|  |
| --- |
| **REPUBLIC OF TURKEY**  **ADANA ALPARSLAN TÜRKEŞ SCIENCE AND TECHNOLOGY UNIVERSITY**  **FACULTY OF ENGINEERING**  **DEPARTMENT OF COMPUTER ENGINEERING**  **ALS DISEASE PREDICTION USING EMG SIGNALS**  **MERVE DENİZ ŞENELEN**  **170101030**  **UNDERGRADUATE STUDENT OF CEN451**  **ADANA 2022** |

****

**REPUBLIC OF TURKEY**

**ADANA ALPARSLAN TÜRKEŞ SCIENCE AND TECHNOLOGY UNIVERSITY**

**FACULTY OF ENGINEERING**

**DEPARTMENT OF COMPUTER ENGINEERING**

**ALS DISEASE PREDICTION USING EMG SIGNALS**

**MERVE DENİZ ŞENELEN**

**170101030**

**UNDERGRADUATE STUDENT OF CEN451**

**SUPERVISOR**

**ASSOC. PROF. DR. YASİN KAYA**

**ADANA 2022**

**ABSTRACT**

**ALS DISEASE PREDICTION USING EMG SIGNALS**

Merve Deniz ŞENELEN

Department of Computer Engineering

Supervisor: Assoc. Prof. Dr. Yasin KAYA

January 2022, 21 pages

This study is about prediction of if a signal is taken from a healthy person or if it comes from a person with ALS disease. EMG(Electromyography), is a medical technique to analyse the electrical response potential of muscles. It is taken with an electromyograph. ALS is an neuromuscular disease. People with ALS have low neurological responses from their muscles, which means they completely their muscle control and muscles themselves ultimately. In medicine, EMG signals are helpful for ALS disease diagnosis. In this study, EMG signals and machine learning is used to predict the likeliness of a signal is coming from an ALS patient. It is seen that with normalization of signals, a healthy person’s EMG signal graphic fluctuates more in time.

**Keywords:** ALS disease diagnosis, EMG, EMG signals, ALS disease prediction

**TABLE OF CONTENTS**

**ABSTRACT……………………………………………………………….………….……….3**

TABLE OF CONTENTS……………………………………………….…………………….4

LIST OF FIGURES………………………..…………………………………………………5

1. INTRODUCTION………………………………………………………………………….6

**1.1. What is ALS?………………………………………………………………………….….7**

**1.2. EMG(Electromyography)……………………………………………………………….8**

2. MATERIALS AND METHODS………………………………………………………..9

**2.1. Tools and Libraries………………………………………………………………………9**

**2.2. Dataset………………………………………………………………………………...…10**

**2.2.1. About Signals………………………………………………………………………….10**

**2.3. Project Structure………………………………………………………………………..12**

**2.3.1.Code Structure……..………………………………………………………………….16**

3. RESULTS AND DISCUSSIONS………………………………………………………17

**3.1. Filtering the Signals…………………………………………………………………….21**

4. CONCLUSIONS…………………………………………………………………….......21

REFERENCES………………………………………………………………………………22

**LIST OF FIGURES**

**Figure 1……………………………………………………………………………………...6**

**Figure 2…………………………………………………………………………………….11**

**Figure 3…………………………………………………………………………………….11**

**Figure 4…………………………………………………………………………………….15**

**Figure 5…………………………………………………………………………………….16**

**Figure 6…………………………………………………………………………………….17**

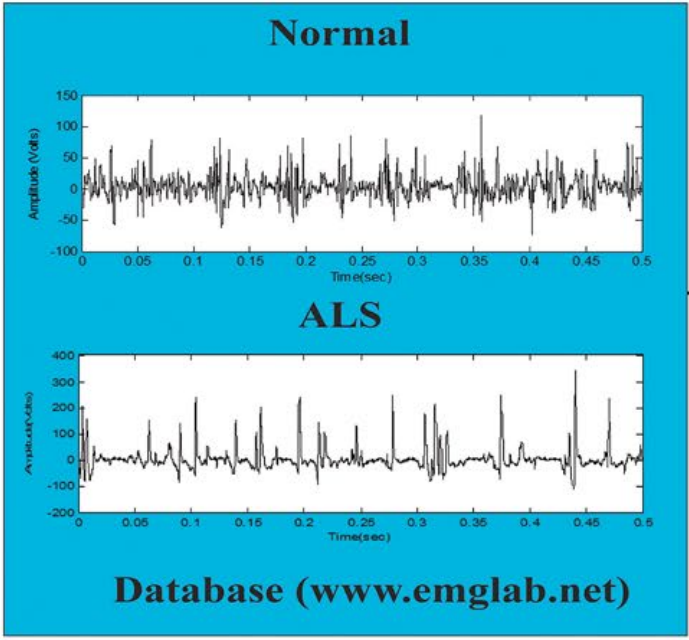
**Figure 7…………………………………………………………………………………….18**

**Figure 8…………………………………………………………………………………….19**

**Figure 9…………………………………………………………………………………….20**

1. **INTRODUCTION**

The aim of this study is to develop a machine learning model that can predict whether randomly chosen EMG signals’ owners are healthy or ALS patients. In other words, the purpose of this poroject is to predict ALS disease using emg signals with LSTM (Long short-term memory). 2 signals from healthy and als people, totally 4 signals are randomly chosen, as it gives us the reasonable prediction results.

More signals can be used by simply adding .bin files into healthy and als folders in the root directory, but it will lead us to take more time for training although it will produce the better performance of prediction. After that, 80% of each of signals is used for training and 20% is used for for testing. What can be known from the visualization (Normalized graph in the Figure 3) is that healthy signals fluctuates much with time where as there are plain part which doesn't change much. We can say if the prediction result is close to 0, they are signals from ALS people and if close to 1 or larger than 1, they come from healthy people.

**Figure 1: Normal EMG signal and ALS patient’s EMG signal.**

Deep learning libraries Keras and TensorFlow, and LSTM(Long Short Term Memory) architecture is used in this study. This model has three layers of networks, which will be discussed further in Section 2, with activation function being tanh. The project was implemented using Python version 3.9.7 .

* 1. **What is ALS?**

Amyotrophic Lateral Sclerosis (ALS) is a nervous system disease that causes muscle weakness that significantly affects physical activity. It is a neurological disorder. The longest-living ALS patient known is the famous scientist Stephen HAWKING. He has been fighting this disease for 48 years. ALS is a type of motor neuron disease that causes nerve cells to deteriorate and die over time.

In most cases, the cause of ALS is unknown. A very few cases are hereditary. People who have ALS have low muscle response. Ultimately, ALS takes control of the muscles needed to speak, move, eat, and breathe. In ALS, the nerve cells that control the muscle gradually die. ALS is incurable and will result in absolute death. ALS is difficult to diagnose early, because the symptoms can be confused with other neurological diseases.

Although there is no definitive cure for the ALS disease today, some symptoms can be treated with early diagnosis and muscle dominance is achieved for a longer period of time. With current treatments, the patient's quality of life is improved. From the onset of the disease, the survival time differs. While 5 percent of the patients live for more than 20 years, the average survival time from the onset of the disease is 4-6 years. However, there are many patients who live 10 years or more.

* 1. **EMG(Electromyography)**

For the diagnosis of ALS, the examination method called EMG (electromyography), MRI (Magnetic resonance imaging), blood and urine tests, lumbar puncture (removal of spinal fluid from the waist) and muscle biopsy are applied.

EMG(Electromyography), or ENMG(Electroneuromyography), is a test which is used to measure the natural electrical activity of muscles. The word itself, consists of shortening of the terms electro (electrical), neuro (nerve), myo (muscle) and graphy (writing). During this test, fine needles are inserted into various muscle groups. These electrodes measure the electrical activity of the muscles during rest and contraction.

Most accurate prediction of muscle disorders using EMG signals is performed using quantitative analysis rather than qualitative techniques, because EMG signals are chaotic and non-stationary in nature. Efficient examination of EMG signals is performed by following the steps such as extraction of features, feature reduction and development of classifcation networks. In this study, neural networks are used for examining EMG signals.

How is an EMG signal generated? Muscle fibers generate electric activity whenever muscles are active. EMG signals are recorded by placing electrodes close to the muscle groups. When the muscle is activated, the length of the muscle decreases and the muscle, skin and electrodes move with respect to one another.

With the type of muscle abnormality that can be seen on the EMG, the doctor can diagnose ALS or consider other diseases that cause similar complaints. Also, if you are diagnosed with ALS, EMG findings can provide information to guide your exercises. Thanks to EMG, it is determined whether the cause of the disease is in the muscles or in the nerves that feed the muscles. EMG is a method used to detect dysfunctions in nerves, to diagnose diseases affecting nerves and muscles, or to determine the severity of damage.

1. **MATERIALS AND METHODS**
   1. **Tools and Libraries**

Keras and TensorFlow was used in this study. Keras is a neural network library and TensorFlow is a library which is used in various tasks in machine learning. As far as other libraries; matplotlib.pyplot helped to do visualization through graphs. Numpy and pandas are the libraries which was used for processing arrays and data in this project.

MATLAB’s EMGlab tool was used to view the signals. EMGlab is an open source tool. Long Short Term Memory model was used for training in keras library, and sklearn.metrics is used for measuring the training results. Sklearn is a shortening for scikit learn. Scikit-Learn is a higher level library that includes implementations of several machine learning algorithms, so one can define a model object in a single line or a few lines of code, then use it to fit a set of points or predict a value. Tensorflow is mainly used for deep learning while Scikit-Learn is used for machine learning.

LSTM is an artificial recurrent neural network model which is used in deep learning. Recurrent neural networks (RNN) are a class of artificial neural networks in which the connections between nodes form a directed loop. They are a class of neural networks that allow previous outputs to be used as inputs while having hidden states. RNN models are mostly used in natural language processing and speech recognition. Artificial Neural Networks consist of many cells and these cells work simultaneously to perform complex tasks.

LSTM networks are well suited for classifying, processing and making predictions based on time series data because there can be delays of unknown duration between significant events in a time series. Visual Studio Code was used as IDE.

* 1. **Dataset**

Dataset which was used in this study are EMG signals of patients and healthy people. The muscles which the EMG signals was taken from is Biceps branchii (longhead). All of the EMG signals are in binary format. Each of the signal file had a header(.hea) and a signal(.bin) file.

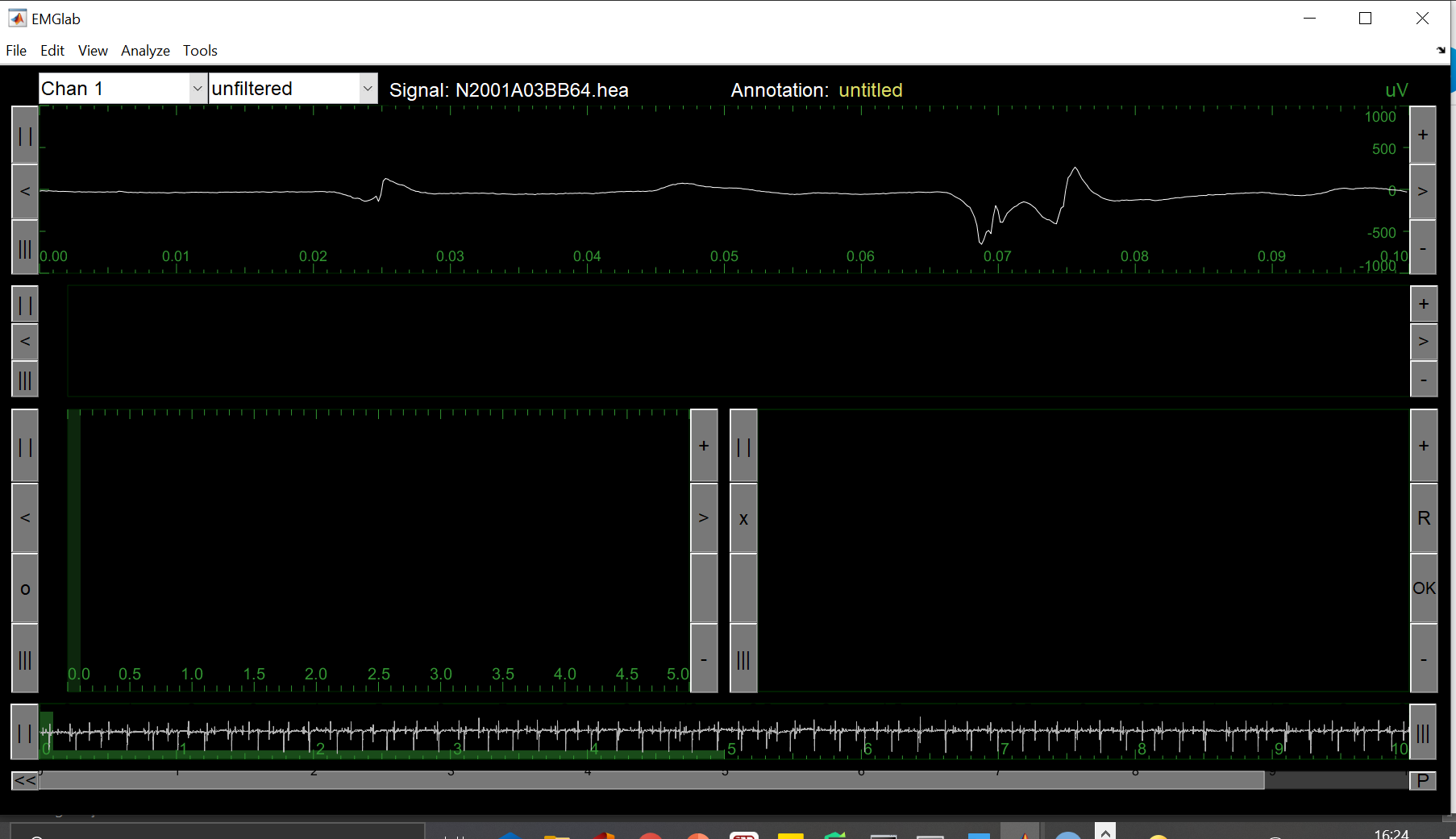
.bin file has the signal itself in binary format and .hea file is the header file which had the name of the contributor, the date EMG was taken, gender of the person, age of the person, muscle name which was the EMG taken from, if the person is diagnosed with ALS or not, if yes, how much time he/she went through in the disease till then, the type of ALS the person has, place EMG was taken, etc.

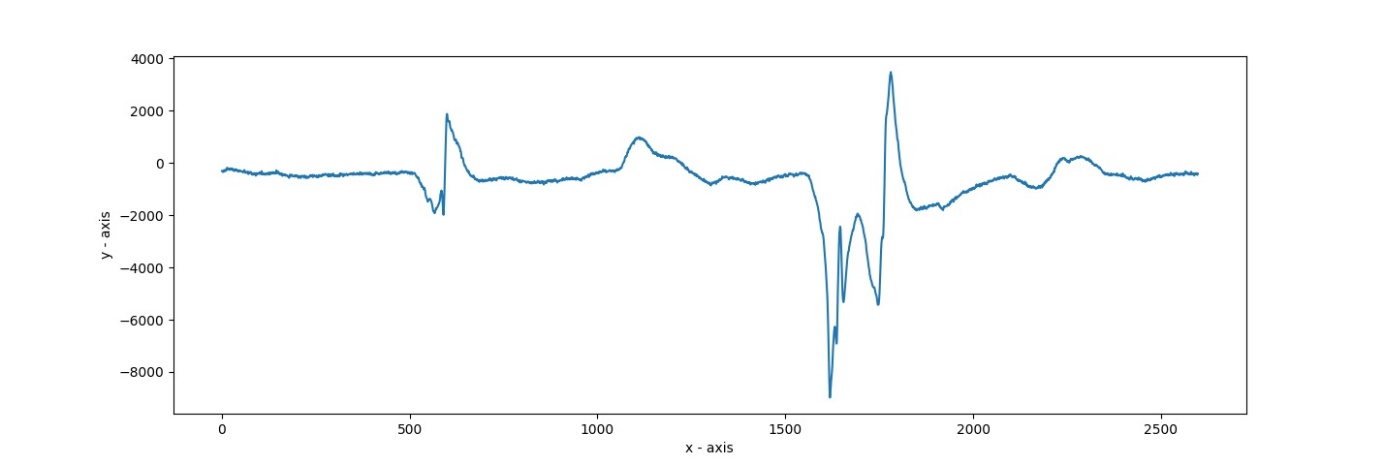
From the code, I only use and read the .bin files since they actually contain the signals. It was not necessary to use the header files for this project. The place which the EMG’s taken from is Dept. Clinical Neurophysiology, Rigshospital, Copenhagen.

* + 1. **About Signals**

EMG signals are chaotic by character, which means they are complex to classify. They are in different lengths, different fluctuations from beginning to the end, etc. In this study, EMG signal processing was considered by using EMGlab.

EMGlab is a MATLAB tool to view signals. MATLAB has features to help for analysing signals using built-in apps for visualizing and preprocessing signals in time, frequency, and time-frequency domains to detect patterns and trends without having to manually write code. For this study, MATLAB’s EMGlab tool, which was an open source, was used to view the signals. In Figure 1, a signal that I viewed in the EMGlab tool can be seen.



 **Figure 2: Signal’s visualization from EMGlab**

**Figure 3: My first graph. They are the matching with the one in EMGlab.**

In the Figure 2, a graph that I was able to create with the help of python library matplotlib.pyplot, I could see that the signal drawing was accurate from my side.

* 1. **Project Structure**

3 layers of networks are used in the LSTM model. Activation function which was used is tanh for these three layers. The important point here is, the data(signals) is rescaled to range [-1, 1] which is suitable for tanh activation function:

X = np.array(X / (max\_val / 2) - 1)

As I am using tanh function in LSTM model, I scaled all values to [-1, 1]. If the values are scaled like that with tanh function, it gives the best performance. Let us suppose, we have series of numbers, like [100, 50, 25, 1]. Here, max val is 100, min val is 1. If we scale these values to [-1, 1], then, we get [1, 0, 0.5, -1].

Mean squred error was used as loss function. The loss function measures the distance between the actual and predicted value of the target. Based on the measured value, it is concluded how close the model is to the desired value. The most commonly used function is the mean squared error.

As optimization function, Adam is used. Optimization is the problem of finding a set of inputs to an objective function that results in a maximum or minimum function evaluation. In other words, it is a procedure for finding the input parameters or arguments to a function that result in the minimum or maximum output of the function.

The most common type of optimization problems encountered in machine learning are **continuous function optimization,** where the input arguments to the function are real-valued numeric values, like floating point values. The output from the function is also a real-valued evaluation of the input values.

Optimization is the challenging problem that underlies many machine learning algorithms like what is tried to achieve in this study; training artificial neural networks. There are perhaps hundreds of popular optimization algorithms, and perhaps tens of algorithms to choose from in popular scientific code libraries. The Adaptive Movement Estimation algorithm, or Adam for short, is said to be the best among adaptative optimizers in most of the cases.

Therefore, it was chosen as most suitable for this project. This algorithm simply estimates moments and uses them to optimize a function.

100 epochs was used in this project, as it was the most recommended, considering every factor, including 4 signals are used at the first time. But we can change the epochs considering the PC’s processing power. Epoch is the number of how many times all of the data goes through neural network in one turn. In other words how many times it goes from training in one turn.

While the model is being trained, not all of the data are included in the training at the same time. They take part in education in a certain number of parts. The first piece is trained, the performance of the model is tested, and the weights are updated according to the success with backpropagation. Then the model is retrained with the new training set and the weights are updated again. This process is repeated at each training step to try to calculate the most appropriate weight values ​​for the model. Each of these training steps is called an epoch.

Since the most suitable weight values ​​to solve the problem in deep learning are calculated step by step, the performance will be low in the first epochs, and the performance will increase as the number of epochs increases. However, after a certain step, the learning situation of the model will decrease considerably.

The model usually takes a long time to train; There are models that take days or months to train. This is common in deep learning. For this reason, it is tried to shorten the training process as much as possible with other hyper parameters.

The size of the epoch number also varies according to the problem type. For example, the number of epochs in Recurrent Neural Networks, like the one in this project, with a learned

pattern should be kept larger than other models. As the number of epochs increases, the performance of the model increases significantly. Training can be terminated at these points,

as performance will increase in very small units after a certain epoch. Amount of dataset is also important in defining epoch. As of dataset size, in the first tries, 4 signals are used for training, this will be further discussed in below.

Why tanh function is used in this project instead of ReLu? Isn't relu expected to perform better ? In general, no. RELU will perform better on many problems but not all problems.

Also, I found out that tanh function performs better then sigmoid function as well.

Hyperbolic Tangent or in short ‘tanh’ is very similar to the sigmoid function. It is centered at zero and has a range between -1 and +1. A con of tanh function is that it is computationally expensive.

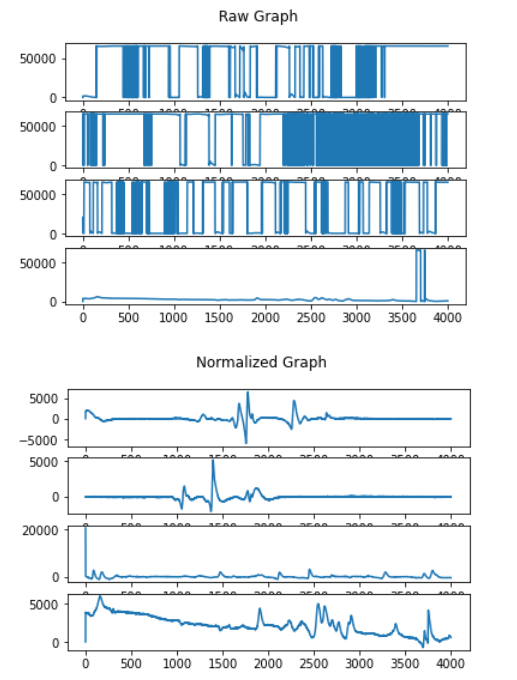
Basically, an activation function is used to map the input to the output. This activation function helps a neural network to learn complex relationships and patterns in data.

I had to define a step size for data, to do the training. Step value keeps the length of each input data that goes into training model. I defined this value 40. How did I define the step size? If the solution is changing rapidly, the step size should be chosen small. Inversely, if the solution is changing slowly, we should choose bigger step size. The existence and uniqueness of the solution of the problem must be considered in the step size selection.

I used 4 signals, 2 ALS and 2 healty, but I can simply put more signals in the root directory to get use of them. In order to achieve more accuracy in a short time, and considering the PC power and remaining time, this was what I decided to use as step size, signal number and train/test ratio. I took %80 percent of each signal for training, and %20 percent of each signal for testing.

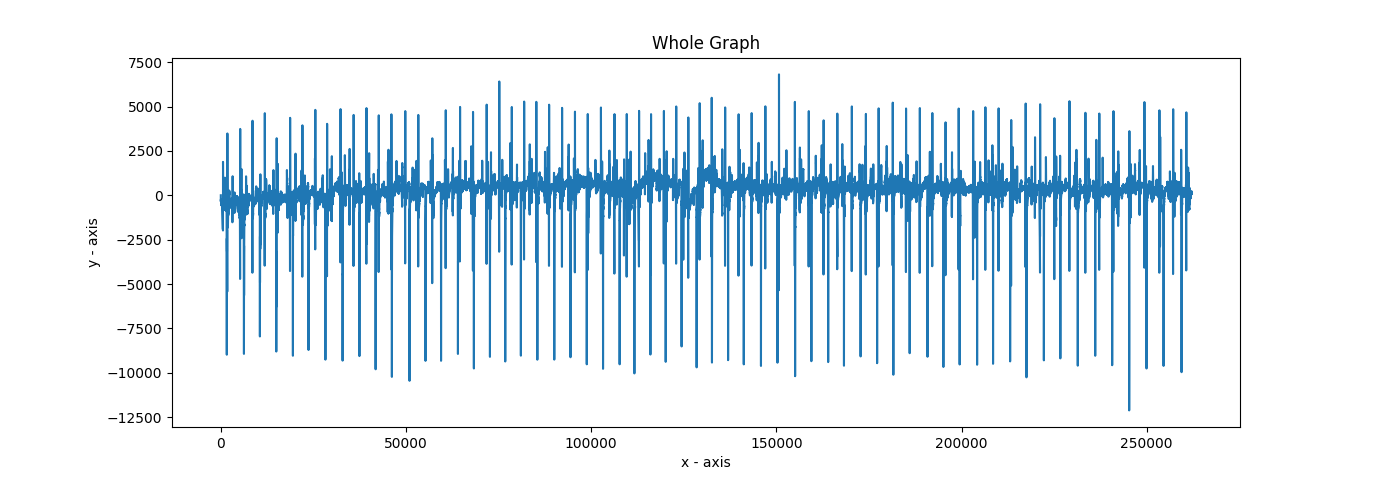
To demonstrate that I am taking correct signals from healthy and als folders and help to analyze the signals visually, I draw eight graphs, namely, 'Raw graph' and 'Normalized graph'. These can be seen at Figure 4 in the next page.

Because first of all, I looked into signal, using EMGlab. EMGlab is an EMG signal processing/viewing tool, for usage in MATLAB. In this study, EMGlab only helped me to view the signal. For Raw graph, I used raw data from .bin file whereas normalized data are used for Normalized graph. The normalized values make graph which is exactly same to one which I could see in Matlab EMGLab earlier. This can be seen in the Figure 2 and Figure 3, on page 11.

What we can know from the visualization (Normalized graph below) is that healthy signals fluctuates more with time.

**Figure 4: Four signals’ visualisations as raw graphs and normalized graphs.**

As can be seen in the Figure 4, first two of each graphs are coming from two ALS patients, second two of each graphs are coming from two healthy people’s muscles. The important point in these graphs to examine is that in first two signals, which belong to an ALS patient, are not fluctuating as much as the ones that belong to a healthy person’s EMG signal, which are the second pair of signals here. Remember that ALS patients’ muscle have low muscle electro potential, this can be reread in the part “2.1.What is ALS?” on page 7.

The whole of the signal in the Figure 2 and Figure 3 can be seen in the Figure 5 below.

**Figure 5: Whole graph of the previous signal at Figure 2 and 3.**

* + 1. **Code Structure**

To build a univariate sequence for my ML model, I defined two functions. “buid\_sequence” and “split\_series” are that global function for deep learning model. “ split\_series” function is intended to deal with real-time signals, but as I am using signals in .bin file, I don't use this function here.

Step value is length of each input data. n\_steps=4000 is recommended if the performance of computer is good enough for training my model. Train test split percent is 80/20. I rescaled data to range [-1, 1] which is suitable to tanh activation function. It seems very important in my project because tanh function is said to best fit for data in range of [-1, 1].

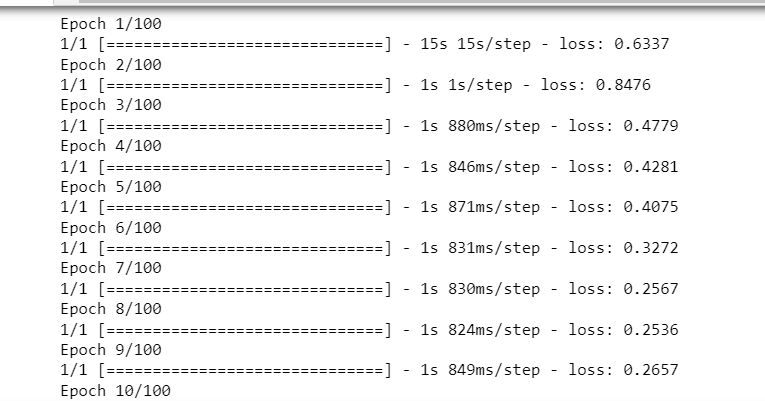
I reshaped my dataset to make it available for my LSTM model:

n\_features = 1

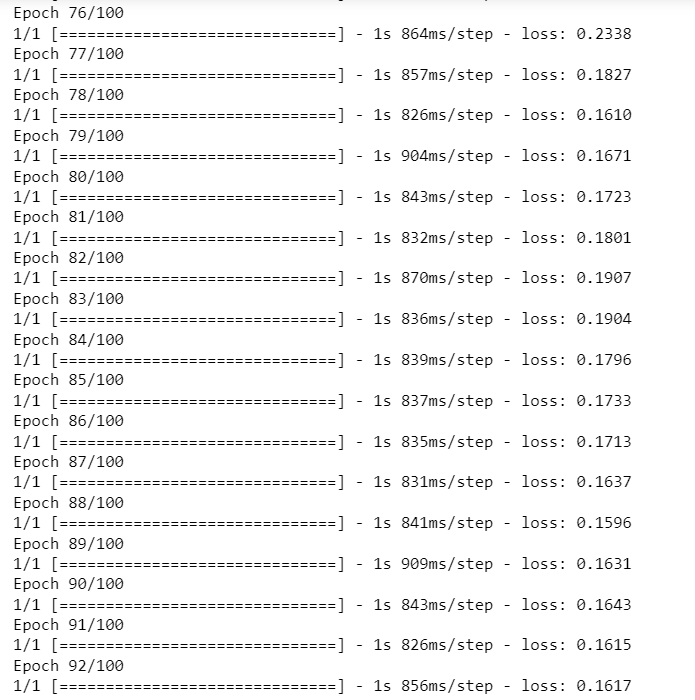
X = X.reshape(X.shape[0], X.shape[1], n\_features)

While fitting the model with training dataset, we can change the number of epochs. epochs=100 is recommended, but it should be adjusted according to the amount of dataset and performance of computer.

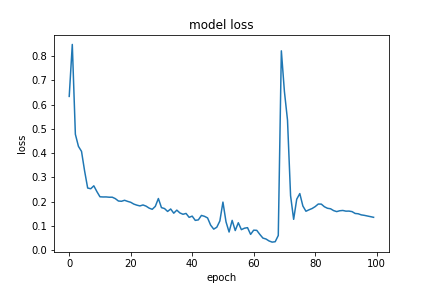
1. **RESULTS AND DISCUSSIONS**

Training model loss results can be seen in the next two figures(Figures 6,7). Next to current epoch, the loss can be seen. Also, overall model loss can be seen in the model loss graph in Figure 8.

**Figure 6: Loss values for each epoch in between first and ninth one.**



**Figure 7: Loss values for epochs from 76th epoch to 92nd epoch.**

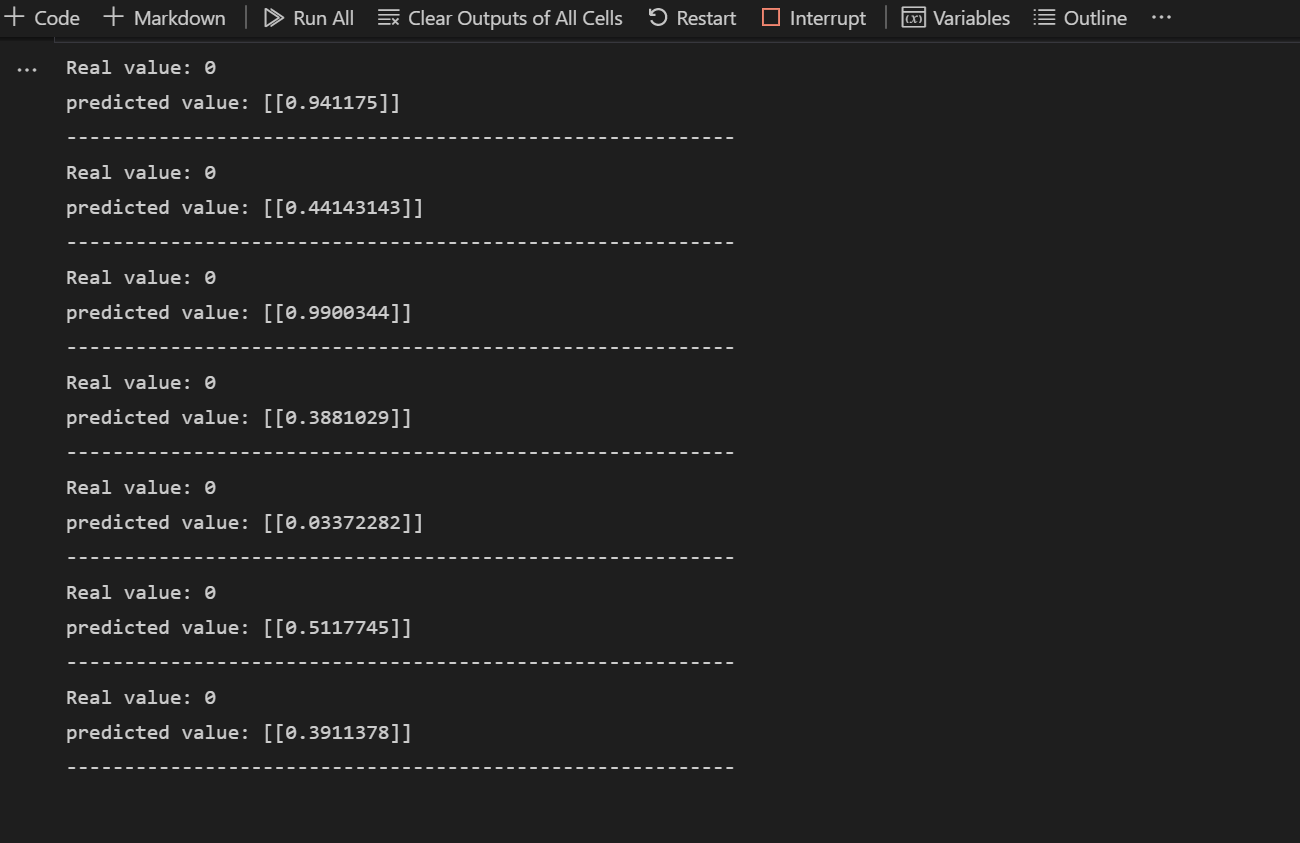


**Figure 8: Model loss graph.**

Loss is a number indicating how bad the model's prediction was on a single example. If the model's prediction is perfect, the loss is zero; otherwise, the loss is greater. The lower the loss, the better a model. The Loss Function is one of the important components of Neural Networks. Loss is said to be the prediction error of a neural network. And the method to calculate the loss is called Loss Function. In this study, mean squared error was used as loss function.

Having a low accuracy but a high loss would mean that the model makes big errors in most of the data. But, if both loss and accuracy are low, it means the model makes small errors in most of the data. However, if they're both high, it makes big errors in some of the data.

As can be seen in the Figure 8, if the predicted value is close to zero more than it’s close to 1, it means the model predicted with low loss, which means it was succesful. Since we can say two values out of seven is closer to 1, accuracy is 5/7, which means %71. We can say if the prediction result is close to 0, they are signals from ALS people and if close to 1 or larger than 1, they come from healthy people.

****

**Figure 9: Prediction results.**

More signals can be used by simply adding .bin files into healthy and als folders in the root directory, but it will lead us to take more time for training, although it will produce better performance of prediction. I used 4 signals, but I can simply put more signals in the root directory to get use of them. In order to achieve more accuracy in a short time, and considering the PC power and remaining time, this was what I decided to use as step size, signal number and train/test ratio.

**3.1 Filtering the Signals**

For filtering the EMG signals, I wrote two files of Python code under the name of EMGfilters.py and handleSignals.py . When I actually imported and used the filters that I created, I came to see that the filters did not effect the results. I came to know that the reason was the EMG signals already had filters when they were being taken in the first place. Every header(.hea) file of every signal has a low pass and a high pass filter in them.

1. **CONCLUSIONS**

As conclusion, considering the defined variable values like epoch number and step size, %71 accuracy is reasonable. We can add more signals to train next time with reconsidering PC processing power and time remaining. It is seen that for a little data like 4 signals, 100 epochs and 40 as step size was successful, considering time cost. For higher performance, training can be considered to be done on a PC with GPU set up. And for that to happen, also the implementation should be set with tensorflow-gpu. Also, more EMG signals should be put into directory and trained for the next time. That way, the accuracy would be much higher.

ALS is a progressive degradation and damage of motor neuron cells in the neuromuscular system, which afects the muscle movement. At present, the treatment of ALS is still a challenging assignment for the clinicians and researchers. Electromyography is a clinical investigation method to identify the abnormalities present in the neuromuscular system. The manual or visual analysis of EMG signals is highly complex owing to the non-stationary and chaotic nature of electromyograms. Therefore, the development of automatic diagnostic tools is required to predict and identify the abnormalities.

**REFERENCES**

*Scientific Paper:*

*“Automated diagnosis of amyotrophic lateral sclerosis using electromyograms and firefly algorithm based neural networks with fractional position update”, A. Bakiya, K. Kamalanand, V. Rajinikanth*

*WEB Sources:*

[*https://www.drerhanozel.com/tr/saglik-rehberi/90-amyotrofik-lateral-skleroz-als*](https://www.drerhanozel.com/tr/saglik-rehberi/90-amyotrofik-lateral-skleroz-als)

[*https://www.memorial.com.tr/tani-ve-testler/elektromiyografi-emg-nedir*](https://www.memorial.com.tr/tani-ve-testler/elektromiyografi-emg-nedir)

[*https://www.acibadem.com.tr/ilgi-alani/als-amyotrofik-lateral-skleroz/#genel-tanitim*](https://www.acibadem.com.tr/ilgi-alani/als-amyotrofik-lateral-skleroz/#genel-tanitim)

[*http://colinraffel.com/wiki/neural\_network\_hyperparameters*](http://colinraffel.com/wiki/neural_network_hyperparameters) *,*

[*Jason Brownlee*](https://machinelearningmastery.com/author/jasonb/)*, 2016, How to Grid Search Hyperparameters for Deep Learning Models in Python With Keras,*[*https://machinelearningmastery.com/grid-search-hyperparameters-deep-learning-models-python-keras/,*](https://machinelearningmastery.com/grid-search-hyperparameters-deep-learning-models-python-keras/,)

*https://towardsdatascience.com/a-quick-guide-to-activation-functions-in-deep-learning-4042e7addd5b*