% Clear workspace

clear; close all; clc;

% Images Datapath – Please modify your path accordingly

datapath='dataset';

% Image Datastore

imds=imageDatastore(datapath, ...

'IncludeSubfolders',true, ...

'LabelSource','foldernames');

% Determine the split up

total\_split=countEachLabel(imds)

% Number of Images

num\_images=length(imds.Labels);

% Visualize random images

perm=randperm(num\_images,6);

figure;

for idx=1:length(perm)

subplot(2,3,idx);

imshow(imread(imds.Files{perm(idx)}));

title(sprintf('%s',imds.Labels(perm(idx))))

% Number of folds

num\_folds=10;

% Loop for each fold

for fold\_idx=1:num\_folds

fprintf('Processing %d among %d folds \n',fold\_idx,num\_folds);

% Test Indices for current fold

test\_idx=fold\_idx:num\_folds:num\_images;

% Test cases for current fold

imdsTest = subset(imds,test\_idx);

% Train indices for current fold

train\_idx=setdiff(1:length(imds.Files),test\_idx);

% Train cases for current fold

imdsTrain = subset(imds,train\_idx);

% ResNet Architecture

net=resnet50;

lgraph = layerGraph(net);

clear net;

% Number of categories

numClasses = numel(categories(imdsTrain.Labels));

% New Learnable Layer

newLearnableLayer = fullyConnectedLayer(numClasses, ...

'Name','new\_fc', ...

'WeightLearnRateFactor',10, ...

'BiasLearnRateFactor',10);

% Replacing the last layers with new layers

lgraph = replaceLayer(lgraph,'fc1000',newLearnableLayer);

newsoftmaxLayer = softmaxLayer('Name','new\_softmax');

lgraph = replaceLayer(lgraph,'fc1000\_softmax',newsoftmaxLayer);

newClassLayer = classificationLayer('Name','new\_classoutput');

lgraph = replaceLayer(lgraph,'ClassificationLayer\_fc1000',newClassLayer);

% Preprocessing Technique

imdsTrain.ReadFcn = @(filename)preprocess\_Xray(filename);

imdsTest.ReadFcn = @(filename)preprocess\_Xray(filename);

% Training Options, we choose a small mini-batch size due to limited images

options = trainingOptions('adam',...

'MaxEpochs',30,'MiniBatchSize',8,...

'Shuffle','every-epoch', ...

'InitialLearnRate',1e-4, ...

'Verbose',false, ...

'Plots','training-progress');

% Data Augumentation

augmenter = imageDataAugmenter( ...

'RandRotation',[-5 5],'RandXReflection',1,...

'RandYReflection',1,'RandXShear',[-0.05 0.05],'RandYShear',[-0.05 0.05]);

% Resizing all training images to [224 224] for ResNet architecture

auimds = augmentedImageDatastore([224 224],imdsTrain,'DataAugmentation',augmenter);

% Training

netTransfer = trainNetwork(auimds,lgraph,options);

% Resizing all testing images to [224 224] for ResNet architecture

augtestimds = augmentedImageDatastore([224 224],imdsTest);

% Testing and their corresponding Labels and Posterior for each Case

[predicted\_labels(test\_idx),posterior(test\_idx,:)] = classify(netTransfer,augtestimds);

% Save the Independent ResNet Architectures obtained for each Fold

save(sprintf('ResNet50\_%d\_among\_%d\_folds',fold\_idx,num\_folds),'netTransfer','test\_idx','train\_idx');

% Clearing unnecessary variables

clearvars -except fold\_idx num\_folds num\_images predicted\_labels posterior imds netTransfer;

end