AM5023- PHYSIOLOGICAL MEASUREMENTS AND INSTRUMENTATION LABORATORY

HUMAN IN LOOP CYBER PHYSICAL SYSTEMS - LABORATORY REPORT

Submitted by: DINESH KUMAR M

Registration no: AM23M022

DEPARTMENT OF APPLIED MECHANICS & BIOMEDICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY, MADRAS

USE OF IMU SENSORS TO CLASSIFY BOXING PUNCH

AIM
Automatic classification of boxing punches to help athletes improving their punch performance.

OBJECTIVE
- To record the IMU signal and use machine learning model to classify boxing punch

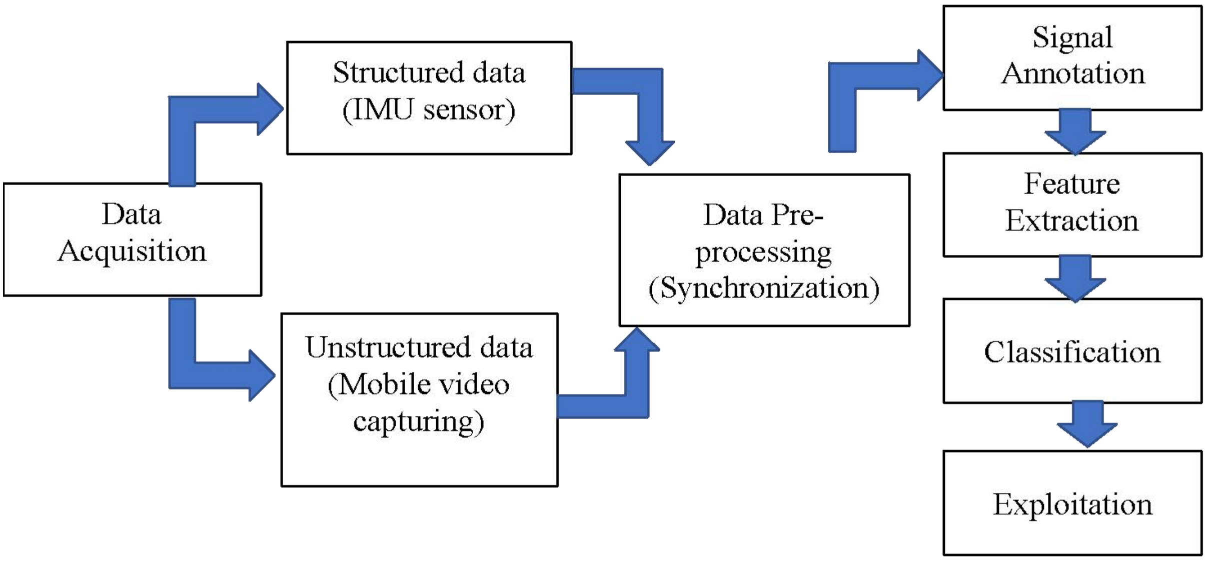
APPARATUS REQUIRED
- MMRL – METAMOTIONRL IMU Sensor
- Metabase mobile application (Data acquisition)
- Computer system with programming language
- Velcro wrist strap
- Sampling Frequency=200Hz

THEORY
Boxing is a physically demanding combat sport. Boxers rely on a combination of strength, coordination, speed, and stamina to succeed in impacting the opponent while evading an adversary's punches. A successful performance requires the ability to deliver precise punches above the belt, to the head or the torso, without being punched back. There are four main types attacking techniques using punches: Jab, cross, hook and uppercut. From punch classification, we get a lot of useful information about their effectiveness, repeatability, strength and individual profile.

Wearable inertial sensors are fast becoming a validated technology to provide data for athlete performance analysis in a range of sports. The advancement in technology resulted in relatively low-cost wearable, unobtrusive inertial sensors and global positioning system (GPS) units and are more readily available for coaching teams. Machine learning and artificial intelligence (AI) techniques are then used to process output data from these sensors. Such techniques consist of classification models to predict boxer's performance. An abundance of scientific literature and commercialized technology have used signals obtained by wearable inertial sensors to identify activity type and intensity in sport and general living.

The monitoring of acceleration or magnitude of sporting motions is often realized by wearing inertial measurement units (IMUs) on the wrist/hand and shank/foot. As a faster, more reliable and cost-efficient strategy for activity and motion analysis, wearable IMUs benefit a lot from the significant reduction in sensor volume and price and are playing a more and more important role in the field of sports analytics. Commonly, an IMU consists of an accelerometer for measuring linear acceleration, a gyroscope for angular acceleration and sometimes a magnetometer for a magnetic field. Moreover, three-dimensional sensors are favorable due to their ability to capture parameters along the three axes and provide detailed and useful component data for orientation and kinematics studies. The combination of multifarious sensors ensures the system robustness and accuracy of captured data and then improves the validity and reliability of activity detection and analysis.

FLOW CHART



METHOD

- Wrap the IMU sensor on the wrist to acquire data.
- Integrating both IMU sensor and Metabase mobile application through Bluetooth connectivity.
- Configure the IMU sensor (200 Hz sampling frequency, ±16 g accelerometer, ±2000 deg/s gyroscope)
- Record the signals like accelerometer (x,y,z axis) and gyroscope(x,y,z axis). Simultaneously record the video through mobile camera.
- Wireless transmission of csv file from IMU sensor to PC for processing.
- Synchronized both acquired signal and captured video.
- Extracting temporal or spectral or spectrotemporal features.
- Classify the punches using machine learning model.

RESULTS

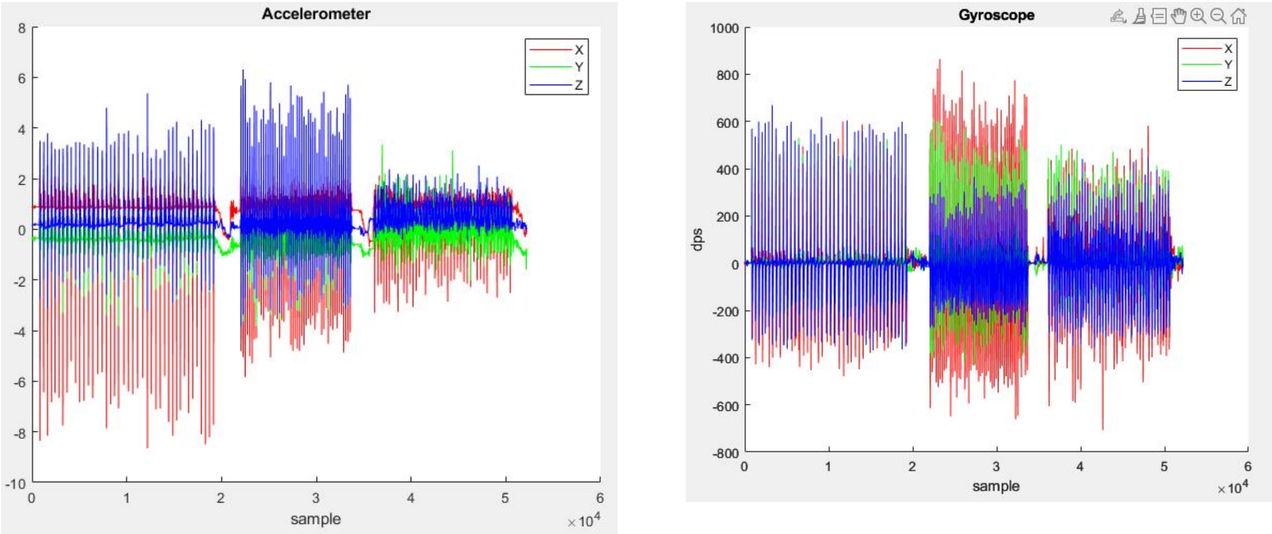


Fig 1: Jab punch signal

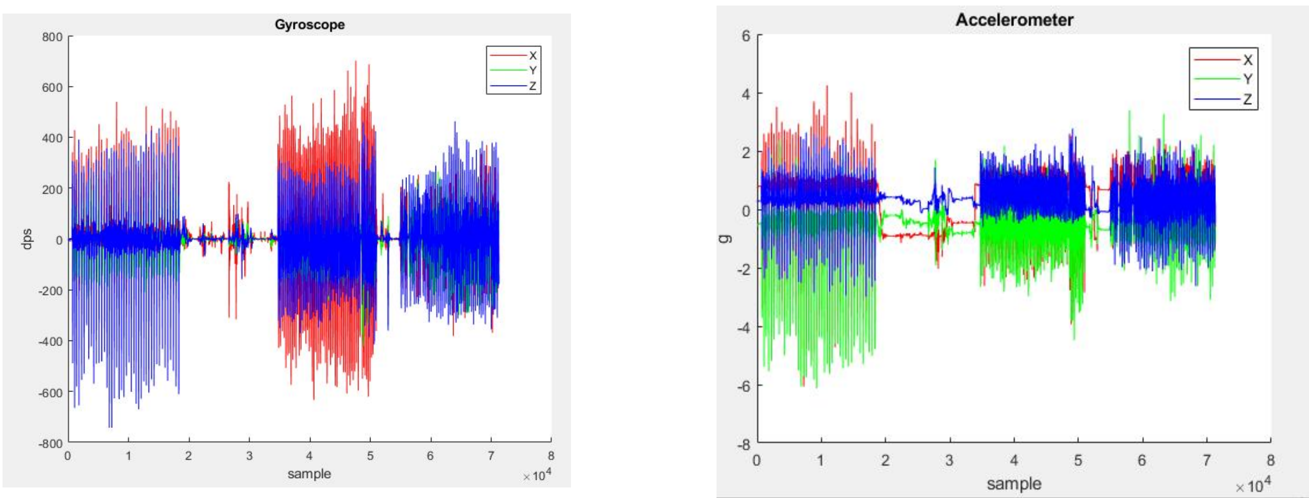


Fig 2: Cross punch signal

MACHINE LEARNING MODEL: SVM KERNEL

Class 0 = No punch

Class 1 = Jab

Class 2 = Hook

Learning Model:

In [1]:
```
import pandas as pd
import numpy as np
from sklearn.svm import SVC
from sklearn.metrics import classification_report, confusion_matrix
import matplotlib.pyplot as plt
get_ipython().run_line_magic('matplotlib', 'inline')
```

In [2]:
```
irisdata = pd.read_csv('F:\\IITM\\Physiological measurements lab\\boxing imu\\feattimepyth.csv')
```

In [3]:
```
from sklearn.model_selection import train_test_split
X = irisdata.drop('class', axis=1)
y = irisdata['class']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20)
```

In [4]:
```
kernels = ['Polynomial', 'RBF', 'Sigmoid','Linear']#A function which returns the corresponding SVC model
def getClassifier(ktype):
    if ktype == 0:
        # Polynomial kernal
        return SVC(kernel='poly', degree=8, gamma="auto")
    elif ktype == 1:
        # Radial Basis Function kernal
        return SVC(kernel='rbf', gamma="auto")
    elif ktype == 2:
        # Sigmoid kernal
        return SVC(kernel='sigmoid', gamma="auto")
    elif ktype == 3:
        # Linear kernal
        return SVC(kernel='linear', gamma="auto")


for i in range(4):
    # Separate data into test and training sets
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20)# Train a SVC model using different kernal
    svclassifier = getClassifier(i)
    svclassifier.fit(X_train, y_train)# Make prediction
    y_pred = svclassifier.predict(X_test)# Evaluate our model
    print("Evaluation:", kernels[i], "kernel")
    print(classification_report(y_test,y_pred))
```

```
Evaluation: Polynomial kernel
              precision    recall  f1-score   support

           0       1.00      0.97      0.98        30
           1       1.00      1.00      1.00        23
           2       0.97      1.00      0.98        32

    accuracy                           0.99        85
   macro avg       0.99      0.99      0.99        85
weighted avg       0.99      0.99      0.99        85

Evaluation: RBF kernel
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        22
           1       0.00      0.00      0.00        32
           2       0.36      1.00      0.53        31

    accuracy                           0.36        85
   macro avg       0.12      0.33      0.18        85
weighted avg       0.13      0.36      0.19        85

Evaluation: Sigmoid kernel
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        24
           1       0.00      0.00      0.00        28
           2       0.39      1.00      0.56        33

    accuracy                           0.39        85
   macro avg       0.13      0.33      0.19        85
weighted avg       0.15      0.39      0.22        85

Evaluation: Linear kernel
              precision    recall  f1-score   support

           0       0.94      0.97      0.95        31
           1       1.00      1.00      1.00        18
           2       0.97      0.94      0.96        36

    accuracy                           0.96        85
   macro avg       0.97      0.97      0.97        85
weighted avg       0.97      0.96      0.96        85
```

In [5]:

```
from sklearn.model_selection import GridSearchCV


param_grid = {'C': [0.1,1, 10, 100], 'gamma': [1,0.1,0.01,0.001],'kernel': ['rbf', 'poly', 'sigmoid']}

grid = GridSearchCV(SVC(),param_grid,refit=True,verbose=2)
grid.fit(X_train,y_train)


print(grid.best_estimator_)


grid_predictions = grid.predict(X_test)
print(confusion_matrix(y_test,grid_predictions))
print(classification_report(y_test,grid_predictions))

plt.hist(grid_predictions)
plt.show()
```

```
SVC(C=0.1, gamma=1, kernel='poly')
[[30  0  1]
 [ 0 18  0]
 [ 1  0 35]]
              precision    recall  f1-score   support

           0       0.97      0.97      0.97        31
           1       1.00      1.00      1.00        18
           2       0.97      0.97      0.97        36

    accuracy                           0.98        85
   macro avg       0.98      0.98      0.98        85
weighted avg       0.98      0.98      0.98        85
```

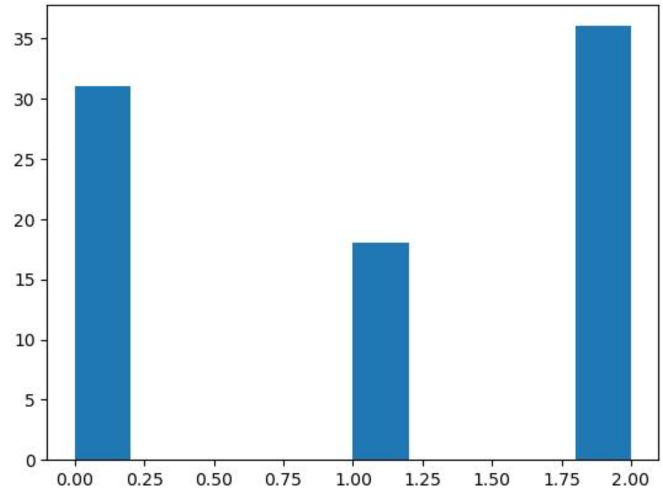


Figure: Histogram

In [6]:
```
xx = pd.read_csv("F:\\IITM\Physiological measurements lab\\boxing imu\\Testingtimepyth.csv")
yy = pd.read_csv("F:\\IITM\Physiological measurements lab\\boxing imu\Testing outputpyth.csv")
grid_predictions1 = grid.predict(xx)
print(confusion_matrix(yy,grid_predictions1))
print(classification_report(yy,grid_predictions1))
print(grid_predictions1)
print(yy)

plt.hist(grid_predictions1)
plt.show()


xx1 = pd.read_csv("F:\\IITM\Physiological measurements lab\\boxing imu\\Testingtimepyth.csv")
yy1 = pd.read_csv("F:\\IITM\Physiological measurements lab\\boxing imu\\Testing outputpyth.csv")
grid_predictions2 = grid.predict(xx1)
print(confusion_matrix(yy1,grid_predictions2))
print(classification_report(yy1,grid_predictions2))
print(grid_predictions2)
```

```
[[18  2  4]
 [ 0 12  0]
 [ 2  0 10]]
              precision    recall  f1-score   support

           0       0.90      0.75      0.82        24
           1       0.86      1.00      0.92        12
           2       0.71      0.83      0.77        12

    accuracy                           0.83        48
   macro avg       0.82      0.86      0.84        48
weighted avg       0.84      0.83      0.83        48
```

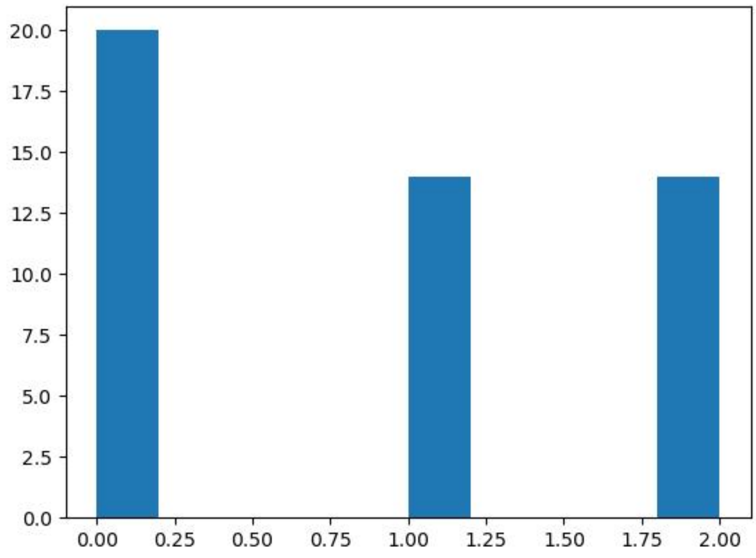


Figure: Histogram

HYPERPARAMETERS

Kernel = 'poly'
C = 0.1
Gamma = 1

CONCLUSION

The results indicate that the machine learning models can classify boxer punches with a high degree of accuracy. Therefore, automatic punch classification can be used to evaluate and automatically label complex pad-work combinations executed in training, bypassing cumbersome notional analysis and video labelling. There is potential for the punch types to be paired with other biofeedback metrics such as punch impact acceleration, velocity, approach angle, torso angular velocity, and rotation angle, calories burnt and fatigue level.