Ministry of Education and Science of the Republic of Kazakhstan
Suleyman Demirel University

UDC.....

Aitimov Kuanish

Speech Recognition of Kazakh Language

THESIS

Presented in Partial Fulfillment for the
Degree of Master of Science in Computing Systems and Software
(degree code: 6M070400)
Department of Computer Sciences
Faculty of Engineering and Natural Sciences

Supervisor: **Cemil Turan**

Kaskelen, 2020

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Aitimov Kuanish

Abstract

In this research, an overage implementation of speech recognition revised. Here you can find algorithms and logic of how to convert digital signal into reasonable text. In common, all the signals recorded in m4a file format. You also will learn how to convert sound waves into digital signals, so you can use it wherever you like. In our days IT, technologies are expanding in an exponential level. There are many robots, which can understand human speech. Why we can't create one of our own? Kazakh language will also be presented in technology world. That is our goal!

Аңдатпа

Бұл мақалада сөйлеуді танудың орташа шешімдер қарастырылады. Мұнда сандық сигналды саналы мәтінге қалай түрлендірудің алгоритмдері мен логикаларын таба аласыз. Жалпы, барлық сигналдар m4a файл форматында жазылған. Толқын (сигнал) мақалада сипатталған. Сондай-ақ, сіз дыбыстық толқындарды сандық сигналдарға қалай түрлендіруге болатындығын білесіз, сондықтан оны өзіңіз қалаған жерде пайдалана аласыз. Біздің заманымызда IT, технологиялар экспоненциалды деңгейде дамуда. Адамның сөйлеуін түсінетін көптеген роботтар бар. Неліктен біз өзіміздің роботымызды жасай алмадық? Сондай-ақ, қазақ тілі бірнеше технологияларда ұсынылады. Бұл біздің мақсатымыз!

Аннотация

В данной статье описывается средняя реализация распознавания речи. Здесь вы найдете алгоритмы и логику того, как развивается распознавание речи. В основном речь записана в формате m4a file. Вы также узнаете, как преобразовать звук в цифровую форму, так что вы можете использовать его в любой программе. В наши дни IT-технологии развиваются огромными шагами. Есть много роботов, которые могут понимать человеческую речь. Почему мы не можем просто создать свой собственный! Казахская речь также будет представлена в нескольких технологиях. Это наша цель!

Acknowledgements

Thanks to the Suleman Demirel University, taking knowledge from this university was a difficult task but worth. Thanks to supervisor Mr. Cemil Turan for supporting me in a hard times and giving me advises. Without him i think this work would not be accomplished at all. Especially i want to thank my colleagues at work. Instead of pushing me with work, they always gave me a time to practice and study. In the end as all sons do, thanks to parents who were always been with me. In hard and good times!

Contents

1. Introduction

1.1 Motivation

When we speak about speech recognition problem, there are many topics, which should be discussing. First of all the sound itself. How to convert sound into digital form? Then, even if we can convert sound into digital signal, we must compare two of them, after that another problem appears. Comparing two sound is good, but what if signal is not hundred percent correct and have some differences, how to handle those differences. After handling those differences, there is main and huge problem. The language is not consist of one or two words. There are many thousands of words there. In combination, they form sentences. Therefore, this is much difficult task to solve. Among the sentences, our model must identify each word with a 100 percent accuracy! Another problem I forgot to mention is the noise problem. Program will always include noise to its algorithms. In addition, this causes more tension to hardware and to execution time. So anyway, we must find algorithm, which can filter noise from actual speech.

Now let us see what tools we have to solve our problems. In our case, we have only laptop. Of course laptop, it consist of camera, microphone, and software (in my case it is Windows). We do not need to pay attention how microphone works. If it records correctly and sound is clear, it is more than enough! What we truly need, is a program, platform, or IDE to create and test our own program. I picked "MATLAB" to start my testing and coding of my work. MATLAB is the simplest and well-developed language to work with. However, anybody can choose another tool as you wish! By the way we are going to test some algorithms on Python. Because Python has some great features to implement deep learning and deep learning is one of the ways we are going to use in this research.

I want to explain why I picked MATLAB and PYTHON. Let's revise MATLAB first. I am using student version R2019b Update 1(9.7.0.1216025) 64-bit(win64). First, if we talk about deep learning (machine learning) we have to admit that it consist of supervised and unsupervised learning. In our case, we are going to use supervised learning. In addition, for this reason MATLAB helps us with apps "classification learning" and "regression learning". If I or anybody else will ask you what the difference between classification and regression learning is, what will be your answer? Well the difference is that classification learning predicts categorical responses, while regression learning predicts continuous responses. So moreover, there are many speech recognition techniques.
Are we talking about speaker recognition, command recognition or sentence recognition? For the first step, because we are still searching the way we will test command recognition. At least we will try to recognize the given command. Then we will slightly move to sentence recognition. Therefore, our journey begins with MATLAB and its toolboxes.

1.2 Aims and Objectives

Is to create basic program. Which will recognise Kazakh language. It will help to solve language problem in the Republic of Kazakhstan if not all then at least most. The program could be integrated in social services and tourism industry for foreigners. Because of that any citizen of the republic will not suffer misunderstanding and confusion.

1.3 Thesis Outline

The first chapter is about searching for an algorithm. Which algorithms and methods are most useful to solve my problem. The second chapter is about supervised and unsupervised learning. All others will guide you into machine learning concept of recognizing any language. And in the last chapter we will talk about concepts of Kazakh language itself.

2. Preliminaries

In my work i mostly were using internet resources. Today technology as google helps everyone to search for any solution you want. So at the end of the thesis I left all useful links where i took information from. Also during my work i faced many problems on my way. The beginners road is always hard. Before i began i was searching for a tools and IDE. Was studying which one is the best and comfortable for me. As a programmer I already have experience. But, even in that case the task was difficult. If i choose java, there were no classes or libraries which could help. Even the microphone was a problem. As i learned after that, all windows microphones are using c library. So, if you wanted to write you program on any other language, you got to use c library anyway! I had no experience with c. Therefore, it was not an option to me. Thank god that we have supervisors! My supervisor which name i wrote on the title toughed me the way. I want to say thanks to him again!

3. Concepts of a sounds.

3.1 How to convert sound into digital form

Nowadays, technology allows us to convert sound into digital form. We can record and create sounds, which we like. However, how that process is happening. Let us revise, because we need to understand which features we can take from a sound to be useful in our speech recognition problem. Therefore, before we begin we need to understand what digital form means. Digital form is the data, which consists of binary numbers. Because computers understand only zeros and ones.
Microphone takes sounds. In other words, microphone takes vibrations. Vibrations caused by micro sand, which are jumping around in the tiny conductor box. By changing the resistance, frequencies and amplitudes saved in the computer memory.

3.2 Concepts of correlation.

When everything done and all data collected, there is one problem, which occurs. If we can compare integers between each other, we can compare text between each other, but is there a way to compare two sounds, or is it possible to compare many of them simultaneously?
That is where correlation concept comes out. What is correlation? If we speak about correlation, we also speak about correlation coefficient. About how similar sounds are between each other? Despite of that, correlations also can be positive, negative and no correlation at all. Positive correlation is when the set of data is well predictable or forms increasing straight graph line, which means that given data is much the same as the original data. Negative correlation is the complete opposite, which means forms negative line. However, sometimes it also could be

used in some cases, when you need to identify the complete opposite result. In addition, the last one "no correlation" is the simplest! Given data is just do not make any sense, and do not form anything.
Example of correlation on MATLAB:

```
filename1='ae.wav';
audiowrite(filename1,y,Fs);
filename2='B.wav';
audiowrite(filename2,x,Fs);
plot(xcorr(y,x));
disp(max(xcorr(y,x)));
```

3.3 Noise problem

Sometimes, when you record an audio, you also record the environment sound. It is just inevitable. Some scientist consider the noise as a chaos. In addition, this chaos is mostly hardens your work by providing you unnecessary data that you do not want to compare or correlate. In that case, we must use noise remover algorithm in order to get rid of unnecessary data. I need to add, that we are taking sound from a closed rooms, not on the street. Therefore, in that case we need to remove data frames, which are giving small amplitudes. I want to call it "removing a silence". Let all chaos appear as '0'!
Solution: To remove the noise we need to recognize the noise first. Because machine doesn't know what noise is! Therefore, we need to teach our machine to recognize it. For the reason our first step is to record the noise first before we make any speech recognition process. After we give an example of a noise to the machine, we need to subtract that signals every time when we record a sound! Lets see how it looks like in code.

```
Fs = sample_rate;            % Sampling Frequency (Hz)
Fn = Fs/2;                   % Nyquist Frequency (Hz)
Wp = 1000/Fn;                % Passband Frequency (Normalised)
Ws = 1010/Fn;                % Stopband Frequency (Normalised)
Rp =   1;                    % Passband Ripple (dB)
Rs = 150;                    % Stopband Ripple (dB)
```

```
[n,Ws] = cheb2ord(Wp,Ws,Rp,Rs); % Filter Order
[z,p,k] = cheby2(n,Rs,Ws,'low');% Filter Design
[soslp,glp] = zp2sos(z,p,k);  %Convert ToSecond-Order-Section
                              For Stability
figure(3)
freqz(soslp, 2^16, Fs)          % Filter Bode Plot
filtered_sound = filtfilt(soslp, glp, sample_data);
sound(filtered_sound, sample_rate)
```

4. Deep learning

4.1 Neurons

After all ingredients are set, we need to teach our machine to react to certain commands! Before this, we need to understand what machine learning itself mean! Machine learning is one of the computer science part, which studies the algorithms based on neural network of a human. Let me clarify my position. If we imagine a human brain, then human brain consist of neurons. In machine learning algorithms, those neurons are functions. The set of functions forms complete neural network. Neurons itself does not do much without set. Moreover, the set itself consist of three parts (layers). Input layer, hidden layer, output layer. Let me provide an example. Imagine we are teaching a student to cook a pancake. In this example, we assume our student as a machine. This student given ingredients. Let us say egg, butter, pan, milk, sugar. As a supervisor, we provide an example for a student, to make him repeat. According to our example, the student repeat all our steps and makes it right. OK now revise. Those ingredients were an input layer, our example was hidden layer, and the result of a student is output layer. I hope my example is clears understanding a neural network concept.
Now, I think it is time to explain the neurons. What are they, and how they made? Neurons are not as difficult as you imagine. Do not make term 'neural network' scare you! For example, let's see how two input neuron looks like. (see Figure 4.1)

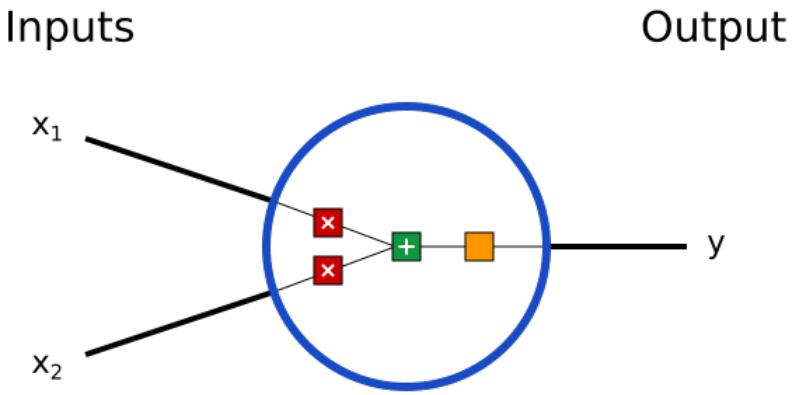


Figure 4.1: neuron

What is happening here? Well the neuron takes two input variables, makes some math with them and outputs a variable. In addition, complete set of them, connected to each other forms a complete neural network.

4.2 Supervised and Unsupervised learning

Do you remember our example with a student who makes cupcakes? So this one is example of a supervised learning. We provide example of how result must look like. If we talk about unsupervised learning, the example will be slightly different. In this case, we don't have example of how pancake must look like. Instead, we have a taster. Taster will tell us, is it tastes like pancake or not, and give positive respond or negative according to student's cook. This one is an example of unsupervised learning.

This time we should chose, which algorithm we would use. According to example above, unsupervised is not suitable for us, because we have supervisor and example of how output must be. Now let us see supervised learning more deeply!

There are several algorithms in supervised learning. Support Vector Machine, Naïve Bayes, Nearest Neighbor, Neural Networks. Let's revise each of them step by step and pick most suitable for us.

Support Vector Machine (SVM) is one of the supervised algorithms, which used to classify data. It can be used in both regression model and classification model. How does it work? On the image below is an example of classes of triangles and

circles. 4.2 Our task is to draw a line, which will separate them between each

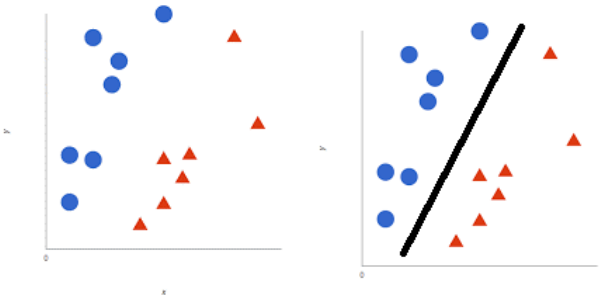


Figure 4.2: trained and untrained svm

other. Approximately the result would be like this on the second figure:4.2

However, sometimes you will have data, which will not be as same as above example. There are some examples, which I want you to try to separate! 4.3

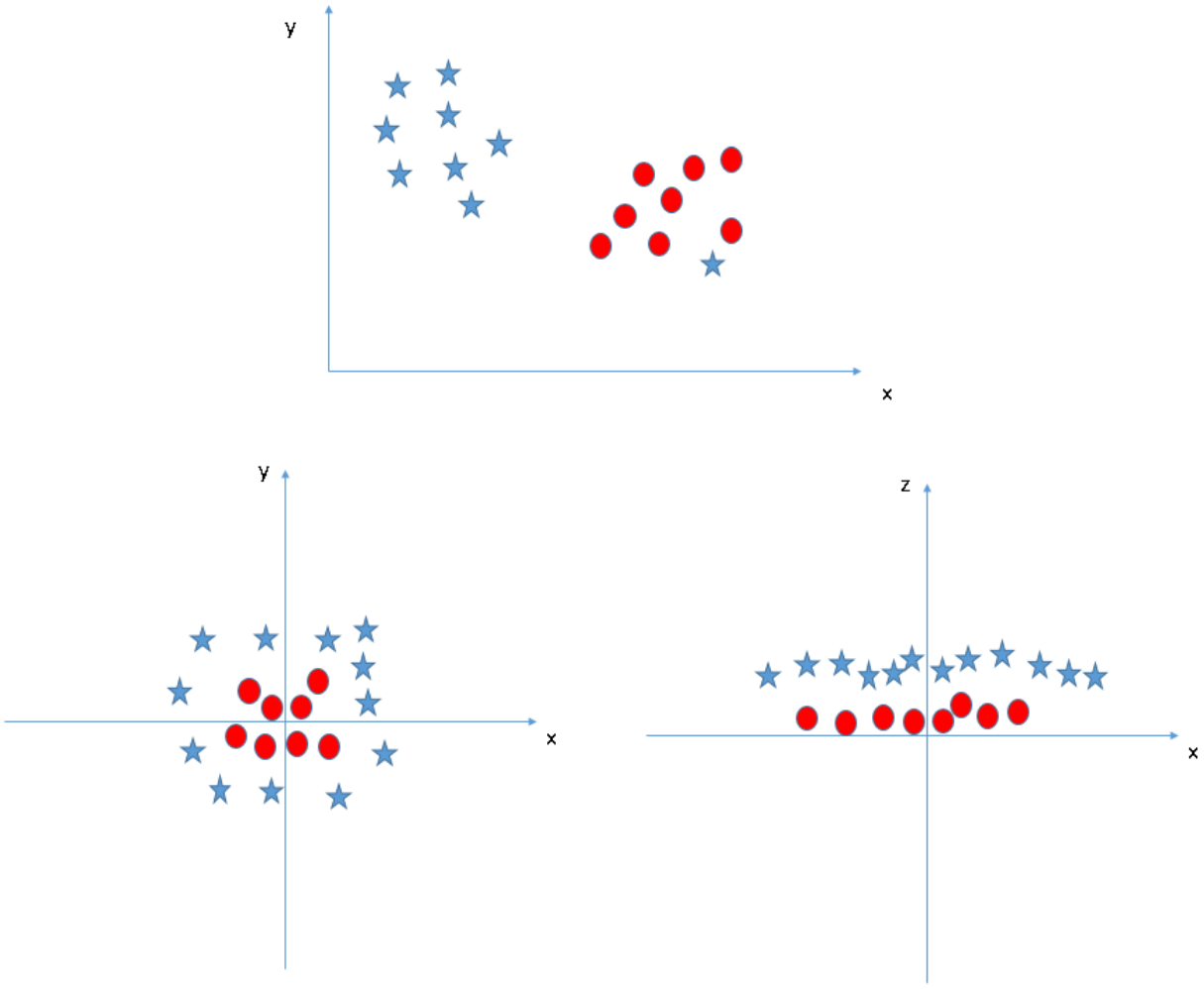


Figure 4.3: non-linear data

For these particular examples, we have a problem. For this, we have a solution. A Kernel Trick. What does it mean? A Kernel trick helps us to separate

inseparable examples as shown on figures above (4.3). Of course, we could draw a separating line or circle manually, but the best case is to have an algorithm, which will implement the hard work itself.

4.3 Naïve Bayes

Naïve Bayes this is prediction algorithm in machine learning the formula is as follows:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{4.1}$$

Bayes probability theorem is identifying the probability according to several amount of events and conditions that might be related to the event!
P(A|B): posterior probability P(A): prior probability P(B|A): likelihood P(B): Predictor prior probability.
Example:

```
Good weather to play football
weather     play football
sunny     no
overcast yes
sunny     no
sunny     yes
overcast yes
rainy     no
rainy     yes
sunny     yes
sunny     no
overcast yes
rainy     no
```

Problem! Will players play on rainy days?
P(Yes | rainy) = P(rainy | Yes) * P(Yes) / P (rainy)
P(rainy | Yes) = 1/3
P (rainy) = 3/11
P(Yes) = 1/11 now calculate:

P(Yes | rainy) = 1/3 * 1/11 / 3/11 = 1/3 * 1/3 = 1/9(0.111) too low probability!

4.4 Nearest Neighbor and K-means.

K-nearest neighbor is a supervised machine learning algorithm that can be used to solve both classification and regression problems. The K-means algorithm assumes that similar things exist in close proximity.

Notice in the image above 4.3 that most of the time, similar data points are close to each other. The KNN algorithm hinges on this assumption being true enough for the algorithm to be useful. KNN captures the idea of similarity (sometimes called distance, proximity, or closeness) with some mathematics we might have learned in our childhood— calculating the distance between points on a graph.
The KNN Algorithm:

- Load the data
- Initialize K to your chosen number of neighbors
- Calculate the distance between the query example and the current example from the data.
- Add the distance and the index of the example to an ordered collection
- Sort the ordered collection of distances and indices from smallest to largest (in ascending order) by the distances
- Pick the first K entries from the sorted collection
- Get the labels of the selected K entries
- If regression, return the mean of the K labels
- If classification, return the mode of the K labels

Choosing the right value for K. To select the K that's right for your data, we run the KNN algorithm several times with different values of K and choose the K

that reduces the number of errors we encounter while maintaining the algorithm's ability to accurately make predictions when it's given data hasn't seen before.
K means algorithm – How it works! The input: K, set of points x1. . . xn;(K is an input!)
Place centroids to random locations C1. . . Cn; First we gona run through our data set and for each individual we gona find nearest centroid!
-for each point Xi:
Find nearest centroid dj argmin D(Xi,Cj) –Euclidean distance between instance Xi and cluster center dj. Assign the point Xi to cluster j.
The next step we run over our cluster K centroids:
For each cluster j = 1 . . . K:
And for each centroid recompute its position! For a = 1..d

$$Cj = 1/Nj_(xicj)Xi(a) \tag{4.2}$$

Now new centroid d j = mean of all points Xi assigned to cluster j in previous step! Keep running until no point is changing cluster members!
Well, now it is time to choose. Actually any of these algorithm works, but with their own special result. Because we are doing command based recognition, lets stay with regression model learning!

4.5 Neural Network

In the previous topics, we discussed about how neural networks built. We have learned that every neural network consists of sets of functions. The simplest model is as follows:

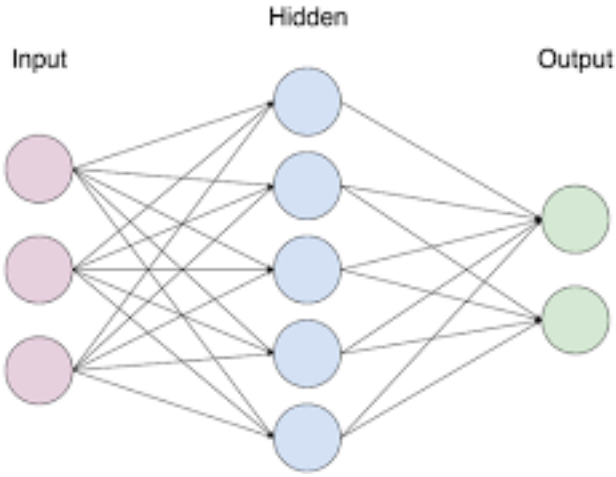


Figure 4.4: neural network

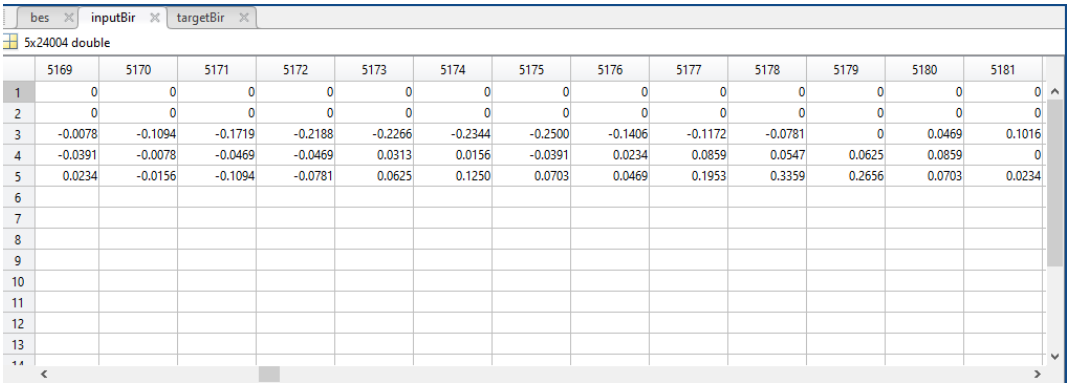
In this diagram the neural network is processing the images. Whenever we put on some closed shape, it fills it with surrounding color. Using this concept, we must input some sound and make a text output. For now, text output is more than enough. So, our input will be sound and our output will be text.

```
recObj = audiorecorder;  % Getting recObject class
disp('start speaking') % warn user to speak
recordblocking(recObj,5); % record sound for 5  sec
f = getaudiodata(recObj); % getting sound as array
word = getWord(f); % Here is our function which we will implement
later
disp('end of recording'); % warn about end of recording

plot(f); % plot the diagram of a sound.
disp(word); % display the word
```

Remember, we did not include here the noise removal function yet. Read the previous chapters about noise removal. Soon we will add all our elements together and complete whole project. So, in this getWord() function we execute our neural network to return us the word we need! But remember! We got to train our model first, before we use it! Let us go to the training part then! Training part: In order to create model, we must create some amount of examples. By word examples I meant outputs, remember we are doing command-based recognition, so our outputs must be given to machine! Lets assume we will have five outputs.

We are going to use Kazakh numbers for commands. So, in total 5 outputs. “бір”,”екі”,”үш”,”төрт”,”бес”. Those are going to be our outputs. Now as long as we have outputs we need to have some models for each our output. I have prerecorded each sound and already put them into an array (or set of data if you wish). This is how input data looks like: Not all of the data fit unfortunately.



	5169	5170	5171	5172	5173	5174	5175	5176	5177	5178	5179	5180	5181
1	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0
3	-0.0078	-0.1094	-0.1719	-0.2188	-0.2266	-0.2344	-0.2500	-0.1406	-0.1172	-0.0781	0	0.0469	0.1016
4	-0.0391	-0.0078	-0.0469	-0.0469	0.0313	0.0156	-0.0391	0.0234	0.0859	0.0547	0.0625	0.0859	0
5	0.0234	-0.0156	-0.1094	-0.0781	0.0625	0.1250	0.0703	0.0469	0.1953	0.3359	0.2656	0.0703	0.0234
6													
7													
8													
9													
10													
11													
12													
13													

Figure 4.5: dataset

However, this is how it is. From the first line till the last are sounds. For the first row it is “bir”, second “eki”, third “ush”, fourth “tort”, fifth “bes”. Now the next step is to extract the features! For this reason I have also recorded a bunch of sounds for each number. In my case five recordings for each number. Now lets extract the features!

Mel Frequency Cepstral Coefficient (MFCC) is method to extract the features we need. This method is used to identify and classify objects by selecting common features of any object. But before doing that you need provide multiple examples to save the features. Let me explain in other ways! Imagine you have a million pictures on your computer. Instead of comparing every picture where is you, better to identify common features of every picture where is you. For example identifying the change of gradients from color to another. Also known as spectral analysis but in our case we will not use that method. So, by using this analysis you can create a program which will identify your skin color, shirts, width and height. That algorithm helps to decrease amount of comparing and executes very fast. Returning to our case. In our problem, features will be the same frequencies from each word. The identical frequencies we will find, and then more accurate our machine will work. Steps at glance: Well the first step is to compare every sound and subtract every difference they and save the similarities. Remember, the similarities could be in the different amount of time. More we have examples,

then accurate the result will be.

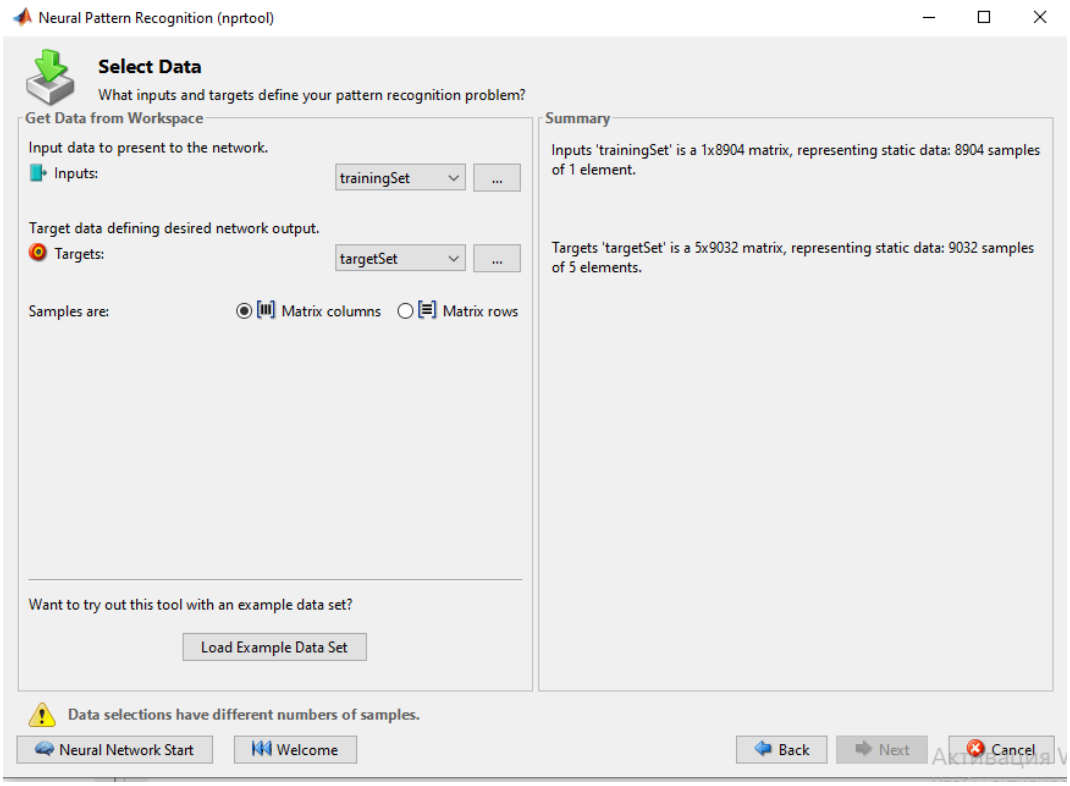


Figure 4.6: Neural Net Pattern Recognizer

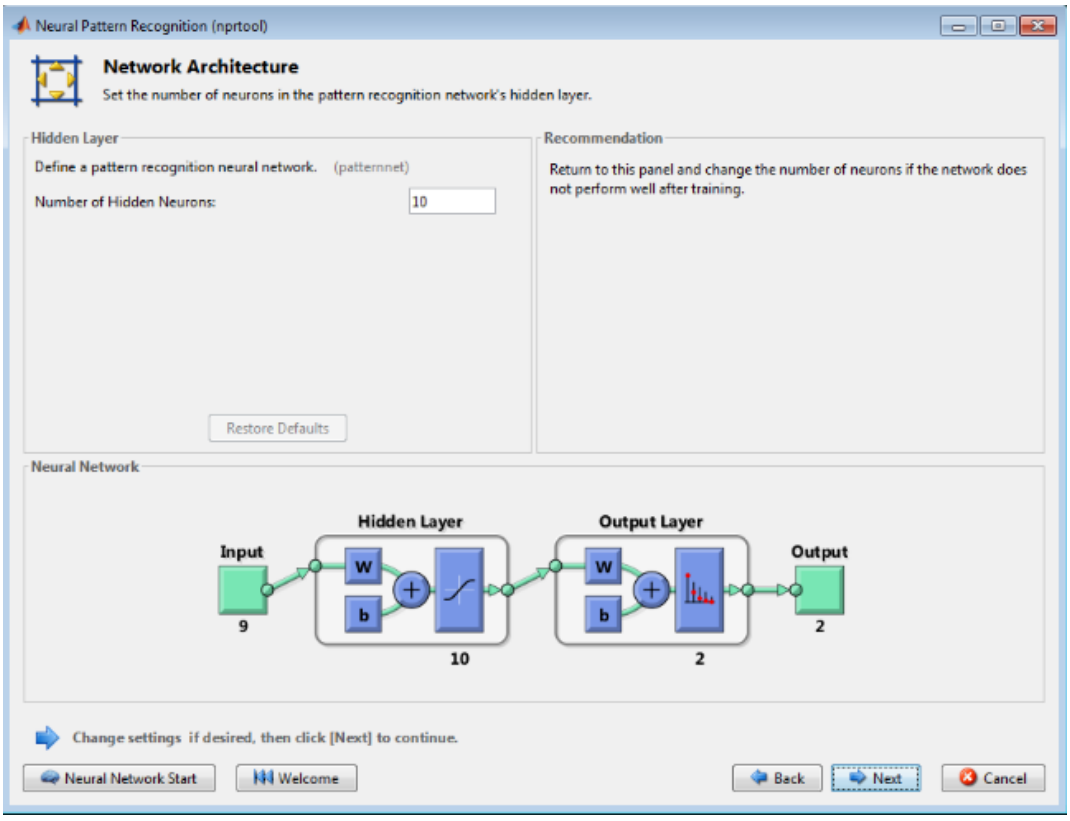


Figure 4.7: inputs and target

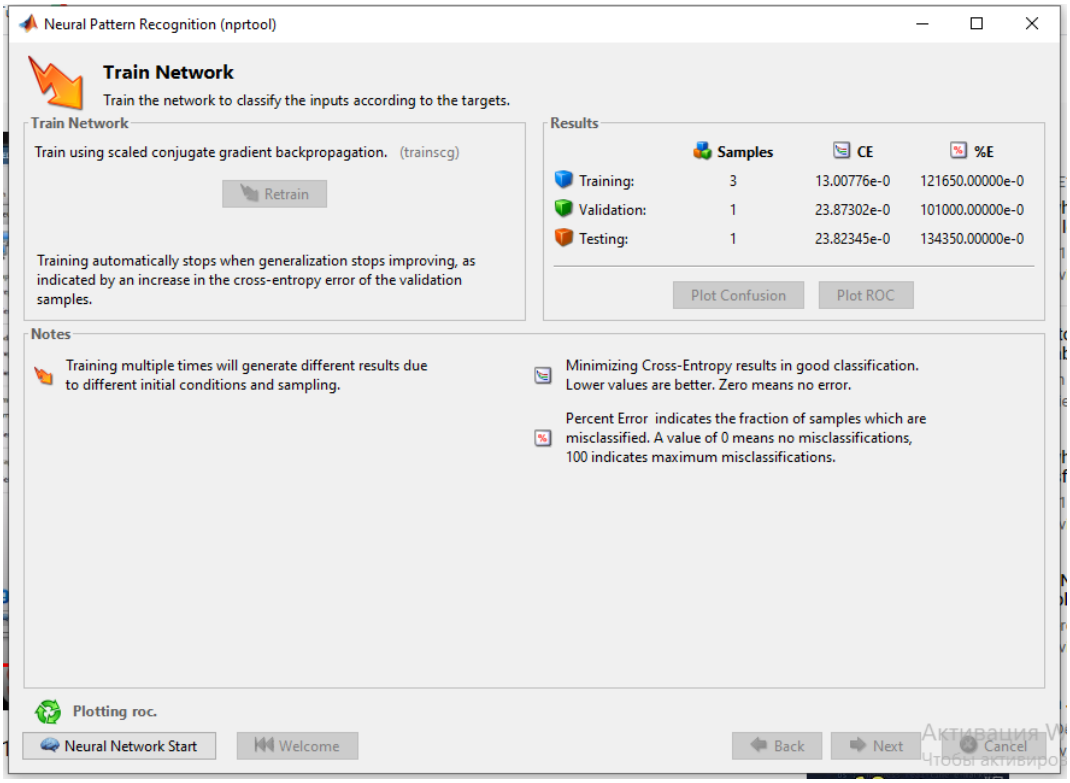


Figure 4.8: settings

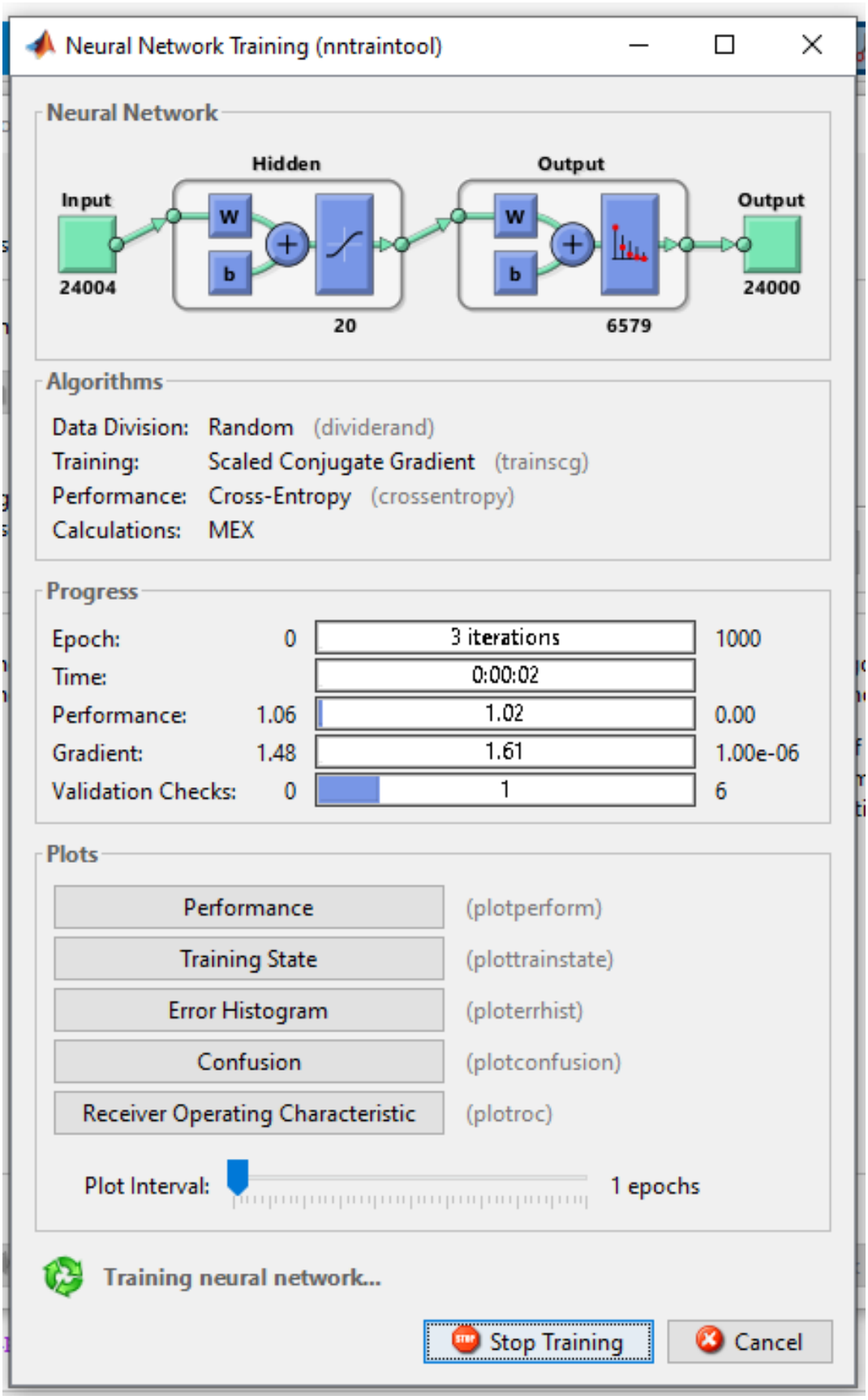


Figure 4.9: training process

After we complete training, we can create matlab function according to the training set.

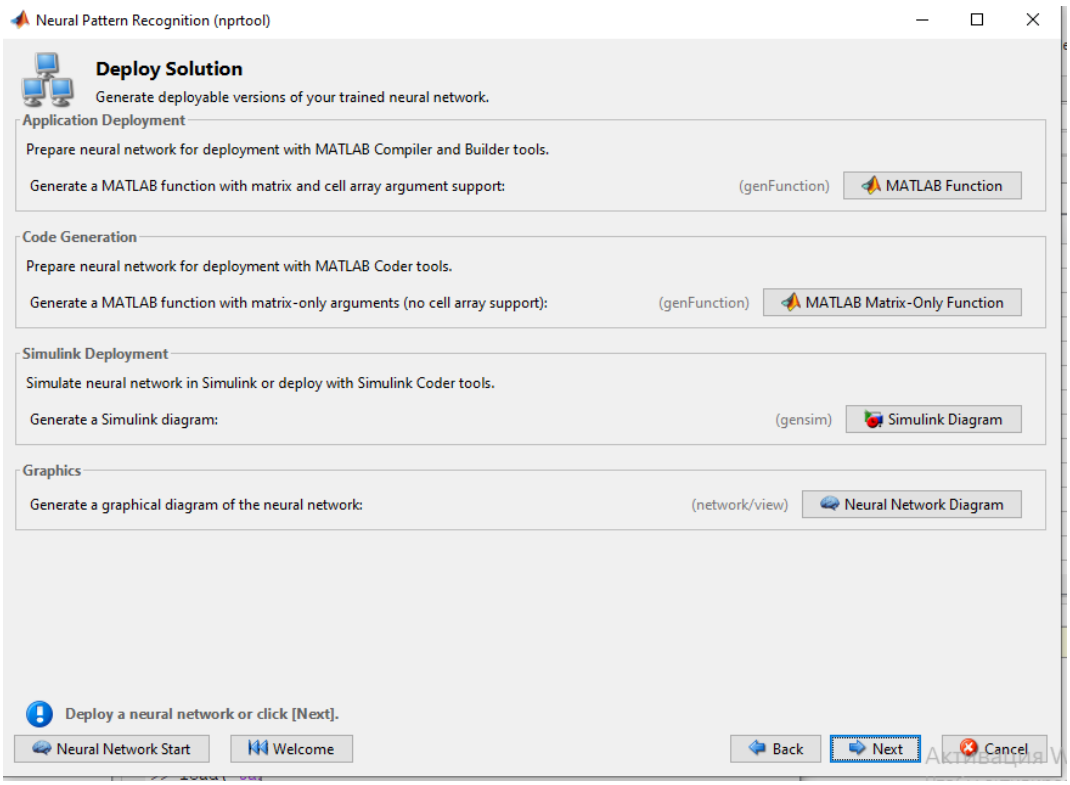


Figure 4.10: creating function

5. Features of Kazakh language.

5.1 Concepts of extracting features.

Imagine that you have been solving a problem where you should identify trees on pictures! Also the amount of images will approximately be one million. For a human it is disgusting task, but for a machine it is simple. Everything in our nature has a feature. The machine will take the difference of gradients of the picture. There is no reason for a machine to remember all the pixels of the tree. if the gradients changes correctly according to algorithm, it is not a big deal. Let me explain feature extraction in more simple way! Imagine that you have to identify watermelons among all the fruits! The task is simple. If fruit has green striped color then it is a watermelon! The exact same algorithm is implemented in feature extraction.(fig:5.1) If we talk about features of texts and sounds, so

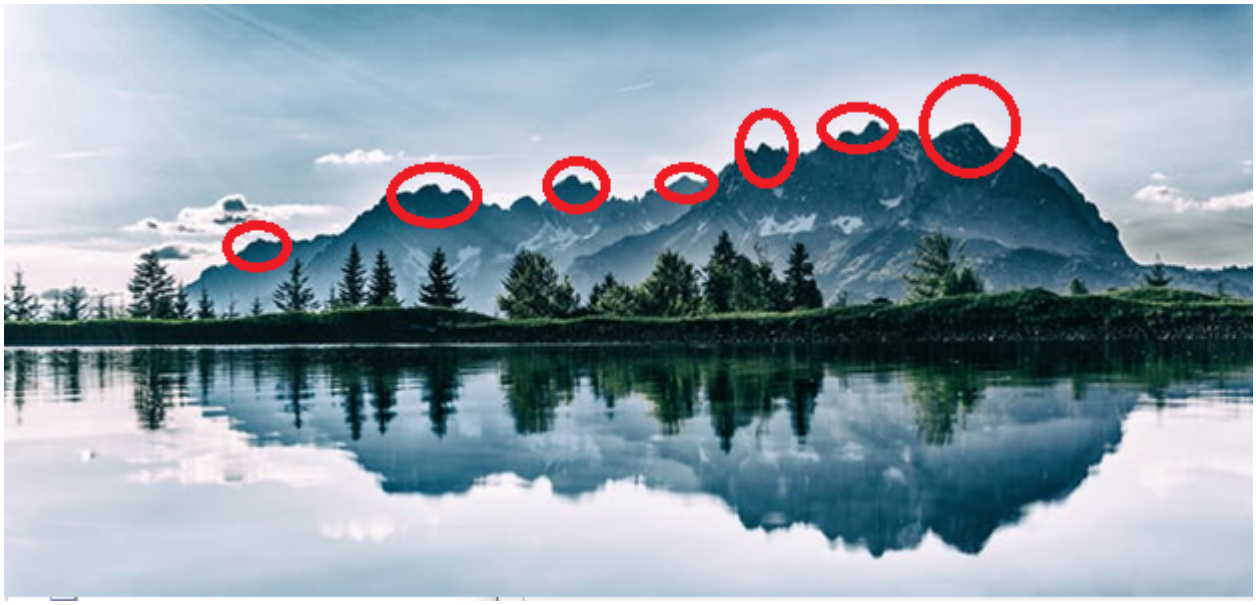


Figure 5.1: get the important features

there is a different approach. Implementation steps:

- The first one is an observation method. Our machine is looking for a changes according to a text.
- The second is similarities method. So in this case we provide a bunch of almost same signals to the machine. And it pics the features by itself. The features which are present in all signals!

5.2 MFCC feature extraction.

Mel-frequency cepstral coefficients, is used in speech recognition algorithms. The main formula is :(fig:5.2) According to formula we already have word sequences

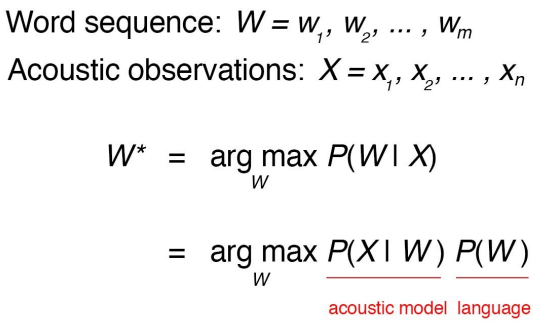


Figure 5.2: MFCC main formula

words = [“bir”,”eki”,”ush”,tort”,”bes”]. And the acoustic:(fig:5.3) Now let us perform and make set of training data for MFCC! For this we will extract features of every sound we have and place them in a set called training set. To use mfcc functions in MATLAB it requires Audio Toolbox. So you need to install this app first!(fig:5.4)

```
fs = 8000;
[audioIn,fs] = audioread('bes1.m4a');
[coeffs,delta,deltaDelta,loc] = mfcc(audioIn,fs);
disp(coeffs);
```

The result of the mfcc coefficient of one audio file. As you can see in the figure above, the amount of 0 values are pretty much! And we need to get rid of that. For this let’s remove unnecessary silence that we have! Write the code fs = 8000; [audioIn,fs] = audioread(’bes1.m4a’);

indexOfLoud = audioIn > 0; onlyLoudParts = audioIn(indexOfLoud);

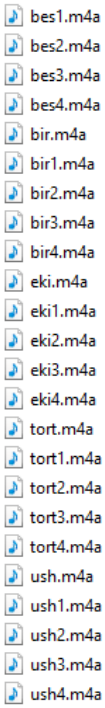


Figure 5.3: sounds

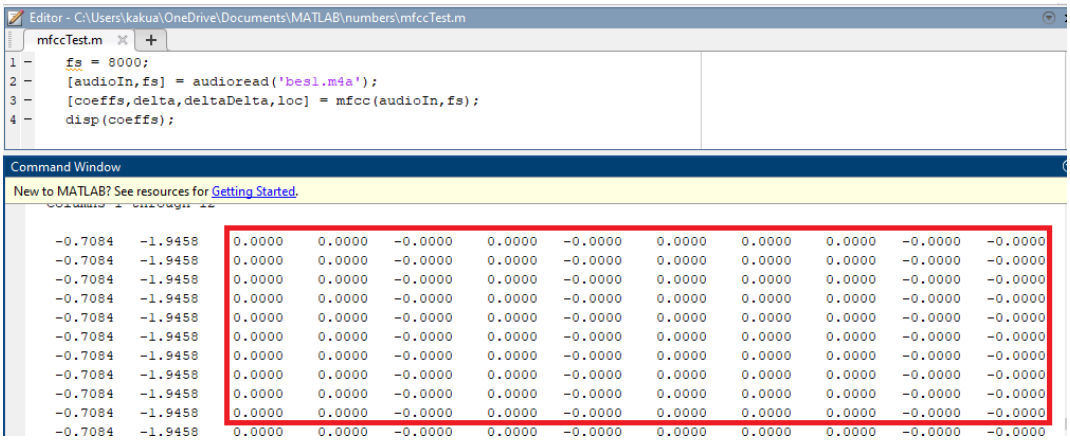


Figure 5.4: coefficents of sound bes1.m4a

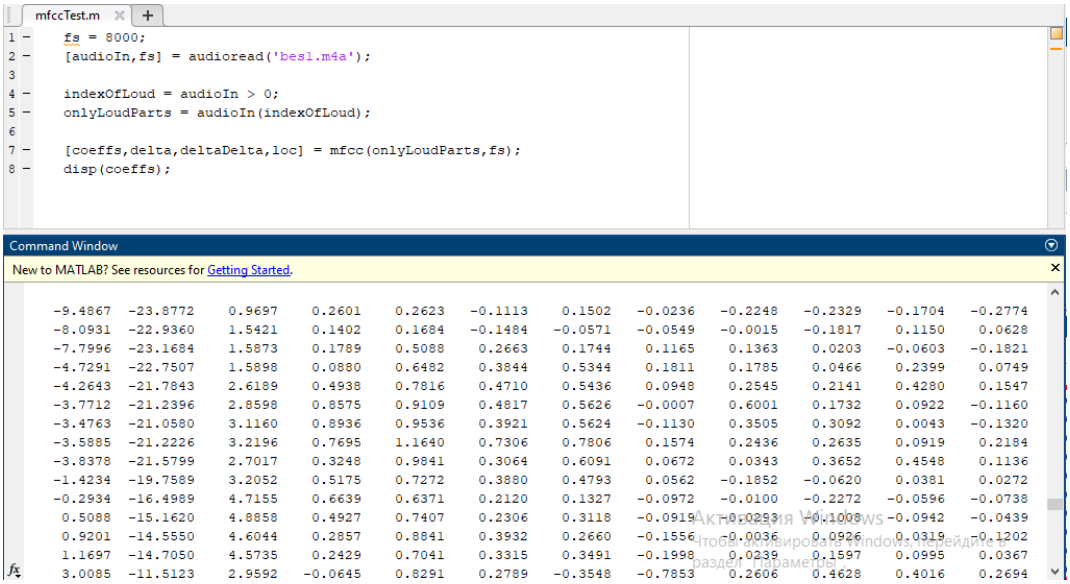


Figure 5.5: coefficents of sound bes1.m4a with silence removed

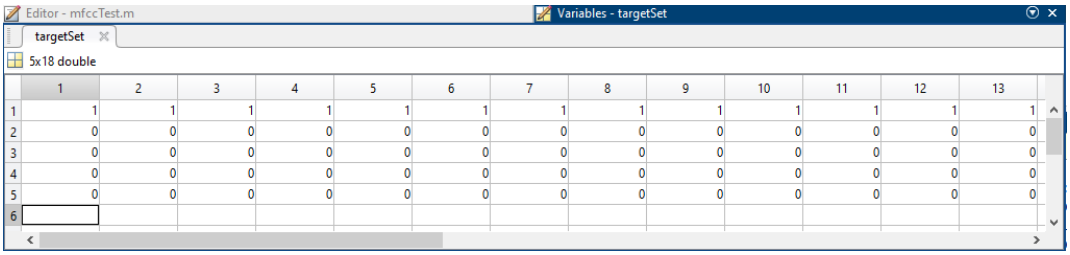


Figure 5.6: targetSet

[coeffs,delta,deltaDelta,loc] = mfcc(onlyLoudParts,fs); disp(coeffs);

As you can see now (fig:5.5) we don't have unnecessary zeros in our file. Perform this operation for every sounds step by step, and do not mix it up! Because later you will have problem sorting them in the training set. After performing try to save it immediately in training set, so everything is placed in order. After our training set is over it is time to create or target set. My target set will look like this:(fig:5.6) As you can see, the first row is filled up with ones. It is because we placed the coefficients of sounds "bir" in first place. Because of first sixty coefficients are from sounds "bir", "bir1", "bir2", "bir3", "bir4" the first row is filled with ones. After the sixtieth frame ones are going down to second row and etc. according to order. After we complete target set and the training set it is time to prepare neural network. Go to the apps and select Neural Pattern Recognition app! (fig:5.7) After window appears, select our input data and output data! As an input we select 'trainingSet' and as an output 'targetSet'! (fig:5.8, fig:5.9) After everything is set, it is time to get into the next step! Press next and train.

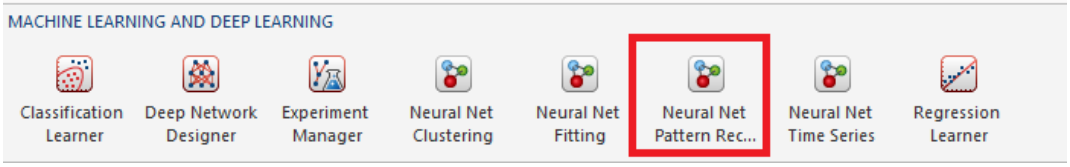


Figure 5.7: Neural net pattern recogintion app

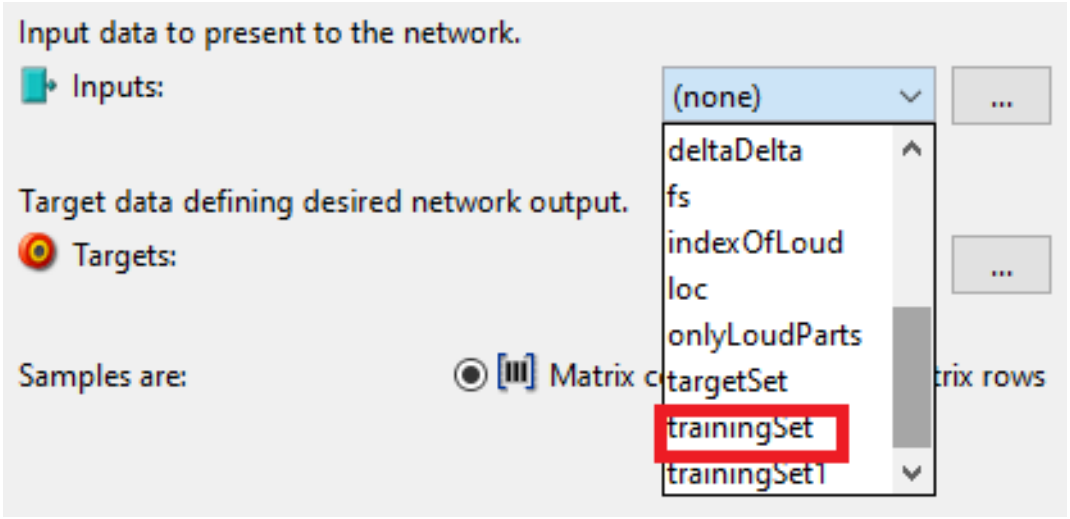


Figure 5.8: input select

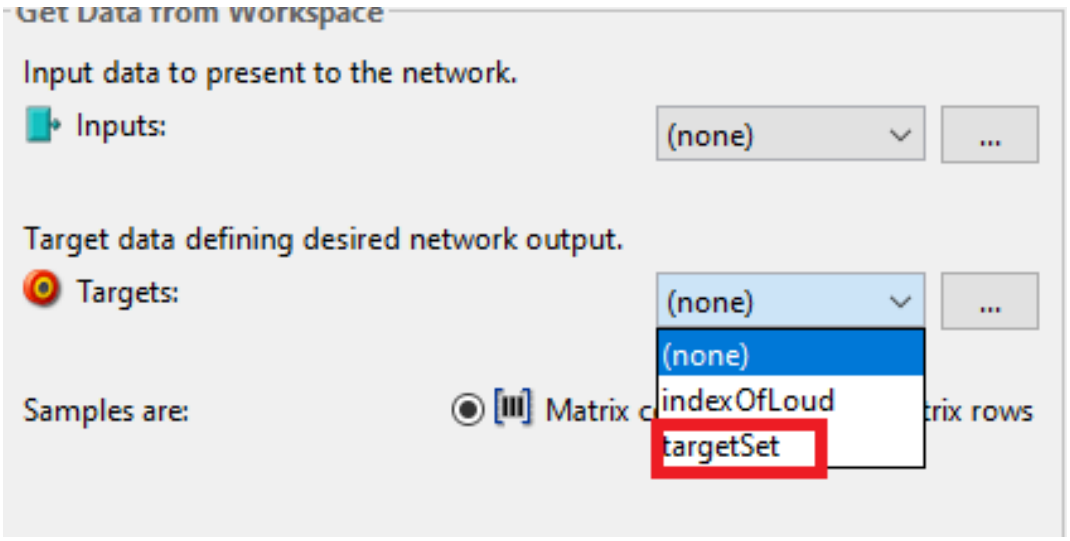
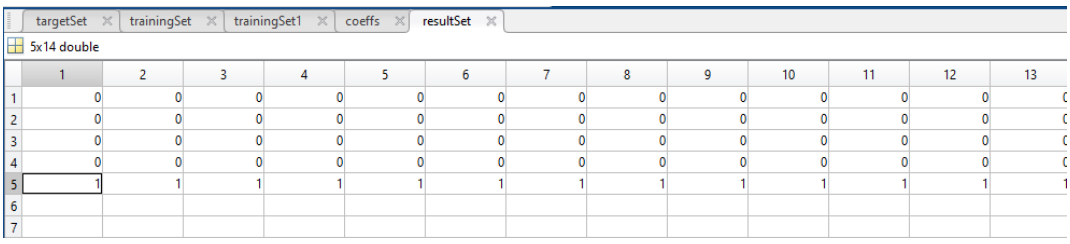


Figure 5.9: output select



	1	2	3	4	5	6	7	8	9	10	11	12	13
1	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0
5	1	1	1	1	1	1	1	1	1	1	1	1	1
6													
7													

Figure 5.10: resultSet

By the way, the number of neurons in my case is 20, but you may set any amount you want. After neural network is successfully trained save the it as the matlab function! Now the moment of triumph! Let's use our function in our problem. Will work correctly or not, let test!

```
fs = 8000;
[resultSet,fs] = audioread(bes.m4a');
Result = TraindedNN(resultSet);
```

TrainedNN is a function we made before! Now let us look to the result set!(fig:5.10) So, as you see, the result shows us ones on fifth row. It means that features matching the signal of sound file "bes.m4a" and the program works well! Well, executes perfectly. But the problem is that we used our audio file. Audio file does not have any differences in frequencies and periods. Therefore, we need it to test on actual voice using microphone.

6. Integration with application

6.1 MATLAB Application Designer

Now when we have accomplished main function, it is time to integrate them with an application. We are going to use application designer to make a user-friendly application. MATLAB provides us tool AppDesigner to make it easy for us to integrate and test. Because we are familiar with programming we could simply avoid this step, but most of the user are not programmers and they using user-friendly interfaces where everything is logically and visually understandable. We are not an exception either and are going to make some interface to make it easy for non programmers! First go the "Apps" tab and select "AppDesigner". After

Figure 6.1: select designer

you select you will be provided a user interface to design the future application. Select an empty blank application! On the left side select necessary item for your application! In my case I am going to use buttons, diagrams and labels, but you may add anything you want. Well my design will look like this:(fig:6.3) So, let me explain each element's purpose. The button will trigger our microphone for 3 seconds. The diagram will plot the sound we just speak. In addition, the text field will write our word. Now press the button by right click and select go to callbacks. Application will automatically create a button function for us where we can implement our coding. Before we start coding the button function, we need to add functions to the app! To add functions you firstly have to change the mode to code view. Here in the figure (fig:6.4) shown how to add functions.

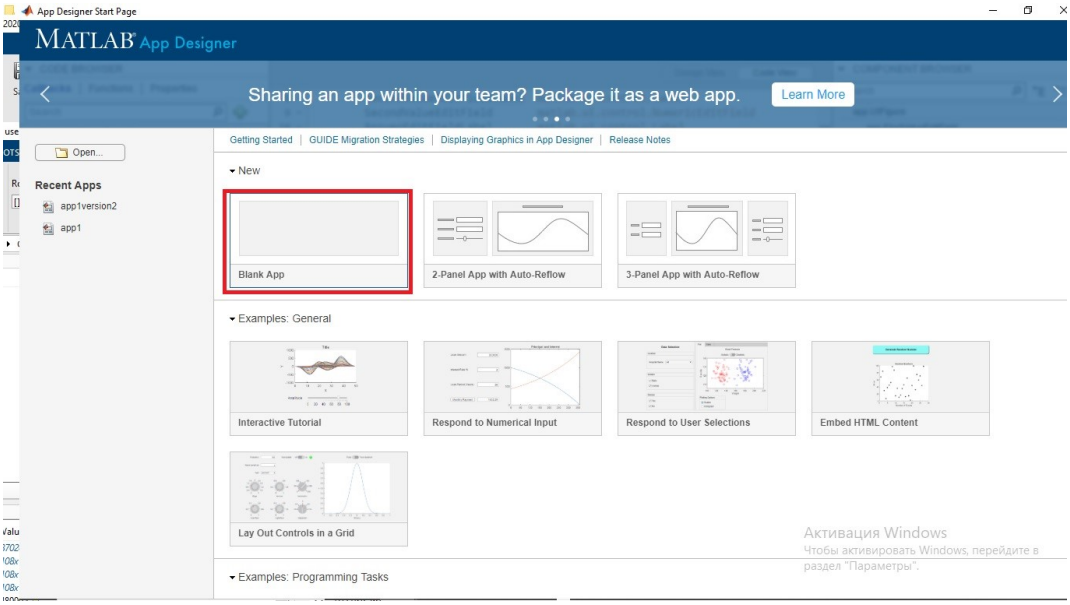


Figure 6.2: blank application

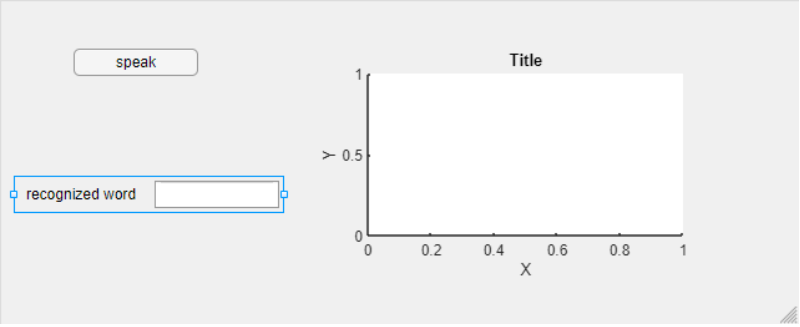


Figure 6.3: design

If you press plus button, you will give an option to select what kind of functions you want to create! Let us select public functions. By default, MATLAB creates this code:

```
        methods (Access = public)
        function results = func(app)
        end
    end
```

Program gave function called 'func'. To change the name just edit it! I am going to copy and paste all the functions we have created before, but most important. It is our 'TrainedNN()' function. Without it all the processes is not so much useful. Therefore, after you copy your necessary functions, we may continue to write logic to our 'speak' button.

```
recordblocking(recObj,5);
```

Figure 6.4: functions

```
data = getaudiodata(recObj);
res = TrainedNN(data);
x1 = res(1,:);
x2 = res(2,:);
x3 = res(3,:);
x4 = res(4,:);
x5 = res(5,:);
maxVal = max([x1,x2,x3,x4,x5]);
if max(x1)>= maxVal
            app.recognizedwordEditField.Value = "bir";
elseif max(x2)>= maxVal
    app.recognizedwordEditField.Value = "eki";
elseif max(x3)>= maxVal
    app.recognizedwordEditField.Value = "ush";
elseif max(x4)>= maxVal
    app.recognizedwordEditField.Value = "tort";
else max(x5)>= maxVal
    app.recognizedwordEditField.Value = "bes";
end
```

Now test it! If program works correctly it will react to any voice you create and put a string to edit text. By the way set edit text to editable false. The

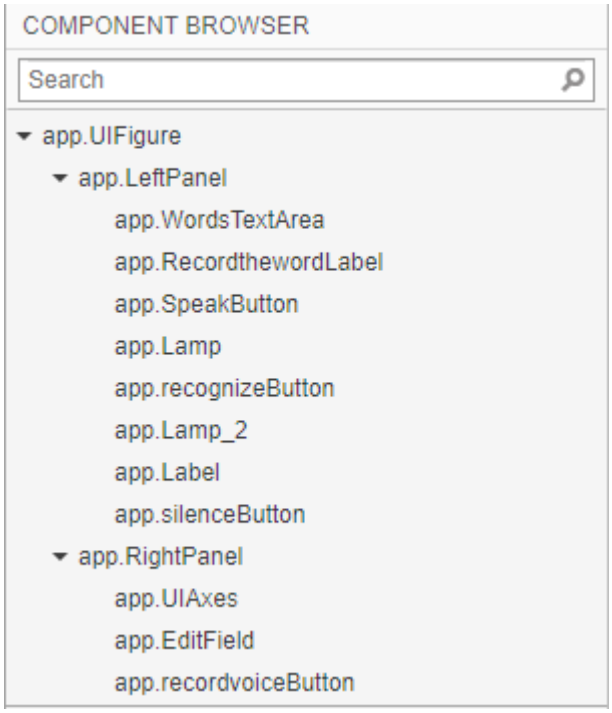


Figure 6.5: component browser

edit option is not needed in our case. As you can see the diagram of ours is not showing anything. Got to solve this mess. So in order to access items of your application you can look for them on the right tab of the application. See it in figure. It called component browser. The figure is an example of how you can find your elements. In actual project there are only three items! And you need to select "app.UIAxes" element. Just add to the button function you just created below code!

```
plot(app.UIAxes,data,'-r');
```

Yes. It is that simple! Only one line of code needed. Moreover, the output will be as follows (fig:6.6).

Of course, remember that diagram will appear as soon as you speak! Maybe you mentioned already, but in the diagram we have a lot of silence. The silence as discussed before is an unnecessary data that we can get rid of. So let's move on and remove that silence for better execution!

Add another function called removeSilence()! I forgot to mention. Before we make another function let us record the silence itself. In my case recorded sound will be "silence.m4a". Now we may start the code.

```
function results = removeSilence(data)
```

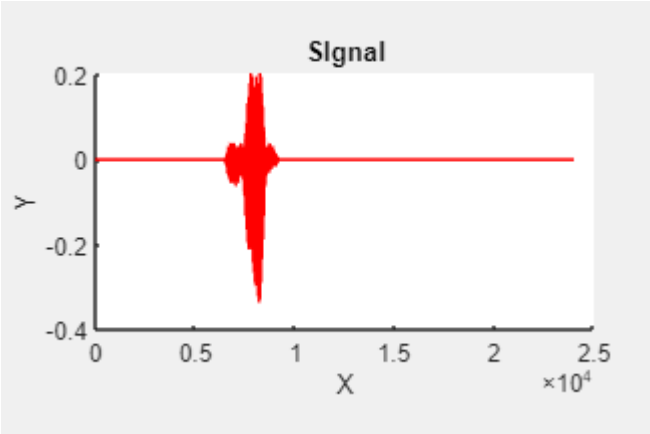


Figure 6.6: plot

```
    [MyData,8000] = getaudiodata("silence.m4a");
    results = data - MyData;
end
```

Remember! We are recording sounds for three seconds. So, the recorded silence must be also for three seconds. Otherwise program will return a mistake of matrix differences! When we finish the function, it is easy now to insert it in the actual main code! Put right after the recording of a sound.

```
    recordblocking(recObj,5);
data = getaudiodata(recObj);
data = removeSilence(data);
res = TrainedNN(data);
```

Now it is time to say that our work is complete. It is not perfect, but gives us look of how recognitions work. This one was only the beginning. It is not GOOGLE level speech recognition, but a graduate level project. Nevertheless, it done as my research. In addition, I am looking forward of making a full advanced recognition system in future.

7. Summary

This project is developed as a fundamental help for further research. Any other researcher can take my work and based on my results create and develop more advanced applications. This is not a GOOGLE level recognition program but a program of graduate student.
There is used only Mel-frequency cepstral coefficients algorithm to recognize the speech. The process is a command speech recognition, when program reacts on a certain word you speak. Also there are much helpful information of how to create user-friendly interface. Nevertheless, if you are a complete beginner in MATLAB programming it is a much more helpful tool for you. Because thesis is written exactly for beginners who just started education in machine learning. Every step is provided with code example and figure.
In the end if your program is working fine according to my algorithm. I want to congratulate you. Because, it means that you are not a beginner any more and have gained knowledge provided in this work. Also it is a pleasant for me too, that my work somehow helped someone on his journey of speech recognition.

8. Conclusion

Kazakh language has specific pronunciation. You write the sound what you hear. If you take for example french or english languages, there are several problems. Each letter pronounced differently than when it is in the word itself. Let's take for example simple word "jentlmen" and compare with french version "monsieur". Next let's see the transcription [jentlmen]. Ok, english transcription is good, but what about french [musiu]. As we can see there are several letters that are not pronounced in french language. So it much harder to implement french speech into digital form. If we take kazakh language for example there are less words which you pronounce differently how it writes. Some kazakh names for example "Ерканат" but sounds like [Ерғанат]. One letter just transformed to another. In the end i want to say that kazakh speech recognitions are the most easiest ones and have huge potential.

A. Source codes

trainedModel.m

```
    //inputs
x1_step1.keep = [269 270 272 273 274 275 276 277 278 279 280 281....
x1_step2.xoffset = [0;0;-0.0078125;-0.0703125;-0.1171875;;....
x1_step2.gain = [256;256;256;28.4444444444444;17.0666666666667;....
x1_step2.ymin = -1;

//layer 1
b1 = [-1.4000409813359917521;-1.2529949148794679026;....
IW1_1 = [0.0023176903715810275204 0.0033936827451215918934  ....

//layer 2
b2 = [-0.73147278363070677987;0.88017806052273106499;....
LW2_1 = [-0.26856722133092808535 -0.96729555631456820564 ....

//output
y1_step1.xrows = 24000;
y1_step1.keep = [269 270 272 273 274 275 276 277 278 279 280  ....
y1_step1.remove = [1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 ....
y1_step1.constants = [0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;....
% Format Input Arguments
isCellX = iscell(X);
if ~isCellX
    X = {X};
end
```

```
% Dimensions
TS = size(X,2); % timesteps
if ~isempty(X)
    Q = size(X{1},2); % samples/series
else
    Q = 0;
end

% Allocate Outputs
Y = cell(1,TS);

% Time loop
for ts=1:TS

    % Input 1
    temp = removeconstantrows_apply(X{1,ts},x1_step1);
    Xp1 = mapminmax_apply(temp,x1_step2);

    % Layer 1
    a1 = tansig_apply(repmat(b1,1,Q) + IW1_1*Xp1);

    % Layer 2
    a2 = softmax_apply(repmat(b2,1,Q) + LW2_1*a1);

    % Output 1
    Y{1,ts} = removeconstantrows_reverse(a2,y1_step1);
end

% Final Delay States
Xf = cell(1,0);
Af = cell(2,0);

% Format Output Arguments
if ~isCellX
```

```
    Y = cell2mat(Y);
end
end

% ===== MODULE FUNCTIONS ========

% Map Minimum and Maximum Input Processing Function
function y = mapminmax_apply(x,settings)
y = bsxfun(@minus,x,settings.xoffset);
y = bsxfun(@times,y,settings.gain);
y = bsxfun(@plus,y,settings.ymin);
end

% Remove Constants Input Processing Function
function y = removeconstantrows_apply(x,settings)
y = x(settings.keep,:);
end

% Competitive Soft Transfer Function
function a = softmax_apply(n,~)
if isa(n,'gpuArray')
    a = iSoftmaxApplyGPU(n);
else
    a = iSoftmaxApplyCPU(n);
end
end
function a = iSoftmaxApplyCPU(n)
nmax = max(n,[],1);
n = bsxfun(@minus,n,nmax);
numerator = exp(n);
denominator = sum(numerator,1);
denominator(denominator == 0) = 1;
a = bsxfun(@rdivide,numerator,denominator);
end
```

```
function a = iSoftmaxApplyGPU(n)
nmax = max(n,[],1);
numerator = arrayfun(@iSoftmaxApplyGPUHelper1,n,nmax);
denominator = sum(numerator,1);
a = arrayfun(@iSoftmaxApplyGPUHelper2,numerator,denominator);
end
function numerator = iSoftmaxApplyGPUHelper1(n,nmax)
numerator = exp(n - nmax);
end
function a = iSoftmaxApplyGPUHelper2(numerator,denominator)
if (denominator == 0)
    a = numerator;
else
    a = numerator ./ denominator;
end
end

% Sigmoid Symmetric Transfer Function
function a = tansig_apply(n,~)
a = 2 ./ (1 + exp(-2*n)) - 1;
end

% Remove Constants Output Reverse-Processing Function
function x = removeconstantrows_reverse(y,settings)
Q = size(y,2);
x = nan(settings.xrows,Q,'like',y);
x(settings.keep,:) = y;
x(settings.remove,:) = repmat(settings.constants,1,Q);
end
```

app1.matlab

```
    classdef app1version2 < matlab.apps.AppBase

    % Properties that correspond to app components
```

```
properties (Access = public)
    UIFigure            matlab.ui.Figure
    GridLayout          matlab.ui.container.GridLayout
    LeftPanel           matlab.ui.container.Panel
    WordsTextAreaLabel  matlab.ui.control.Label
    WordsTextArea       matlab.ui.control.TextArea
    RecordthewordLabel  matlab.ui.control.Label
    SpeakButton         matlab.ui.control.Button
    LampLabel           matlab.ui.control.Label
    Lamp                matlab.ui.control.Lamp
    recognizeButton     matlab.ui.control.Button
    Lamp_2Label         matlab.ui.control.Label
    Lamp_2              matlab.ui.control.Lamp
    Label               matlab.ui.control.Label
    silenceButton       matlab.ui.control.Button
    RightPanel          matlab.ui.container.Panel
    UIAxes              matlab.ui.control.UIAxes
    EditFieldLabel      matlab.ui.control.Label
    EditField           matlab.ui.control.EditField
    recordvoiceButton   matlab.ui.control.Button
end

% Properties that correspond to apps with auto-reflow
properties (Access = private)
    onePanelWidth = 576;
end


properties (Access = public)
    Property % Description
    recObj = audiorecorder;
    silence;
    counter = 1;
    words = ["","","","",""];
```

```
    s1;
    s2;
    s3;
    s4;
    s5;
end

methods (Access = public)


    function results = func(app)
        fs = 48000;
        [data] = app;

        data = data(1:end,1)/max(data(1:end,1));

        % do framing
        f_d = 0.025;
        f_size = round(f_d * fs);
        n = length(data);
        n_f = floor(n/f_size);  %no. of frames
        temp = 0;
        for i = 1 : n_f

           frames(i,:) = data(temp + 1 : temp + f_size);
           temp = temp + f_size;
        end

        % silence removal based on max amplitude
        m_amp = abs(max(frames,[],2));\\
        % find maximum of each frame
        id = find(m_amp > 0.03);
        % finding ID of frames with max amp > 0.03
        fr_ws = frames(id,:); % frames without silence
```

```
        % reconstruct signal
        data_r = reshape(fr_ws',1,[]);

    end
end

methods (Access = private)

end


% Callbacks that handle component events
methods (Access = private)

    % Button pushed function: recordvoiceButton
    function recordvoiceButtonPushed(app, event)

        curText = app.WordsTextArea.Value;
        word = app.EditField.Value;
        app.WordsTextArea.Value = curText+" "+word;
        app.words(app.counter) = word;

    end

    % Button pushed function: SpeakButton
    function SpeakButtonPushed(app, event)
        if (app.counter<=5) && (app.words(app.counter)~="")
            app.LampLabel.Text = "запись";
            app.Lamp.Color= [1 0 0];
            recordblocking(app.recObj,3);
            app.LampLabel.Text = "";
            app.Lamp.Color= [0 1 0];
            f = getaudiodata(app.recObj);
```

```
    f=f-app.silence;
    plot(app.UIAxes,f,'-r');
else
    msgbox('write word you want to record!');
end
switch app.counter
    case 1
        app.s1 = f;
        app.s1=app.s1';
        app.s1=app.s1(1,:);
        app.s1=app.s1';
    case 2
        app.s2 = f;
        app.s2=app.s2';
        app.s2=app.s2(1,:);
        app.s2=app.s2';
    case 3
        app.s3 = f;
        app.s3=app.s3';
        app.s3=app.s3(1,:);
        app.s3=app.s3';
    case 4
        app.s4 = f;
        app.s4=app.s4';
        app.s4=app.s4(1,:);
        app.s4=app.s4';
    case 5
        app.s5 = f;
        app.s5=app.s5';
        app.s5=app.s5(1,:);
        app.s5=app.s5';
    otherwise
        msgbox('library is full!');
end
```

```
    app.counter=app.counter+1;
end

% Button pushed function: recognizeButton
function recognizeButtonPushed(app, event)

        app.Lamp_2.Color= [1 0 0];
        recordblocking(app.recObj,3);
        app.Lamp_2Label.Text = "";
        app.Lamp_2.Color= [0 1 0];
        f = getaudiodata(app.recObj);
        f=f';
        f=f(1,:);
        f=f';
        plot(app.UIAxes,f,'-r');
        x1 = max(xcorr(f,app.s1));
        x2 = max(xcorr(f,app.s2));
        x3 = max(xcorr(f,app.s3));
        x4 = max(xcorr(f,app.s4));
        x5 = max(xcorr(f,app.s5));
        a=[x1,x2,x3,x4,x5];
        maximum = find(a==max(a));
        app.Label.Text = app.words(maximum);
end

% Button pushed function: silenceButton
function silenceButtonPushed(app, event)
    recordblocking(app.recObj,3);
    app.silence = getaudiodata(app.recObj);
end

% Changes arrangement of the app based on UIFigure width
function updateAppLayout(app, event)
    currentFigureWidth = app.UIFigure.Position(3);
```

```
            if(currentFigureWidth <= app.onePanelWidth)
                % Change to a 2x1 grid
                app.GridLayout.RowHeight = {480, 480};
                app.GridLayout.ColumnWidth = {'1x'};
                app.RightPanel.Layout.Row = 2;
                app.RightPanel.Layout.Column = 1;
            else
                % Change to a 1x2 grid
                app.GridLayout.RowHeight = {'1x'};
                app.GridLayout.ColumnWidth = {220, '1x'};
                app.RightPanel.Layout.Row = 1;
                app.RightPanel.Layout.Column = 2;
            end
        end
    end

    % Component initialization
    methods (Access = private)

        % Create UIFigure and components
        function createComponents(app)

            % Create UIFigure and hide until all components are
            created
            app.UIFigure = uifigure('Visible', 'off');
            app.UIFigure.AutoResizeChildren = 'off';
            app.UIFigure.Position = [100 100 640 480];
            app.UIFigure.Name = 'UI Figure';
            app.UIFigure.SizeChangedFcn =
            createCallbackFcn(app, @updateAppLayout, true);

            % Create GridLayout
            app.GridLayout = uigridlayout(app.UIFigure);
            app.GridLayout.ColumnWidth = {220, '1x'};
```

```
app.GridLayout.RowHeight = {'1x'};
app.GridLayout.ColumnSpacing = 0;
app.GridLayout.RowSpacing = 0;
app.GridLayout.Padding = [0 0 0 0];
app.GridLayout.Scrollable = 'on';

% Create LeftPanel
app.LeftPanel = uipanel(app.GridLayout);
app.LeftPanel.Layout.Row = 1;
app.LeftPanel.Layout.Column = 1;

% Create WordsTextAreaLabel
app.WordsTextAreaLabel = uilabel(app.LeftPanel);
app.WordsTextAreaLabel.HorizontalAlignment = 'right';
app.WordsTextAreaLabel.Position = [22 441 40 22];
app.WordsTextAreaLabel.Text = 'Words';

% Create WordsTextArea
app.WordsTextArea = uitextarea(app.LeftPanel);
app.WordsTextArea.Position = [76 405 138 60];

% Create RecordthewordLabel
app.RecordthewordLabel = uilabel(app.LeftPanel);
app.RecordthewordLabel.Position = [76 372 94 22];
app.RecordthewordLabel.Text = 'Record the word';

% Create SpeakButton
app.SpeakButton = uibutton(app.LeftPanel, 'push');
app.SpeakButton.ButtonPushedFcn =
createCallbackFcn(app, @SpeakButtonPushed, true);
app.SpeakButton.Position = [76 317 100 22];
app.SpeakButton.Text = 'Speak!';

% Create LampLabel
```

```
app.LampLabel = uilabel(app.LeftPanel);
app.LampLabel.HorizontalAlignment = 'right';
app.LampLabel.Position = [71 269 60 22];
app.LampLabel.Text = '';

% Create Lamp
app.Lamp = uilamp(app.LeftPanel);
app.Lamp.Position = [154 267 26 26];

% Create recognizeButton
app.recognizeButton = uibutton(app.LeftPanel, 'push');
app.recognizeButton.ButtonPushedFcn =
createCallbackFcn(app, @recognizeButtonPushed, true);
app.recognizeButton.Position = [74 209 100 22];
app.recognizeButton.Text = 'recognize';

% Create Lamp_2Label
app.Lamp_2Label = uilabel(app.LeftPanel);
app.Lamp_2Label.HorizontalAlignment = 'right';
app.Lamp_2Label.Position = [69 161 60 22];
app.Lamp_2Label.Text = '';

% Create Lamp_2
app.Lamp_2 = uilamp(app.LeftPanel);
app.Lamp_2.Position = [152 159 26 26];

% Create Label
app.Label = uilabel(app.LeftPanel);
app.Label.Position = [69 48 107 81];
app.Label.Text = '';

% Create silenceButton
app.silenceButton = uibutton(app.LeftPanel, 'push');
app.silenceButton.ButtonPushedFcn =
```

```
createCallbackFcn(app, @silenceButtonPushed, true);
app.silenceButton.Position = [76 107 100 22];
app.silenceButton.Text = 'silence';

% Create RightPanel
app.RightPanel = uipanel(app.GridLayout);
app.RightPanel.Layout.Row = 1;
app.RightPanel.Layout.Column = 2;

% Create UIAxes
app.UIAxes = uiaxes(app.RightPanel);
title(app.UIAxes, 'SIgnal')
xlabel(app.UIAxes, 'X')
ylabel(app.UIAxes, 'Y')
app.UIAxes.PlotBoxAspectRatio = [1.94615384615385 1 1];
app.UIAxes.Position = [60 251 300 185];

% Create EditFieldLabel
app.EditFieldLabel = uilabel(app.RightPanel);
app.EditFieldLabel.HorizontalAlignment = 'right';
app.EditFieldLabel.Position = [49 176 95 22];
app.EditFieldLabel.Text = {'запишите слово'; ''};

% Create EditField
app.EditField = uieditfield(app.RightPanel, 'text');
app.EditField.HandleVisibility = 'off';
app.EditField.Position = [159 176 100 22];

% Create recordvoiceButton
app.recordvoiceButton =
uibutton(app.RightPanel, 'push');
app.recordvoiceButton.ButtonPushedFcn =
createCallbackFcn(app, @recordvoiceButtonPushed, true);
app.recordvoiceButton.Position = [60 128 100 22];
```

```
            app.recordvoiceButton.Text = 'record voice';

            % Show the figure after all components are created
            app.UIFigure.Visible = 'on';
        end
    end

    % App creation and deletion
    methods (Access = public)

        % Construct app
        function app = app1version2

            % Create UIFigure and components
            createComponents(app)

            % Register the app with App Designer
            registerApp(app, app.UIFigure)

            if nargout == 0
                clear app
            end
        end

        % Code that executes before app deletion
        function delete(app)

            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```

B. References

- Detecting Outbreaks and Significant Changes in Signals. https://nl.mathworks.com/help/signal/examples/detecting-outbreaks-and-signif html
- An Algorithm to Remove Noise from audio Signal by Noise Subtraction. https://link.springer.com/chapter
- Understanding Support Vector Machine(SVM) algorithm from examples (along with code) https://www.analyticsvidhya.com/blog/2017/09/unders
- Machine Learning for Beginners: An Introduction to Neural Networks https://towardsdatascience.com/machine-learning-for-beginners-an-introd
- Machine Learning Project 1: ANN based Speech Recognition System in MATLAB https://www.youtube.com/watch?v=sH4DMWb-SBM
- Machine Learning in MATLAB https://www.mathworks.com/help/stats/machine-learning-in-matlab.html
- Can you help remove the noise from this audio file? https://www.mathworks.com/matlabcentral/answers/357022-can-you-help-remove-the-noise-fr