TED University Fall Term

2021-2022

Senior Design Project II – CMPE 492

Final Report

Beray AKAR- 43612955016

Barış PINAR -53590733858

Öykü BİLİK -14074215802

Atakan MAVZER- 26387632580

Supervised by Tolga Kurtuluş Çapın

Project Website
https://barispinara.wixsite.com/neurolysis

Content

1.Introduction:

EMG is a non-invasive procedure involving the detection, recording and interpretation of the electrical activity of groups of muscles at rest (static) and during activity (dynamic). EMG devices are taking electrical signals from the athlete's muscle and transmitting these signals to the system.Clinical EMG derived muscle data provides valuable insights related to the athletic performance injury risks and training efficiency. The main goal of the project is to implement machine learning models to improve surface ElectroMyoGraphy (sEMG) applications on muscle fatigue analysis of athletes.

In order to perform the fatigue analysis, we firstly need to do modification and calculation features on raw EMG data for implementing the Machine Learning model. The Signal Processing field is taking raw EMG data from Amazon AWS bucket system and it starts to eliminate unnecessary data(noise) or NaN(packet loss) values from EMG data. In the beginning of the project, we were using some contraction detector algorithms in the system, however the company developed a label system for the training system. Basically, physiotherapists who are customers are given a time range of contraction zones. Therefore, we also implemented a new algorithm for new labeled data. Another important part in the Signal Processing is feature calculations. There are 3 main features which we are using in the project and these features are MNF , RMS and Bandlimit MNF. These features mainly calculate the changes between frequency and PowerEMGSpectrum values. We will mention these features in the reports more detaily. After extracting the feature values from EMG data, The Signal Processing transfers this created new DataFrame into S3 Bucket system.

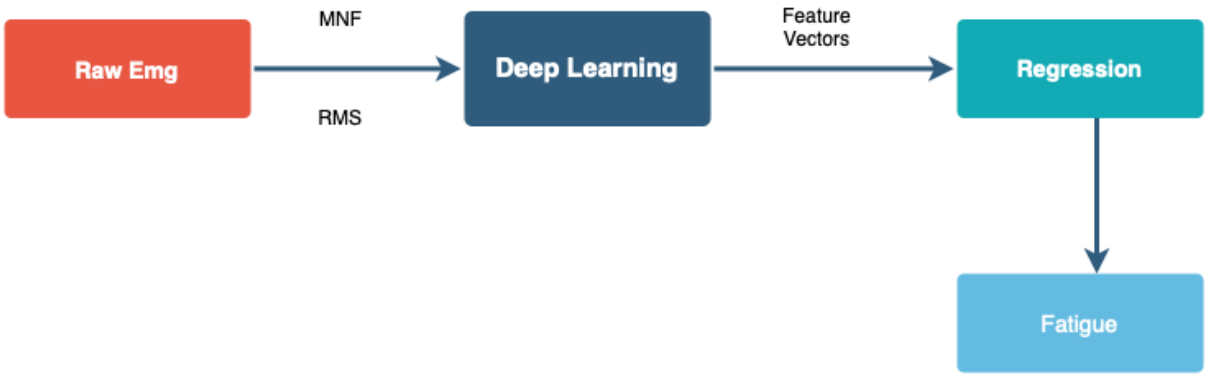
After the signal processing part parses the signals that are coming from the EMG device, datasets on which the model can be applied for the Machine Learning models are created.Three different machine learning models; LSTM(Long Short Term Memory), SVM(Support Vector Machine) and Linear Regression, were applied on the datasets of different dimension.Models are first applied and trained on small-sized datasets.After the models were trained on small-sized datasets, they were retrained on the overlapped versions of these datasets with frame sizes of 50 and 61.In the same way, muscle groups were selected

by applying certain combinations while the models were fit.After testing different muscle groups on the model, the most suitable group was selected and the model was trained on that group, and a linear regression model was applied on the results obtained.The first of the models we used, the deep learning model LSTM, was used to perform feature extraction in the project.After the feature vectors were created by training the LSTM model, the SVM model was created.Here, the SVM was taken as a base for the deep learning model and the accuracy was tried to be brought closer to it.The feature vectors obtained from the LSTM model were inserted into the linear regression model to tune the fatigue coefficients.After the appropriate Machine Learning Models are applied, the fatigue analysis results will be shown in a dashboard.

The dashboard displays the physiology and athletic state of the players with both live and post-event evaluations. In our project Web GUI is created for users. Users can choose an athlete and display their analysis result. To give the most amount of information in a chart to monitor the player, we chose two different graphs. First one is a combination of a line chart with a bar chart. Line chart gives information about the status of the player through time and bar chart gives information about the possible fatigue increasing dates, such as match day. Second chart is a calendar heatmap. This chart's main goal is to give an idea about the player in a simplistic but in an effective way. In our frontend, two of these charts can be found. Reason for two charts is that the second chart is an average of the whole team/users players. This will further enhance the monitoring process, since the second chart can give a baseline for the selected player.

2.Final Design Details:

2.1 Machine Learning Part

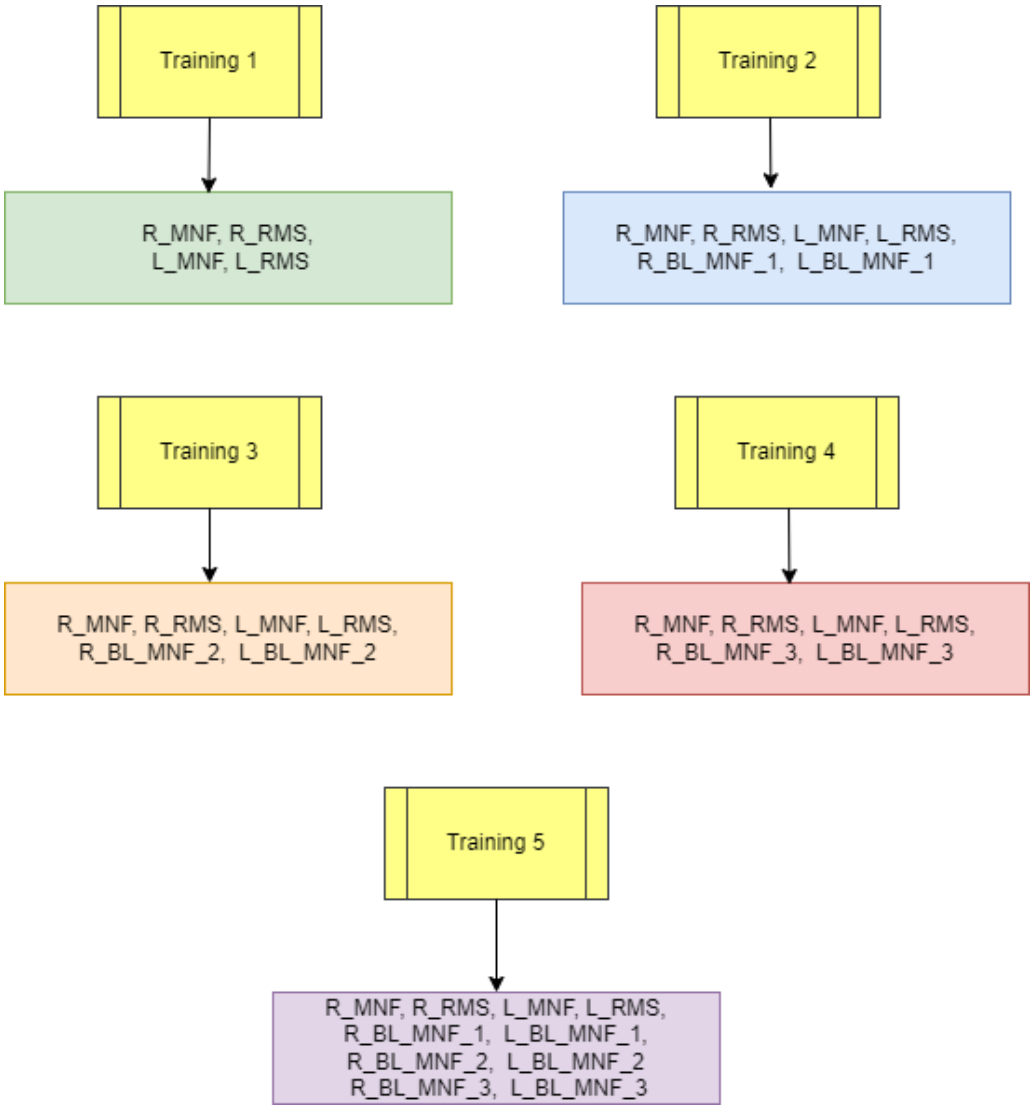


Thanks to the EMG device, raw EMG data is collected from the athletes.From this data, 2 features, RMS(power) and MNF(conductivity), are extracted. Root mean square (RMS) and mean frequency (MNF) are indicators of muscle fatigue.The RMS(root mean square) value refers to the size of the waveform over time. According to studies, it has been found that there is an inverse correlation between power and conductivity in a static motion. However, the main goal of the project is to analyze cumulative fatigue. Therefore, there is a need to utilize these features for developing our model.

2.1.1 Selection of the band

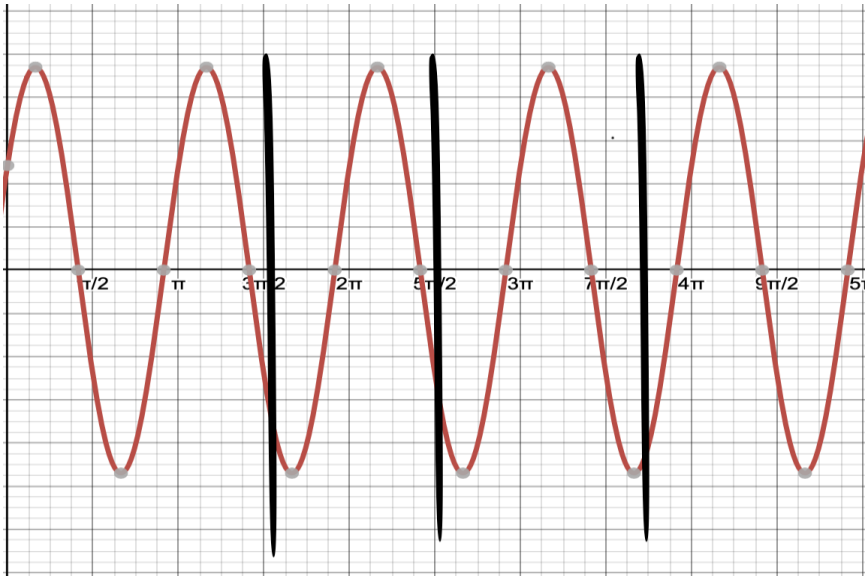
At the beginning of the design, dataset consist of R_MNF (Mean Frequency of right muscle), L_MNF (Mean Frequency of left muscle), R_RMS (Root mean square of right muscle), L_RMS (Root Mean Square of left muscle) Application was made to develop a model of RMS and MNF features about right and left muscles. As a result of the researches, it was observed that the right and left muscles should be synchronized with each other in terms of RMS and MNF values and our models just used these features to predict fatigue according to match day. In order to make the model more inclusive, the features that change according to the bandwidths were also included in the model. According to this approach, implementation of bandwidths is applied. The bandwidth values have been moved over the MNF feature. This process can be called feature selection to develop a more successful model. It is actually aimed to reach the desired accuracy value with the training obtained according to the different selected features. In the shown below figure, there were 5 training approaches to find best features for the model. Training 1 represents a training using without bandwidths features. Training 2 includes R_BL_MNF1, L_BL_MNF1 which means bandwidth 1 values. Training 3 contains R_BL_MNF2, L_BL_MNF2 which means bandwidth 2 values. Training 4 includes R_BL_MNF3, L_BL_MNF3 which means bandwidth 3 values. Last training version which

consists of all of the features (without bandwidths ,bandwidth 1, bandwidth 2, bandwidth 3 features)

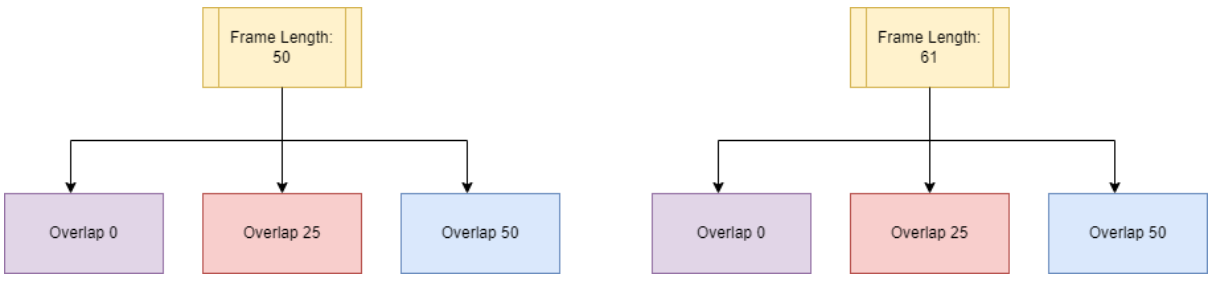


The model's work on different features has also enabled the change of the input shape attribute.

The conductivity of the muscle is assessed by dividing sub-bands.The values we extracted such as band_1 and band_2 from these sub-bands are extracted.While creating the bands, it was tried to find the most suitable band value by tuning with the band intervals and frequency.After the created bands were fitted one by one, the most suitable band was selected as band_1.



On the other hand, trying different lengths of existing features in the existing feature selection approach is another approach implemented in the project. Frame size and overlap parameters that are applied in signal processing which is crucial for developing model.The length of the existing features on the signal side has been designed to be 61 and 50. It was aimed to generalize the model and increase the accuracy value according to these two different length variables which means frame size was 61 and 50 in different experiments. In the first approach, the length of the feature obtained in feature extraction is divided into 50 arrays.In the second approach, the lengths are divided into 61 arrays. At the same time, these two experiments were detailed with different overlap approaches in themselves. The overlap being 0, 25 and 50 has been implemented in these two experiments. In the below figure related experiments are shown.



In these two main experiments, the above-mentioned feature selection approach (taking 5 different trainings) was applied. It means that the frame length 50 and overlap 0 experiment is applied for 5 different training approaches as without bandwidth, bandwidth 1, bandwidth 2, bandwidth 3 and finally for all features.

Experimenting with different overlap values as 0, 25 and 50 plays a critical role in the development of the ML model. The increasing number of overlaps is a factor affecting the rate of entering the details of the data. For example, if we assume that the frame size is 61 and the

overlap value is 50; it means that we put the values in the partition of the 61 array of each array into more training numbers. There is no doubt that; if the numbers of training are increasing which means it has an effect on the issues about overfitting and underfitting problems. Increasing numbers of training the related model leads to the problem that increases the memorization of the model after a certain point, and in this case, it is called underfitting. To avoid the problem about underfitting, the dataset should be shuffled before the training starts and the dataset is split into parts. It is aimed to find the optimal solution of this problem, which is experienced in large data groups, with the overlap approach. Therefore, Model design with multiple overlap sizes and frame lengths has been achieved to balance the training results and reduce the problems mentioned above. Experimenting the model over LSTM and Bidirectional LSTM in this way has approached the correct fatigue analysis.

2.1.2 Deep Learning Model

A deep learning model was trained to classify whether the training was pre-match or post-match.An accuracy value was obtained as a result of the trained LSTM. In order to obtain a suitable accuracy value, band intervals were tuned and different frequency values were given. Before the feature vector is passed to the regression, obtaining the accuracy value in the model runs over a certain epoch and loss function. It is aimed to predict the working success of the model and the match day correctly. After applying LSTM and Bidirectional LSTM with related fatigue values that taken by giving the theoretical fatigue values calculated by a function added to the fatigue value, which determines the fatigue decay of how many matches they played in 24 days, and how many minutes they played in the match, to the regression model and this mathematical approach is used in fatigue prediction.

A feature vector was obtained by training the LSTM and Bidirectional LSTM models.The resulting feature vector can give information about whether the training is done pre-match or post-match. Since feature vectors are the equivalent of vectors of explanatory variables that are used in statistical procedures such as linear regression. We applied these vector formats to predict related match days. It was an effective approach for our classification problem. The feature vector obtained by taking all training (timestamp values) of 14 subjects, X_train and the above-mentioned theoretical fatigue values, is inserted into linear regression.

To explain the process in more detail, we use the knowledge a model has gathered from training on a specific task to solve a different but related task.The model can use what it learned from the prior task to help it learn the new one faster.This model is named as Transfer

Learning in Data Science.Here our prior tasks can be thought of as classifying pre-match and post-match.After classifying the pre-match and post-match values, a feature vector is created with the outputs.Fatigue size has been tried to be estimated with these feature vectors.

2.1.3 Linear Regression

A regression model is trained by taking the outputs of the Transfer Learning model. As a result of the applied regression model, a fatigue value is obtained. In the Linear Regression Model, the extent to which one or more variables affect one or more variables is examined.The regression model not only shows the functional form of the relationship between two or more variables, but also provides predictions about the other when the value of one of the variables is known.If there is a relationship between the variables, the degree of the relationship is expressed as a mathematical function.This function is called the regression function.Here, it is aimed to analyze how much fatigue values are affected by the feature vector. The reason for taking exponential is a step made to approximate the value.Since this is an area where lactate tests should be done normally, it is tried to approach the most accurate fatigue value as much as possible with this approach.
Linear Regression equation is expressed as:

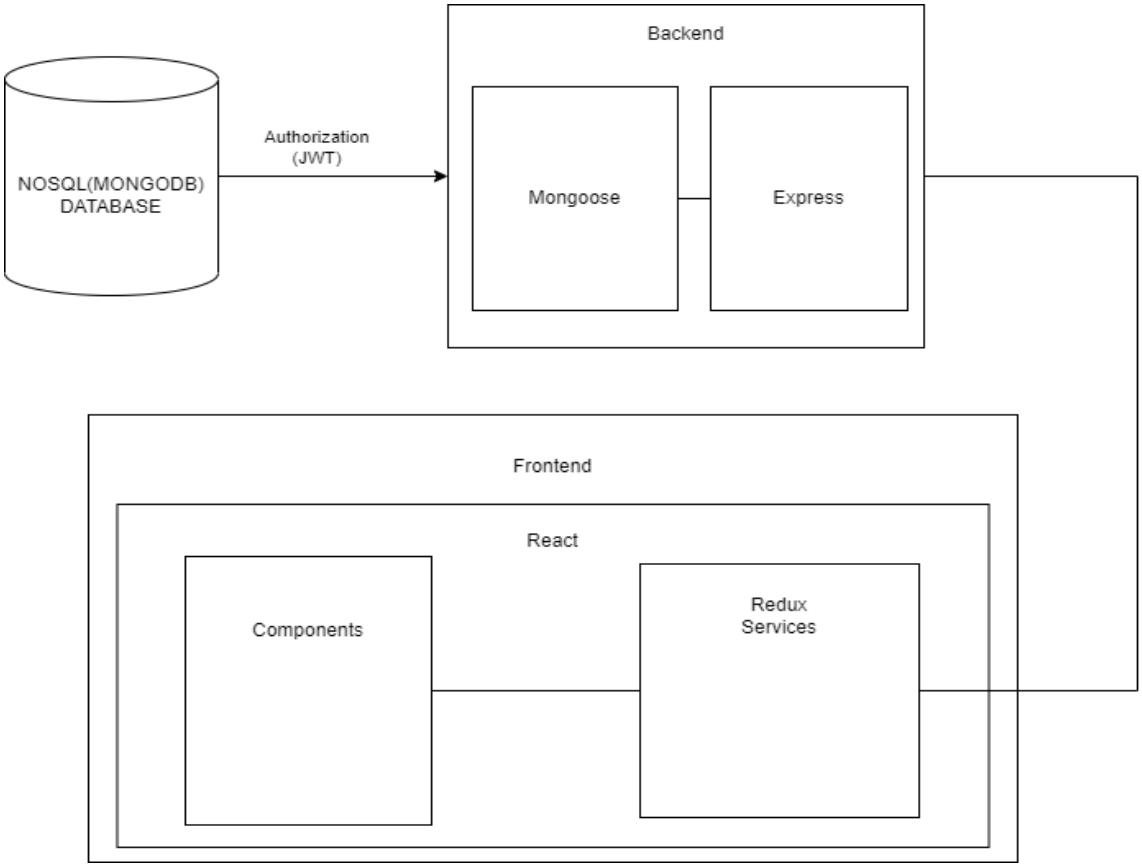
$$y = a \exp\left(\frac{e^{x-b}}{c} \right)$$

a: initial value
x:the day we are in
b:how many days before the training
c:a value that can be scaled by frequency

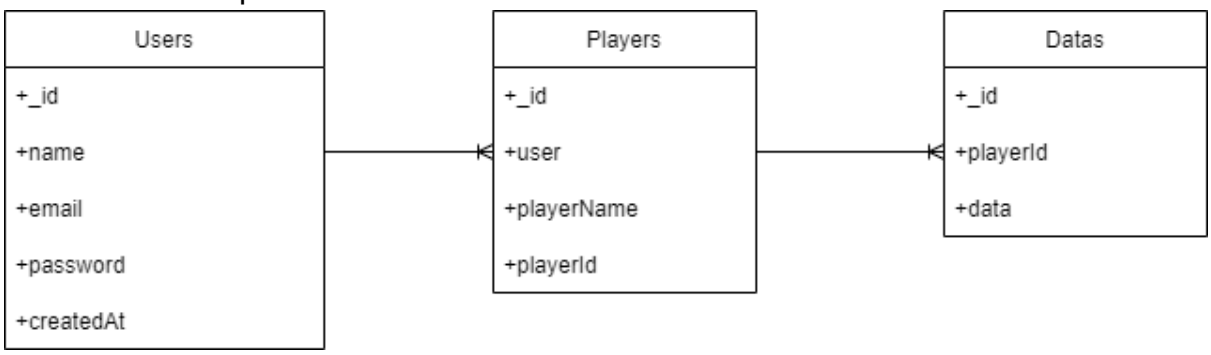
Since we have used Linear Regression , we have used the ln() function for linearization.We are trying to test the accuracy of the outputs by applying linear regression to the function outputs using the feature vector outputs obtained by training the deep learning model.As a result, the accuracy of the values obtained is estimated by linearization of an unknown nonlinear function. MSE (Mean Square Error) is used to determine the performance of the model. Actually, our goal is tuning this formula as much as possible. Feature vector helps for linear regression process to make closer with real fatigue approximation.

2.2 Application Part

2.2.1 Architecture



- **Database:** NoSql database consists of 3 documents.



  - Every user can have many players bound to them with user id relationships. Users also have a name, email and password field.
  - Players document consists of files that have their own _id, user which is _id field in Users document. PlayerId is the id that Neurocess company uses to identify players. We also used this ID to identify which data players have. Every player can have multiple datas

- Datas document has playerId and data. playerId is a key from players. data field is a JSON object which is an array of array of 2-tuples such as [[“03/03/2022”,50],[“04/03/2022”,60]]

- **Backend:**
  - *Routes*:
    - /api/users:
      - POST “/”
      - POST “/login”
      - GET “/data”
    - /api/players:
      - POST “/set”, requires authorization
      - GET “/get”, requires authorization
      - POST “/setPublic”
      - POST “/getPublic”
    - /api/data:
      - POST “/”, requires authorization
      - GET “/”, requires authorization
      - POST “/get”
      - POST “/getPlayer”
      - POST “/set”

  - *Controllers*:
    - Controllers are the final layer of the backend which are the actions that Routes take.

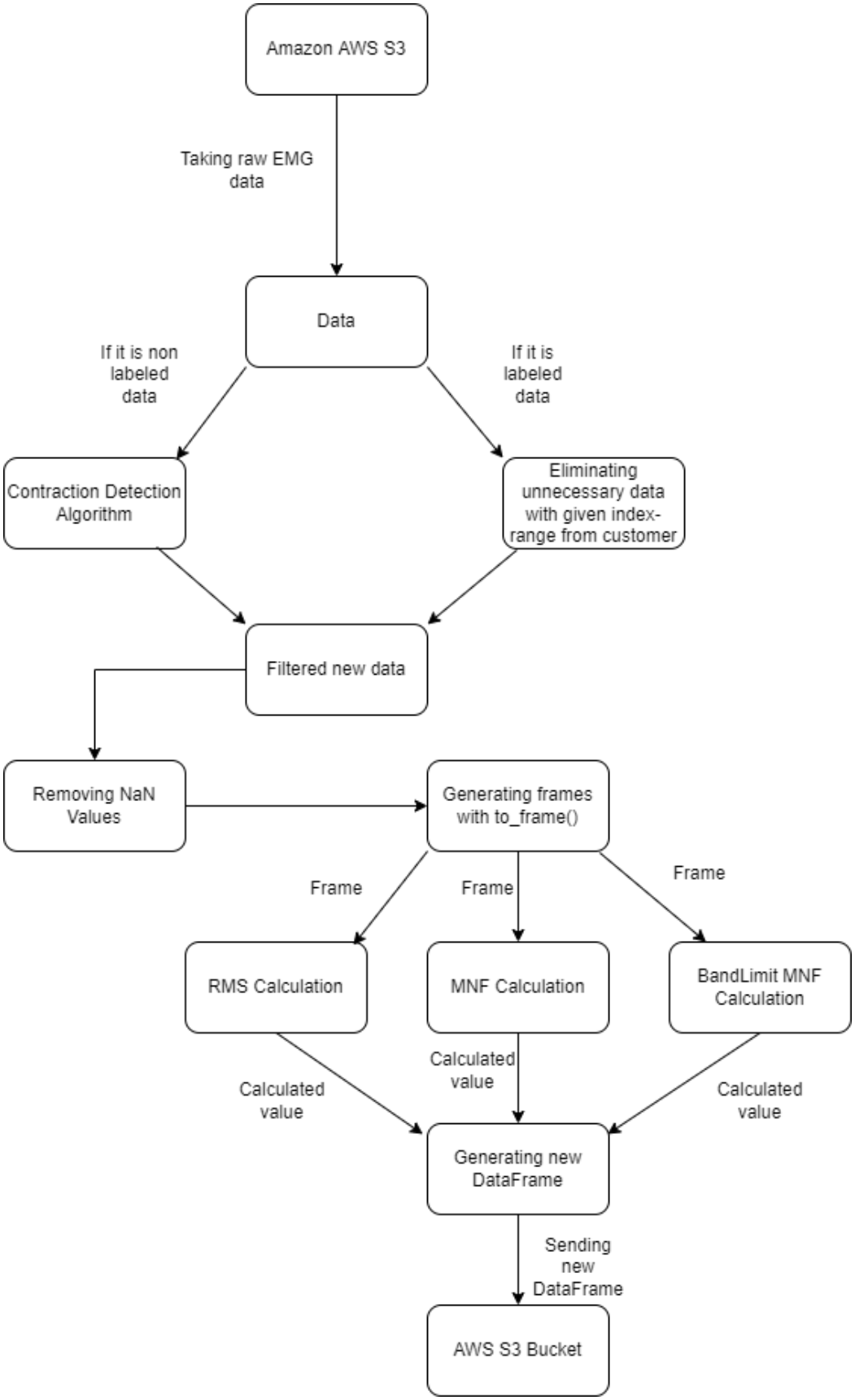
  - *Models (Schema):* Models control the file type that will be inserted and queried by the backend to MongoDB database.
    - userModel:
      - Requires name, email, password
    - playerModel:
      - Requires userId, playerId, playerName
    - dataModel:
      - Requires playerId, data

- **Frontend:**
  - *Login Page:*
    - Login Page is created with a component library called Material UI.
      - Features:
        - Redirect to Dashboard.
        - Alert when wrong form information.
  - *Dashboard Page:*
    - Dashboard Page is created with an admin template and bootstrap called Core UI.
      - Features:
        - Selecting different players.

        - Monitoring line and heatmap graphs for selected players.
        - Comparing average of all of the users players with the selected player through heatmap.
        - Logout
- *State:*
    - All of the data that required by the application is stored and accessed through a state container called Redux-Toolkit
        - Services:
            - Operation: Access database through the backend to get files from documents.
        - Slices:
            - Operation: Using AsyncThunkAPI to access the redux services and initiate the service to store it into state.
    - Types:
        - auth: To get user related states.
        - players: To get users players related states.
        - data: To get players data related states.

2.3 Signal Processing

2.3.1 Architecture


Amazon AWS S3
Taking raw EMG data
Data
If it is non labeled data
If it is labeled data
Contraction Detection Algorithm
Eliminating unnecessary data with given index-range from customer
Filtered new data
Removing NaN Values
Generating frames with to_frame()
Frame
Frame
Frame
RMS Calculation
MNF Calculation
BandLimit MNF Calculation
Calculated value
Calculated value
Calculated value
Generating new DataFrame
Sending new DataFrame
AWS S3 Bucket

2.3.2 To_Frame Method

The To_Frame() method is one of the crucial parts of the Signal Processing field. Before doing contraction detection or calculation features, we are separating our data into frames. This method needs 2 parameters which are size of frame and overlap value. For example, we are using 128 size and 32 overlap values in our project. Methods take the first 128 index value from data DataFrame and append it into an array. The second frame will start from the 96th index value from data DataFrame which means it is overlapping the value during the to_frame process. After this separation process, it returns an array which contains all created frames.

2.3.3 Contraction Detection

After taking the raw EMG data from S3 Bucket, the signal processing system starts with detecting contraction parts on the data. There are several methods for finding contractions for different types of data. As we mentioned in the Introduction, Neurocess Limited company has improved their system. This new system is basically, customers are giving contraction time zones to the company from UI and the company can eliminate unnecessary data more easily thanks to given time zones values. If the data is the old type, we are using a contraction_detection() method on the data because there is no given time zone value on this type of data. This function basically follows changes of data value and it tries to detect contraction zones. If the data is a new type which is a label system, then we just basically select index ranges on DataFrame on given time zones.

2.3.4 Elimination of NaN values

The current EMG sensors are sending their data to the mini-pc with UDP protocol. The UDP protocol is a fast way to send data but it is not a reliable protocol therefore packet losses can occur during the gathering data process. NaN values are representing packet losses on DataFrame and it causes crashes on Machine Learning models. For eliminating the NaN values on the data, we are basically using replaceWithMean() method. This function is replacing the NaN value with the mean value of the frame.

2.3.5 Feature Calculations

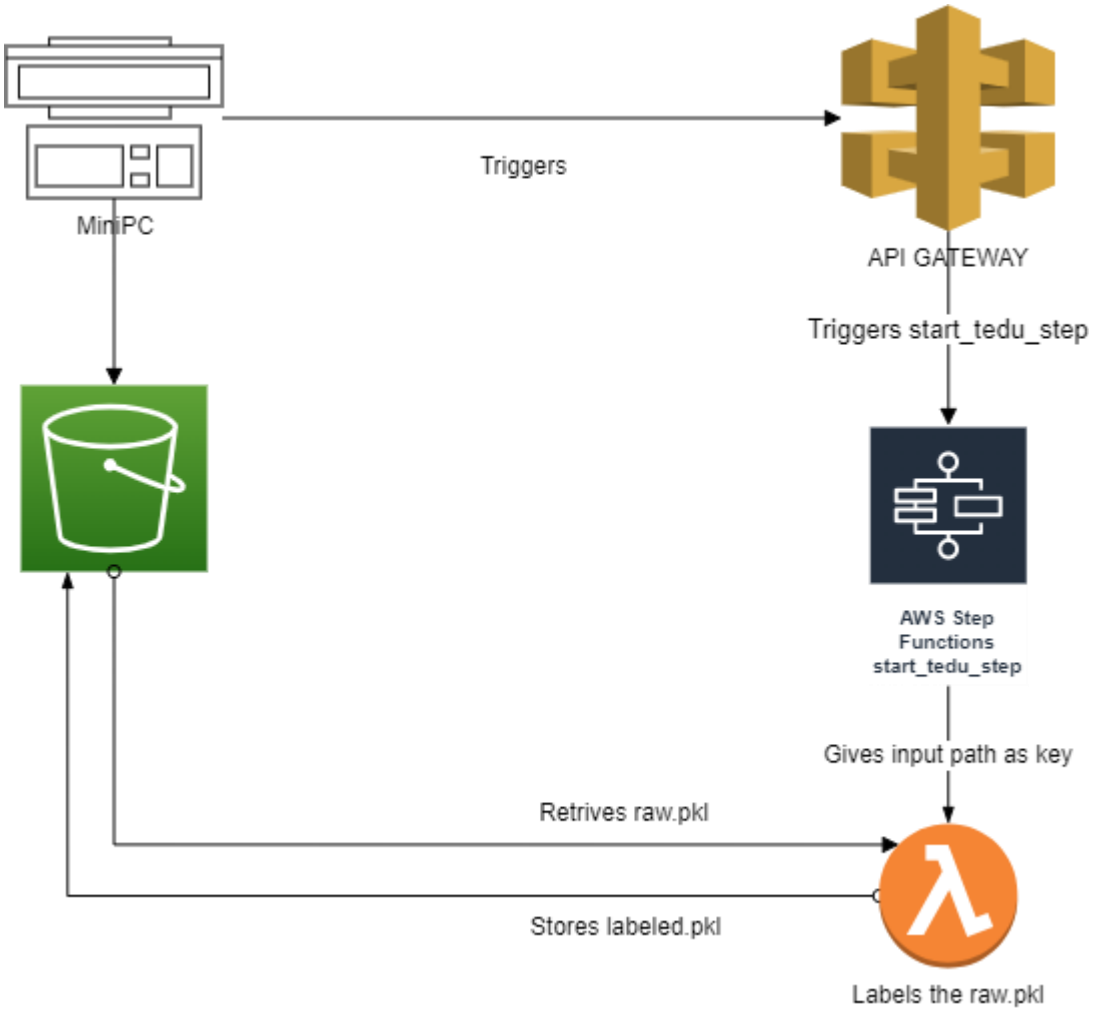
We are currently using 3 features on the project which are RMS , MNF and Bandlimit MNF features. These features are not complicated but require calculation for our Machine Learning models. After taking the frame array from to_frame() method, the system calls feature methods for each element frame in the frame array. RMS and MNF functions give 1 calculated value for each given frame. However, the Band Limit MNF feature works differently; this feature can return several numbers of values according to given parameters. These parameters are related to the requirement of the Machine Learning field but mostly we give 3 frequency range to the band limit MNF feature that's why it returns 3 different calculated values for each given frame in a dictionary format.

2.3.6 AWS Integration

In our project, there are several fields which are Signal Processing , Machine Learning and User Interface. We are using Amazon AWS services for transferring necessary data files between these fields. After making calculations and generating new DataFrame in the Signal Processing part, the system sends this data in the S3 Bucket system with boto3 library.

2.4 Amazon Side Signal Processing:

2.4.5 **Architecture**



Process starts with the raw emg measurements by the minipc. After the miniPC completes its measurements, the data is then sent and stored into the Amazon S3 bucket. After the storage operation, the miniPC then triggers the API Gateway to commence the labeling operation.

API is responsible for triggering the “start_tedu_step” step function and the step this function is responsible for starting the lambda function that our signal processing codes reside and passing a parameter “path” which is a folder path for the raw.pkl to be processed.

Lambda function then retrieves the raw data and uses Signal Processing to make it into labeled.pkl.

3.Engineering Solutions, Contemporary Issues and Social Impacts:

The purpose of "Neurolysis" is to predict the fatigue levels of players using data collected from players during training.Since the project is sponsored by Neurocess company, the required data to collect from athletes is provided by the company.After the company shared with us the signals it receives from the muscles of athletes who have trained repeatedly,the transmitted signals were processed and the parts that contracted on the muscles were identified and parsed.Different Machine Learning models were applied after the parsed signals were converted into datasets where machine learning model could be applied.As a result of the models applied, we aim to measure the fatigue levels of the muscles of the athletes.At the same time, we aim that the models we use and the outputs we achieve will be a recommendation for other people in the industry who can work for the same purpose.Thanks to the outputs obtained, company employees and physiotherapists will be able to analyze the muscle fatigue of the athletes more easily.When the model works successfully, the risk of injury to the athletes will be reduced.Apart from the muscle data of the athletes, their physical characteristics are also kept in the system and never shared with third parties, and it is important to keep them confidential.The interface created at the end of the project, where we display the data of the athletes, cannot be used by people other than physiotherapists and admin users.

4.Tools and Technologies Used:

1. **Python**

   We have used python to predict fatigue from a given dataset.

   We have used Python libraries to apply Machine Learning Models in a given dataset.Frequently used libraries :

   - **Keras :** Keras is a deep learning API written in Python.Keras is the most essential part of our libraries.We have used Keras for LSTM,Bidirectional LSTM and their preprocessing.Keras makes it very easy to define and train models.
   - **Tensorflow :** Tensorflow is an open source deep learning library.Thanks to its flexible structure, it provides more opportunity to deploy arrangements using one or more CPUs, GPUs, regardless of platform, with an API.In our project, first of all we have applied model on small dataset and ran the model on

CPU.When the dataset grows, the model is now run on the GPU, which we think will be more efficient in terms of performance.That's why tensorflow library was used.

- **Pandas :** The Pandas library is used to organize data and make analyzes easier and faster.Pandas has some data structures for easy analysis of data.DataFrame structure was used frequently in our project.
- **Numpy :** NumPy's array type is useful for numerical work. For example, the Numpy library is used in the project to manipulate matrices.
- **Pickle** : Pickle is one of the useful libraries for transfering pandas DataFrames. It basically creates binary files and also decreases the size of data.
- **OS :** In many cases, we have to add automated systems in our project. The OS library is helping to give commands to the terminal dynamically which we mostly used for our testing processes.
- **Scipy :** Scipy library is used for scientific computing and technical computing in the feature calculation part.

2. **Spyder :** We used Spyder IDE to manipulate and visualize data such as confusion matrix and column charts.Also the variable explorer part was very useful in terms of checking whether we got the data correctly.
3. **Cuda :** It is a parallel programming platform that highly contributes to the computing performance of the computer.We have used Cuda to run our model.
4. **Javascript:**
    a. **Framework:** React
    b. **Dependencies**:
        i. Material UI
        ii. Core UI
        iii. axios
        iv. chart-js
        v. jscharting
        vi. Redux-toolkit
        vii. express
        viii. mongoose
        ix. jsonwebtoken
        x. bcryptjs
        xi. JEST

xii. Supertest

5. The Proposed Algorithms

5.1 Signal Processing Algorithms

5.1.1 **RMS**

RMS feature is a type of time based feature and it basically calculates the root mean square value of each frame

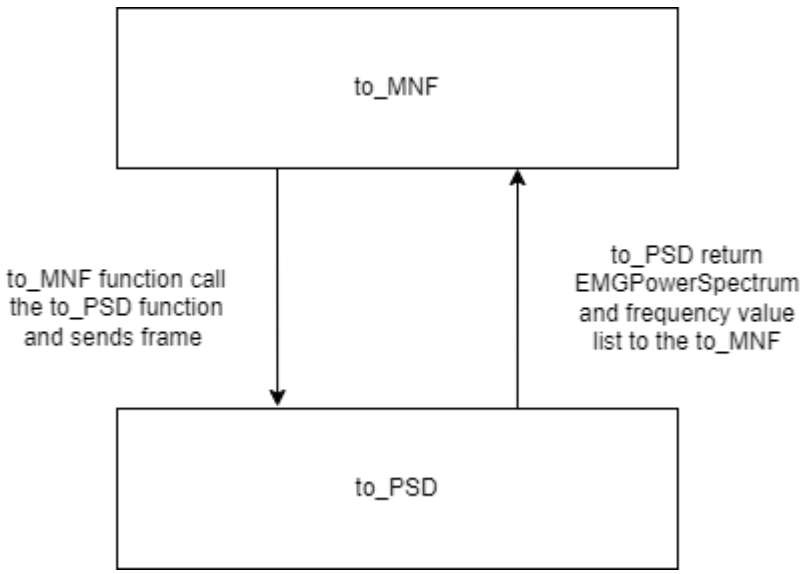
$$RMS = |a(\sqrt{b})|$$

a = mean calculation

b = EMG value in the frame array

We are calculating the changes of average EMG value on each frame. Therefore, we can analyze changes on the data based on time periods.

5.1.2 **MNF**



MNF feature is a type of frequency based feature. MNF functions calls the to_PSD which calculates frequency and EMGPowerSpectrum values for each element in the given frame. to_PSD uses scipy.signal.welch library for calculating EMGPowerSpectrum and frequency values. Then it returns an array. After that MNF functions made calculations as shown in the figure

$$MNF = \frac{sum(a * b)}{sum(b)}$$

a = frequency value of each element in the frame
b = EMGPowerSpectrum value of each element in the frame

5.1.3 **Bandlimit MNF**

Bandlimit MNF feature is a type of frequency based feature and it is similar to MNF feature calculation; it is more sensitive than MNF method. Bandlimit MNF calculates the frames with given frequency ranges as parameters. Therefore, it becomes a more sensitive feature method but the calculation equation is the same with the MNF method. This method can return more than 1 value for each value. Number of returned values is related to the given parameter. Currently we are using 3 different ranges which are [0-180] , [180,270] and [240,360]. Therefore, it returns 3 value for each given frame

5.2 Match-Day Algorithm

In our Machine Learning models, we also need y_true value for calculating mean square error. Therefore, we developed a match-day API system for taking real match days of selected sports teams. This algorithm basically scrapes all match day dates with a given time range and its return date in a json file. After taking this json file, we basically calculate the time difference between training day and match day based on day. The difference value range is between +3 and -3 which means that, if the closest match day is before training day, then it gets plus value otherwise it gets minus value. On the other hand, if the time difference is higher than 3 day before or after training day then it gets -200 value. This -200 value is eliminated in the Machine Learning part because we can’t predict fatigue value correctly.

5.2 Machine Learning Algorithms

5.2.1 Preprocessing of data

Normalization of data: Normalization is a process that translates data into a range between 0 and 1. In our dataset, we applied normalization in range [0,10] since our dataset values are very large, we wanted a more realistic normalization approach. Normalization doesn't make assumptions about the distribution of your data. Our time series tend to be always up or down. Therefore, normalization is one of the effective approaches to find maximum and minimum values in our time series. Before applying each model, a normalization method is used for our data.

PCA (Principal Component Analysis):

The DBScan model requires Principal Component Analysis(PCA) which DBScan is an unsupervised learning method. With this method, also known as dimension reduction, the computational expense burden can be reduced. Examining correlation matrices shows that it is one of the situations where we should most often apply PCA application among highly correlated features. According to our observations, it shows that explained variance ratio helps to determine eigenvectors and eigenvalues. Projection of eigenvalues is the application of PCA. so we try to detect the features with high correlation in the dataset and catch the ones with the same variance values. Explained variance ratio is used to determine the quantity for the number of dimensions that can be reduced. According to our experiments, %90 or %95 effective results to find dimension reduction. As a result of this preprocessing phase, we have reduced the cost of training all dimensions.

5.2.2 Hyperparameter Tuning

It tried to tune the model by constantly changing some parameters such as the number of epochs we used in the model. This process can be called model tuning. It is one of the optimization phases of our project. Since there is big data that is dealing with, some of the configuration variables changed for training of our data. This tuning process is applied from all of the feature selection modes and it also shows that bandwith1 features are more good at giving high accuracy value because our goal was to maximize our predictive accuracy.

According to our experiments, Bidirectional LSTM uses some units as dense. And our experiments show that the model is more effective with the “dense_1” layer. For instance: ”dense_2” layer represents our Softmax dense.

When examining Bidirectional model LSTM summary, we analyzed model dense values. The Penultimate density was taken and smoother data was obtained in the feature vector creation. This situation is one of the processes of selecting our dense variable in the model tuning.

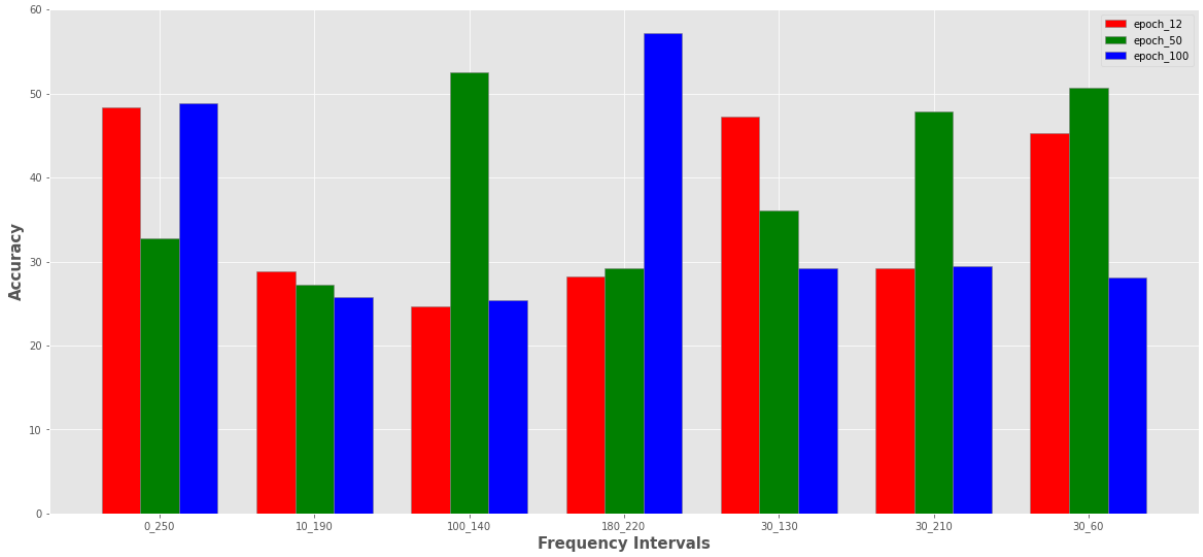
Layer (type)	Output Shape	Param #
bidirectional (Bidirectional)	(None, 64)	10240
dense (Dense)	(None, 16)	1040
dense_1 (Dense)	(None, 8)	136
dense_2 (Dense)	(None, 6)	54

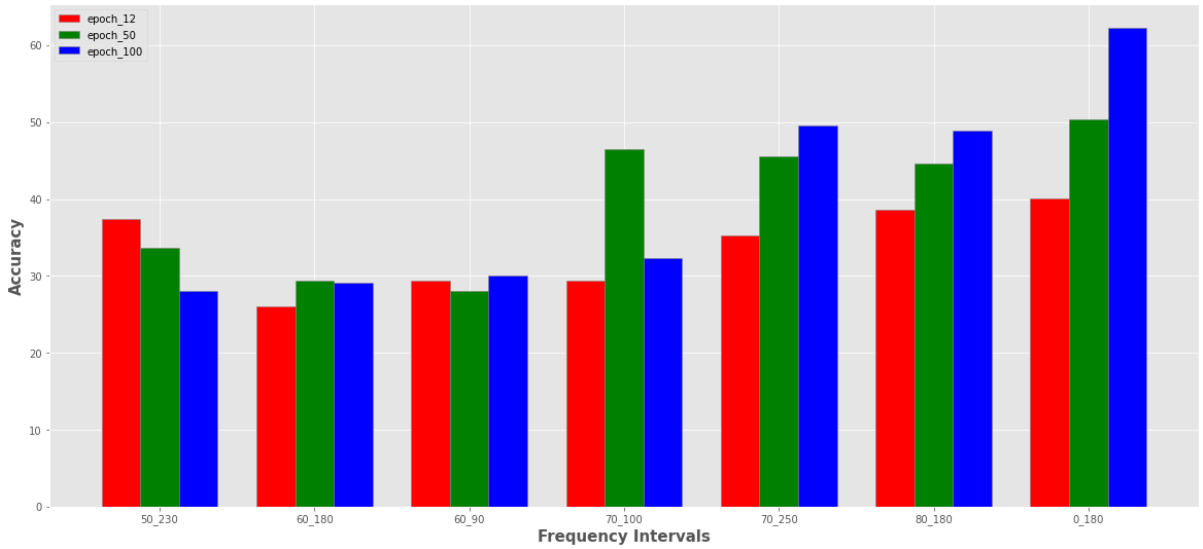
Experimenting Epochs: Another experiment for parameter tuning was tuning of the epoch value. Experiments are designed from a low epoch value and gradually increase the number of epochs while tuning the epoch number to its required value. According to the results obtained, with the increase in the number of epochs too much, the model started to memorize itself after a point. This was an undesirable situation. However, it has been observed that the generalization probability of the model is low because the number of trainings is low in a model operated at a very low epoch. Therefore, observing the loss values for each epoch, analyzing the ideal epoch number is an essential stage for tuning process . Possibility of overfitting prevented with this way. Gradually increased the epochs in order to improve the results we got with small epochs during the creation of the model. At the end, this experiment is completed.

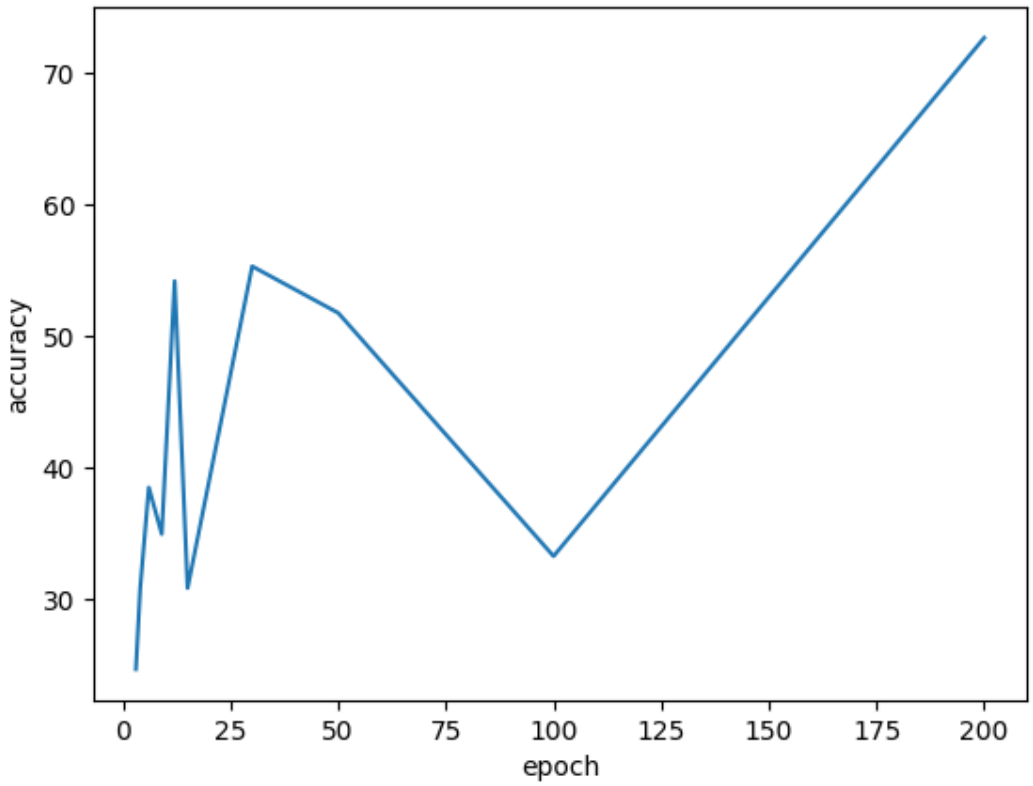
Experimenting Frequencies: Frequencies values are another tuning parameter for our model. In the signal processing part, model tuning is implemented with different frequency ranges, with different bandwidth values of the “toFrame” function in a received frame area. Some of the frequencies values are [0,180], [0,180], [0,250], [0,360], [10,190], [30,130], [30,210], [50,230], [50,230], [60,180], [70,250], [80,180], [100,140], [180, 220] frequencies interval. Our tuning training shows that we analyzed that model tuning results were better at lower and medium frequency values, without keeping the frequency value range too high.

Frequency Intervals : ['0_250', '10_190', '100_140', '180_220', '30_130', '30_210', '30_60', '50_230', '60_180' ,'60_90', '70_100', '70_250', '80_180','0_180']

In order to better analyze these frequency ranges, when the graphs below were created, it was seen that the best values were between 0_180 and this range was used in the model.







6.2.3 LSTM Model

LSTM(long short-term memory) used in the field of Deep Learning. It is a variety of recurrent neural networks (RNNs) that are capable of learning long-term dependencies, especially in sequence prediction problems. According to researchers, LSTM is one of the best approaches for time series data. LSTM is a model that can be easily used in time series estimations.Also , LSTM provides automatic testing. The model fits once on training data then again on the full time series dataset. Our dataset aims to be an overfitting situation, LSTM is also useful for preventing overfitting problems.

In LSTM, it is possible to validate and observe loss during each epoch of training on the validation data. epoch is a hyperparameter that

After applying preprocessing of data(normalization process), OneHotEncoder structure is used to determine our label and It makes the representation of categorical data more impressive and easy. It is used as the y parameter in the output we expect to train the data.

Before model fitting in the model creation part, LSTM uses Dense values.Dense layer is the regular deeply connected neural network layer. It is the most popular and often utilized layer.

The Dense layer does the following on the input and returns the result.This model uses three Dense Layers (e.g : 16,8,2).As a result of the experiments, it is more appropriate to give the multiples of 2(e.g : 16,8,2), since large matrix multiplications are made while calculating the layer values.Thus, the model has been made to give faster results.Since it is used an array with an size 50, the dense value should not above the 50.The reason why the dense values are decreasing gradually is to reduce the size of the array.

Model compiling process is created by the “adam” optimizer and the "categorical_crossentropy" loss function. In the training part creation our dataset length, datasize frame size which can be 61 or 50 and lastly in terms of feature selection.

After applying feature selection opportunities, model structure is created with mentioned dense values with activation functions which are tanh and softmax. Tanh uses sigmoid and hyperbolic approach as activation function. Different activation functions use inner dense layers and it has an effect on the performance. LSTM like many of the models which consists of recurrent structures uses activations and mostly, the number of these activations are two as recurrent activation and output activation. According to model development logic, developer normally can use these possibilities. Actually, LSTM uses Softmax as output activation. After adding two tanh and activation functions. There is a requirement for deleting activation function for refreshing.

5.2.3.1 Model Fitting for LSTM and Bidirectional LSTM

Some of the parameters that applied for model developing and fitting processes as below:

***Epochs*:** While the model is being trained, not all of the data are included in the training at the same time. Therefore, we divide the data into certain parts to get it into training. A part of it is trained and the performance of the model is tested. This process is repeated at each training step to try to calculate the most appropriate weight values for the model. Since epoch is a hyperparameter, it is defined by the developer of the model. Each of these training steps is called an “epoch”. In the experiments, first the epoch value of 5 was taken in the small dataset and the model was trained.When the large dataset is passed, first the epoch value is taken as 5 and the model is trained, and the model is run again with the values of 20, 30 and 50.Considering the trained models, it was decided to train the final version of the model with an epoch value of 200 as mentioned. The value with the smallest validation loss was found and that value was taken as the best epoch.

Verbose: While training the model in each epoch, saving these results to an .h5 file; If the verbose value is set to 0, it prints the final result. Verbose value of 1 prints all epoch values. In our observations, we applied verbose as 1 to analyze all epoch training results.

Validation split: It is used to find overfitting and underfitting regions in our dataset and it is helpful to determine when we will stop our model. In a situation where we are not sure of the number of epochs then we applied a validation split variable. According to our observed results, where the best validation loss value is obtained, it may not always give the best result in the testing data.

Softmax: Softmax is a mathematical function that turns a vector of integers into a vector of probabilities, with the probability of each value proportional to the vector's relative scale. Softmax should always equal the output dimension.

Tanh: Due to the saturation of relu, the tanh value was used as an activation function during the addition of the first two dents. It contributes positively to the performance of the model in the experiments.

5.2.4 Bidirectional LSTM

Bidirectional LSTM is an upgrade over LSTMs. In bidirectional LSTMs, each training sequence is presented forward and backward so as to separate recurrent nets. The output layer for both sequences is the same. Bidirectional LSTMs have comprehensive knowledge of every point in a sequence, including everything that comes before and after it. The reason for using Bidirectional LSTM is because it is working on a problem with time series, it is a necessary step. Trying to analyze the forward and backward data of existing time series is one of the reasons we work on this model. 61.2 structure has been used since data is derived from 2 seconds. Actually, our goal was analyzing 2 seconds of data. Since MNF and RMS are features that come out of small frames, the more we divide the data, the more advantageous it is for us. It was also observed in this section that the data works better in 61 frame size. On the other hand, Bidirectional LSTM can run back and forth connections over this existing structure.The feature vector values obtained from the applied bidirectional LSTM model give information about whether it will be before or after the match day, in terms of classifying it.The process performed here is called the transfer learning process, as mentioned before.Here, we create a feature vector and actually make a regression to the transfer learning model. Regression output becomes fatigue value.

5.2.5 Support Vector Machine (SVM)

Support Vector Machine (SVM) is an intelligent supervised machine learning program for computation or regression.However, it is mostly used in classification problems. One of the advantages of SVM is that there is no need to train with epoch values. Because of this situation, we were able to more easily observe the large dataset over the GPU.SVM has 3 most used hyper parameters.These parameters are:

- **C(Regularization) :**C is the penalty parameter, which represents misclassification or error term. The SVM optimisation uses the misclassification or error term to determine how much mistake is acceptable. This is how the trade-off between decision boundary and misclassification term can be managed.
- **Gamma**:It defines how far it influences the calculation of plausible lines of separation. When gamma is higher, nearby points will have high influence; low gamma means far away points are also considered to get the decision boundary.
- **Kernels:** It defines how far it influences the calculation of plausible lines of separation.When gamma is high, local dots have a large impact; when gamma is low, distant points are also considered when determining the decision border.

In the development of the SVM model, with the feature selection approach mentioned above, a model was developed over SVM for the overlap values of 0, 25 and 50 where the frame size is 50 and 61.Since the basic idea of SVM is based on the approximation with mapping the data into a high dimensional feature. When we evaluated the SVM model over lstm and bidirectional lstm, the success rate of SVM was lower than LSTM and Bidirectional LSTM since we were working on a problem with time series.

5.2.6 DBSCAN (Density-based spatial clustering)

We have developed models with specific output labels in LSTM and SVM models. In this section, we applied clustering methods. Clustering machine learning algorithms are popular solutions for unsupervised learning areas. It is one of the clustering algorithms. There is no doubt that most of the clustering methods are based on distance. However, DBScan relies on density. The reason why we applied this model is that we wanted to test how similar results we could find without knowing our labels.

The DBScan algorithm requires parameters such as eps, minimum_sample. “eps” is the most extreme distance between two examples for one to be considered as in the neighborhood of the other. We utilized the default value as 0.5 for eps, by trial and error, whichever eps value gave better results, we worked on that value in the model. “metric” is another parameter that calculates distance between instances in a dataset. In our experiments, euclidean and cosine matrices are tried. In the last step euclidean is applied.

After labeling results are derived and mapping algorithms are applied. Therefore, this model approach is based on trending. However, through our implementation shows that this model just give an idea in terms of statistics. Therefore, this is not directly fit our model.

6. Final Status of the Project

Model Discussion of Models

A machine learning technique in which a model trained for one task is redesigned in a second related task can be called the Transfer Learning Model.Here, it is desired to get a fatigue output by using the outputs we obtained thanks to the feature vector in another model.In this way, it is aimed to obtain a model that shows higher success and learns faster with less training data by using previous knowledge with transfer learning.NormTo summarize the design in the project again, 2 different features, mnf and rms, are obtained by making feature selection from the raw data coming from the emg device.Here, after treating the rms value as power and the mnf value as conductivity, it is divided into sub-bands to better understand the conductivity.After the appropriate band value is selected, a deep learning model is trained.The purpose of the deep learning model here is to understand whether the incoming training data is pre-match or post-match.To understand this, a feature vector is created by combining the training data and match values by applying an LSTM model.As a result of the LSTM model, the feature vector we have trained becomes able to give you whether the training is pre-match or post-match.The Bidirectional LSTM model is also designed in this project. The Bidirectional LSTM is just an extension of the LSTM model.Instead of training a single model, bidirectional LSTM introduces two. The first model learns the input sequence, whereas the second model learns the opposite of that sequence.

The purpose of using Bidirectional LSTM is that we have a time-series data and we want to learn the data of it bidirectionally, forward and backward.The reason why (61,2)- 61 represent the time series length, 2 represents the channel- is used while doing this process is

that the data we have is 2 seconds(2048 ms) The incoming 2 second data was created with the toFrame function of 61 size. Through this implementation aimed splitting into smaller frames. in this way, the detail factor in the frames is transferred. The backward and forward connections between the frames were calculated by subtracting the mnf and rms values from the 61 frame data we obtained from the 2 second data and inserting them into the bidirectional LSTM.

Additionally, a separate learning from scratch is performed for each task in machine learning, while features, weights, etc. obtained from previously trained models with transfer learning used for a new task.

Weights in pre-trained models contain a lot of information. Therefore, by using this information and fine-tuning, the new model is trained faster.The advantages of the Transfer Learning model and one of the reasons why it is used in the project is the faster training time.Weights in pre-trained models contain a lot of information. Therefore, by using this information and fine-tuning, the new model is trained faster.In this way, a successful model was obtained without using very large epoch values.First, after giving small epoch values, this value was gradually increased and decreased and the appropriate model was obtained.

Another advantage of the transfer learning model is that the model is trained using less data.The biggest disadvantage of models created from scratch is that a large-scale dataset is required for training.Generating these datasets takes a significant amount of time. Instead, fine-tuning pre-trained models results in higher performances using less data.Another advantage is the efficiency.A simple process of adding new fully connected layer(s) to pre-trained models seems to improve success.

After training the obtained feature vector with the deep learning model, these feature vector outputs were put into a linear regression model.By tuning the function parameters obtained from this model, the fatigue value, which is the aim of the project, was tried to be estimated.In the established regression model, it is necessary to investigate the fit of the data (observations) to the regression line.

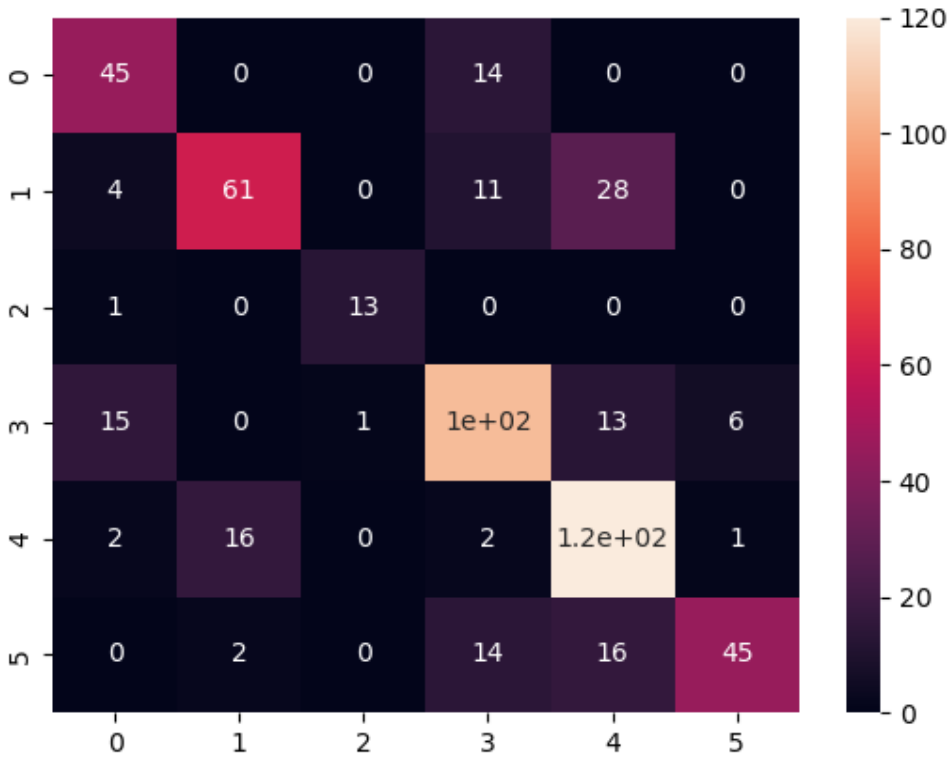
Our metrics that used to measure model performance:

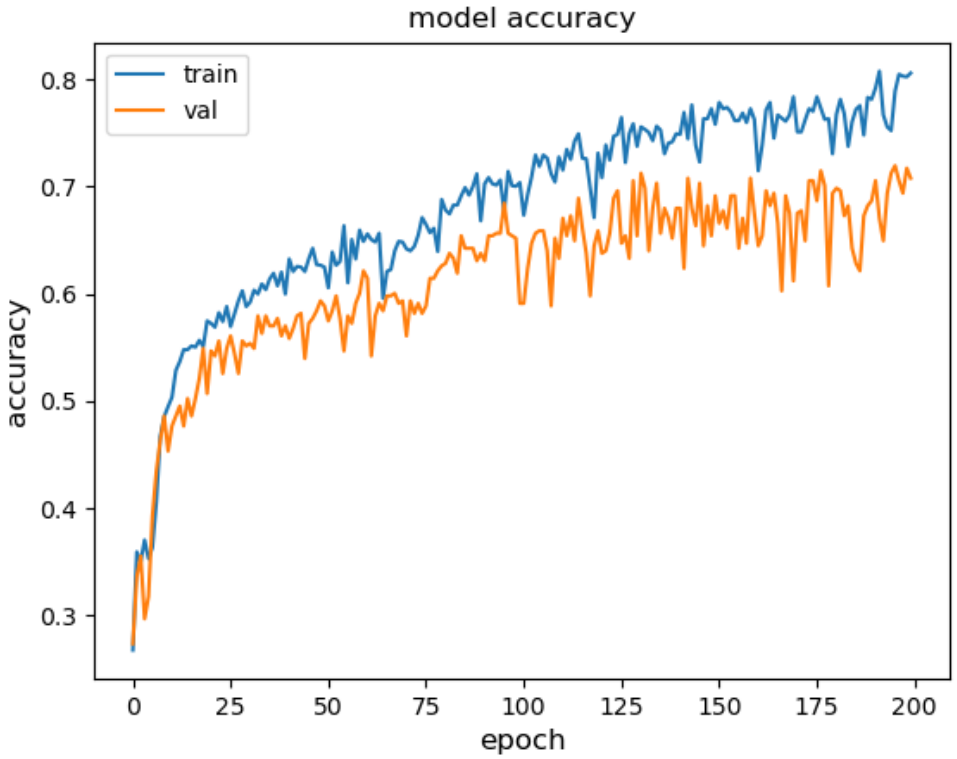
- Confusion Matrix
- Accuracy values (Number of correct prediction/Total number of predictions)
- F1 Score values (2*(precision*recall)/(precision+recall))
- MSE

Bidirectional LSTM:

Our best result is given in 0_180 frequencies interval.At the beginning of the project, our model performance did not work well in terms of it's accuracy value. Like all of the machine learning models, after completing model, there is a need of tuning stage that discussed in Deep Learning section. Various tuning operations were determined and the model was run a large number of them, and the lstm structure that gave the best performance was created.

Accuracy: Bidirectional LSTM → 72.71028037383176





Support Vector Machine (SVM):

Support vector machine (SVM) is a statistical learning theory-based machine learning approach. For small samples and non-linear models, SVM constructs a high-dimensional hyperplane and classifies samples by computing the maximum distance between training data points on the hyperplane.SVM has improved stability and fewer training parameters in small-sample model training.For this reason, large data was not used while applying SVM.SVM is memory efficient because it uses a subset of training points (called support vectors) in the Decision function.

Accuracy and F1-Score, Experimenting of different possible Kernel values Linear, RBF, Sigmoid, Polynomial Results:

Linear Kernel Confusion Matrix, Accuracy, F1-Score:

```
Evaluation: Linear kernel
=== Test Set Performance for 0th Model
Accuracy is: 64.523364485981308
f1:  57.375395919878976
[[ 4 12  1 18 14  6]
 [ 9 18  2 25 21  1]
 [ 6  0  2  6  2  1]
 [21 23  2 19 34 12]
 [26 23  0 22 27 13]
 [16  8  1 16 12  5]]
```

RBF Kernel Confusion Matrix, Accuracy, F1-Score:

```
Evaluation: RBF kernel
=== Test Set Performance for 1th Model
Accuracy is: 49.401869158878505
f1:  63.164142127044954
[[  0   0   0   0  55   0]
 [  0   0   0   0  76   0]
 [  0   0   0   0  17   0]
 [  0   0   0   2 109   0]
 [  0   0   0   0 111   0]
 [  0   0   0   0  58   0]]
```

Sigmoid Kernel Confusion Matrix, Accuracy, F1-Score:

```
Evaluation: Sigmoid kernel
=== Test Set Performance for 2th Model
Accuracy is: 59.934579439252335
f1:  41.187384044526894
[[  0   0   0   0  55   0]
 [  0   0   0   0  76   0]
 [  0   0   0   0  17   0]
 [  0   0   0   0 111   0]
 [  0   0   0   0 111   0]
 [  0   0   0   0  58   0]]
```

Polynomial Kernel Confusion Matrix, Accuracy, F1-Score:

```
Evaluation: Polynomial kernel
=== Test Set Performance for 3th Model
Accuracy is: 61.19626168224299
f1:  32.467192019882546
[[13  8  2 16 12  4]
 [11 13  2 22 21  7]
 [ 1  3  0  6  5  2]
 [14 23  6 30 27 11]
 [14 26  4 21 36 10]
 [ 9 13  6 12 15  3]]
```

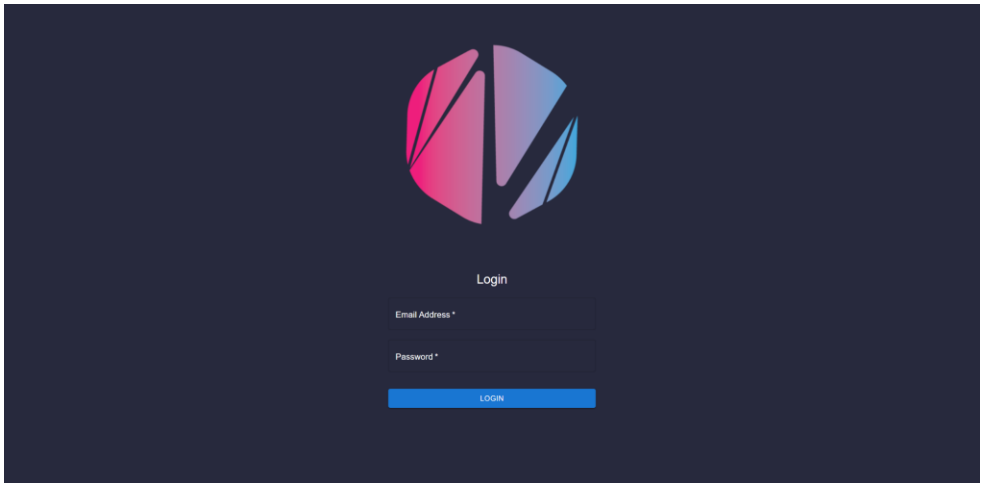
According to select best hyperparemeters C = 1, gamma = 0.001 and Confusion Matrix as follows:

```
SVC(C=1, gamma=0.001)
[[ 0  3  0 23 29  0]
 [ 0  3  0 30 43  0]
 [ 0  0  0  7 10  0]
 [ 0  2  0 42 67  0]
 [ 0  4  0 30 77  0]
 [ 0  2  0 21 35  0]]
```

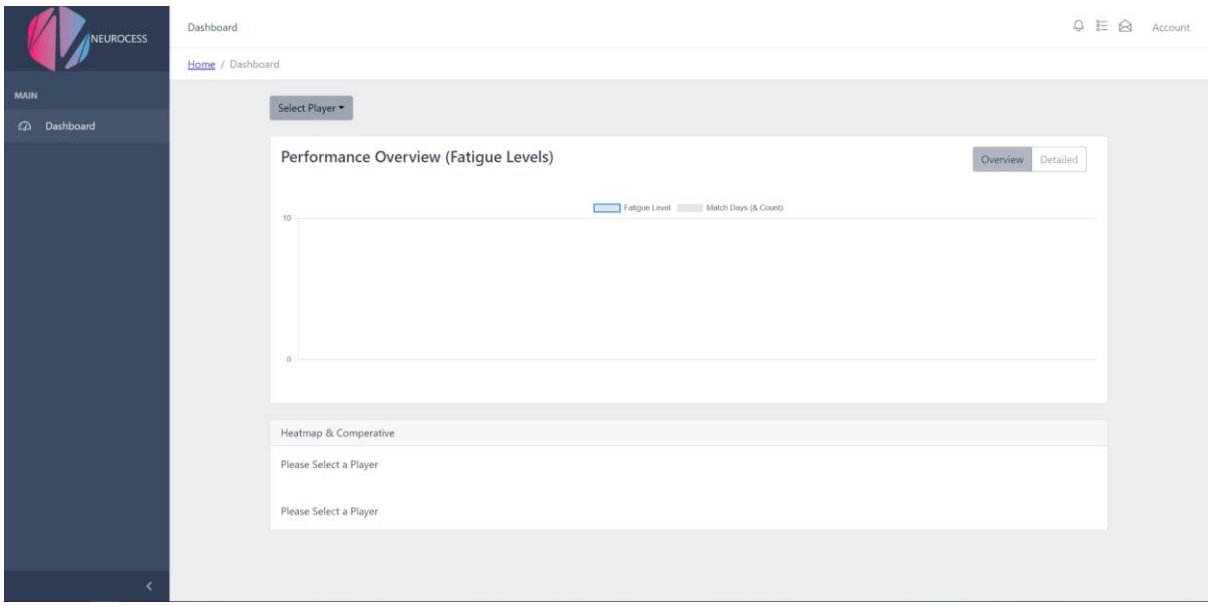
Since we are dealing with time series problems, our model creation does not fit very well with SVM. Therefore, our main model is based on Bidirectional LSTM.

7.UI User Guide

- **Login Page**: Our login page consists of 2 forms to be filled for authentication purposes. After entering correct information, the user can proceed to the dashboard.

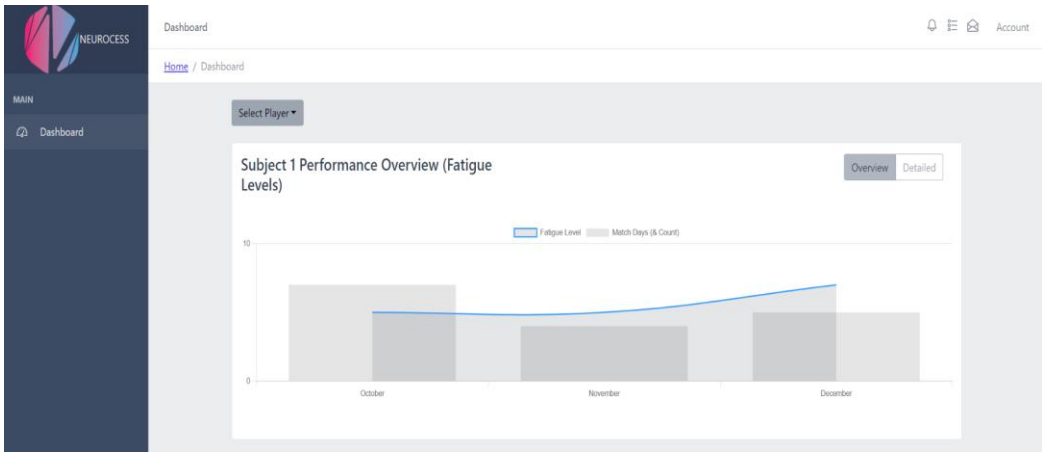


- **Dashboard**: After the user logged in, the dashboard page can be accessed by the user. To see the data of the subject of the users choice, the user has to select the subject from the dropdown section.

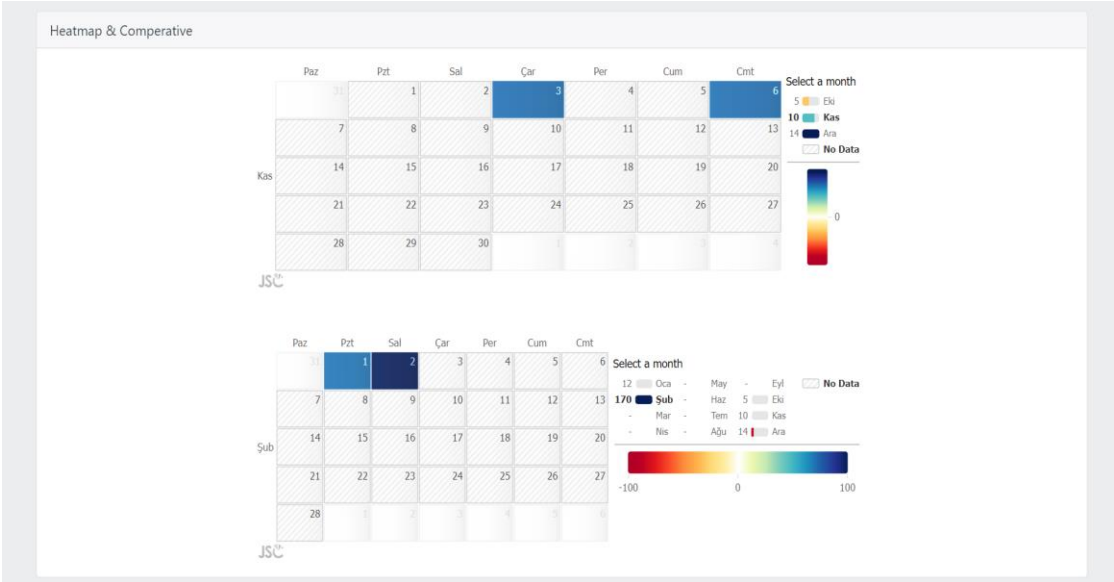


After selecting the “Player” the user can see the Overview graph, which has the fatigue level on a line chart and days of the match as a bar chart, which the team of the

player belongs to.



The user is also able to see selected players fatigue heatmap through the data points which is gathered by Neurocess equipment. Also can compare the results with the average of all the subjects through another heatmap.



(All of the data shown in the guidelines are mocked)

To logout from the site, users can use the top right button (“Account”) to access logout functionality.

8.Testing Details and Results

8.1 Machine Learning

What we expect as the final product within the scope of the graduation project is actually an analysis, not a product.The analysis results obtained are left to the interpretation of the user.Likewise, due to the current working procedure of the company, physiotherapists and data scientists of the company test the ml outputs by reporting the analysis outputs and making the necessary intuitions.In this case, it is not possible to make a clear comment by comparing the current state of fatigue values based on the results obtained from Machine Learning models.In the ML part of the project, it is aimed to create advisory outputs for those who will examine the analysis.For this reason, instead of making a clear inference about which values work well and which values work poorly, it is aimed to create a foresight for the people in the sector.

In line with the explanations above, in the ML part of the project, the tests were made not on the correctness or inaccuracy of the outputs, but on whether the model works better.While there was a non-dynamic code design before, it has been made dynamic for the model to work more efficiently.For example, we parameterized some important values that we used in the model instead of entering them as hard coded.Thus, instead of changing the parameters one by one when the dataset grows, this is no longer necessary thanks to the dynamic structure created.

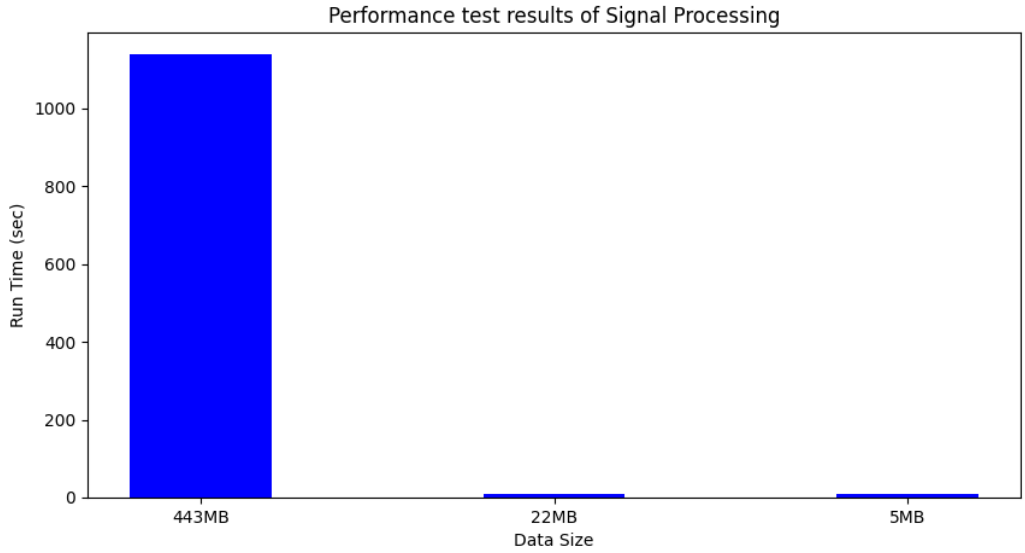
As another test metric, we can find the epoch's ideal values.During model training, we aimed to find the lowest accuracy value by starting from a small value and increasing the epoch values.The lowest accuracy value was taken as the best epoch value.This process can be seen as a process of testing the model by trying it somewhere.Likewise, the existing dataset in the model is divided into two parts, training and testing.With the use of testing data, the model has been more generalized.20%(validation split) of the dataset was used for testing and 80% for training.

8.2 Signal Processing Test Results

Signal processing is one of the crucial parts of the project that’s why it needs to be tested in many ways. In the signal processing field, there are several types of tests such as; performance and dataframe

Performance Test:

The average duration of training is about 10 ~ 20 MB which mostly equals 1 or 2 minutes training. Therefore, making the performance test was a little bit difficult on the signal processing side because we didn't have big size data in a single training. However, we requested special data from the Neurocess company and they prepared big data for us which is 443MB. After taking special data we calculated these data and calculate run-times



According to the test result, there is an important difference between big data and usual training data. However, as we have mentioned in the first paragraph, 443 MB is a very unusual size of data for our project. On the other hand, we also tested that the signal processing system doesn't give errors with big data inputs, it works fluently.

DataFrame Accuracy Tests:

DataFrame is the most important product in the Signal Processing field and there are several methods for testing accuracy of DataFrames.

Size test is one of the test methods for DataFrames, we are separating our feature array according to required sizes. In general, we use 50 and 61 sizes for DataFrames arrays. According to test results;

Trainings	MNF(50)	RMS(50)	BL_MNF(50)	MNF(61)	RMS(61)	BL_MNF(61)
Subject1_t1	Passed	Passed	Passed	Passed	Passed	Passed
Subject2_t3	Passed	Passed	Passed	Passed	Passed	Passed
Subject6_t1	Passed	Passed	Passed	Passed	Passed	Passed
Subject8_t2	Passed	Passed	Passed	Passed	Passed	Passed

System separating the array values correctly on different sizes of data and features.

Another method for the DataFrame accuracy test is detecting NaN values. The EMG sensors are sending their data with UDP protocol and sometimes package loss can occur in training. These losses represent NaN values in DataFrame. For eliminating these NaN values we are using several functions such as double_nan_checker() and replaceWithMean(). However, we created another test class for testing NaN values in the last merged DataFrame because any single NaN value in the data causes crash for Machine Learning models. According to the test results.-

Trainings	MNF(50)	RMS(50)	BL_MNF(50)	MNF(61)	RMS(61)	BL_MNF(61)
Subject1_t1	Nan not found	Nan not found	Nan not found	Nan not found	Nan not found	Nan not found
Subject2_t3	Nan not found	Nan not found	Nan not found	Nan not found	Nan not found	Nan not found
Subject6_t1	Nan not found	Nan not found	Nan not found	Nan not found	Nan not found	Nan not found
Subject8_t2	Nan not found	Nan not found	Nan not found	Nan not found	Nan not found	Nan not found

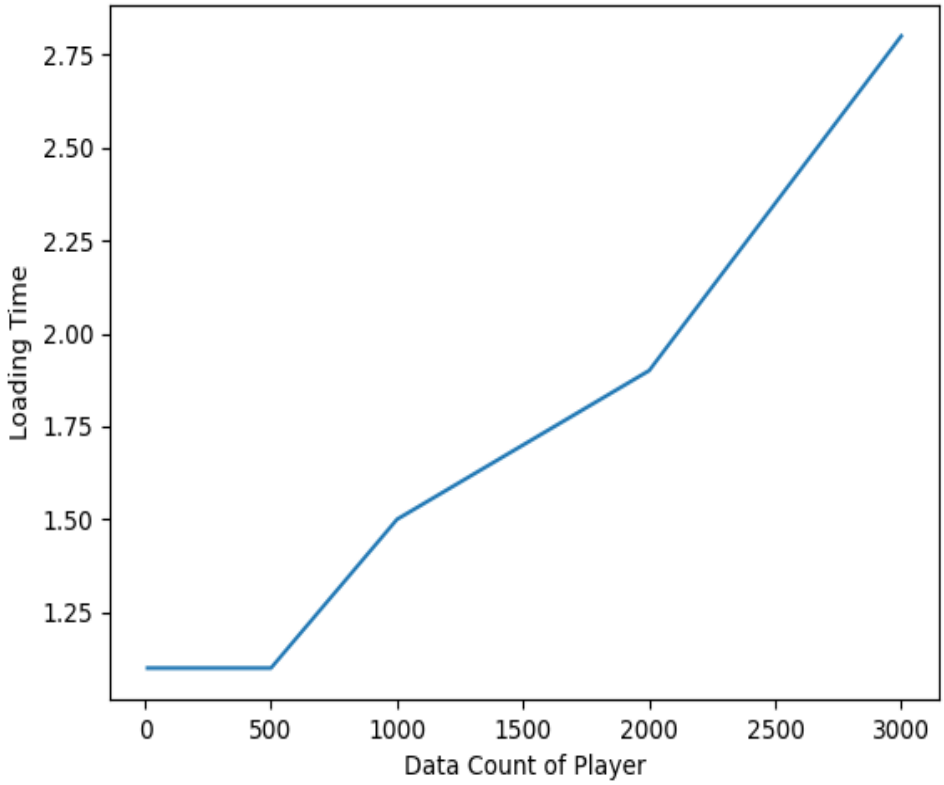
Frontend/Backend Testing Results:

- **Backend:**
  - **Routes:** Accessibility of the routes of our API is tested using unit testing methods throughout the development process with POSTMAN

software. Aim of the routes test is to constantly test whether these new routes are accessible and functionalities and limitations such as “authorization middleware” of the routes are within desired model.

- Results
  - /api/users: All routes Passed
  - /api/data: All routes Passed
  - /api/players: All routes Passed

- **System Testing**: In the late stage development of our application, base features are constantly monitored using Selenium to automate the testing on the frontend while our API mocked the possible scenarios. To achieve this, python script logs in to the dashboard and selects players, while API is used to insert or delete data from our database. After selecting players, the script looks for the correct heatmap result for a given month.
  - This test is used mainly in the late stage of the frontend development, to see new features not braking the overall structur**e.**
- **Performance Testing**:
  - Heatmap Chart Loading Time:



9.Conclusion

In our Senior Project , we have shown that machine learning algorithms such as, Neural Network(LSTM and Bidirectional LSTM),Support Vector Machine and Linear Regression we will especially work on Deep Learning can be used efficiently and effectively in order to identify the fatigue value of the player by classifying them.One of the critical issues while training a neural network on the sample data is Overfitting.Since this is a machine learning project, the model and parameters were run many times and the parameters were tuned to achieve the best results.When a neural network model is trained with more epochs than is required, the training model learns patterns that are highly particular to sample data.The model should be trained for an ideal number of epochs to reduce overfitting and increase the neural network's generalization capacity.For this reason, the best epoch selection was made for the model to give more accurate results.Deep Learning models have been tested locally with certain epoch values, and after necessary updates have been made in Keras, they have become transferable to EC2.

10. Glossary :

Feature: It is a measurable property of the object you're trying to analyze.

Sampen (Sample Entropy): Sample Entropy is a measurement of complexity at a given timeframe

Contraction: It is the activation of tension-generating sites within muscle cells.

Raw data: It is data that comes from a sEMG device.

Physio: These people help people affected by injury, illness or disability through movement and exercise, manual therapy, education and advice.

Unsupervised Learning: It uses machine learning algorithms to analyze and cluster unlabeled datasets.

Supervised Learning : It is the machine learning task of learning a function that maps an input to an output based on example input-output pairs.

SVM : Support Vector Machine

LSTM : Long short term memory

11.References :

Brownlee, J. (2020, June 23). *Softmax activation function with python*. Machine Learning Mastery. Retrieved May 25, 2022, from https://machinelearningmastery.com/softmax-activation-function-with-python/

Keras - dense layer. Tutorials Point. (n.d.). Retrieved May 25, 2022, from https://www.tutorialspoint.com/keras/keras_dense_layer.htm

SVM hyperparameter tuning using GRIDSEARCHCV. Velocity Business Solutions Limited. (2020, February 19). Retrieved May 25, 2022, from https://www.vebuso.com/2020/03/svm-hyperparameter-tuning-using-gridsearchcv/