BREACTOR
Software Design Document

Abdelrahaman Shahata , Ahmed Hany, Doaa Hamid,
Nader El-sheikh
Supervised by: Dr. Sarah Nabil
Teaching Assistant : Eng. Menna Gamil

February 12, 2018

1 Introduction

Most of Drivers can be affected by many abnormal behaviors while they are driving, which can cause accidents. As Driver can drive for too many hours, which It can make the driver feel sleepy or carry a mobile phone that leads to have a lack of focus in the road and the surrounded environment can cause the driver or passenger death. We searched about a solution to this problem to avoid or reduce the amount of accidents and loss of lives in accidents and we found that the brain signals of driver is the best way to detect state of driver like the driver is feeling drowsy or not , by reading driver's brain signals , The System can compare between normal behavior and drowsy behavior and send alerts to warn the driver to take an action to concentrate on the road. We found that the best processing technique that can be used to recognize mental states from EEG signals in brain computer interface. The EEG signal reflects activation of head musculature, eye movement, interference from nearby electrical devices and changing conductivity in electrodes due to movement. The Main focus of our system is to read the EEG signal and get rid of noise from the signal (pre-processing) and then classify signal and detect the abnormal behavior by Support Vector Machine (SVM) Algorithm and we detect and begin to send warning messages to driver to stop or to take care of road.

1.1 Purpose

The Purpose of Software Design Document is to provide full description for BREACTOR, This document show most of functionality and system design and system architecture used for BREACTOR and how BREACTOR is interacting with user.

1.2 Scope

1.2.1 Goal

BREACTOR's Goal is to detect the drowsiness of the driver or abnormal behavior.

1.2.2 Objective

BREACTOR's Objective is to save drivers life's as to avoid accident happening. BREACTOR focus on detecting brain signals of driver as it's the best way to detect drowsiness of driver while driving , as we found that most of solutions of this problem like camera detecting will not detect the drowsiness 100 percent. On the other hand , detecting the brain signals will solve this problem easily.

1.3 Overview

Some extra details about our project is interacting with Driver as in the setup phase for our program their will be some questions for user to take a survey about what user prefer the way to pay attention to traffic and road if system detect that he/she is getting asleep or feeling stressed system will allow user to choose from options as to play music or giving some instructions.

1.4 Definitions and Acronyms

Term	Definition
Support Vector Machine (SVM)	Algorithm Used for classifying the signals.
K Nearest Neighbour (KNN)	Algorithm used for classifying the signals.

2 System Overview

First step as in the Figure 1. the emotive must be ready to work and start reading signals, After that the signals will be filtered from noise, to extract the specific signals, then we use extracted signals to classify in which class drowsiness or awake where we will use some algorithm to classify the signals . Finally the signal will be saved and checked, if signal Classification result is drowsiness the system will alert the driver and repeat the whole process every one minute on the other hand if the signal is normal the process will be repeated every 5 minutes.

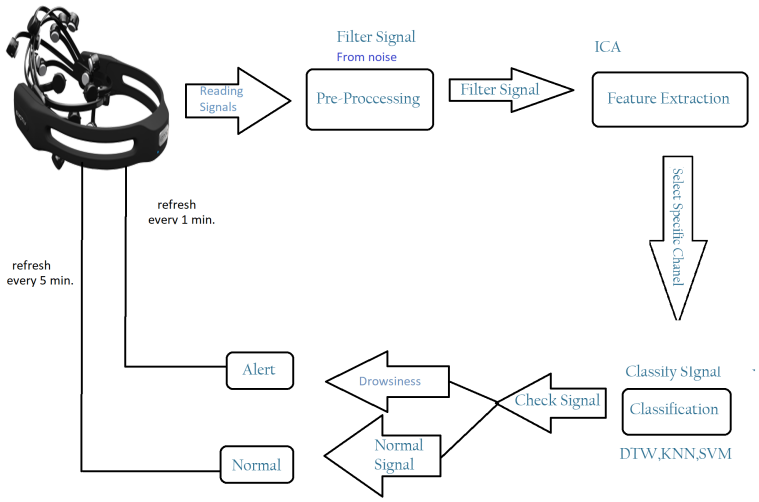
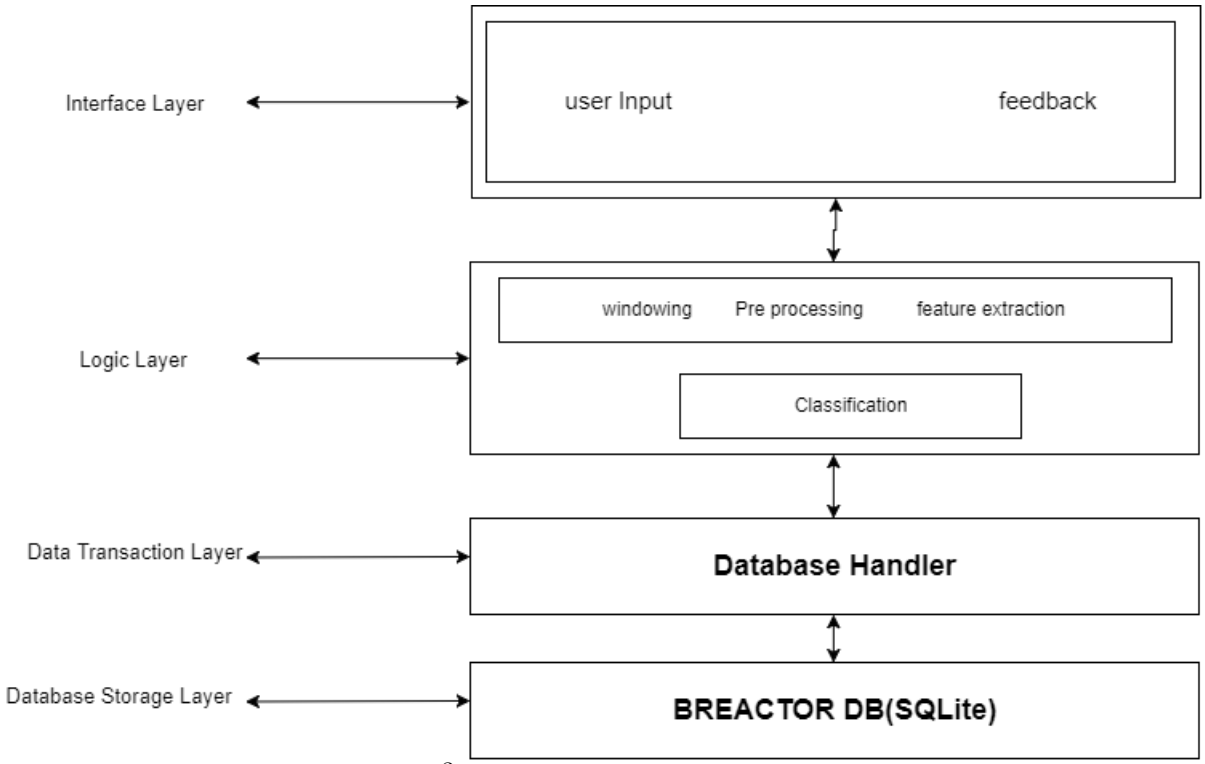


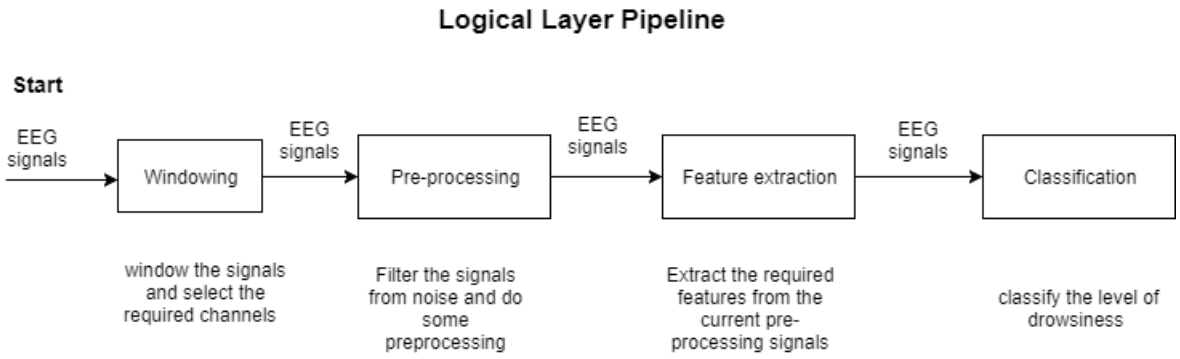
Figure 1: BREACTOR's Overview Diagram

3 System Architecture

3.1 Architectural Diagram

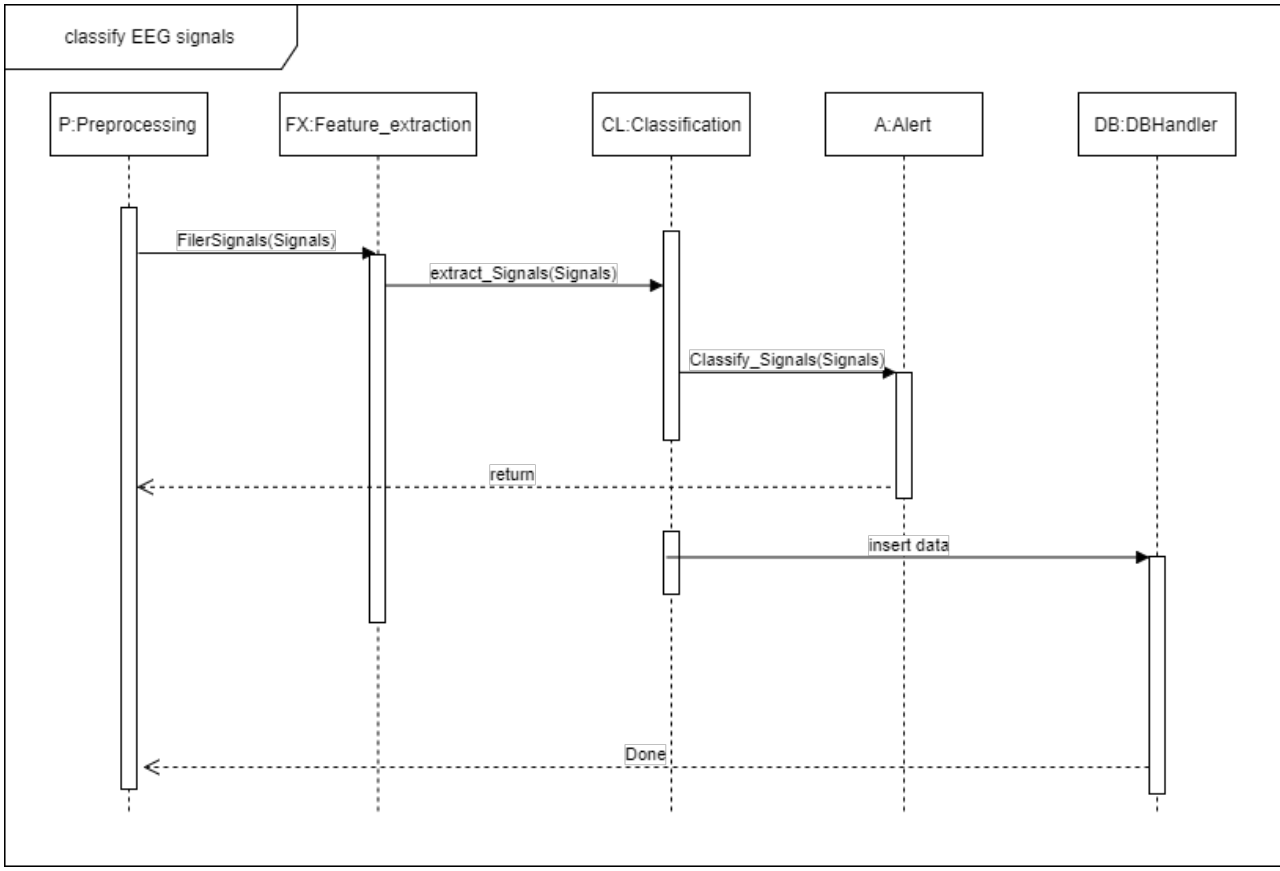


3.2 Logic Layer Pipeline Diagram

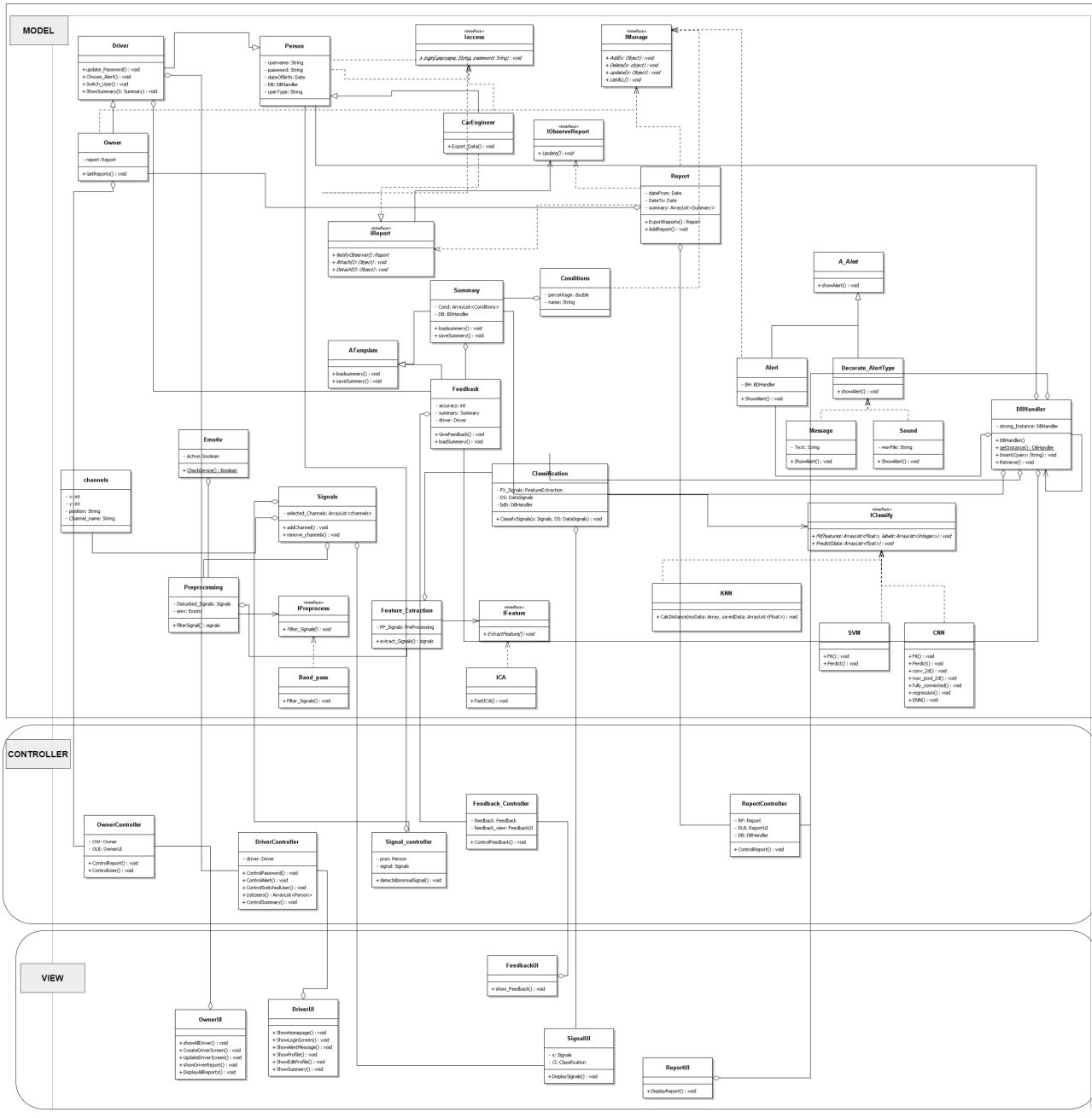


3.3 Decomposition Description

3.3.1 Sequence Diagram



3.3.2 Class Diagram



Interface layer	
Component	**Description**
User Input	This handle the user data input from the brain.
feedback	This is BREACTOR feedback that is given to the user .

Logic layer (Top)	
Component	**Description**
Windowing	Handle the cutting of exact needed parts from the signals.
Pre-processing	Handle the noise filtering and frequency band selection from the window signals.
Feature extraction	Handle the extraction of required data for the next layer.

Logic layer (Bottom)	
Component	**Description**
Database Handler	Handle insert and select from the database.

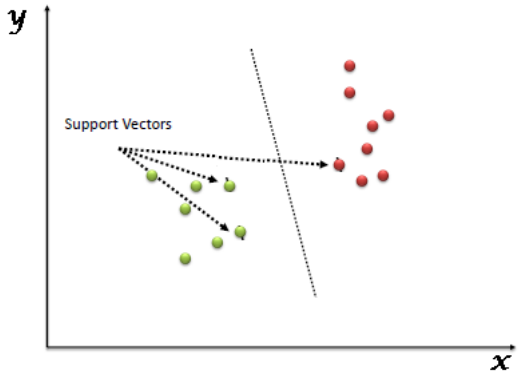
Data Transaction layer	
Component	**Description**
BREACTOR DB	The systems actual database .

3.4 Design Rationale

Layered architecture separate user interface. Logic layer, data Transaction and data storage which gives the system the flexibility, Maintainability and scalability. It allow further work in the development phase and makes it reusable. On the other side it have some disadvantages which are the complexity to applications and decrease performance little bit.

Some algorithms parts that is mentioned to be used in similar problems as:

Support vector machine (SVM): its one of the most common used supervised algorithm due to its capability to manage large data and it is used for regression and classification. The algorithm use a kernel trick to transform data into high dimension space. We used binary SVM to identify either awake or sleep.
it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well (look at the below snapshot).



Support Vectors are simply the co-ordinates of individual observation. Support Vector Machine is a frontier which best segregates the two classes (hyper-plane/ line).

K nearest neighbors (KNN): a very simple algorithm that store all scenarios and predict the class based on the distance functions and the k value. Its also a supervised algorithm.
KNN is an approach to data classification that estimates how likely a data point is to be a member of one group or the other depending on what group the data points nearest to it are in.
The k-nearest-neighbor is an example of a "lazy learner" algorithm, meaning that it does not build a model using the training set until a query of the data set is performed.

Naive Bayes (NB): its an efficient probability classifier and independent algorithm.it is also used in some papers to classify drowsiness.
The Naive Bayes Classifier technique is based on the so-called Bayesian theorem and is particularly suited when the dimensional of the inputs is high. Despite its simplicity, Naive Bayes can often outperform more sophisticated classification methods.

Discrete Wavelet Transform (DWT): an algorithm that is used in feature extraction to make a decomposition of EEG signals into sub-bands.

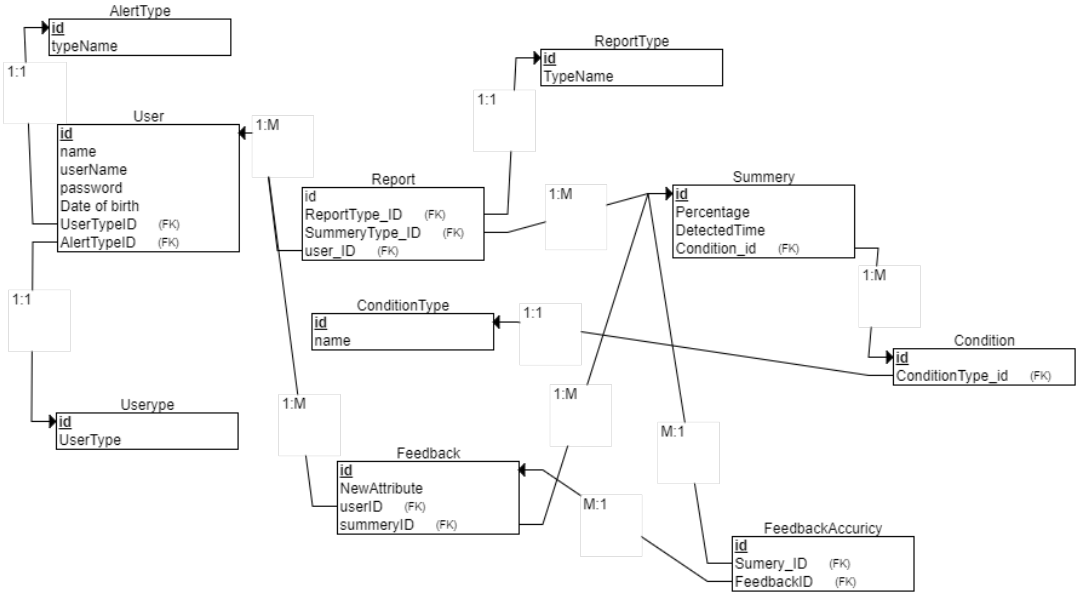
Dynamic Time Wrapping (DTW): an algorithm that is widely used in EEG to measure between two sequences of data and detect the best match. It also has its own optimization and weight functions.

Convolutional Neural Network (CNN): As we know the neural network is an assembly of several artificial neurons that produce nonlinear boundaries. Neural networks require a mid to high-rig machine to handle this using GPU for real-time classifications. It takes the input with some biases, create pool with feature maps (which is also called hidden layer) and repeat this step with minimum of 2 hidden layers then its all converted into a fully layer then the output.

Each created hidden layer becomes the next input (Activation function). The second step is to compare output to intended output (The cost or loss function). After that, an optimization function is used to minimize cost. Finally, each epoch is created by adding the feed forward to the back propagation (goes backward and manipulate the weights).
Independent Component Analysis (ICA): it is a statistical algorithm that cut the complex data into parts. It can separate the needed signals and channels from the unneeded and filter the signals.

4 Data Design

4.1 Database Design



Explain how the information domain of your system is transformed into data structures. Describe how the major data or system entities are stored, processed and organized. List any databases or data storage items.

4.2 Database Description

4.2.1 Table User :

User	
Attribute	**Description**
ID	Serialization for Users in System .
Name	Name of User
User Name	user's name to be unique in the system
Password	to identify that it's user's account
Date of Birth	to differs between ages of users
User Type ID	to identify type of user (Owner or Driver)
Alert Type ID	to identify type of alert used by user

4.2.2 Table Condition :

Condition	
Attribute	**Description**
ID	Serialization for condition types in System .
Condition Type ID	getting id of condition types

4.2.3 Table Summary :

Summary	
Attribute	**Description**
ID	Serialization for Summaries in System .
Percentage	Detected Percentage of Drowsiness
Detected Time	time of detection
Condition ID	Condition type of this percentage

4.2.4 Table Report :

Report	
Attribute	**Description**
ID	Serialization for Report in System .
Report Type ID	Specifying type of Report
Summary Type ID	Getting Summary of user
User ID	to identify that it's user's account

Alphabetically list the system entities or major data along with their types and descriptions. If you provided a functional description in Section 3.2, list al the functions and function parameters. If you provided an OO description, list the objects and its attributes, methods and method parameters.

5 Component Design

5.1 Biosppy (PRE-PROCESSING)

First, we can find the pre-processing stage that changed Signals as we use biosppy which is a toolbox for biosignal processing, it supports many types of biosignals like EEG , EMG , ECG.

5.1.1 Parameters

Signal (Array) , Sampling-rate (int , float , optional) , show (bool , optional)

5.1.2 Returns

TS (Array) Signal Time in seconds , filtered (Array) filtered signals ,Theta (Array) , Alpha Low (Array) , Alpha High (Array) , Beta (Array) , Gamma (Array).

5.2 DTW (FEATURE EXTRACTION)

FastDTW algorithm uses recursive approach to determine the warping path between two time series.

5.2.1 Parameters

It takes a radius parameter with temporal sequences as input, which is used to apply a so called radius window. Radius window used to restrict the calculation for warping path in cost matrix
x query time series
y template time series
dist distance function received form distance function module by passing the distance name
radius integer value implements so called radius window
pattern name of the step pattern, see step patterns section for names
normalized set to true to get normalized warping distance

5.3 SVM (CLASSIFICATION)

The implementation is based on libsvm. The fit time complexity is more than quadratic with the number of samples which makes it hard to scale to dataset

with more than a couple of 10000 samples.
The multiclass support is handled according to a one-vs-one scheme.
For details on the precise mathematical formulation of the provided kernel functions and how gamma, coef0 and degree affect each other, see the corresponding section in the narrative documentation.

5.3.1 Parameters predict(X)

Perform classification on samples in X.
For an one-class model, +1 or -1 is returned.

X : array-like, sparse matrix, shape ($n_samples, n_features$)
$For kernel = precomputed, the expected shape of X is [n_samples_test, n_samples_train]$

5.3.2 Returns:

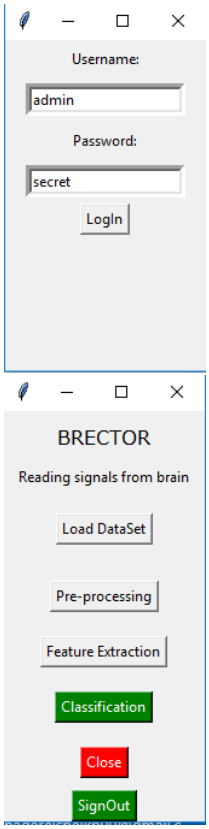
$y_pred : array, shape(n_samples,)$
$Class labels for samples in X.$

6 Humnan Interface Design

6.1 Overview of User Interface

The user can create his own account where he can choose the type of alert to wake him up. The driver only needs to wear the device to see his drowsiness level, Then a feedback will be displayed to the user by his drowsiness level in the last trip also the driver can give his feedback about his drowsiness level in the last trip. Moreover the alert will be repeated for example if the driver is extremely tiered it will be repeated every 30 seconds but if the driver is tiered it will be repeated every 5 minutes

6.2 Screen Images



Display screenshots showing the interface from the user s perspective. These can be handdrawn or you can use an automated drawing tool. Just make them as accurate as possible. (Graph paper works wel .)

6.3 Screen Objects and Actions

A discussion of screen objects and actions associated with those objects.

7 Requirements Matrix

Provide a cross reference that traces components and data structures to the requirements in your SRS document. Use a tabular format to show which system components satisfy each of the functional requirements from the SRS. Refer to the functional requirements by the numbers/codes that you gave them in the SRS.

8 APPENDICES

This section is optional. Appendices may be included, either directly or by reference, to provide supporting details that could aid in the understanding of the Software Design Document.

9 References

dissertationbib