**ME 466 Introduction to AI**

**Fall 2021**

**Take-home Final Exam Submitted on: 19 January 2022**

**Name: Korkut Emre ARSLANTÜRK**

**Student ID:250206039**

**Grade:**

|  |  |
| --- | --- |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
| 7 |  |
| Σ |  |

I hereby declare that the paper I am submitting under this cover is product of my own efforts only. Even if I worked on some of the problems together with my classmates, I prepared this paper on my own, without looking at any other classmate’s paper. I am knowledgeable about everything that is written under this cover, and I am prepared to explain any scientific/technical content written here if a short oral examination about this paper is conducted by the instructor. I am aware of the serious consequences of cheating.

**Signature:** metin, tablo içeren bir resim

Açıklama otomatik olarak oluşturuldu

**TABLE OF CONTENTS**

[1. Introduction 3](#_Toc19349)

[2. Body of The Report 4](#_Toc19350)

[3. Conclusion 14](#_Toc19351)

# INTRODUCTION

We have 3 different problems in this assignment.

# BODY OF THE REPORT

## Question 1

## Question 2

## Question 3

## Question 4

## Question 5

## Question 6

## Question 7

The dataset contains walking data of 7 different people of different genders. Matrix data’s each row matches a 5-second recording of the acceleration data, which is sampled at 25 Hertz, as the participant is walking. The participant number is kept in the variable participants for each row. Also, the gender of the participant is kept in the variable gender in the corresponding row. The aim of this study is to design a neural network algorithm to find the user and gender of the participants from the frequency information in the walking data.

It was asked to be created as a vector whose labels have 7 elements and all the elements are -1, which person belongs to that element is 1. For this purpose, the data was requested to be obtained by DFT.

First, the data set was loaded, and the labels were converted to the desired format.

load finalq7.mat

Targets=-1\*ones(2,420);

for k=1:420

Targets(gender(k),k)=1;

end

1. Then, for gravitational acceleration, the mean was asked to be removed. The mean of each row is subtracted.

for j = 1:420

data(j,:) = data(j,:) - mean(data(j,:));

end

1. Then, the calculation of the DFT of each row was performed. Each row consists of 125 features. The FFT function was used to perform the Fourier transform. Since the amplitude information will be needed, the abs function was used and the absolute value was taken. Then, since the negative and positive components on the frequency domain are symmetrical (have the same value), only their frequencies on one side were taken. That is, the first 63 elements of the 125-element vector were taken and assigned to the row of the other matrix. Thus, each row had its own Fourier transform assigned to the row of another matrix.

DFT\_mat = zeros(420,63);

for j = 1:420

DFT = abs(fft(data(j,:),125));

DFT\_mat(j,:) = DFT(1:63);

end

1. After that, a standard normalization process was applied. All the matrix features were decompressed between 0 and 1. The controlled value was subtracted from the minimum value of the row under consideration. The result was divided by the difference between max and min. Thus, all values are normalized between 0 and 1.

for i=1:63

DFT\_mat(i,:) = (DFT\_mat(i,:) - min(DFT\_mat(i,:))) / (max(DFT\_mat(i,:)) - min(DFT\_mat(i,:)));

end

1. To be able to select random, the random permutation of the instances between 1 and 420 were taken. Then, 250 instances were defined for training, 50 instances for validation, and the remaining 120 instances for testing.

idx = randperm(420);

training\_idx = idx(1:250);

val\_idx = idx(251:300);

test\_idx = idx(301:end);

Then it is asked to develop a neural network algorithm. Since the feature number is 63, a layer consisting of 63 neurons was defined for the Input layer. Also, because of there are 7 participants, the output layer consists of 7 neurons. For the model to converge faster, the hidden layer was set to a high (150). Also, the learning rate was specified. Then, the Wih and Woh were determined randomly. In there, multiplied by 0.02 when subtracting 0.5 decimals to get a value between -0.5 and 0.5. Subsequently, Training, TrainingTargets,Validation, ValidationTargets, Test, TestTargets defined. Also, number of training, validation, test is determined.

num\_input = 63;

num\_hidden = 150;

num\_output = 2;

eta=0.0001;

Wih=0.02\*(rand(num\_input+1,num\_hidden)-0.5);

Who=0.02\*(rand(num\_hidden+1,num\_output)-0.5);

f=DFT\_mat;

Training= f(training\_idx,:)';

TrainingTargets = Targets(:,training\_idx);

Validation = f(val\_idx,:)';

ValidationTargets = Targets(:,val\_idx);

Test = f(test\_idx,:)';

TestTargets = Targets(:,test\_idx);

num\_training=250;

num\_validation=50;

num\_test=120;

relE=inf;

prevE=inf;

trErrorhistory=[];

valErrorhistory=[];

epoch=0;

The validation loss is getting smaller and smaller. Training will stop when the loss starts to increase. If the previous error is less than the current error, it means that the error is increased. Therefore, the condition for continuing while was determined by subtracting the current error from the previous error. For the process does not to take too long time to converge, the equation does was not written with ‘0’ because after values such as 10-8, it starts not to converge. In while loop, allocated data for the train process is specified by their weight and their output. Values of v and o are estimated for the hidden layer. Subsequently, it goes to the output layer from tanh. Also, the error is calculated. The weights are updated with the weight update equations because the desired targets are not output. The derivative of the neuron function is calculated using the tangent hyperbola. Finally, the test is carried out in a single line for validation.

while (relE>0.01)

trE=0;

for i=1:num\_training

v=Wih'\*[Training(:,i);-1];

o=tanh(v);

vv=Who'\*[o;-1];

oo=tanh(vv);

trE=trE+sum((oo-TrainingTargets(:,i)).^2);

deltao=(oo-TrainingTargets(:,i)).\*(1-oo.^2);

Who=Who+(-eta\*[o;-1]\*deltao');

deltah=(Who\*deltao).\*(1-[o;-1].^2);

Wih=Wih+(-eta\*[Training(:,i);-1]\*deltah(1:end-1)');

end

epoch=epoch+1;

trE=trE/num\_training;

val=tanh(Who'\*[tanh(Wih'\*[Validation;-1\*ones(1,num\_validation)]);-1\*ones(1,num\_validation)]);

E=sum(sum((val-ValidationTargets).^2))/num\_validation;

relE=(prevE-E)/E;

prevE=E;

trErrorhistory=[trErrorhistory,trE];

valErrorhistory=[valErrorhistory,E];

plot(trErrorhistory)

hold on;

plot(valErrorhistory)

hold off;

pause(0.1)

end

est=tanh(Who'\*[tanh(Wih'\*[Test;-1\*ones(1,num\_test)]);-1\*ones(1,num\_test)]);

Finding the maximum value of each column in the generated data determines the activity.

A confusion matrix has been created. Classification accuracy was found by considering the elements in the main diagonal of the confusion matrix.

conf=zeros(2);

for k=1:num\_test

[~,I]=max(est(:,k));

[~,J]=max(TestTargets(:,k));

conf(J,I)=conf(J,I)+1;

end

conf

temp = 0;

for i = 1:2

temp = temp + conf(i,i);

end

100\*temp/120

SONUÇLAR VE YORUMLARI **PCA**

İİ) In this part, neural network design is desired to estimate the gender of the participants by using 63 features as input. Since many operations are the same as the previous part, not all decode, but the differences between them will be explained. The labels of the previous part were changed and a 2-element label was determined because there are 2 genders. For the male it is -1, for the female, it is 1.

Targets=-1\*ones(2,420);

When the input neuron does not change, the output neuron becomes 2 because of the 2 genders. Accordingly, the model was trained.

num\_input = 63;

num\_hidden = 150;

num\_output = 2;

eta=0.0001;

Also, the confusion matrix is changed. The confusion matrix is being made 2x2.

conf=zeros(2);

for k=1:num\_test

[~,I]=max(est(:,k));

[~,J]=max(TestTargets(:,k));

conf(J,I)=conf(J,I)+1;

end

conf

temp = 0;

for i = 1:2

temp = temp + conf(i,i);

end

100\*temp/120

.

SONUÇLAR VE YORUMLARI **PCA**

(e)

In this part, a train with a k-nearest neighbor classifier is requested.

First, the data set was loaded. Then, for gravitational acceleration, the mean was asked to be removed. The mean of each row is subtracted. After that, a standard normalization process was applied. All the matrix features were decompressed between 0 and 1. The controlled value was subtracted from the minimum value of the row under consideration. The result was divided by the difference between max and min. Thus, all values are normalized between 0 and 1.

load finalq7.mat

for j = 1:420

data(j,:) = data(j,:) - mean(data(j,:));

end

DFT\_mat = zeros(420,63);

for j = 1:420

DFT = abs(fft(data(j,:),125));

DFT\_mat(j,:) = DFT(1:63);

end

for i=1:63

DFT\_mat(i,:) = (DFT\_mat(i,:) - min(DFT\_mat(i,:))) / (max(DFT\_mat(i,:)) - min(DFT\_mat(i,:)));

end

From the 420 instances, randomly select 300 instances for training. Using the same 63 features, train a k-nearest neighbor classifier for k = 5 (In MATLAB, it takes a single line of code to do this if you install the Statistics and Machine Learning Toolbox. Simply use the fitcknn() function. There is a similar built-in function in Python as well). Then, test the classifier using the remaining 120 instances. Report the confusion matrix

Sonrasında, soruda istenildiği gibi 420 instances’tan randomly 300 tanesi training için ayrılırken kalanı test için kullanılmıştır. Sonrasında labellar true\_class’ta tutulmuştur.

idx = randperm(420);

training\_idx = idx(1:300);

test\_idx = idx(301:end);

true\_class = participants(test\_idx);

The training data, labels, and K value were given as 5 by calling the “fitcknn” function and entering it. The classifier is returned by the function. Then, the resulting classifier is used in the predict function. Then, the MSE value is calculated for the prediction and the label that should be. Then the confusion matrix has been created. Classification accuracy was found by considering the elements in the main diagonal of the confusion matrix.

KNN = fitcknn(DFT\_mat(training\_idx,:),participants(training\_idx)','NumNeighbors',5);

label = predict(KNN,DFT\_mat(test\_idx,:));

MSE = mean((label-true\_class).^2)

conf = zeros(7);

for i = 1:120

conf(label(i),true\_class(i)) = conf(label(i),true\_class(i)) + 1;

end

conf

temp = 0;

for i = 1:7

temp = temp + conf(i,i);

end

100\*temp/120

SONUÇLAR VE YORUMLARI **PCA**

# CONCLUSION

This assignment includes three problems.