PREDICTION OF MUSCULAR PARALYSIS DISEASE BASED ON USING NEURAL TANGENT KERNAL WITH MACHINE LEARNING TECHNIQUES

22AIE114 Introduction to Electrical and Electronics Engineering

Submitted by

Batch A Group Number 14

1. A P Devanampriya – CB.SC.U4AIE23001
2. Mathivanan S – CB.SC.U4AIE23042
3. Patel Srikari Shasi – CB.SC.U4AIE23053
4. T Rohith Balaji – CB.SC.U4AIE23069

2023 B. Tech CSE – AI

Amrita School of Artificial Intelligence

Amrita Vishwa Vidyapeetham
Coimbatore – 641 112

June 2024

Table of Contents

ABSTRACT

Common symptoms experienced by individuals affected with Corona Virus Disease 2019 (COVID-19) include muscular fatigue along with paralysis. To prevent such neuro-muscular illnesses and treat them effectively, early detection and diagnosis are essential. Electromyography (EMG) [1] signals provide significant perceptions in neuro-muscular function, and are hence used in healthcare facilities for disease diagnosis. But, the existing methods for muscular paralysis prediction are limited by constraints such as feature degradation and inefficient feature selection, bringing about imperfect classification performance. This paper aims to put forward a hybrid feature extraction method with Neural Tangent Kernel (NTK) classifier to address these shortcomings, and examine if EMG signals can be used in NTK.

INTRODUCTION

In 2019, around 77 million people worldwide were infected with the coronavirus disease and around 7 million people died from it. Many researchers have been working on the detection, early detection, diagnosis and prevention of CA group of experts in leading muscular paralysis has proposed a new definition and categorization of muscle-sdiseasesdache, new loss of taste or smell, sore throat, runny nose, nausea or vomiting and diarrhoea along with myalgia, frailty and fatigue. Even after recovering from COVID-19, many patients suffer from fatigue, shortness of breath, cognitive difficulties, chest pain, palpitations, muscle and joint pain, headaches and temperature dysregulation. Fatigue is the persistent feeling of being burnt out, generally caused by muscle weakness, loss of energy and tiredness [4]. The diagnosis and treatment of COVID and post-COVID patients is greatly facilitated by the detection of muscular fatigue and muscle paralysis. The function of muscles and nerves is assessed with the medical EMG test. These signals are analysed to detect diseases such as muscle degeneration disorders and nerve diseases. Neuro-muscular diseases include motor nuclei in the brain, spinal cord cells, nerve roots and spinal nerves that can cause muscle weakness. Damage to the peripheral nervous system is exemplified by neuropathies, which include data from the brain and spine to all other parts of the body. [3]Transient numbness, tingling, stabbing sensations, allergic reactions, muscle weakness and other symptoms such as burning pain, muscle paralysis are experienced by patients with muscle diseases. Therefore, EMG signals are studied to detect and treat these diseases [5]Nevertheless, muscle weakness is a consequence of spinal cord injury (SCI) and other clinical indications. Therefore, it is important to understand how spinal cord injury affects the survival and functionality of the motor unit [6]. As more information has become available, the categorization of paralysed muscles has improved. There are a number of approaches, with the main defining features being age of onset, body composition and etymology. A new definition and categorization of musculoskeletal disorders has been proposed by a group of experts in leading muscle paralysis [7]. Abnormal long-term disturbances of EMG function, joint contraction of antagonist muscles and excessive tension of non-functioning muscles have been observed in EMG recording [8]

Limitations of current techniques include poorer classification performance in predicting muscle paralysis and selection of irrelevant features. This study experiments with Neural Tangent Kernels to work with EMG signals, while DWT is used to reduce feature degradation. The CNN approach efficiently analyses the feature relationship for classification. The results are validated against the state-of-the-art methods using the UCI EMG-Lower Limb Dataset in terms of critical features such as F-measure, accuracy, precision, recall and error rate.

The report is organised as follows: A review of current methods for muscle paralysis prediction is presented in Section 2. Section 3 contains the implementation of Convolutional Neural Network (CNN), Neural Tangent Kernel (NTK) and Linear Regression, a mathematical method. Section 4 describes the validation of CNN and NTK with respect to different parameters. Finally, Section 5 discusses the conclusions of the research study..

BACKGROUND THEORY

The EMG signals are utilized to examine the musculoskeletal data in order to detect diseases. Several machine learning techniques have been used to identify certain activities and illness that causes muscle paralysis. Gautam et al. [9] created MyoNet, a long-term recurrent convolution network based on transfer learning. Three stages are involved in the method: FE, joint angle prediction, and movement categorization. Transfer learning was used to train the model end-to-end for knee joint prediction angle in order to maximize computing efficiency and memory. The performance of the created technique in the prediction was analyzed using UCI dataset. According to the investigation, MyoNet outperforms current techniques in terms of accuracy across a range of subject postures. The generated model's overfitting issue has an impact on the model's functionality.

An adaptive local binary pattern (aLBP) approach was presented by Ertugrul et al. [10] for the interpretation of EMG data. The adaptive neighbor value was signed for analysis using the down sampling and smoothing coefficient approach. The simulation approach employed the UCI EMG dataset, and the outcomes demonstrated that aLBP performs better in the analysis. The method's memory requirements and processing cost were modest for the EMG analysis. The necessary characteristics were 833 Pers Ubiquit Comput (2023) 27:831-844 in order to choose the created method's efficient performance in the EMG analysis.

Huang et al. [11] presented an integrated classification algorithm based on random equilibrium sampling and an association rule-based FS method. The multi-disease categorization was carried out using the stochastic equalization method. The experimental procedure made use of a variety of UCI medical data sets, which demonstrated that rule-based FS is capable of handling imbalanced datasets. The developed method's effective operation required the necessary properties.

Fractional order calculus was used by Miljkovic et al. [12] in conjunction with linear and non-linear moving window filters to remove artifacts from EMG signals. The developed technique successfully eliminated EMG signal artifacts. The results of the simulation confirmed that the confirmed that the created approach has a higher classification accuracy than the methods that are already in use. An efficient classification method is needed to improve signal detection's efficacy.

The log-likelihood approach was developed by Spanias et al. [22] to eliminate the disruption in the EMG signal. The EMG signal disturbance was eliminated using the single threshold that was obtained from the training set. The signal was classified using the linear discriminant analysis (LDA) classifier, which is based on log-likelihood. The performance of the LDA approach, which has a smaller prediction error and a low false-positive value in the threshold, was examined using the EMG signal dataset.
To effectively classify the signal, the pertinent features must be chosen.

METHODOLOGY

Data Preprocessing

Dataset

The dataset contains 22 samples, 11 normal and 11 with some pathology on knee. Each patient has 3 different kinds of samples: march, extension of the leg from sitting position and flexion of the knee stand up. Each sample has 5 columns of data taken from the lower limb. Ch1 electrode is connected to the rectus femoris (RF) muscle, Ch2 to biceps femoris (BF), Ch3 to vastus internus (VM), Ch4 to semitendinosus (ST), and Ch5 to knee flexion(goniometer).

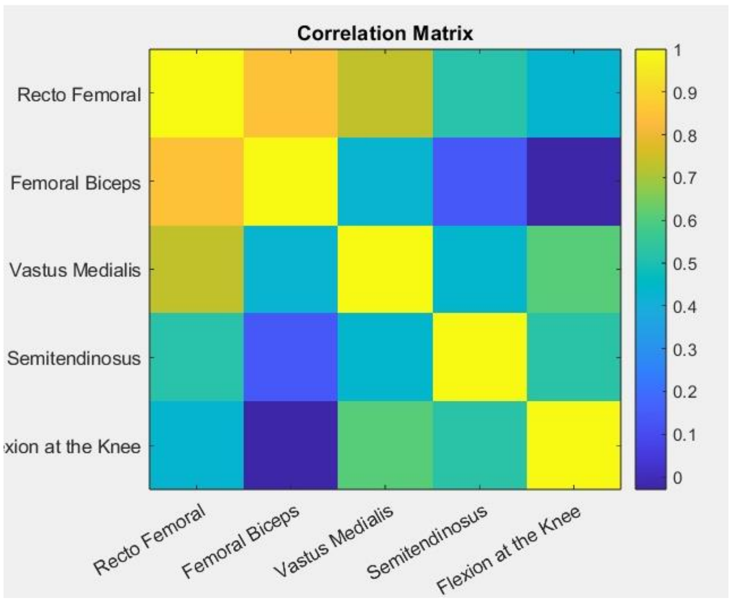
Data Analysis

1) MAR
The CSV file is read by matlab and particular numerical columns are extracted. The chosen facts are converted into a numerical array for similarly analysis.

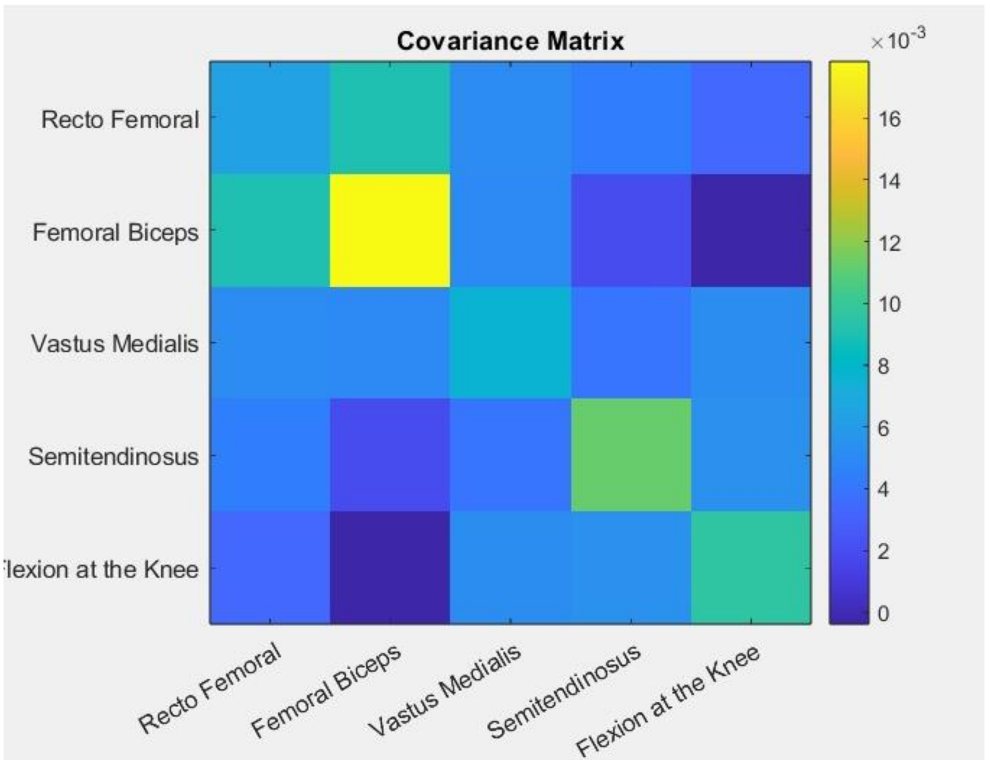
Covariance and Correlation Calculation: The covariance matrix is computed, which shows how pairs of variables alternate together. The correlation matrix, which normalizes covariance by using the variables' trendy deviations, ensuing in a degree of linear courting strength is calculated manually.

Visualization: Heatmaps for each the covariance and correlation matrices, with suitable labels for the axes are created. Color bars and titles to the figures are added for better interpretability. This approach allows for a comprehensive evaluation of the relationships among the selected numerical variables, facilitating insights into their interdependencies and capacity patterns.

Result

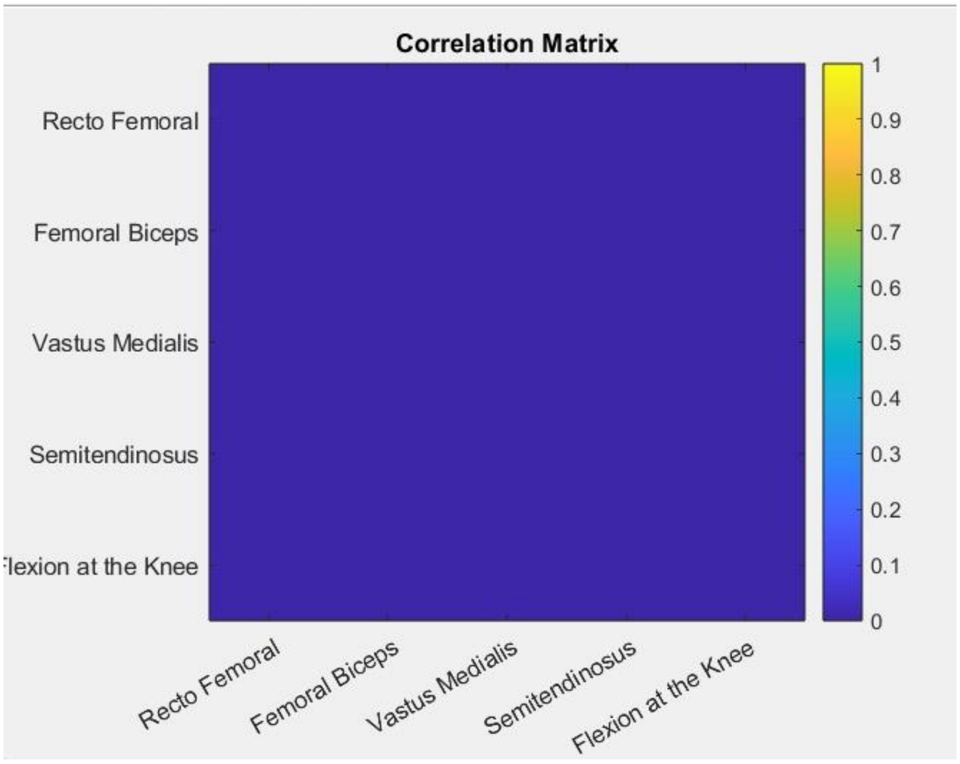


The heat map represents the correlation coefficients of the variables. Heat map of vastus capitis medialis semitendinosus knee flexion.
The heat map uses a color gradient to represent the value of the correlation coefficient: the scale on the right (color bar) ranges from 0 to 1. high correlation value. The diagonal line (from left to right) shows the correlation of each variable with itself, which is always 1 (yellow). There is a good relationship between the variables. Correlation: Darker (closer to blue) cells indicate a weak or no correlation between variables. This means that the relationship between these two variables is low. Success. Relationship.

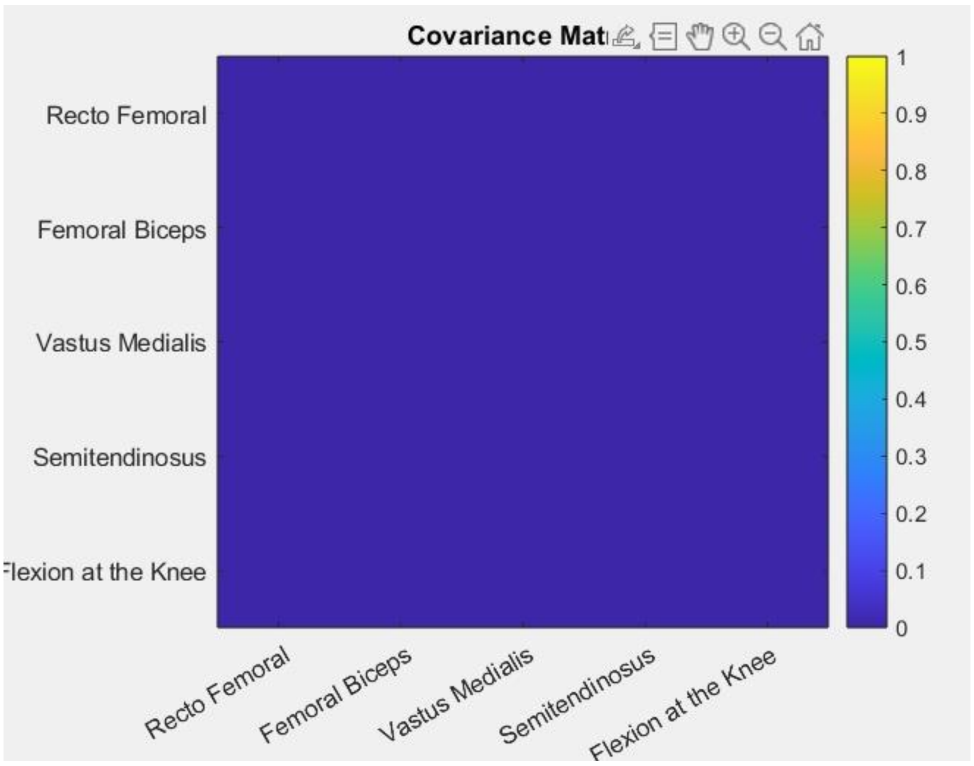


The temperature represents the sum of the variances of the variables. Femoral Biceps Femoris Semitendinosus Knee Flexion Heat Map:- Color: The heat map uses color gradients to represent different parameters: The scale on the right (color bar) ranges from 0 to 16 × 10 ×. Covariance values of variables on the y-axis. Cells represent different forces. Blue means these two variables have a lower difference. This information is used to understand the differences between variables in the data set. Visualizations provide an intuitive way to see how different variables in the data change together. Color gradients help quickly identify strong and weak relationships and facilitate deeper data analysis and interpretation.

2) PIE

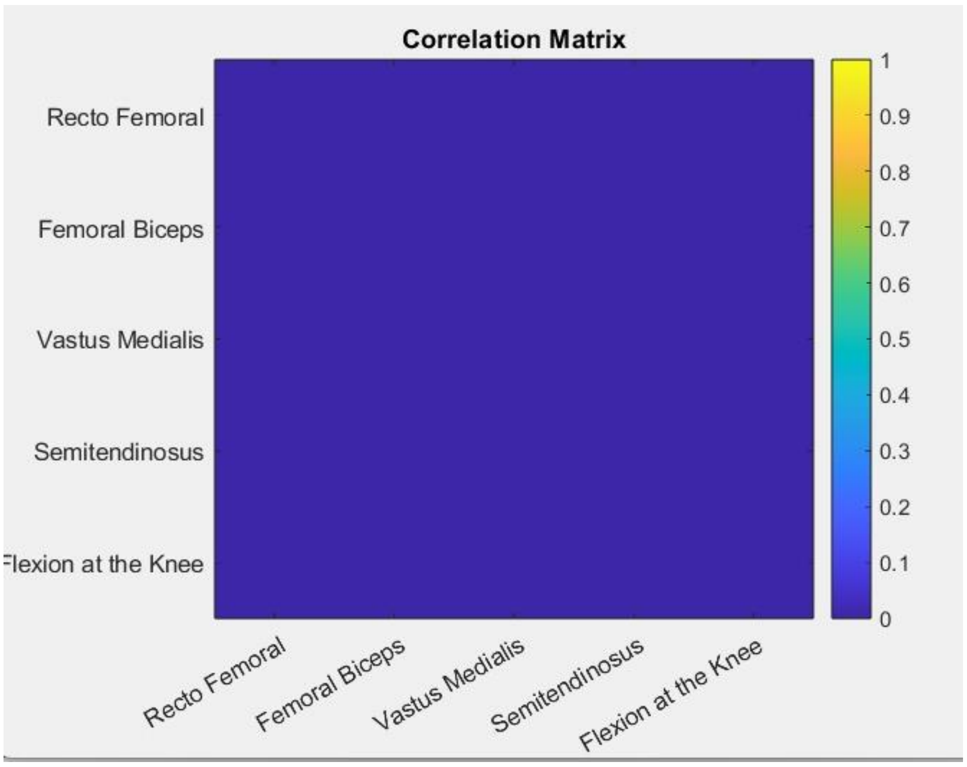


This diagram is another visualization, this time of the visualization of the relationship matrix. However, unlike the previous correlation matrix heat map, this graph shows little correlation between all variables. The detailed description is as follows: Heat Map Overview Title: The title at the top is “Correlation Matrix”, indicating that the heat map represents the correlation coefficients of the variables. Biceps Femoral Medial Semitendo Knee Flexion Heatmap Color: The heatmap uses a color gradient to represent the value of the correlation coefficient: The scale (color bar) on the right ranges from 0 to 1. The value is close to 1. The right angle (from left to right) should show all the differences relative to itself, which is always 1. Low. A low correlation may indicate a data problem, such as incorrect prior data or errors in calculating the correlation. It helps analyze the relationship between a pair of variables in data that do not have a linear relationship. Analyzing relationships such as univariate correlation or multivariate analysis. Uniformly low rates require checking data and calculations to ensure accuracy.



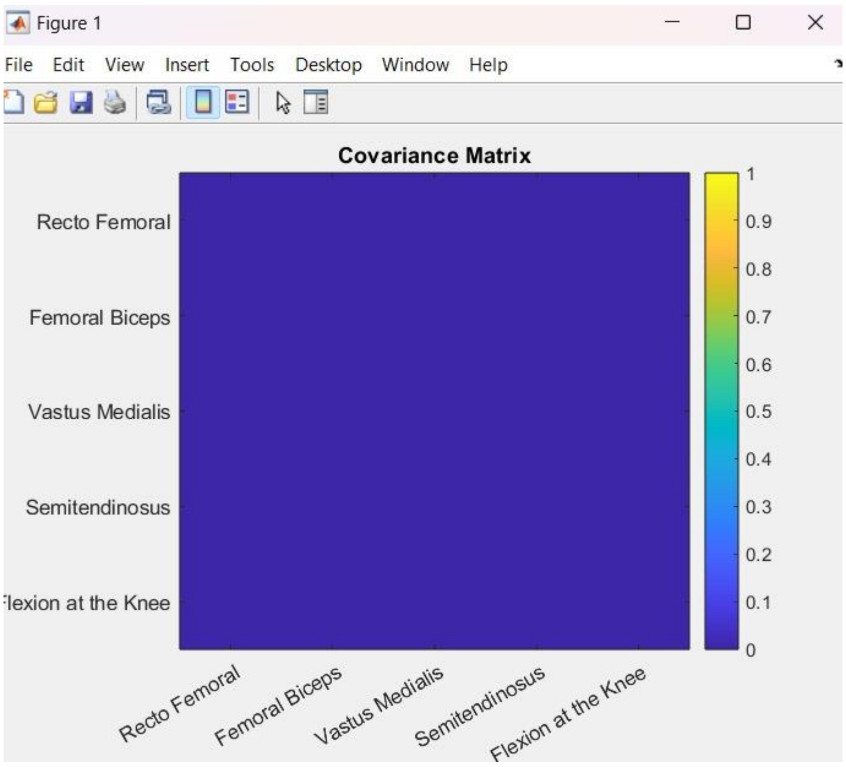
A covariance matrix is a square matrix that gives the difference between each pair of elements of a given set of variables. Difference between differences. Analysis: Thermal image is usually dark blue; This indicates that the difference between movement (knee flexion) with each muscle pair mentioned is very small or close to zero. Variation: Muscle movements and muscular movements are different from each other. This means that the activity of one muscle group does not predict the activity of another muscle group or knee flexion. Rehabilitation and Physical Therapy: Can be used to create specific exercises that target isolated muscles. The heat map gives a visual representation of the difference between different muscles and movements (knee flexion), showing that these differences have little difference, they are independent of each other in this regard.

3)SEN



A correlation matrix is a table that shows the correlation coefficients between variables. Each cell in the table shows the relationship between two variables.

Colour representation: Dark blue: Indicates a low value close to 0. The correlation between muscle and movement (knee flexion) is very low or close to zero. This means that the activity of one muscle group does not predict the activity of another muscle group or knee flexion. br>Rehabilitation and Physical Therapy: Can be used to create specific exercises that target isolated muscles. The heat map provides a visual representation of the relationship between different muscle groups and movements (knee flexion), showing that these variables are low-correlated, thus independent of each other. This is useful for identifying muscle coordination and creating training or rehabilitation plans.



Right Color bar ranging from 0 to 1 indicates the size of the variable value. It seems that the difference between these variables is small, which means that the correlation between these variables in the data set is small. The fact that the difference value is almost zero indicates that the difference is not consistent.

Normalization

Normalization scales the data to a standard range, usually between 0 and 1. This process is essential because it ensures that all features contribute equally to the model, avoiding differences due to biased scales. For example, if one feature ranges from 0 to 1 while another ranges from 0 to 1000, the model may disproportionately be biased towards the larger-scale feature. In this process, the minimum and maximum values of each feature in the data set are calculated. Then, the normalization formula(value - minimum) / (maximum - minimum), for each value in the dataset, that scales the data to the range [0, 1].The identical scaling is applied for training and test data to match.

Random Sampling

By selecting a subset of data from a larger dataset in a method that provides each potential sample an equal chance of being selected. it involves reducing the amount of data. simply reduces bias in the selected sample.

Discrete Wavelet Transform

Daubechies 4 (db4) Wavelet: db4 wavelets have been used in this process. Daubechies wavelets are a family of orthogonal wavelets characterized by a contraction stimulus and a defined decay time. This particular wavelet, db4 has a decay time of 4, which means it can represent polynomials up to degree 3.

Its process includes:

a. Decomposition:

Apply high-pass and low-pass filters are to the original signal to obtain the approximation (A1) and detail (D1) coefficients. Then the coefficients are down sampled and the data size is reduced by a factor of 2.

The obtained approximation coefficients (A1) are decomposed into new approximation (A2) and detail (D2) coefficients, and filters and down sampling is done to obtain D2.

Repeat the process of estimating the coefficients (A2) obtained in the previous step to obtain the coefficients A3 and D3, and use filters and down sampling to obtain D3.

b. Reconstruction:

After decomposition, the signal can be reconstructed from the approximation (An) and detail (Dn) coefficients using the Inverse Discrete Wavelet Transform (IDWT).

Start with the highest level of detail coefficients (D3), reconstruct the approximation at that level (A3), then proceed iteratively to reconstruct the signal at lower levels until reaching the original signal.

Here, db4, L3 DWT is used to extract features from signals with a high degree of time-frequency localization, making it suitable for classification. DWT with db4 wavelet can efficiently compress signals while preserving essential information, and here in biomedical signal analysis, db4, L3 DWT helps in extracting relevant features from EMG (Electromyography) signals for disease diagnosis.

Sliding Window Function:

The sliding window function is a preprocessing step used to create sequences of fixed length from the input data for CNN training. Here's how it works:

Input Parameters:

data: The input dataset containing sequential data.

Window size: The desired length of the sliding window.

stride: The number of data points to move the window forward.

Process:

The function initializes empty lists to store windowed data and corresponding labels. It iterates over the input data, moving the sliding window with a specified stride. For each iteration, it extracts a window of window size length from the input data.

The windowed data is appended to the windowed data list.

The label is extracted from the last sample in the window and appended to the labels list. This process continues until all possible windows are extracted from the input data.

CONVOLUTIONAL NEURAL NETWORK (CNN)

Convolutional Neural Networks (CNNs) are a type of deep neural network specifically designed for processing structured grid-like data, such as images. They have revolutionized the field of computer vision and are widely used in various applications, including image classification, object detection, and image segmentation. Let's dive into the CNN architecture and their underlying processes:

CNN Architecture:

1. Convolutional Layers:

Convolutional layers are the core building blocks of CNNs. They consist of filters (also known as kernels) that slide over the input data and perform element-wise multiplication with the local regions, followed by summation.

This operation generates feature maps that capture spatial hierarchies of patterns in the input data.

Multiple filters are applied in parallel to extract diverse features from the input, leading to the creation of multiple feature maps.

Pooling Layers:

Pooling layers are used to down sample the feature maps obtained from convolutional layers. They help reduce the spatial dimensions of the feature maps while retaining the most salient information.

Common pooling operations include max pooling and average pooling, where the maximum or average value within each pooling window is retained, respectively.

Activation Function:

Activation functions introduce non-linearity into the CNN model, enabling it to learn complex patterns and relationships in the data.

ReLU, sigmoid, and tanh are generally used in CNNs. ReLU is widely preferred due to its simplicity and effectiveness in mitigating the vanishing gradient problem.

Fully Connected Layers:

Fully connected layers are traditional neural network layers where each neuron is connected to every neuron in the previous layer.

They serve as the final layers of the CNN architecture and are responsible for making predictions based on the extracted features.

Typically, fully connected layers are followed by a softmax activation function in classification tasks to obtain probability distributions over output classes.

Output Layer:

The output layer of a CNN depends on the specific task being performed. For classification tasks, it typically consists of one or more neurons representing different classes, with softmax activation for multi-class classification. Then loss functions are applied. Common loss functions for classification tasks include categorical cross-entropy and binary cross-entropy, while popular optimizers include SGD, Adam, and RMSprop.

For tasks such as object detection or segmentation, the output layer may involve additional processing layers specific to the task.

CNN Process:

Data Preprocessing:

Normalization, to scale the input data to a similar range[0,1] and DWT to decompose the signals into different frequency components, that help in extracting relevant information is applied on all the input data(EMG signals), after handling all the missing values.

Model Construction:

The CNN architecture is defined by stacking convolutional, pooling, activation, and fully connected layers based on the problem requirements.

The input layer has a shape of (200,1). The convolutional layers apply filters to the data to detect features.

Here, two convolutional layers are used, Layer 1 with 32 filters, kernel size 3 using ReLU (Rectified Linear Unit) activation and layer 2 with 64 filters, kernel size 3 using ReLU activation. Pooling layers down sample the feature maps from the convolutional layers, reducing the computational load and capturing the most important features. The pooling layer has the maximum pooling with pool size 2.

The flattening layer converts the 2D matrix of these convolutional layers into a 1D vector. It has 2 fully connected layers, dense layer 1 with 128 neurons, ReLU activation, and dense layer 2 with 64 neurons, ReLU activation. The dropout rate is set to 0.5 to prevent overfitting by randomly deactivating a certain percentage of neurons during training. Softmax activation is applied for the output layer for multi-class classification, corresponding to the two cases.

Model Compilation:

Once constructed, the CNN model is compiled by the categorical cross-entrophy loss function which is suitable for multi-class classification. It measures the dissimilarity between the predicted probabilities and the actual labels. Adam optimizer is applied for optimization for efficient gradient descent, as it adapts the learning rate for each parameter individually, leading to faster convergence and better performance. The learning rate is set to 0.001 and batch size to 32, with 100 epochs. The validation loss is monitored with a patience of 10 epochs to prevent overfitting. This early stopping is applied to halt training if the validation loss stops improveing after a certain number of epochs.

Model Training:

Training involves feeding the labeled training data into the CNN model and adjusting its parameters (weights and biases) iteratively to minimize the loss function.

Model Evaluation:

After training, the CNN model is evaluated using a separate set of labeled test data to assess its performance on unseen examples. To evaluate the model, accuracy, precision, recall and F1-score are taken into consideration. These metrics provide insights into the model’s ability to correctly classify instances across different classes.

LINEAR REGRESSION:

Linear regression is a statistical method used to model the relationship between a dependent variable and one or more independent variables by fitting a linear equation to observed data.

1)Data Loading and Preparation:

a) Two datasets, 'Dataset1.xlsx' and 'Dataset3test.xlsx', are loaded from Excel files.

b) 'Dataset1.xlsx' is used for training, and 'Dataset3test.xlsx' is used for testing.

c) The training dataset consists of 250 samples with multiple features and corresponding labels. These features and labels are separated for both training and test datasets.

2. Initialization:

a) Two random matrices, 'B' and 'A', are initialized.

b) 'B' is used to transform the input features into a higher-dimensional space.

3. Feature Transformation and Mapping:

a) A cosine transformation is applied to the inner product of the input features with the random matrix 'B', creating a feature map. This feature map represents a new higher-dimensional representation of the input data, capturing complex relationships.

4. Kernel Matrix Construction:

a) A composite kernel matrix 'K' is constructed, capturing interactions between the transformed input features and their gradients

$$K = (A \cdot A' + 1)^3$$

5. Parameter Estimation:

a) The parameter 'alpha' is estimated using the Moore-Penrose pseudoinverse ('pinv') of the kernel matrix `K`.

$$\alpha = \text{pinv}(K) \cdot Y$$

6. Prediction:

a) Predictions are made for the test data using the trained linear regression model.

$$\text{label} = \text{round}((A \cdot A' + 1)^3 \cdot \alpha)$$

b) The prediction of the test data is done by the last equation from which the fact of how well the model is trained will be known.

$$y\text{output} = \text{round}((X1 \cdot A' + 1)^3 \cdot \alpha)$$

c) The predicted labels for the test data (`youtput`) are computed using the same polynomial kernel method applied during training.

NEURAL TANGENT KERNAL

1. The Neural Tangent Kernel (NTK) framework provides a way to analyze and optimize neural networks by approximating their behavior using kernel methods. NTK is particularly useful in the infinite-width limit, where the behavior of the network becomes more predictable and linear. This framework helps us understand how neural networks learn and how their parameters change during training.

2. The procedure starts by loading two datasets from Excel files. One dataset is used for training the model, and the other is used for testing. The training dataset consists of 250 samples, each with multiple features (input data) and corresponding labels (output data). The test dataset is similarly structured but is used to evaluate the performance of the trained model. The features (inputs) and labels (outputs) are separated for both the training and test datasets. This separation allows us to use the features to train the model and later predict the labels on the test data.

3. We initialize two random matrices, B and A where B is used to transform the input features into a higher-dimensional space.

4. A is a weight vector that will be used to combine this transformed feature to produce an initial output. This random initialization is crucial because it allows us to start with a diverse set of parameters, which will be fine-tuned during training. We compute the inner product of the input features with the random matrix B. This step projects the input features into a new space. A cosine transformation is applied to the inner product, creating a feature map. This non-linear transformation helps capture complex relationships in the data that a linear model might miss. The feature map is essentially a new representation of the input data in a higher-dimensional space, which will be used for training the model. Using the feature map, we calculate an initial output by multiplying it with the weight vector A. This initial output serves as a baseline prediction before the model parameters are adjusted through training.

5. We construct a composite kernel matrix, k1, which captures the interactions between the transformed input features and their gradients. This matrix includes both the feature map and additional features derived from the gradients of the input features.

6. The purpose of constructing this kernel matrix is to model the training dynamics more effectively, taking into account how changes in the input features affect the output.
7. We use the pseudoinverse of the kernel matrix k1 to calculate the weights that adjust the initial random weights.

8. This step involves solving a least squares problem to minimize the error between the predicted and actual labels. The new weights are a combination of the initial random weights and the adjustment weights, resulting in a more accurate model. The updated weights are used to make predictions on the test data. The test input features are transformed using the updated matrix B, and the feature map is created using the same cosine transformation.

9. The predictions are calculated by multiplying the feature map with the updated weight vector A. These predictions are then scaled, rounded, and converted into binary values to indicate the presence (1) or absence (0) of muscular paralysis.

Equations

$$k1_i = \left[k(i,:), -\sin(\mathrm{IP}_m(i,:) + \tfrac{\pi}{\mathrm{rows}}) \cdot A \cdot X(i,1), \ldots, -\sin(\mathrm{IP}_m(i,:) + \tfrac{\pi}{\mathrm{rows}}) \cdot A \cdot X(i,d-1)\right]$$

The Neural Tangent Kernel (NTK) provides a way to understand the behavior of neural networks by approximating their output using kernel methods. The NTK is defined as:

$$\Theta(x,x') = \nabla\theta f\theta(x)\top\nabla\theta f\theta(x')$$

- fθ is the neural network function with parameters θ.
- x and x' are input data points.
- $\nabla\theta f\theta(x)$ is the gradient of the network output for its parameters for input xx.

In the infinite-width limit, the NTK remains constant during training, allowing the network's training dynamics to be modeled as a linear

Random Initialization of Parameters:

We initialize random matrices BB and AA

$$B \in R^{(d-1)\times n}$$

$$A \in R^{(1\times n)}$$

where d is the number of features in the input data and n is the number of neurons in the hidden layer.

Inner Product and Feature Mapping:

We compute the inner product of the input features and the random matrix

$$\mathrm{IP}_m = X \cdot B$$

Then, we apply a cosine transformation to create the feature map.

$$k = \cos(\mathrm{IP}m + \pi/\mathrm{rows})$$

here rows are the number of training samples.

The initial output is calculated ,

$$\mathrm{output}_0 = k \cdot A^{\top}$$

Kernel Matrix Construction for Training Data:

For each training sample ii, we construct a composite kernel matrix $k1k1$ that captures the interactions between different input features and their gradients

This results in a composite kernel matrix $k1$ for all training samples.

Calculate Weights Using Pseudoinverse:

We calculate the adjustment weights using the pseudoinverse of k1:

$$w1 = \mathrm{pinv}(k1) \cdot (Y - \mathrm{output}_0)$$

We combine the initial random weights into a single vector

$$w0 = [A, B(1,:), B(2,:), \ldots, B(d-1,:)]^\top$$

The updated weights $w$are obtained by adding $w1$ to w0:

$$w = w1 + w0$$

The weights are extracted and updated for the transformed future space.

$$A1 = w(1:n)$$
$$B1 =$$
$$\left[w(n+1:2n)^\top; w(2n+1:3n)^\top; \ldots; w((d-1)n+1:dn)^\top\right]$$

Prediction Calculation:

inner product is computed for the test data using the updated transformation matrix B1:

$$\mathrm{IP}_m = X1 \cdot B1$$

The feature map is created for the test data:

$$k = \cos(\mathrm{IP}_m + \tfrac{\pi}{\mathrm{rows}})$$

The output is calculated:

$$youtput = \frac{k \cdot A1}{1.0 \times 10^8}$$

Where 1.0×10^8 the is chosen based on the nature of the output and to make the output as the integer.

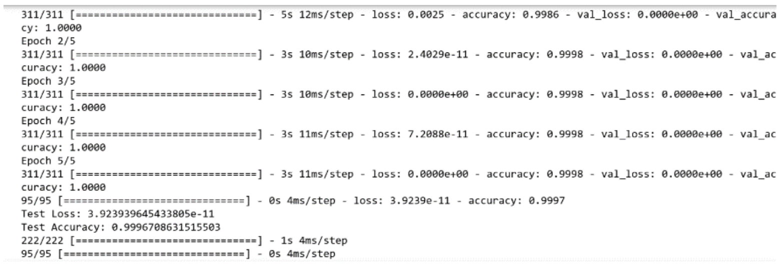
The raw output is rounded and converted to binary values:

$$youtput = \mathrm{round}(youtput)$$
$$\mathrm{binary\_output} = youtput \geq 0$$

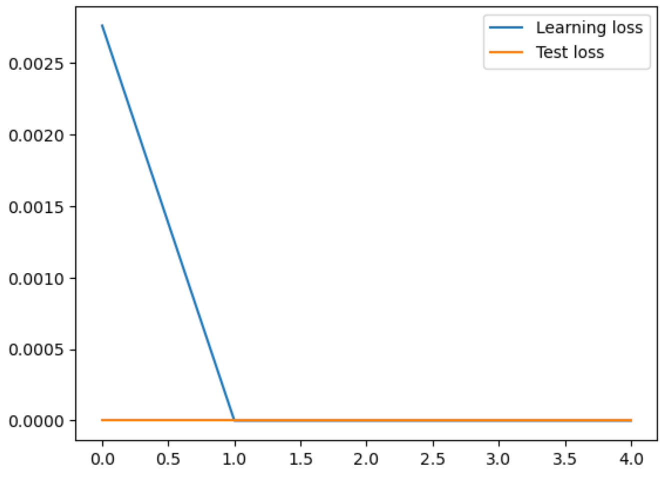
RESULTS AND DISCUSSION

Convolutional Neural Networks

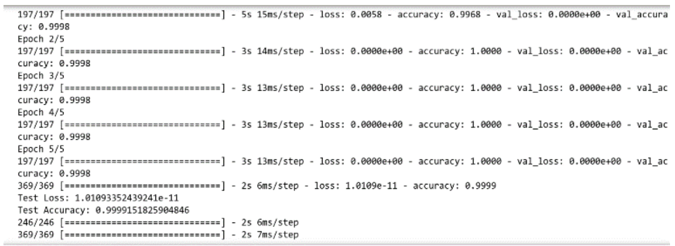
Mar



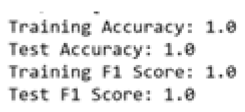
```
311/311 [==============================] - 5s 12ms/step - loss: 0.0025 - accuracy: 0.9986 - val_loss: 0.0000e+00 - val_accura
cy: 1.0000
Epoch 2/5
311/311 [==============================] - 3s 10ms/step - loss: 2.4029e-11 - accuracy: 0.9998 - val_loss: 0.0000e+00 - val_ac
curacy: 1.0000
Epoch 3/5
311/311 [==============================] - 3s 10ms/step - loss: 0.0000e+00 - accuracy: 0.9998 - val_loss: 0.0000e+00 - val_ac
curacy: 1.0000
Epoch 4/5
311/311 [==============================] - 3s 11ms/step - loss: 7.2088e-11 - accuracy: 0.9998 - val_loss: 0.0000e+00 - val_ac
curacy: 1.0000
Epoch 5/5
311/311 [==============================] - 3s 11ms/step - loss: 0.0000e+00 - accuracy: 0.9998 - val_loss: 0.0000e+00 - val_ac
curacy: 1.0000
95/95 [==============================] - 0s 4ms/step - loss: 3.9239e-11 - accuracy: 0.9997
Test Loss: 3.923939645433805e-11
Test Accuracy: 0.9996708631515503
222/222 [==============================] - 1s 4ms/step
95/95 [==============================] - 0s 4ms/step
```



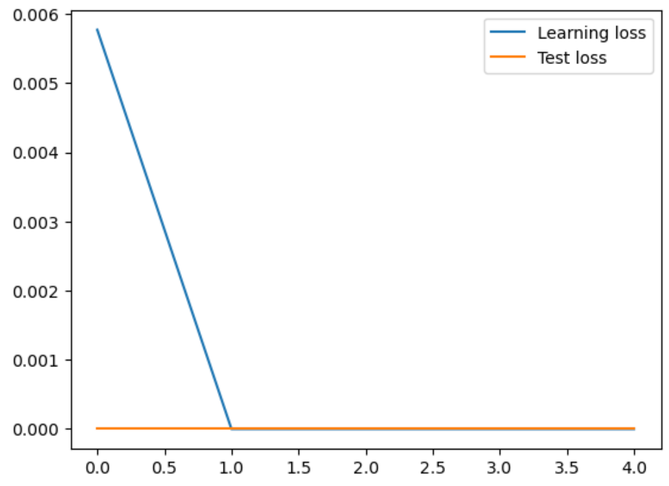
Sen



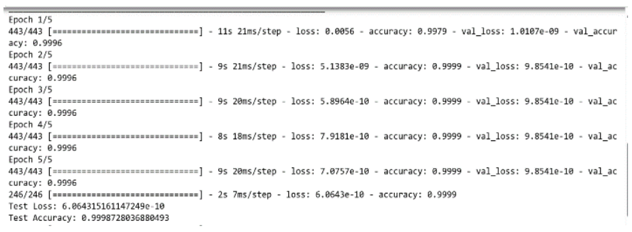
```
197/197 [==============================] - 5s 15ms/step - loss: 0.0058 - accuracy: 0.9968 - val_loss: 0.0000e+00 - val_accura
cy: 0.9998
Epoch 2/5
197/197 [==============================] - 3s 14ms/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - val_ac
curacy: 0.9998
Epoch 3/5
197/197 [==============================] - 3s 13ms/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - val_ac
curacy: 0.9998
Epoch 4/5
197/197 [==============================] - 3s 13ms/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - val_ac
curacy: 0.9998
Epoch 5/5
197/197 [==============================] - 3s 13ms/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - val_ac
curacy: 0.9998
369/369 [==============================] - 2s 6ms/step - loss: 1.0109e-11 - accuracy: 0.9999
Test Loss: 1.01093352439241e-11
Test Accuracy: 0.9999151825904846
246/246 [==============================] - 2s 6ms/step
369/369 [==============================] - 2s 7ms/step
```



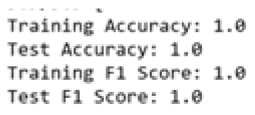
```
Training Accuracy: 1.0
Test Accuracy: 1.0
Training F1 Score: 1.0
Test F1 Score: 1.0
```



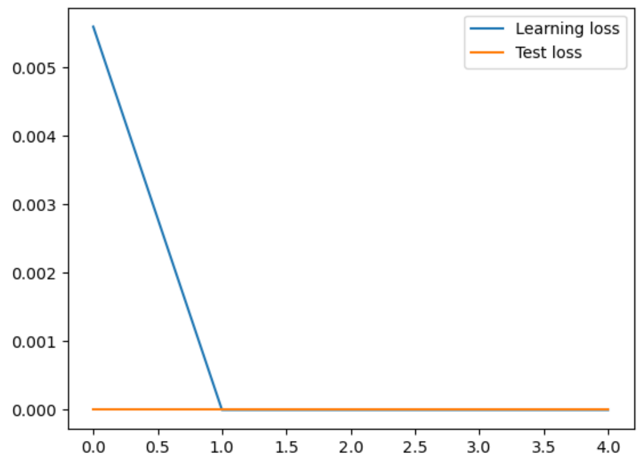
Pie:



```
Epoch 1/5
443/443 [==============================] - 11s 21ms/step - loss: 0.0056 - accuracy: 0.9979 - val_loss: 1.0107e-09 - val_accur
acy: 0.9996
Epoch 2/5
443/443 [==============================] - 9s 21ms/step - loss: 5.1383e-09 - accuracy: 0.9999 - val_loss: 9.8541e-10 - val_ac
curacy: 0.9996
Epoch 3/5
443/443 [==============================] - 9s 20ms/step - loss: 5.8964e-10 - accuracy: 0.9999 - val_loss: 9.8541e-10 - val_ac
curacy: 0.9996
Epoch 4/5
443/443 [==============================] - 8s 18ms/step - loss: 7.9181e-10 - accuracy: 0.9999 - val_loss: 9.8541e-10 - val_ac
curacy: 0.9996
Epoch 5/5
443/443 [==============================] - 9s 20ms/step - loss: 7.0757e-10 - accuracy: 0.9999 - val_loss: 9.8541e-10 - val_ac
curacy: 0.9996
246/246 [==============================] - 2s 7ms/step - loss: 6.0643e-10 - accuracy: 0.9999
Test Loss: 6.064315161147249e-10
Test Accuracy: 0.9998728036880493
```

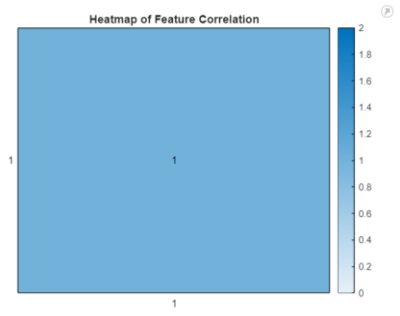


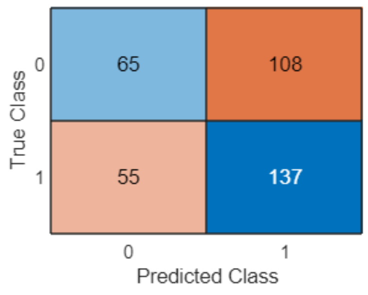
```
Training Accuracy: 1.0
Test Accuracy: 1.0
Training F1 Score: 1.0
Test F1 Score: 1.0
```



KERNAL REGRESSION:

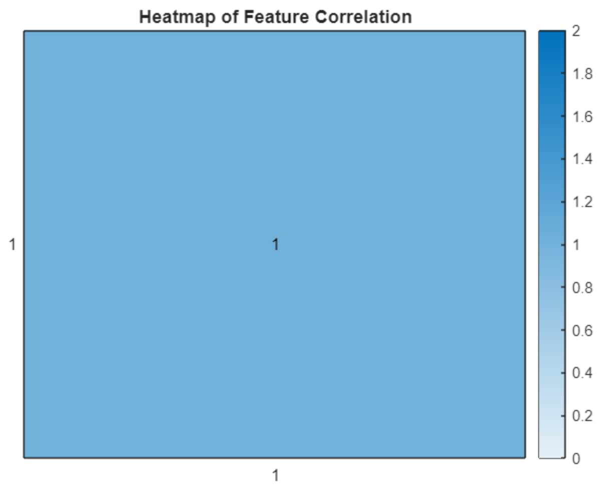
Pie:

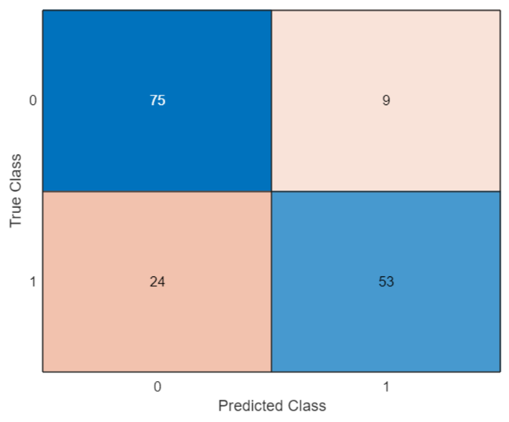




```
Mean Absolute Error (MAE): 0.446575

Mean Squared Error (MSE): 0.446575

Root Mean Squared Error (RMSE): 0.668263

Accuracy: 55.3425%
```

Sen:

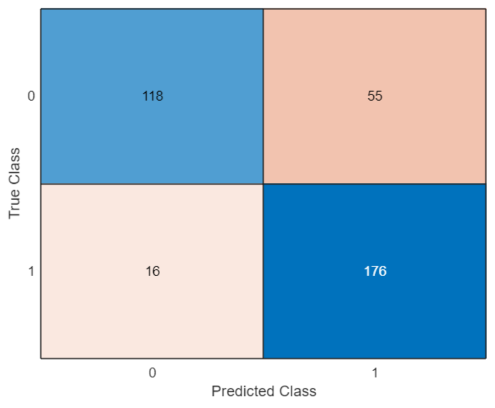


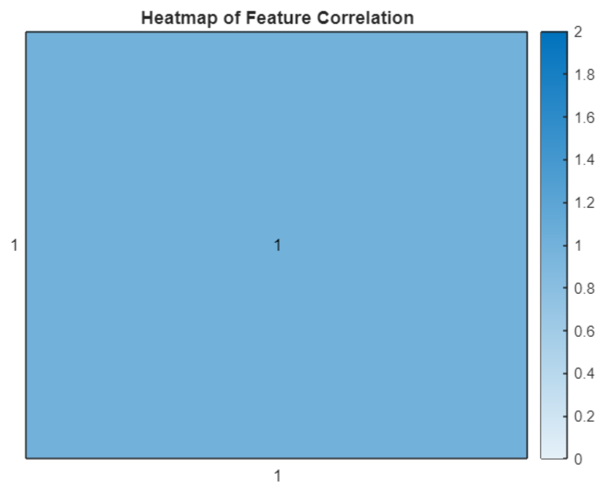


```
Mean Absolute Error (MAE): 0.194521

Mean Squared Error (MSE): 0.194521

Root Mean Squared Error (RMSE): 0.441045

Accuracy: 80.5479%
```

Mar:

```
Mean Absolute Error (MAE): 0.204969

Mean Squared Error (MSE): 0.204969

Root Mean Squared Error (RMSE): 0.452735

Accuracy: 79.5031%
```





Neural Tangent Kernel

Pie

```
Mean Absolute Error (MAE): 0.400000

Mean Squared Error (MSE): 0.400000

Root Mean Squared Error (RMSE): 0.632456
```

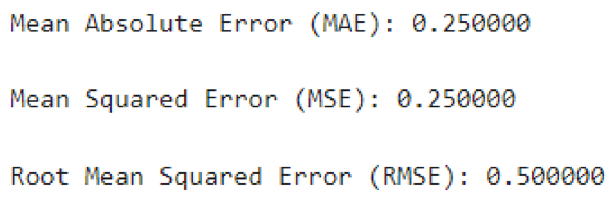
```
Accuracy: 60%
```

Mar

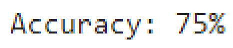
```
Mean Absolute Error (MAE): 0.400000

Mean Squared Error (MSE): 0.400000

Root Mean Squared Error (RMSE): 0.632456
```

```
Accuracy: 60%
```

Sen



```
Mean Absolute Error (MAE): 0.250000

Mean Squared Error (MSE): 0.250000

Root Mean Squared Error (RMSE): 0.500000
```



```
Accuracy: 75%
```

CONCLUSION

As machine learning is so effective, it has been included into the current disease diagnosing technique. For the purpose of predicting the disease of muscular paralysis, CNN method turned out to be the most efficient. kernel Regression and NTK have lower accuracy than CNN . In this case, CNN- is utilized as a classifier, adapting to the data to be classified. Utilizing the UCI EMG-Lower Limb Dataset, the suggested method’s performance is assessed.

REFERENCES

[1] Sanchez,Oscar and Sotelo,Jose. (2014). EMG dataset in Lower Limb. UCI Machine Learning Repository. https://doi.org/10.24432/C5ZW3P..

[2] Grassin-Delyle S, Roquencourt C, Moine P, Saffroy G, Carn S,Heming N, Fleuriet J, Salvator H, Naline E, Couderc LJ, Devillier,, Metabolomics of exhaled breath in critically ill COVID-19, patients: a pilot study., P EBioMedicine 63:103154, 2020.

[3] Subramani, P., K, S., B, K.R. et al. Prediction of muscular paralysis disease based on hybrid feature extraction with machine learning technique for COVID-19 and post-COVID-19 patients. Pers Ubiquit Comput 27, 831–844, https://doi.org/10.1007/s0077, 2023.

[4] A. KamalM, Assessment and characterisation of post-COVID-19 manifestations., Int J Clin n of post-COVID-19 manifestations. Int J Clin, 2020.

[5] Kehri V, Techniques of EMG signal analysis and classification of neuromuscular diseases, Paris: International Conference on Communication and Signal Processing 2016 (ICCASP 2016). Atlantis Press, 2016.

[6] Li X, Jahanmiri-Nezhad F, Rymer WZ, Zhou P, An examination of the motor unit number index (MUNIX) in muscles paralyzed by spinal injury., y. IEEE Trans Inf Technol Biomed 16(6): 1143-1149, 2012.

[7] S. IM, Dystonia–new advances in classification, genetics, pathophysiology and treatment, Acta Neurol Scand 129:13-19, 2014.

[8] H. M. Quartarone A, Emerging concepts in the physiological basis of dystonia., Mov Disord 28(7):958–967, 2013.

[9] Li P, Wang Y, Peppelenbosch MP, Ma Z, Pan Q, Systematically comparing COVID-19 with 2009 influenza pandemic for hospitalized patients., Int J Infect Dis 102:375–380, 2020.

[10] ÖF, A novel approach for SEMG signal classification with adap
ɱput 54(7):1137–1146, 2016.

[11] ChenxiH,XinH,YuF,JianfengX,YiQ,PengjunZ,LinF,HuaY,Yilu X, Jiahang L , Sample imbalance disease classification model based on association rule feature selection. Pattern Recogn, Lett 133:280–286, 2020.

[[12] Miljković N, Popović N, Djordjević O, Konstantinović L, Šekara TB, ECG artifact cancellation in surface EMG signals by fractional order calculus application., Comput Methods ProgBiomed 140:259–264, 2017.

[13] Li P, Wang Y, Peppelenbosch MP, Ma Z, Pan Q, Systematically comparing COVID-19 with 2009 influenza pandemic for hospitalized patients., Int J Infect Dis 102:375–380, 2020.

[14] P. E. H. L. Spanias JA, Detection of and compensation for EMG disturbances for powered lower limb prosthesis control., IEEE Transac Neural Syst Rehab Eng 24(2):226-234, 2015.