clc

clear all

close all

fs = 44.1e3;

duration = 0.5;

N = duration\*fs;

M=1000;

wFake = sin(2\*rand([N,M]) - 1)+...

cos(2\*rand([N,M]) - 1);

wLabels = repelem(categorical("fake"),1000,1);

bReal = filter(1,[1,-0.999],wFake);

bReal = bReal./max(abs(bReal),[],'all');

bReal = sin(bReal)+cos(bReal);

bLabels = repelem(categorical("real"),1000,1);

classNames = ["fake", "real"];

figure(1)

sound(wFake(:,1),fs)

melSpectrogram(wFake(:,1),fs)

figure(2)

sound(bReal(:,1),fs)

melSpectrogram(bReal(:,1),fs)

%% Devide data into Training and Validation Sets

audioTrain = [wFake(:,1:700),bReal(:,1:700)];

labelsTrain = [wLabels(1:700);bLabels(1:700)];

audioValidation = [wFake(:,701:end),bReal(:,701:end)];

labelsValidation = [wLabels(701:end);bLabels(701:end)];

%% Extract features using Feature Extractor

aFE = audioFeatureExtractor(SampleRate=fs, ...

SpectralDescriptorInput="melSpectrum", ...

spectralCentroid=true, ...

spectralSlope=true);

% Training features

featuresTrain = extract(aFE,audioTrain);

[numHopsPerSequence,numFeatures,numSignals] = size(featuresTrain);

% Validation features

featuresValidation = extract(aFE,audioValidation);

featuresValidation = squeeze(num2cell(featuresValidation,[1,2]));

%% DEFINE AND TRAIN NETWORK

% layers

layers = [ ...

sequenceInputLayer(numFeatures)

lstmLayer(50,OutputMode="last")

fullyConnectedLayer(numel(unique(labelsTrain)))

softmaxLayer];

% options

options = trainingOptions("adam", ...

Shuffle="every-epoch", ...

ValidationData={featuresValidation,labelsValidation}, ...

Plots="training-progress", ...

Metrics={"accuracy","loss","rmse"}, ...

Verbose=false);

net = trainnet(featuresTrain,labelsTrain,layers,"crossentropy",options);

% TEST Network

NN = 100;

wFakeTest = sin(2\*rand([N,1]) - 1)+cos(2\*rand([N,1]) - 1);

scores = predict(net,extract(aFE,wFakeTest));

scores2label(scores,classNames)

% We now test the LSTM

%% TEST OVER MIXED Audio Data

NN = 100;

% generate data of white noise

wFakeTest = sin(2\*rand([N,NN]) - 1)+cos(2\*rand([N,1]) - 1);

%generate data of less white noise

bRealTest = filter(1,[1,-0.999],wFakeTest);

bRealTest = bRealTest./max(abs(bRealTest),[],'all');

bRealTest = sin(bRealTest)+cos(bRealTest);

% combined data or fake and real audio

TESTDATA = [wFakeTest,bRealTest];

TESTDATA(end+1,1:NN) = 1;

TESTDATA(end+1,NN+1:2\*NN) = 2;

Str = rand(1,2\*NN);

TESTDATA = [Str; TESTDATA];

TESTDATA = TESTDATA';

% mixed data of real and fake

TESTDATA = sortrows(TESTDATA,1);

TESTDATA = TESTDATA';

% THE MANUAL SCORES

TESTDATA\_M = TESTDATA(end,:);

% Exclude the last row

TESTDATA = TESTDATA(1:end-1,:);

% USE THE TRAINED ALGORITHM TO PREDICT

for i = 1: 2\*NN

scores(i,1) = predict(net,extract(aFE,TESTDATA(:,i)));

T1{i,1} = scores2label(scores(i,1),classNames);

end

for i = 1: 2\*NN

if scores(i,1)==TESTDATA\_M(i)

FinalS(i,1) = 1;

elseif scores(i,1) ~=TESTDATA\_M(i)

FinalS(i,1) = 0;

end

end

%% The algorithm score

SCORE1 = sum(FinalS(:,1))/length(FinalS);

%% We now train MLPs and use them, and compared

layers = [ ...

sequenceInputLayer(numFeatures)

fullyConnectedLayer(numel(unique(labelsTrain)))

reluLayer

softmaxLayer];

net = trainnet(featuresTrain,labelsTrain,layers,"crossentropy",options);

% USE THE TRAINED ALGORITHM TO PREDICT

for i = 1: 2\*NN

scores(i,1) = predict(net,extract(aFE,TESTDATA(:,i)));

T2{i,1} = scores2label(scores(i,1),classNames);

end

for i = 1: 2\*NN

if scores(i,1)==TESTDATA\_M(i)

FinalS(i,1) = 1;

elseif scores(i,1) ~=TESTDATA\_M(i)

FinalS(i,1) = 0;

end

end

%% The algorithm score

SCORE2 = sum(FinalS(:,1))/length(FinalS);

% The results are

%TESTDATA\_M which yields the manually classified audios

% T1 which yields LSTM classified audio

% T2 which yield MLPs classified audio

% SCORE1 which gives the performance of LSTM

% SCORE2 which gives the performance of MLPs