**FILENAME: TRAIN\_ALEXNET**

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%% %%%%%

%%%%% MADE BY; %%%%%

%%%%% DIPESH CHATROLA-16 %%%%%

%%%%% %%%%%

%%%%% SUBJECT - SIGNAL & SYSTEM %%%%%

%%%%% %%%%%

%%%%% MARWADI UNIVERSITY %%%%%

%%%%% %%%%%

%%%%% %%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

close all;

clear all;

clc;

tic;

fprintf('\n==============================================> Simulation Started <==============================================\n');

fprintf('Start Time: %s\n', datestr(now,'HH:MM:SS.FFF\n\n'))

% Load a pre-trained, deep, convolutional network

alex = alexnet;

layers = alex.Layers;

% Loading training images

num\_classes = 2;

maskImages = 'mask';

maskData = dir(maskImages);

for i1 = 3:length(maskData)

filename = [maskImages '\' maskData(i1).name];

fprintf('Processing %s: %d/%d\n',filename,i1-2,length(maskData)-2);

temp\_image = im2double(imread(filename));

Images(:,:,:,i1-2,1) = imresize(temp\_image, [layers(1).InputSize(1:2)]);

Labels(i1-2,1) = categorical({'With Mask'});

end

nomaskImages = 'no\_mask';

nomaskData = dir(nomaskImages);

for i2 = 3:length(nomaskData)

filename = [nomaskImages '\' nomaskData(i2).name];

fprintf('Processing %s: %d/%d\n',filename,i2-2,length(nomaskData)-2);

temp\_image = im2double(imread(filename));

Images(:,:,:,i1-2+i2-2,1) = imresize(temp\_image, [layers(1).InputSize(1:2)]);

Labels(i1-2+i2-2,1) = categorical({'Without Mask'});

end

% Modify the network to use two categories

layers(23) = fullyConnectedLayer(num\_classes);

layers(25) = classificationLayer;

% Set up and split the training data

split\_size = 0.80 ;

total\_samples = numel(Labels);

idx = randperm(total\_samples) ;

XTrain = Images(:,:,:,idx(1:round(split\_size\*total\_samples)));

YTrain = Labels(idx(1:round(split\_size\*total\_samples)));

XTest = Images(:,:,:,idx(round(split\_size\*total\_samples)+1:end));

YTest = Labels(idx(round(split\_size\*total\_samples)+1:end));

% Re-train the Network

opts = trainingOptions('sgdm', 'InitialLearnRate', 0.001, 'MaxEpochs', 10, 'MiniBatchSize', 64);

myNet = trainNetwork(XTrain, YTrain, layers, opts);

% Measure network accuracy

predictedLabels = classify(myNet, XTest);

accuracy = mean(predictedLabels == YTest)

save('MaskDetect','myNet');

toc;

fprintf('End Time: %s\n', datestr(now,'HH:MM:SS.FFF'))

fprintf('\n==============================================> Simulation Ended <==============================================\n');

**FILENAME: LIVEMASK**

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%% %%%%%

%%%%% MADE BY; %%%%%

%%%%% DIPESH CHATROLA-16 %%%%%

%%%%% NIMISH GONDALIYA-24 %%%%%

%%%%% JENISH RAIYANI-42 %%%%%

%%%%% %%%%%

%%%%% SUBJECT - SIGNAL & SYSTEM %%%%%

%%%%% %%%%%

%%%%% MARWADI UNIVERSITY %%%%%

%%%%% %%%%%

%%%%% %%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

close all;

clear all;

clc;

tic;

fprintf('\n==============================================> Simulation Started <==============================================\n');

fprintf('Start Time: %s\n', datestr(now,'HH:MM:SS.FFF\n\n'))

% Loading the trained network

load('MaskDetect.mat');

net = myNet; clear myNet

% Some preliminaries

if ~exist('camera')

camera = webcam; % Connect to the camera

end

cr\_label = '| Real-time Face Mask Detection by DIPESH, NIMISH & JENISH | ';

img\_size = [227,227];

% Loading the face detector

FDetect = vision.CascadeObjectDetector;

while true

picture = camera.snapshot;

orig\_picture = picture;

BB = step(FDetect, picture);

if size(BB,1) >= 1 % if a face is found

for faces\_iter = 1:size(BB,1) % for total number of faces found in the camera

picture\_cropped = imcrop(orig\_picture,BB(faces\_iter,:));

picture\_resized = imresize(picture\_cropped,img\_size);

label = classify(net, picture\_resized); % find out if it has mask or not

label\_text = char(label);

text\_color = 'green';

line\_color = 'g';

if strcmp(label\_text, 'Without Mask')

text\_color = 'red';

line\_color = 'r';

end

picture = insertText(picture,BB(faces\_iter,1:2),label\_text,'FontSize',16,'BoxColor',...

'white','BoxOpacity',0,'TextColor',text\_color); % text label around each face

image(picture); % show the picture

axis off;

end

for faces\_iter = 1:size(BB,1) % bounding boxes around each face

rectangle('Position', BB(faces\_iter,:), 'Linewidth',2,'LineStyle','-','EdgeColor',line\_color);

end

label = strcat(cr\_label,' Detected face(s) = ',num2str(size(BB,1)));

else % if no face found

image(picture); % show the picture

picture = imresize(picture,img\_size);

axis off

label = strcat(cr\_label,' Detected face(s) = 0');

end

label\_cell{1} = label;

label\_cell{2} = strcat('Date: ', date, ', Time: ', datestr(now,'HH:MM:SS.FFF')); % current date and time

title(label\_cell, 'FontSize', 24); % show the label

drawnow;

fprintf('Detected face(s) = %d. Press ''Ctrl+c'' to end...\n', size(BB,1))

end

toc;

fprintf('End Time: %s\n', datestr(now,'HH:MM:SS.FFF'))

fprintf('\n==============================================> Simulation Ended <==============================================\n');