Vision

Training EXAMPLE

```
folderName = "Original";
unzip("MerchData.zip",folderName);
```

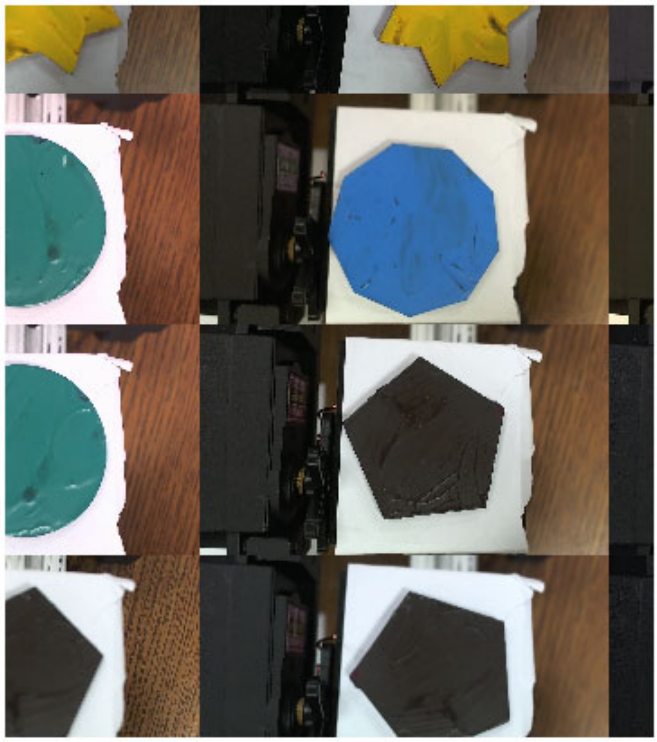
```
Error using matlab.io.internal.archive.checkFilename>throwFileNotFoundError (line 156)
Function UNZIP was unable to find file ''MerchData.zip''.

Error in matlab.io.internal.archive.checkFilename>validateLocalFilename (line 106)
        throwFileNotFoundError(filename, functionName);

Error in matlab.io.internal.archive.checkFilename (line 54)
    fullfilename = validateLocalFilename(filename, validExtensions, functionName);

Error in matlab.io.internal.archive.parseUnArchiveInputs (line 71)
[archiveFilename, url] = matlab.io.internal.archive.checkFilename(archiveFilename, validExtensior

Error in unzip (line 10)
[zipFilename, outputDir, url, urlFilename] = matlab.io.internal.archive.parseUnArchiveInputs( ..
```

```
imds = imageDatastore(folderName, ...
    IncludeSubfolders=true, ...
    LabelSource="foldernames");

numImages = numel(imds.Labels);
idx = randperm(numImages,16);
I = imtile(imds,Frames=idx);
figure
imshow(I)

classNames = categories(imds.Labels)

numClasses = numel(classNames)

[imdsTrain,imdsValidation,imdsTest] = splitEachLabel(imds,0.7,0.15,"randomized");

net = imagePretrainedNetwork("squeezenet",NumClasses=numClasses);

analyzeNetwork(net)

inputSize = networkInputSize(net)

[layerName,learnableNames] = networkHead(net)

net = freezeNetwork(net,LayerNamesToIgnore=layerName);

%% Prepare Images
pixelRange = [-30 30];

imageAugmenter = imageDataAugmenter( ...
    RandXReflection=true, ...
    RandXTranslation=pixelRange, ...
    RandYTranslation=pixelRange);

augimdsTrain = augmentedImageDatastore(inputSize(1:2),imdsTrain, ...
    DataAugmentation=imageAugmenter);

augimdsValidation = augmentedImageDatastore(inputSize(1:2),imdsValidation);
augimdsTest = augmentedImageDatastore(inputSize(1:2),imdsTest);

options = trainingOptions("adam"
```

```
options = trainingOptions( adam , ...
    ValidationData=augimdsValidation, ...
    ValidationFrequency=5, ...
    Plots="training-progress", ...
    Metrics="accuracy", ...
    Verbose=false);

net = trainnet(augimdsTrain,net,"crossentropy",options);

YTest = minibatchpredict(net,augimdsTest);
YTest = scores2label(YTest,classNames);

TTest = imdsTest.Labels;
figure
confusionchart(TTest,YTest)

acc = mean(TTest==YTest)

im = imread("MerchDataTest.jpg");
im = imresize(im,inputSize(1:2));
X = single(im);

if canUseGPU
    X = gpuArray(X);
end
scores = predict(net,X);
[label,score] = scores2label(scores,classNames);

figure
imshow(im)
title(string(label) + " (Score: " + gather(score) + ")")
```

Own Images

```
folderName = "Vision";
unzip("Shapes.zip",folderName);
imds = imageDatastore(folderName, ...
    IncludeSubfolders=true, ...
    LabelSource="foldernames");

numImages = numel(imds.Labels);
idx = randperm(numImages,16);
I = imtile(imds,Frames=idx);
figure
imshow(I)
```



```
classNames = categories(imds.Labels)
```

```
classNames = 4×1 cell
'Blue Nonagon'
'Brown Pentagon'
'Green Circle'
'Yellow Star'
```

```
numClasses = numel(classNames)
```

```
numClasses = 4
```

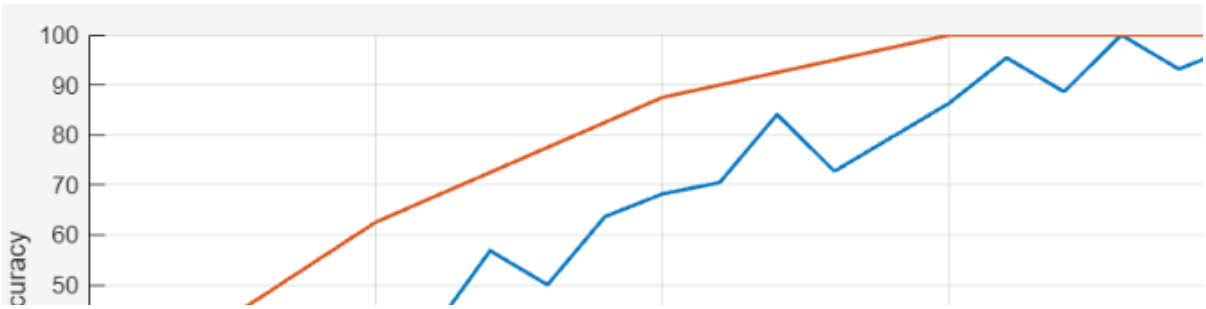
```
[imdsTrain,imdsValidation,imdsTest] = splitEachLabel(imds,0.7,0.15,"randomized");

net = imagePretrainedNetwork("squeezenet",NumClasses=numClasses);

analyzeNetwork(net)

inputSize = networkInputSize(net)
```

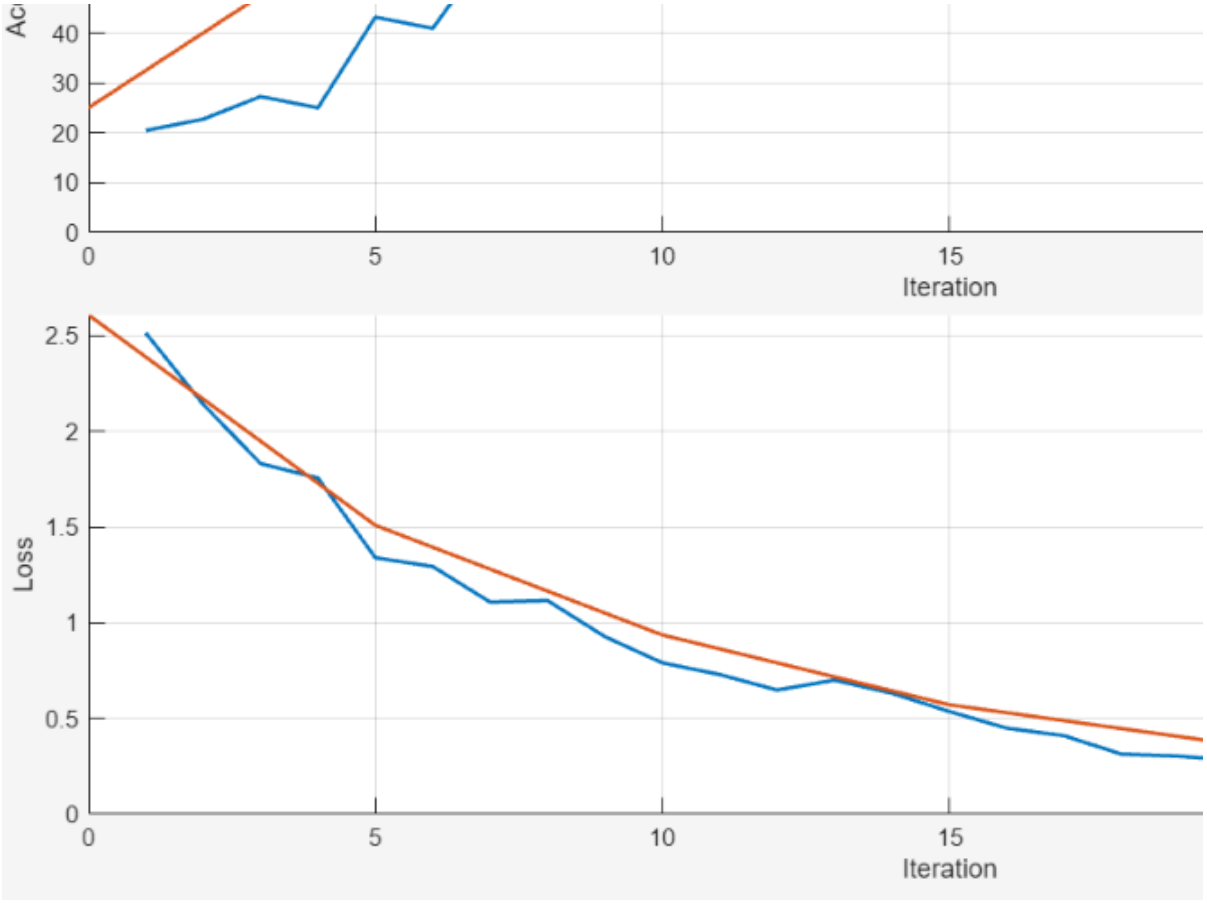
```
inputSize = 1×3
   227   227     3
```

```
[layerName,learnableNames] = networkHead(net)
```

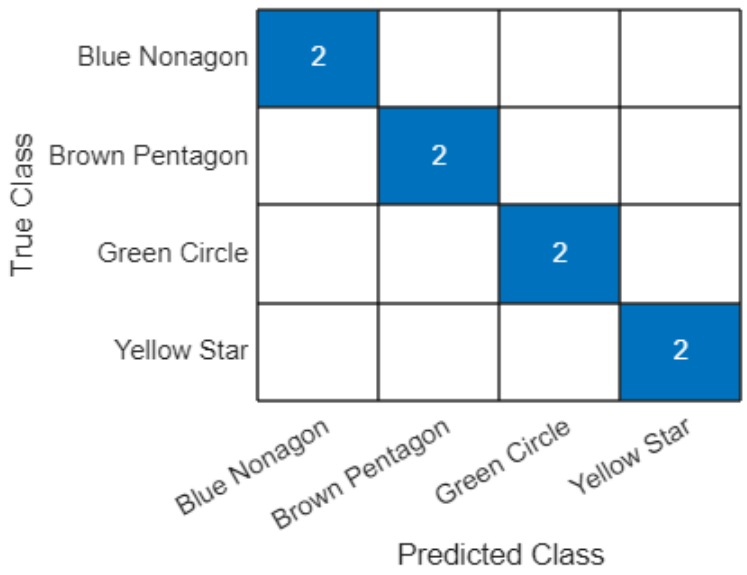
```
layerName = "conv10"
learnableNames = 2×1 string
"conv10/Wei…
"conv10/Bias"
```

```
net = freezeNetwork(net,LayerNamesToIgnore=layerName);

%% Prepare Images
pixelRange = [-30 30];

imageAugmenter = imageDataAugmenter( ...
    RandXReflection=true, ...
    RandXTranslation=pixelRange, ...
    RandYTranslation=pixelRange);

augimdsTrain = augmentedImageDatastore(inputSize(1:2),imdsTrain, ...
    DataAugmentation=imageAugmenter);

augimdsValidation = augmentedImageDatastore(inputSize(1:2),imdsValidation);
augimdsTest = augmentedImageDatastore(inputSize(1:2),imdsTest);

options = trainingOptions("adam", ...
    ValidationData=augimdsValidation, ...
    ValidationFrequency=5, ...
    Plots="training-progress", ...
    Metrics="accuracy", ...
    Verbose=false);

net = trainnet(augimdsTrain,net,"crossentropy",options);
```






```
YTest = minibatchpredict(net,augimdsTest);
YTest = scores2label(YTest,classNames);

TTest = imdsTest.Labels;
figure
confusionchart(TTest,YTest)
```




```
acc = mean(TTest==YTest)
```

```
acc = 1
```

```
% List available webcams
camList = webcamlist;

% Display the list
disp('Available webcams:');
```

```
Available webcams:
```

```
disp(camList);
```

```
    {'USB2.0 HD UVC WebCam'}
```

```
camera = webcam('Logitech HD Pro Webcam C910');
```

```
The device name specified is invalid. The list of valid devices are {USB2.0 HD UVC WebCam}.
```

```
im = camera.snapshot;
im = imresize(im,inputSize(1:2));
X = single(im);

if canUseGPU
    X = gpuArray(X);
end
scores = predict(net,X);
[label,score] = scores2label(scores,classNames);

figure
imshow(im)
title(string(label) + " (Score: " + gather(score) + ")")
```

Taking Photos

```
%%camera = webcam;
camera = webcam('Logitech HD Pro Webcam C910');
nnet = alexnet;
picture = camera.snapshot;
figure(1);
image(picture);
picture = imresize(picture,[227,227]);
label = classify(nnet, picture);
title(upper(char(label)));
```

Camera to Gcode

```
camera = webcam('Logitech HD Pro Webcam C910');
im = camera.snapshot;
im = imresize(im,inputSize(1:2));
X = single(im);

if canUseGPU
    X = gpuArray(X);
end
scores = predict(net,X);
[label,score] = scores2label(scores,classNames);

figure
imshow(im)
title(string(label) + " (Score: " + gather(score) + ")")

%% Gcode
%% Initiate serial port connection
arduino=serialport("COM7",250000,"Timeout",120); % Replace COM7 with the port for your board
in1=readline(arduino);    % String sent from MKS board at startup
in2=readline(arduino);    % String sent from MKS board at startup

% There will be about a 2 second delay before the G code is executed


%% G code goes here
object = label;

if label == classNames(1) %% Blue Nonagon
    writeline(arduino,"G1 X40");  % Write your G code here. One line per command
    writeline(arduino,"G1 X0");  % Write your G code here. One line per command
end

if label == classNames(2) %% Brown Pentagon
    writeline(arduino,"G1 X40");  % Write your G code here. One line per command
    writeline(arduino,"G1 X0");  % Write your G code here. One line per command
end

if label == classNames(3) %% Green Circle
    writeline(arduino,"G1 X40");  % Write your G code here. One line per command
    writeline(arduino,"G1 X0");  % Write your G code here. One line per command
end

if label == classNames(4) %% Orange Triangle
    writeline(arduino,"G1 X40");  % Write your G code here. One line per command
    writeline(arduino,"G1 X0");  % Write your G code here. One line per command
end

if label == classNames(5) %% Purple Star
    writeline(arduino,"G1 X40");  % Write your G code here. One line per command
    writeline(arduino,"G1 X0");  % Write your G code here. One line per command
end

if label == classNames(6) %% Red Hexagon
    writeline(arduino,"G1 X40");  % Write your G code here. One line per command
    writeline(arduino,"G1 X0");  % Write your G code here. One line per command
end
if label == classNames(7) %% Red Square
    writeline(arduino,"G1 X40");  % Write your G code here. One line per command
    writeline(arduino,"G1 X0");  % Write your G code here. One line per command
end
```

```
end

if label == classNames(8) %% Star Red
    writeline(arduino,"G1 X40");  % Write your G code here. One line per command
    writeline(arduino,"G1 X0");  % Write your G code here. One line per command
end

if label == classNames(9) %% Yellow Star
    writeline(arduino,"G1 X40");  % Write your G code here. One line per command
    writeline(arduino,"G1 X0");  % Write your G code here. One line per command
end


%% Gcode to come back to orgin
writeline(arduino,"G1 X0 Y0 Z0");  % Write your G code here. One line per command

%% Finish up
clear a;    % Disconnect and clear the serial port
```

```
camera = webcam('Logitech HD Pro Webcam C910');
im = camera.snapshot;
im = imresize(im,inputSize(1:2));
X = single(im);

if canUseGPU
    X = gpuArray(X);
end
scores = predict(net,X);
[label,score] = scores2label(scores,classNames);

figure
imshow(im)
title(string(label) + " (Score: " + gather(score) + ")")


% There will be about a 2 second delay before the G code is executed

%% G code goes here
object = label;

if object == classNames(1) %% Blue Nonagon
    writeline(arduino,"G1 X160 Y160 Z190");
    writeline(arduino,"G1 X160 Y160 Z190"); %% blue nonagon location
    writeline(arduino,"G4 P1000"); % Wait 1 second
    %% writeline(arduino, ) %% encoder tilt
    writeline(arduino,"G4 P1000"); % Wait 1 second
    writeline(arduino,"G1 X160 Y160 Z190");
    writeline(arduino,"G1 X230 Y230 Z190");
end

if object == classNames(2) %% Brown Pentagon
    writeline(arduino,"G1 X160 Y160 Z190");
    writeline(arduino,"G1 X160 Y160 Z95"); %% brown pentagon location
    writeline(arduino,"G4 P1000"); % Wait 1 second
    %% writeline(arduino, ) %% encoder tilt
    writeline(arduino,"G4 P1000"); % Wait 1 second
    writeline(arduino,"G1 X160 Y160 Z190");
    writeline(arduino,"G1 X230 Y230 Z190");
end

if object == classNames(3) %% Green Circle
    writeline(arduino,"G1 X160 Y160 Z190");
    writeline(arduino,"G1 X160 Y160 Z0"); %% green circle location
    writeline(arduino,"G4 P1000"); % Wait 1 second
    %% writeline(arduino, ) %% encoder tilt
    writeline(arduino,"G4 P1000"); % Wait 1 second
    writeline(arduino,"G1 X160 Y160 Z190");
    writeline(arduino,"G1 X230 Y230 Z190");
end

if object == classNames(4) %% Orange Triangle
    writeline(arduino,"G1 X160 Y160 Z190");
    writeline(arduino,"G1 X80 Y80 Z190"); %% orange triangle location
    writeline(arduino,"G4 P1000"); % Wait 1 second
    %% writeline(arduino, ) %% encoder tilt
    writeline(arduino,"G4 P1000"); % Wait 1 second
    writeline(arduino,"G1 X160 Y160 Z190");
    writeline(arduino,"G1 X230 Y230 Z190");
end
```

```
end

if object == classNames(5) %% Purple Star
    writeline(arduino,"G1 X160 Y160 Z190");
    writeline(arduino,"G1 X80 Y80 Z95"); %% purple star location
    writeline(arduino,"G4 P1000"); % Wait 1 second
    %% writeline(arduino, ) %% encoder tilt
    writeline(arduino,"G4 P1000"); % Wait 1 second
    writeline(arduino,"G1 X160 Y160 Z190");
    writeline(arduino,"G1 X230 Y230 Z190");
end

if object == classNames(6) %% Red Hexagon
    writeline(arduino,"G1 X160 Y160 Z190");
    writeline(arduino,"G1 X80 Y80 Z0"); %% red hexagon location
    writeline(arduino,"G4 P1000"); % Wait 1 second
    %% writeline(arduino, ) %% encoder tilt
    writeline(arduino,"G4 P1000"); % Wait 1 second
    writeline(arduino,"G1 X160 Y160 Z190");
    writeline(arduino,"G1 X230 Y230 Z190");
end
if object == classNames(7) %% Red Square
    writeline(arduino,"G1 X160 Y160 Z190");
    writeline(arduino,"G1 X0 Y0 Z190"); %% red square location
    writeline(arduino,"G4 P1000"); % Wait 1 second
    %% writeline(arduino, ) %% encoder tilt
    writeline(arduino,"G4 P1000"); % Wait 1 second
    writeline(arduino,"G1 X160 Y160 Z190");
    writeline(arduino,"G1 X230 Y230 Z190");
end

if object == classNames(8) %% Star Red
    writeline(arduino,"G1 X160 Y160 Z190");
    writeline(arduino,"G1 X0 Y0 Z95"); %% star red location
    writeline(arduino,"G4 P1000"); % Wait 1 second
    %% writeline(arduino, ) %% encoder tilt
    writeline(arduino,"G4 P1000"); % Wait 1 second
    writeline(arduino,"G1 X160 Y160 Z190");
    writeline(arduino,"G1 X230 Y230 Z190");
end

if object == classNames(9) %% Yellow Star
    writeline(arduino,"G1 X160 Y160 Z190");
    writeline(arduino,"G1 X0 Y0 Z0"); %% yellow satr location
    writeline(arduino,"G4 P1000"); % Wait 1 second
    %% writeline(arduino, ) %% encoder tilt
    writeline(arduino,"G4 P1000"); % Wait 1 second
    writeline(arduino,"G1 X160 Y160 Z190");
    writeline(arduino,"G1 X230 Y230 Z190");
end
```