training and testing images were resized into 224x 224 size

Data Augmentation

Nil

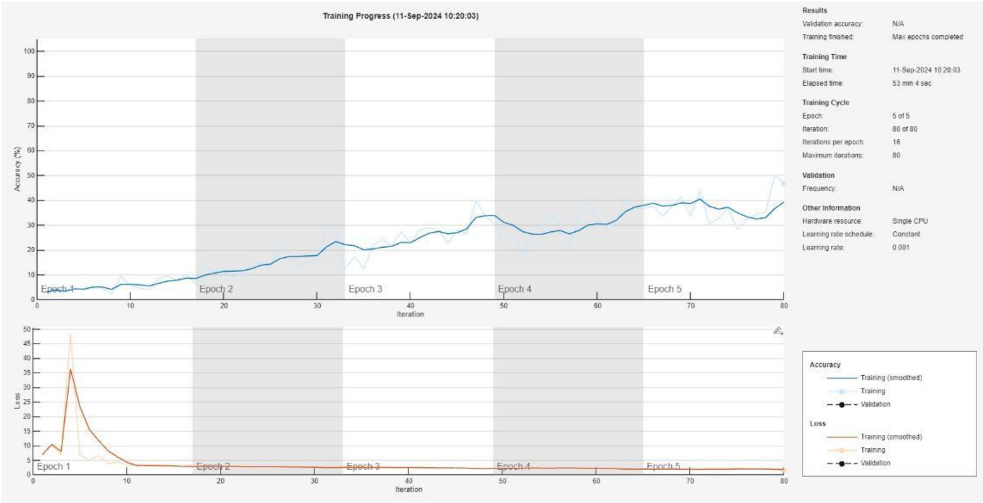
Training

Example:
Network used for training and its details: **VGG16 net**
Training and testing split up details

The dataset consists of over all 2934 images in which 80% taken for training and 20% taken for testing

Training images: 2347, Testing images:
587 Number of epochs:5
Learning rate:0.001

Training progress plot:

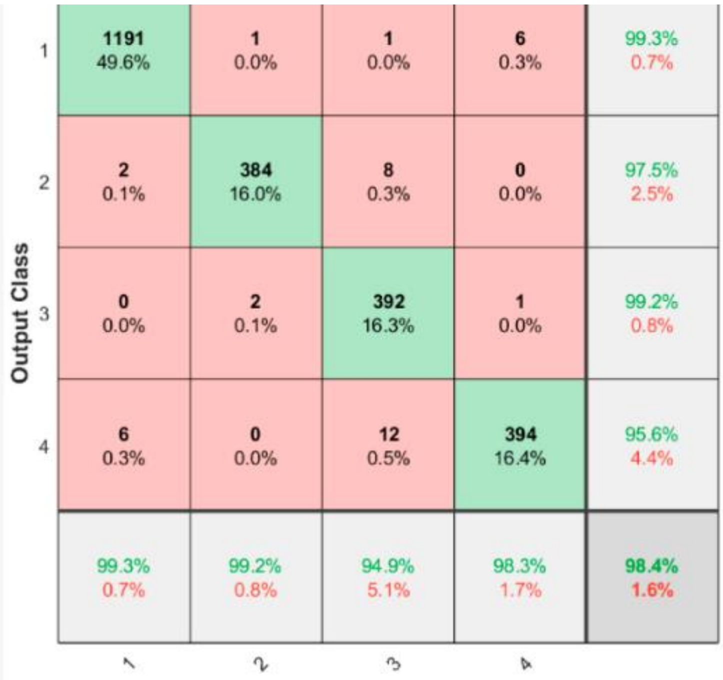


Testing and performance evaluation

- Mention testing accuracy $\text{Accuracy} = \dfrac{TP + TN}{TP + FP + TN + FN}$

- Test Image Accuracy:98.4%

- Confusion matrix diagram



MATLAB Code:

```
clc;
clear;
close all;
```

Loading the vgg16 pretrained network

```
net=vgg16
layers=net.Layers
inputLayer=layers(1)
outputLayer=layers(end)
classes=outputLayer.Classes
```

Dataset

```
imds=imageDatastore('D:\Harish\AI\mech parts',...
    'IncludeSubfolders',true,'LabelSource','foldernames')
```

Split

```
[trainIm,testIm]=splitEachLabel(imds,0.7)
```

Resize

```
traindata=augmentedImageDatastore([224 224 3],trainIm',
ColorPreprocessing="gray2rgb")
testdata=augmentedImageDatastore([224 224
3],testIm,ColorPreprocessing="gray2rgb")
```

modify

```
layers(39)=fullyConnectedLayer(4)
layers(end)=classificationLayer()
options=trainingOptions("adam","Plots","training-progress", ...
    "InitialLearnRate",0.001 ,...
    "MaxEpochs",5);
defectnet=trainNetwork(traindata,layers,options);
```

Test

```
testpred=classify(defectnet,testdata)
testAcc=nnz(testpred == testIm.Labels)/numel(testpred)
cm=plotconfusion(testIm.Labels,testpred)
```

% Load the image

```
img = imread('path/to/image.jpg');
```

% Resize the image to 224x224x3

```
img = imresize(img, [224 224]);
```

% Convert the image to a datastore

```
img_ds = augmentedImageDatastore([224 224 3], img,
ColorPreprocessing="gray2rgb");
```

```
% Get the predicted label for the image
pred = classify(defectnet, img_ds);

% Get the index of the maximum probability
[~, idx] = max(pred);

% Get the corresponding label
label = classes(idx);

% Display the label
fprintf('The image is classified as: %s\n', label);
```

Inference:

- VGG16 is a 16-layer Convolutional Neural Network (CNN) capable of classifying images into 1000 different categories.
- It is recognized as one of the top-performing computer vision models.
- VGG16 is commonly used for object detection and classification tasks.
- Vision-based model identification using VGG16 has demonstrated an accuracy of 98.4%.

Exp. No. 3 **Date:**

Vision Based Model Identification System using Training from Scratch

Objective:

To develop an Artificial Intelligence (AI) system for identifying the models in automotive parts using vision system and train the dataset using training from scratch network

Software required:

MATLAB software

Steps to be followed:

Data Collection

The dataset consists of over all 2138 images in which 70% taken for training and 30% taken for testing

Training images:1500
Testing images:638

Battery

Camshaft

Compressor

Fuel Injector

Data pre-processing

All the training and testing images were resized into 227 x 227 size

Data Augmentation

Nil

Training

Training and testing split up details

The dataset consists of over all 2138 images in which 70% taken for training and 30% taken for testing

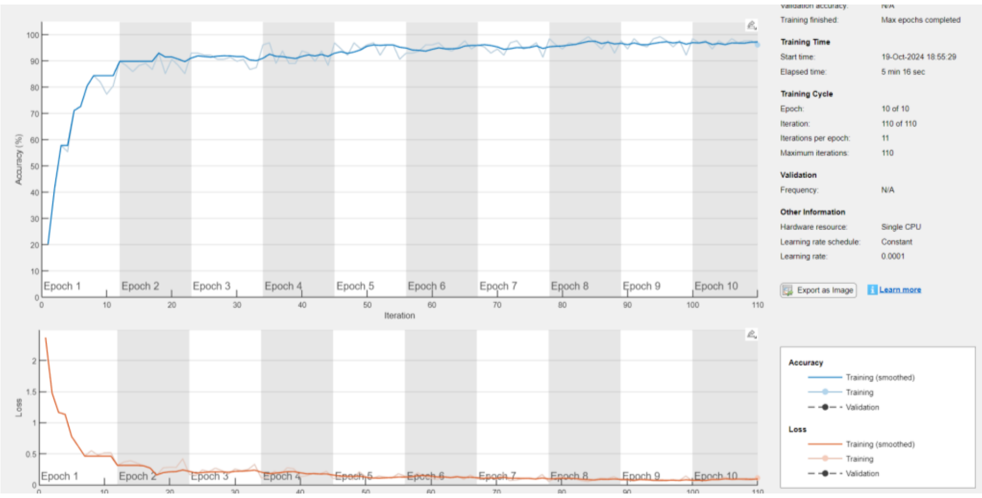
Training images:1500 Testing images:638

Optimization algorithm:Adam

Number of epochs: 20

Learning rate:0.001

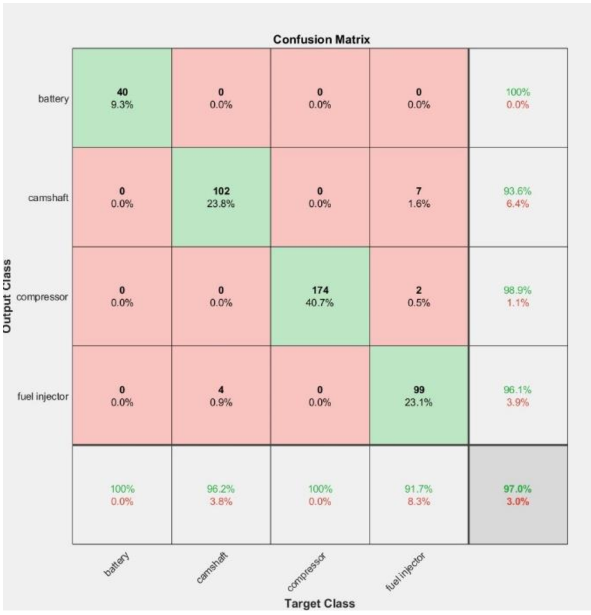
Training progress plot:



Testing and performance evaluation

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN}$$

Test Image Accuracy: 95%



Confusion matrix diagram

MATLAB Code:

```
clc
clear;
close all;
imds=imageDatastore(''D:\Harish\AI\mech parts',...
    'IncludeSubfolders',true,'LabelSource','foldernames')
tbl=countEachLabel(imds);
[trainIm,testIm]=splitEachLabel(imds,0.7);
inputSize=[227 227]
traindata=augmentedImageDatastore([227 227 3],trainIm)
testdata=augmentedImageDatastore([227 227 3],testIm)
layers=[
    imageInputLayer([inputSize 3], ...
        Normalization="rescale-zero-one",Max=2,Min=0);

        convolution2dLayer(3,8,Padding="same")
        batchNormalizationLayer
        reluLayer

        maxPooling2dLayer(2,Stride=2)

        convolution2dLayer(3,16,Padding="same")
        batchNormalizationLayer
        reluLayer

        maxPooling2dLayer(2,Stride=2)

        convolution2dLayer(3,32,Padding="same")
        batchNormalizationLayer
        reluLayer

        maxPooling2dLayer(2,Stride=2)

        convolution2dLayer(3,64,Padding="same")
        batchNormalizationLayer
        reluLayer

        maxPooling2dLayer(2,Stride=2)

        convolution2dLayer(3,128,Padding="same")
        batchNormalizationLayer
        reluLayer

        maxPooling2dLayer(2,Stride=2)
        dropoutLayer
fullyConnectedLayer(2)
```

```
        softmaxLayer
        classificationLayer;

]

    options=trainingOptions("adam","Plots","training-progress", ...
        "InitialLearnRate",0.001 ,...
        "MaxEpochs",5);
    defectnet=trainNetwork(traindata,layers,options);
    testpred=classify(defectnet,testdata)
    testAcc=nnz(testpred == testIm.Labels)/numel(testpred)
    cm=plotconfusion(testIm.Labels,testpred)
```

Inference:

- No pre-trained models are used, you create the entire CNN model.
- Requires a significant dataset of car part images for the model to learn from.
- All aspects, from basic shapes to complex features, are learned from scratch.
- Model effectiveness relies on the training data's quality and comprehensiveness.

Exp. No. 4 **Date:**

Supervised and non-supervised AI learning for machining operation

Objective:

To develop an Artificial Intelligence (AI) system for machining operation in automotive parts using supervised and non-supervised AI learning algorithm

Software required:

MATLAB software

Steps to be followed:

Data Collection

The dataset consists of over all 600 images in which 80% taken for training and 20% taken for testing

Training images:480
Testing images:120

Finished workpiece sample image

Unfinished workpiece sample image

Data pre-processing

Features are extracted from the source objects of finished and unfinished workpieces

Data Augmentation

NIL

Training

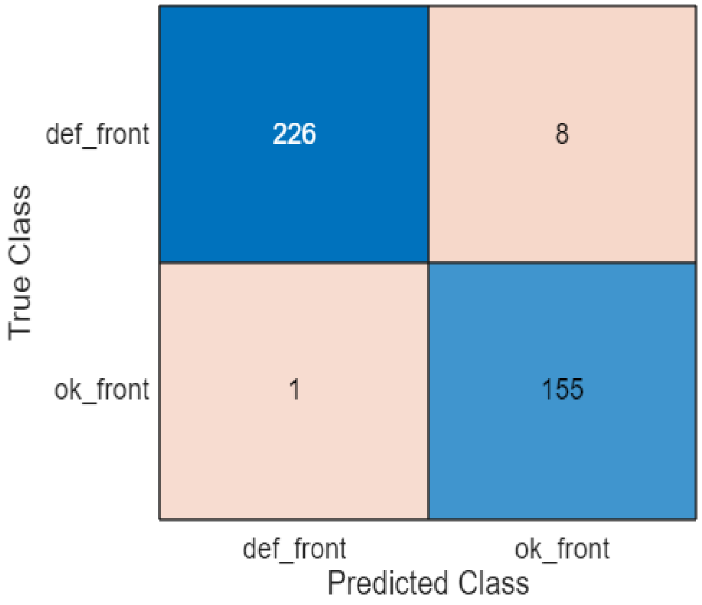
Network used to extract the features:Alexnet
Machine learning algorithm used for training:
Number of train features:

Testing and performance evaluation

Testing Accuracy: 100%

Confusion matrix diagram

.



MATLAB Code:

```
clc;
clear ;
close all;
Read and resize
net=alexnet
layers=net.Layers
imds=imageDatastore('D:\harish\AI\screw',...
    'IncludeSubfolders',true,'LabelSource','foldernames')

[trainIm,testIm]=splitEachLabel(imds,0.8)
```

```
traindata=augmentedImageDatastore([227 227 3],trainIm)
testdata=augmentedImageDatastore([227 227 3],testIm)

trainfeatures=activations(net,traindata, ...
    'fc7','OutputAs','rows');
testfeatures=activations(net,testdata, ...
    'fc7','OutputAs','rows');
classifier=fitcknn(trainfeatures,trainIm.Labels)
predLabels=predict(classifier,testfeatures)
numcorrect=nnz(predLabels==testIm.Labels)
acc=numcorrect/numel(predLabels)
confusionchart(testIm.Labels,predLabels)
```

Inference:

- Models learn from labeled data to predict outcomes for new data
- Models analyze unlabeled data to identify patterns and relationships within the data itself
- Combining supervised and unsupervised learning can improve efficiency in tasks like automotive parts manufacturing.
- This combined approach can potentially reduce defects in manufacturing.

Exp. No. 5 **Date:**

Supervised and non-supervised AI learning for machining operation using Classification Learner App

Objective:

To develop an Artificial Intelligence (AI) system for machining operation in automotive parts using supervised and non-supervised AI learning using Classification Learner App

Software required:

Classification Learner APP in MATLAB software

Steps to be followed:

Data Collection
The dataset consists of over all 800 images in which 80% taken for training and 20% taken for testing

Training images:640
Testing images:160

Finished workpiece sample image

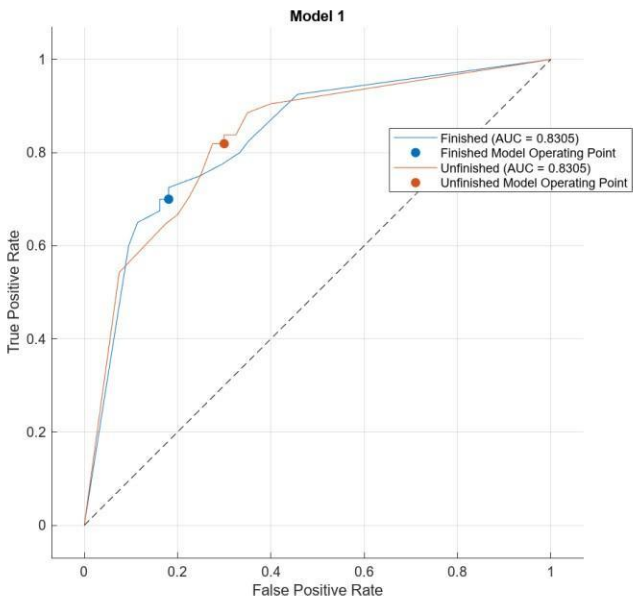
Unfinished workpiece sample image

Data pre-processing
Features are extracted from the source objects of finished and unfinished workpieces

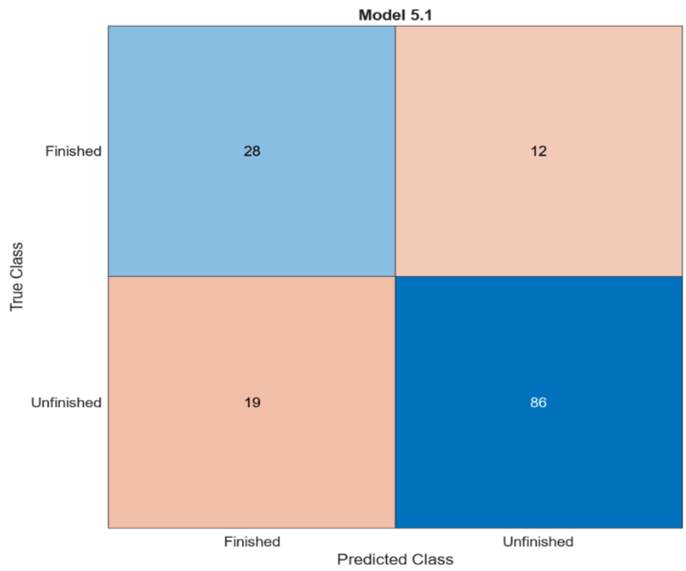
Data Augmentation
Nil
Training using Classification Learner App
Importing the features, mentioning the predictor and response variables, cross-validation, training algorithms, etc.
Testing and performance evaluation

- testing accuracy $\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN}$
- Testing Accuracy: 97.8%

Favorite	Model Number	Model Type	Status	Accuracy (Validation)	Total Cost (Validation)
	1	Tree	Trained	78.62 %	31
	2.1	Tree	Trained	78.62 %	31
	2.2	Tree	Trained	78.62 %	31
	2.3	Tree	Trained	85.52 %	21
	2.4	Discriminant	Failed	-	-
	2.5	Discriminant	Failed	-	-
	2.6	Binary GLM Logistic Regression	Trained	73.79 %	38
	2.7	Efficient Logistic Regression	Trained	74.48 %	37
	2.8	Efficient Linear SVM	Trained	73.79 %	38
	2.9	Naive Bayes	Failed	-	-
	2.10	Naive Bayes	Trained	51.72 %	70
	2.11	SVM	Trained	79.31 %	30
	2.12	SVM	Trained	75.17 %	36
	2.13	SVM	Trained	71.03 %	42
	2.14	SVM	Trained	79.31 %	30
	2.15	SVM	Trained	75.86 %	35
	2.16	SVM	Trained	72.41 %	40
	2.17	KNN	Trained	75.86 %	35
	2.18	KNN	Trained	73.79 %	38

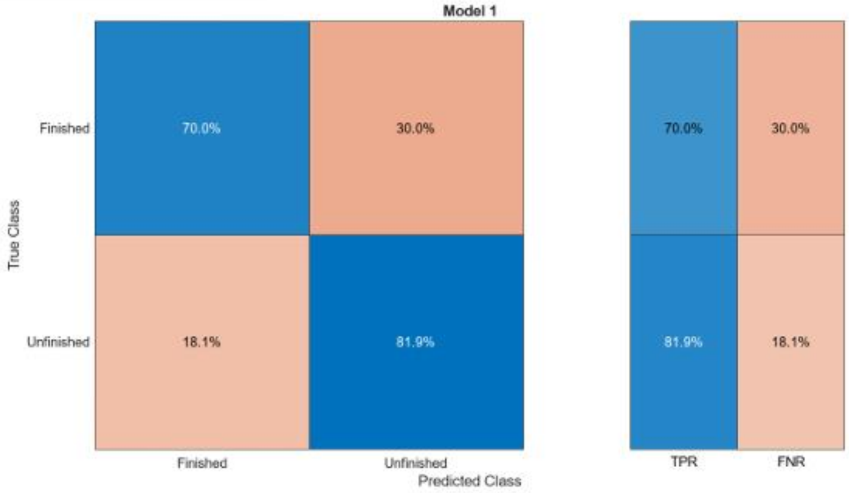


ROC Curve

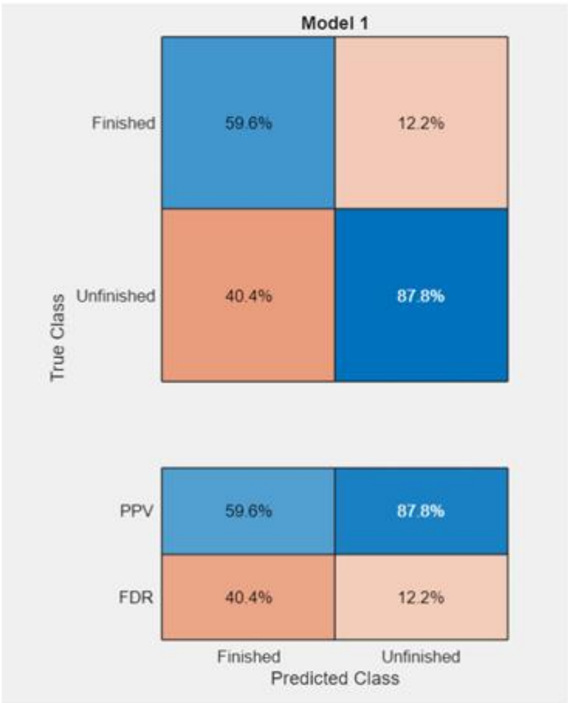


Validation Confusion Matrix

TPR and FNR



PPV and FDR



Inference:

- It can be used to construct an AI system for automotive part machining operations.
- This system can train models to classify various states, including "defective/non-defective" and "finished/unfinished" parts.
- By using this, we can develop intelligent systems for quality control and process optimization in manufacturing.

Exp. No. 6 **Date:**

Image augmentation for improving the performance of deep neural networks

Objective:

To increase the dataset size by applying various transformation using image augmentation techniques for improving the performance of deep neural networks.

Software required:

- MATLAB software

Steps to be followed:

- **Sample image before augmentation**

- **Image Augmentation**
  - Transformations used for image augmentation:
    - RandXReflection
    - RandXScale [1, 1.2]
    - RandRotation [0 45]

MATLAB Code:

```
clc
clear all
close all

%%Loading the image dataset
imds = imageDatastore('F:\BEST-SASTRA\AI\test',...
                                        'IncludeSubfolders',true)
```

```
imshow(read(imds))

%%Define the transformation for augmentation
aug = imageDataAugmenter('RandXReflection',...
                        true,'RandXScale',[1 1.2],'RandRotation',[0,360])

%%Perform image augmentation
augds = augmentedImageDatastore([227 227],...
                        imds,'DataAugmentation',aug)

%%Read the augmented image dataset
mb = read(augds);
imshow(mb.input{5})
```

RESULTS:

Sample image after augmentation

Inference:

- Augment images to help neural networks learn.
- Vary images to improve accuracy.
- Networks recognize objects better.

Exp. No. 7 **Date:**

Vehicle detection using YOLOv4 architecture

Objective:

To develop an Artificial Intelligence (AI) system for vehicle detection using an advanced deep Convolutional Neural Network architecture YOLOV4.

Software required:

- MATLAB software

Steps to be followed:

- **Loading the pre-trained YOLOv4 object detector**
- **Analyze the detector network**
- **Testing the YOLOv4 object detector performance**

MATLAB Code:

```
clc
clear all
close all

%%Loading the YOLOv4 object detector
name = "csp-darknet53-coco"
detector = yolov4ObjectDetector(name)
disp(detector)

%%Analysing the detector network
```

analyzeNetwork(detector.Network)

%%Testing the architecture

img = imread("F:\road.jpg")

imRes = imresize(img,[608 608])

[bboxes,scores,labels] = detect(detector,imRes)

detectedImg = insertObjectAnnotation(imRes,"Rectangle",bboxes,labels)

figure

imshow(detectedImg)

Results:

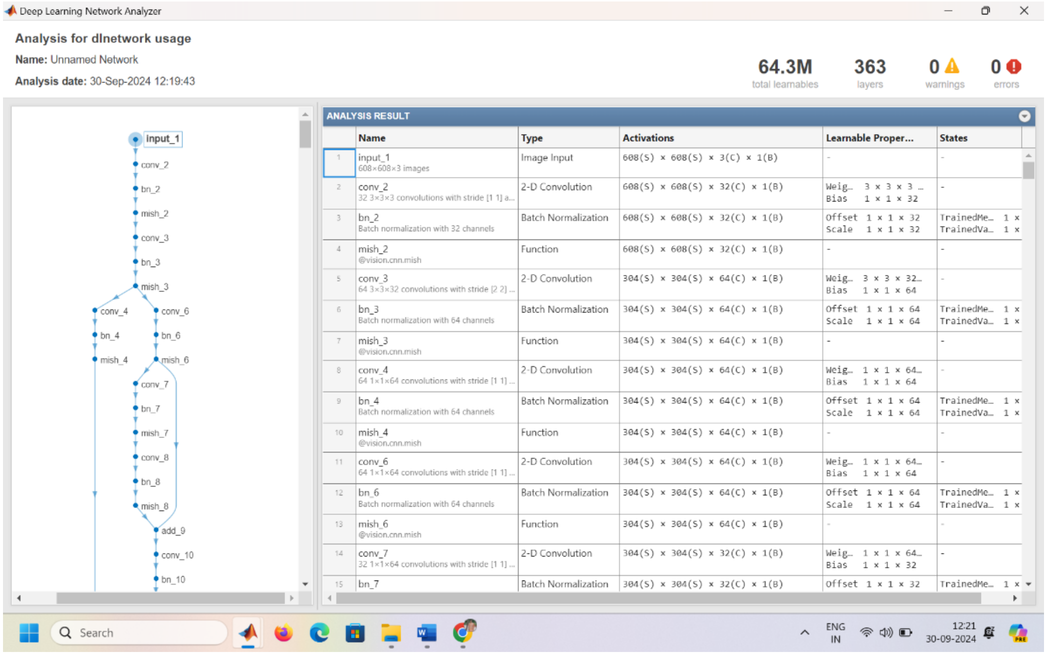
Network details:

```
name = "csp-darknet53-coco"
```

```
detector =
  yolov4ObjectDetector with properties:

             Network: [1×1 dlnetwork]
         AnchorBoxes: {3×1 cell}
          ClassNames: {80×1 cell}
           InputSize: [608 608 3]
    PredictedBoxType: 'axis-aligned'
           ModelName: 'csp-darknet53-coco'
  yolov4ObjectDetector with properties:

             Network: [1×1 dlnetwork]
         AnchorBoxes: {3×1 cell}
          ClassNames: {80×1 cell}
           InputSize: [608 608 3]
    PredictedBoxType: 'axis-aligned'
           ModelName: 'csp-darknet53-coco'
```



Scores for detecting car, truck, person for the given test image

0.78527963

0.98571438

0.95687526

0.56419271

0.86653084

0.94932556

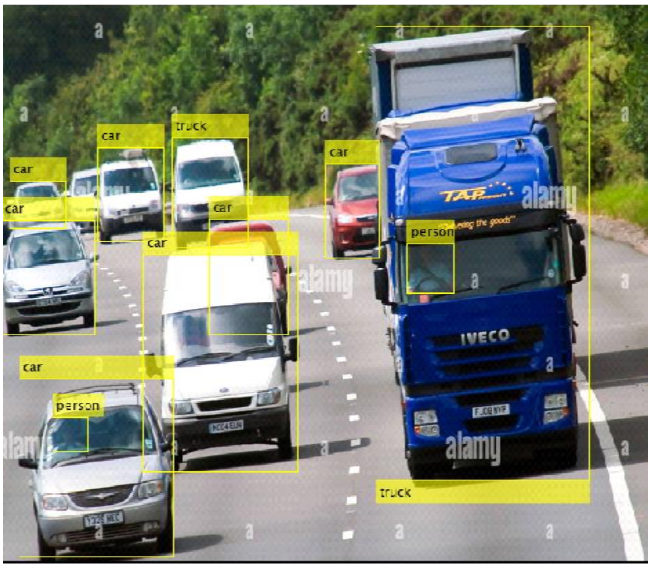
0.80800867

0.98217887

0.83797473

0.77259898

0.94492215



Inference:

- YOLOv4 detects vehicles in images.
- It uses deep learning to locate cars, trucks, etc.
- Accurate detection in real-time.
- Improves safety and traffic management.