A

Project Report

on

HEART RATE ESTIMATION USING PHOTOPLETHYSMOGRAPHY BY NON-INVASIVE TECHNIQUES

Submitted for partial fulfillment of the requirements for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

By

M Drithi (2451-18-733-084)
K Sai Jaideep Patel (2451-18-733-098)
K Sathvika (2451-18-733-116)

Under the guidance of

Mrs. N. SABITHA
Assistant Professor
Department of CSE

M.V.S.R. ENGINEERING COLLEGE
Department of Computer Science and Engineering
(Affiliated to Osmania University & Recognized by AICTE)
Nadergul, Saroor Nagar Mandal, Hyderabad – 501 510
Academic Year: 2021-22

CERTIFICATE

*This is to certify that this Project Seminar Report entitled "***HEART RATE ESTIMATION USING PHOTOPLETHYSMOGRAPHY BY NON-INVASIVE TECHNIQUES***" is a bonafide work carried out by* ***Ms.*** **M Drithi (2451-18-733-084),**
Mr. **K Sai Jaideep Patel (2451-18-733-098),** ***and Ms.*** **K Sathvika (2451-18-733-116)** *in partial fulfillment of the requirements for the award of the degree of* ***Bachelor of Engineering*** *in* ***Computer Science And Engineering*** *from* ***Maturi Venkata Subba Rao (MVSR) Engineering College,*** *affiliated to OSMANIA UNIVERSITY, Hyderabad, during the Academic Year 2021-22 under our guidance and supervision.*

The results embodied in this report have not been submitted to any other university or institute for the award of any degree or diploma.

Internal Guide	**Head of the Department**
N.Sabitha	J.Prasanna Kumar
Assistant Professor	Professor & Head
Department of CSE	Department of CSE
MVSREC.	MVSREC.

I

DECLARATION

This is to certify that the work reported in the present project entitled "**HEART RATE ESTIMATION USING PHOTOPLETHYSMOGRAPHY BY NON-INVASIVE TECHNIQUES**" is a record of bonafide work done by us in the Department of Computer Science and Engineering, Maturi Venkata Subba Rao (MVSR) Engineering College, Osmania University during the Academic Years 2021-22. The reports are based on the project work done entirely by us and not copied from other sources. The results embodied in this project report have not been submitted to any other University or Institute for the award of any degree or diploma.

M Drithi (2451-18-733-084)

K Sai Jaideep Patel (2451-18-733-098)

K Sathvika (2451-18-733-116)

ACKNOWLEDGEMENT

We would like to express our sincere gratitude and indebtedness to our project guide **N. Sabitha** for her valuable suggestions and interest throughout the course of this project

We are also thankful to our principal **Dr. G. Kanaka Durga** and **Prof. J. Prasanna Kumar**,Professor and Head, Department of Computer Science and Engineering, MVSR Engineering College, Hyderabad for providing excellent infrastructure for completing this project successfully as a part of our B.E. Degree (CSE). We would like to thank our project coordinator **P. Subhashini** for their constant monitoring, guidance, and support. We convey our heartfelt thanks to the lab staff for allowing us to use the required equipment whenever needed.

Finally, we would like to take this opportunity to thank our parents for their support throughout the work. We sincerely acknowledge and thank all those who gave directly or indirectly their support in completion of this work.

M Drithi (2451-18-733-084)
K Sai Jaideep Patel (2451-18-733-098)
K Sathvika (2451-18-733-116)

VISION

- To impart technical education of the highest standards, producing competent and confident engineers with an ability to use computer science knowledge to solve societal problems.

MISSION

- To make the learning process exciting, stimulating, and interesting.
- To impart adequate fundamental knowledge and soft skills to students.
- To expose students to advanced computer technologies in order to excel in engineering practices by bringing out the creativity in students.
- To develop economically feasible and socially acceptable software.

PEOs, POs & PSOs

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

The Program Educational Objectives of the undergraduate program in Computer Science & Engineering are to prepare graduates who will:

PEO-1. Achieve recognition through demonstration of technical competence for successful execution of software projects to meet customer business objectives.

PEO-2. Practice lifelong learning by pursuing professional certifications, higher education, or research in the emerging areas of information processing and intelligent systems at the global level.

PEO-3. Contribute to society by understanding the impact of computing using a multidisciplinary and ethical approach

(A) PROGRAM OUTCOMES(POs)

At the end of the program the students (Engineering Graduates) will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using the first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis, and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply to reason informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and teamwork:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

IV

10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principle and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. **Lifelong learning:** Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change.

(B) PROGRAM SPECIFIC OUTCOMES (PSOs)

13. (PSO-1) Demonstrate competence to build effective solutions for computational real-world problems using software and hardware across multi-disciplinary domains.

14. (PSO-2) Adapt to current computing trends for meeting the industrial and societal needs through a holistic professional development leading to pioneering careers or entrepreneurship

COURSE OBJECTIVES AND OUTCOMES

Course objectives

- To enhance practical and professional skills.
- To familiarize tools and techniques of systematic literature survey and documentation
- To expose the students to industry practices and teamwork.
- To encourage students to work with innovative and entrepreneurial ideas

Course outcomes

CO1: Summarize the survey of the recent advancements to infer the problem statements with applications towards society
CO2: Design a software-based solution within the scope of the project.
CO3: Implement test and deploy using contemporary technologies and tools
CO4: Demonstrate qualities necessary for working in a team.
CO5: Generate a suitable technical document for the project.

ABSTRACT

Heart Rate is a key factor that indicates the health condition of a person and many long-term diseases can be identified with continuous monitoring of the Heart Rate. This continuous monitoring isn’t possible with traditional heart rate estimation techniques.

This has been made possible using wearables that use Photoplethysmography based techniques. PPG-based continuous heart rate monitoring is essential in several domains, e.g., for healthcare or fitness applications. PPG-based heart-rate estimations are becoming more and more widely used because of their ease of use over ECG techniques. Machine Learning PPG-based approaches have come into the picture and were found to perform better compared to the classical methods. Our end-to-end learning approach takes the synchronized time-frequency spectra of PPG and provides the estimated heart rate. Taking the raw PPG signals and synchronizing them is out of the scope of this model and will be added later. The model has been trained on the popular PPG-DaLiA dataset.

Keywords: Heart-rate, PPG, Deep Learning, CNN, Random Forests, Time-Frequency Spectrum, PPG-DaLiA

M Drithi (2451-18-733-084)

K Sai Jaideep Patel (2451-18-733-098)

K Sathvika (2451-18-733-116)

Index

LIST OF FIGURES

Chapters:

1. Introduction:

Heart rate is the speed of the heartbeat measured by the number of contractions (beats) of the heart per minute (bpm). The heart rate can vary according to the body's physical needs, including the need to absorb oxygen and excrete carbon dioxide, but is also modulated by a myriad of factors including but not limited to genetics, physical fitness, stress or psychological status, diet, drugs, hormonal status, environment, and disease/illness as well as the interaction between and among these factors. Hence heart rate is considered an important indicator of one’s health conditions and continuous monitoring of heart rate is of prime importance. This continuous heart rate monitoring has been made possible by wearables. Nowadays wearables are becoming an integral part of life for a large number of people. Electrocardiography (ECG) is a precise method of determining the heart rate but is cumbersome in daily life settings. Therefore, wrist-worn heart rate monitors have become widely used recently. Heart rate monitoring is based on photoplethysmography (PPG) in these devices. Various commercial products, such as the Apple Watch, the Fitbit Charge, or the Samsung Simband, include a PPG sensor nowadays.

1.1 Problem statement:

1. “Heart Rate Estimation using Photoplethysmography (PPG)”, understand various application domains and approaches.

2. Train required Machine Learning Models, considering various parameters and approaches for learning.

3. The trained model has to be tested and verified.

1.2 Objectives:

Photoplethysmography (PPG) is a simple, low-cost, non-invasive technology that uses a light source and a photodetector at the surface of the skin to measure the volumetric variations of blood circulation.
Photoplethysmography is an optically obtained signal that is measured with a pulse oximeter which illuminates the skin and measures using the intensities of transmitted and reflected light from the skin. A conventional pulse oximeter monitors the perfusion of blood to the dermis and

subcutaneous tissue of the skin.The most commonly measured value is the heart rate, although advanced applications also use other values, e.g., pulse irregularity, as well as biometric identification or analysis of accurate electrical signals that cause heart contraction, i.e., electrocardiography (ECG). Accurate ECG requires connecting electrodes to the patient's body in several different places, which is inconvenient for the patient, and it can be used only in certain situations. A much more convenient method is measuring the pulse on the wrist by using photoelectric methods. The skin of the wrist is irradiated with single or multicolor light, and then the reflected light is measured. The intensity of the reflected light depends on the absorption of the skin, which depends on the blood volume supplied to the tissues. In this way, the received signal contains information about the current blood supply to the vessels near the measuring device.

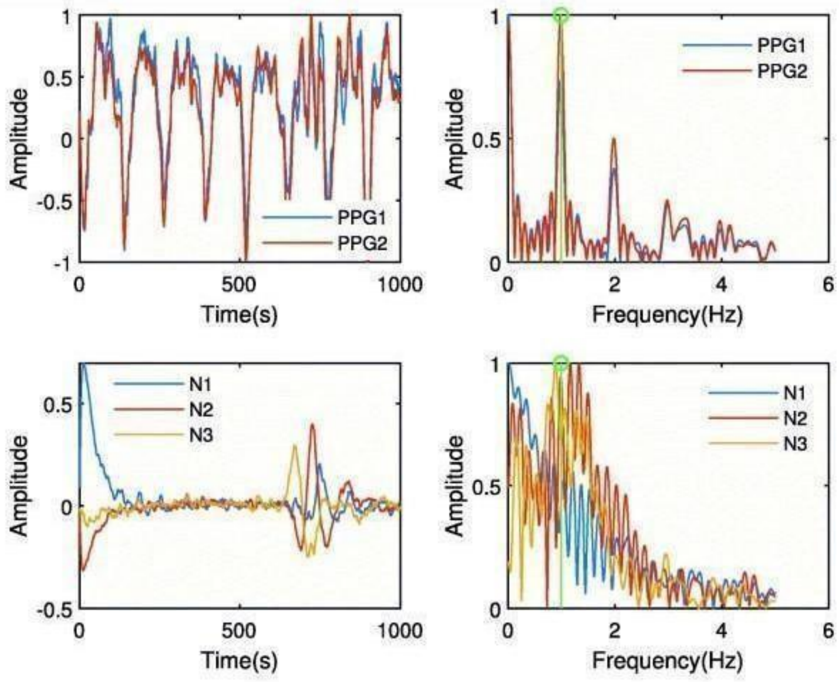
Unfortunately, PPG signals obtained from a moving person's wrist are weak, distorted, and contain noise. The noise level is often higher than a usable PPG signal. Correct analysis of a low-quality PPG signal is a very challenging task and can consume significant processing time, energy, and resources. We'll try to estimate Heart rate using synchronized PPG signals using various Machine Learning Algorithms on the PPG-DaLiA dataset.

1.3 Motivation:

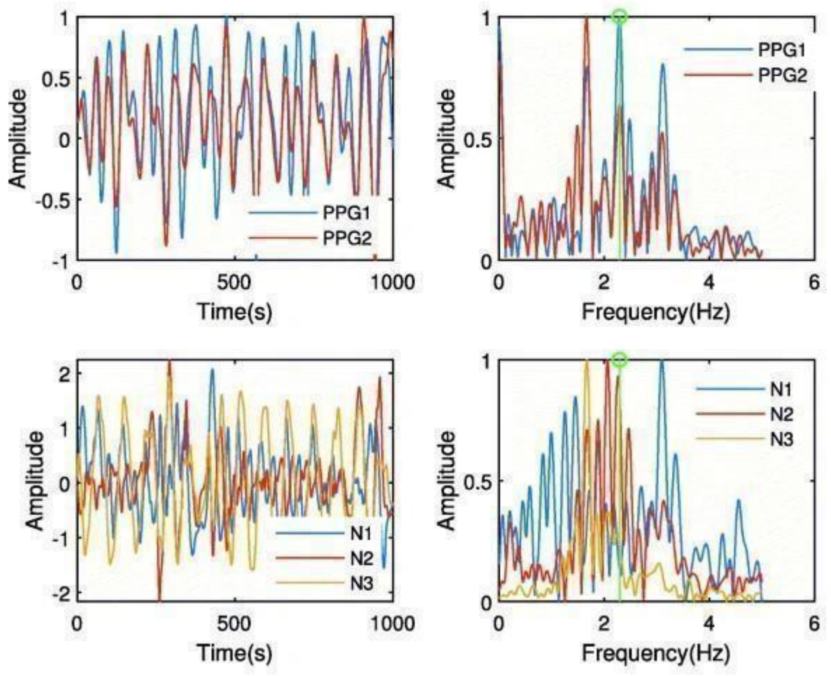
In the last few decades, our world has been transformed into a world of wearables. This transformation is due to the fact that these wearables are light in weight and are operatable in real-time. This equipment provides different functionalities that range from clinical applications like heart and respiration rate monitoring to activity and location recognition as discussed in the research. In these wearables, different types of sensors are used but sensors which use electrodes, accelerometers, oximeters, etc. are more common and they sense different biosignals like sound, motion, heart rate, etc. . Among sensors for biosignals, the use of PPG-based systems is widespread and it appears in clinical applications like obstructive sleep apnea detection in children and respiration influence on arterial pressure, etc. and non-clinical applications like chewing rate, personal authentication, and driver drowsiness, etc.

During physical activity, PPG channels (channels that are used for HR calculation) and motion channels (acceleration (ACC) channels) become correlated. Hence, PPG channels are directly affected due to noise content added due to ACC channels. However, these noise contents are different from other noises like environment noise as they have high amplitude and are able to alter the signal morphology, hence changing its spectral contents. These special types of noise contents which arise due to physical activity are known as motion artifacts (MAs). Moreover, MAs can add contents in which spectral contents may overlap or become too close to actual HR. Due to the high probability of proximity of HR and MA spectral peaks and spectral peak randomness, removal of MAs is the single most important preprocessing step in classical signal processing to calculate HR calculation from PPG.

Fig - 1

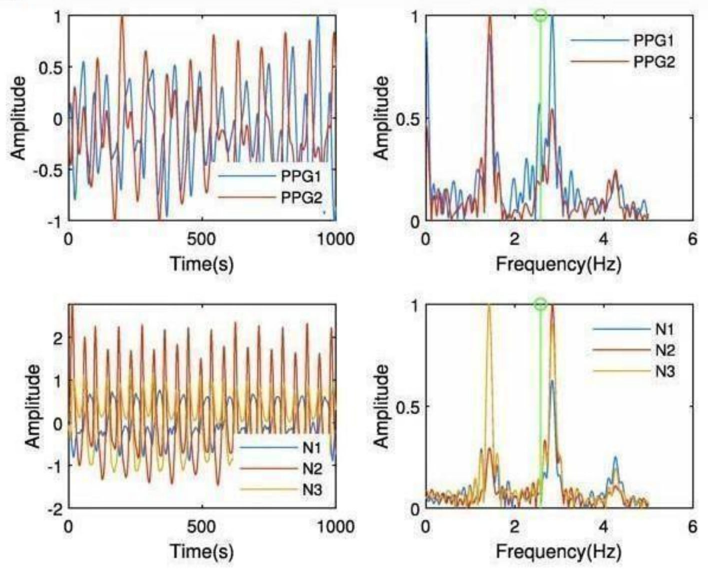


(a) PPG and ACC signals with their spectra. Green peaks (peaks associated with actual heart rate) overlaps with peaks in both PPG1 and PPG2 channels. In the Figure, PPG1 and PPG 2 are PPG channels and N1, N2 and N3 represent the acceleration channels.

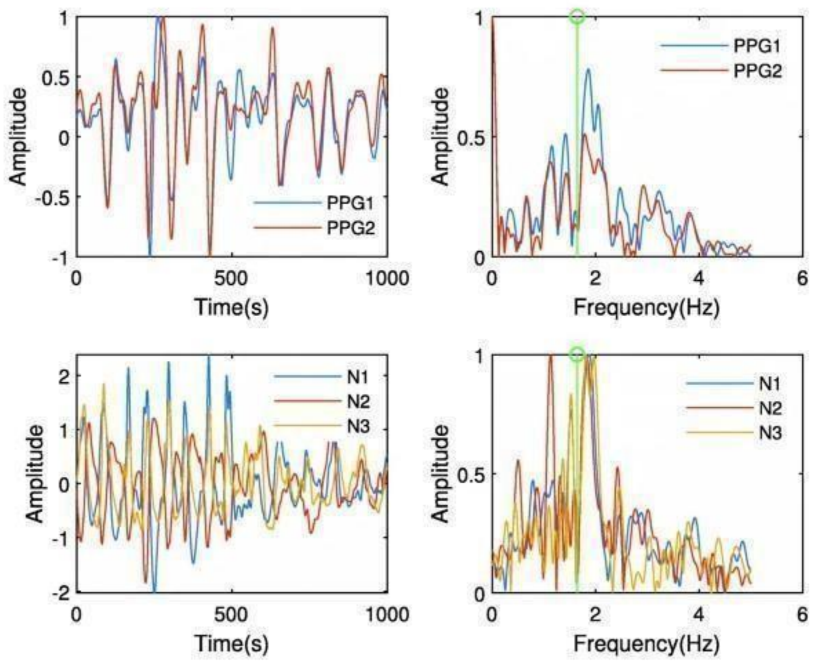


(b) PPG and ACC signals with their spectra. Major peak in PPG1 represents the actual heart rate(Green peaks). And one of the peaks in PPG2 channels represents the actual heart rate(Green peaks). Moreover, spectral distribution of PPG1 and PPG2 shows that MAs have affected both channels differently. In the Figure, PPG1 and PPG 2 are PPG channels and N1, N2 and N3 represent the acceleration channels.

Fig-2



(a) PPG and ACC signals with their spectra. One of peak in PPG1 represents the actual heart rate(Green peaks). PPG2 channels is devoid of actual heart rate peak. In the Figure, PPG1 and PPG 2 are PPG channels and N1, N2 and N3 represent the acceleration channels.



(b) PPG and ACC signals with their spectra. Peak representing the actual heart rate is absent in both PPG channels. In the Figure, PPG1 and PPG 2 are PPG channels and N1, N2 and N3 represent the acceleration channels.

For this case, our project provides an ideal solution where it predicts the Heart rate of a person accurately by removing motion artifacts.

1.4 SCOPE:

PRODUCT SCOPE:

Predicting the heart rate of a person using Machine Learning models where the classical approach of PPG fails because In the classical approach, predicting the heart rate accurately is difficult as the person cannot be at rest all the time. We can also estimate abnormalities in the heart rate by analyzing the electrocardiogram waves.

There are many wearables out there that provide the measure of Heart Rate, But the accuracy of predicting the heart rate drops drastically while performing physical activities. But the Machine Learning models predict the

Heart rates of a person accurately even while the person is performing physical activities by removing the motion artifacts.

SCOPE FOR FUTURE WORK:

- Making this an end-to-end project with signals being collected, synchronized, and fed to the model for more intensive testing.
- Making this model end-to-end requires hardware that this data has been processed and synchronized under, which was out of the scope of this particular project, but can be implemented by obtaining the required hardware. This allows for more intensive testing and opens doors for ML-based PPG heart rate estimation being integrated across wearables.

1.5 Software Requirements:

Operating System: macOS, Windows. **Technology**: Machine Learning

Programming Language: Python

Coding Platform: VS code (visual studio code) and Google Colab

Libraries: Keras**,**sklearn, jolib, matplotlib.pyplot**,** NumPy, scipy, math, tqdm, Dense, Dropout, Activation, Conv1D, Reshape, MaxPooling1D, Flatten

Python:

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales.
Python's design philosophy emphasizes code readability with its notable use of significant indentation. Its language constructs and object-oriented

approach aim to help programmers write clear, logical code for small and large-scale projects.

Imutils:

Imutils are a series of convenient functions to make basic image processing functions such as translation, rotation, resizing, skeletonization, and displaying Matplotlib images easier with OpenCV and both Python 2.7 and Python 3.

Numpy:

A library for the Python programming language. adding support for large. multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate with

Scikit-Learn:

Scikit -Learn is the most useful and robust library for machine learning in python

scipy. spatial:

scipy. spatial can compute triangulations, Voronoi diagrams, and convex hulls of a set of points, by leveraging the Qhull library. Moreover, it contains KDTree implementations for nearest-neighbor point queries and utilities for distance computations in various metrics.

TensorFlow:

TensorFlow is an end-to-end open-source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries, and community resources that lets researchers push the state-of-the-art in ML, and provides a collection of workflows with intuitive, high-level APIs to create machine learning models in numerous languages.

VS Code (visual studio code):

Visual Studio Code is a freeware source-code editor made by Microsoft for Windows, Linux, and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality.

Microsoft has released Visual Studio Code's source code on the Microsoft/vscode(CodOSS) repository of GitHub, under the permissive MIT License, while the releases by Microsoft are freeware.

In the Stack Overflow 2019 Developer Survey, Visual Studio Code was ranked the most popular developer environment tool, with 50.7% of 87,317 respondents reporting that they use it.

Visual Studio Code was first announced on April 29, 2015, by Microsoft at the 2015 Build conference. A Preview build was released shortly thereafter. Visual Studio Code is a source code editor that can be used with a variety of programming languages, including Java, JavaScript, Go, Node.js, Python, and C++. It is based on the Electron framework, which is used to develop Node.js Web applications that run on the Blink layout engine. Visual Studio Code employs the same editor component (codenamed "Monaco") used in Azure DevOps (formerly called Visual Studio Online and Visual Studio Team Services).

Instead of a project system, it allows users to open one or more directories, which can then be saved in workspaces for future reuse. This allows it to operate as a language-agnostic code editor for any language. It supports a number of programming languages and a set of features that differs per language. Unwanted files and folders can be excluded from the project tree via the settings. Many Visual Studio Code features are not exposed through menus or the user interface but can be accessed via the command palette. Visual Studio Code can be extended via extensions, available through a central repository.

This includes additions to the editor and language support. A notable feature is the ability to create extensions that add support for new languages, themes, and debuggers, perform static code analysis and add code lines using the Language Server Protocol.

Visual Studio Code includes multiple extensions for FTP, allowing the software to be used as a free alternative for web development. Code can be synced between the editor and the server, without downloading any extra software.

Visual Studio Code allows users to set the code page in which the active document is saved, the newline character, and the programming language of the active document. This allows it to be used on any platform, in any locale, and for any given programming language.

Google Colab:

Google Colab is a great platform for deep learning projects with large datasets, and it supports both GPU and TPU instances, which makes it a perfect tool for deep learning and data analytics enthusiasts because of computational limitations on local machines as access to faster GPUs and TPUs means we spend less time waiting while our code is running

Hardware Requirements:

Processor: Latest version or any updated Processor.

RAM: Minimum 4 GB.

System Type: 64-bit OS, x64-based processor.

2. Literature Survey

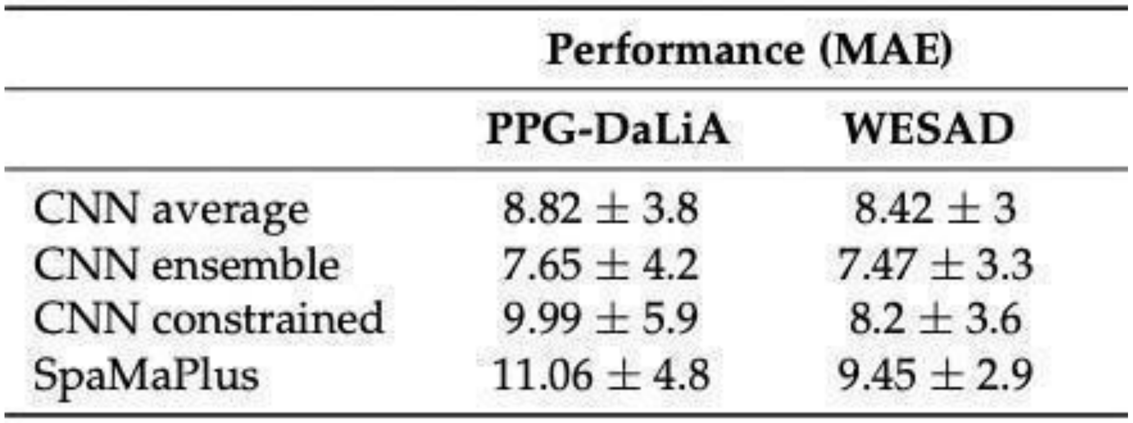
2.1 Existing system:

The relationship between the PPG and Heart Rate is well obtained, by using PPG to measure the speed of blood flow, commonly known as the pulse wave velocity (PWV), when the arteries are contracted, blood travels faster and exerts more pressure, similarly when the arteries are relaxed or elastic, the blood travels slower and exerts lower pressure, this can be understood by predicting the PTT pulse transit time, which is the amount of time taken by pulse originating at the heart to reach the peripheral point of the body. This corresponds to one heartbeat and thus the Heart Rate can be found in beats per minute (bpm). The main problem is having a perfect dataset because the PPG signals have a lot of noise and are easily affected by external factors. Moreover, recording the signals in a controlled environment is also not advisable because we might want to test them under different conditions and the results might not be accurate. The following articles are explored and considered,

Deep PPG: Large-Scale Heart rate Estimation with Convolutional Neural Networks [1]

In this article several Machine Learning models over different datasets are stated below,

1. IEEE Accurate Heart Rate Monitoring [11]
2. PPG DaLiA [2]
3. WESAD [3] Several variations of CNN models have been trained and discussed in the paper and the metrics are shown below,



	Performance (MAE)	
	PPG-DaLiA	WESAD
CNN average	8.82 ± 3.8	8.42 ± 3
CNN ensemble	7.65 ± 4.2	7.47 ± 3.3
CNN constrained	9.99 ± 5.9	8.2 ± 3.6
SpaMaPlus	11.06 ± 4.8	9.45 ± 2.9

in Table.1. Performance Metrics for different variations of CNN models.

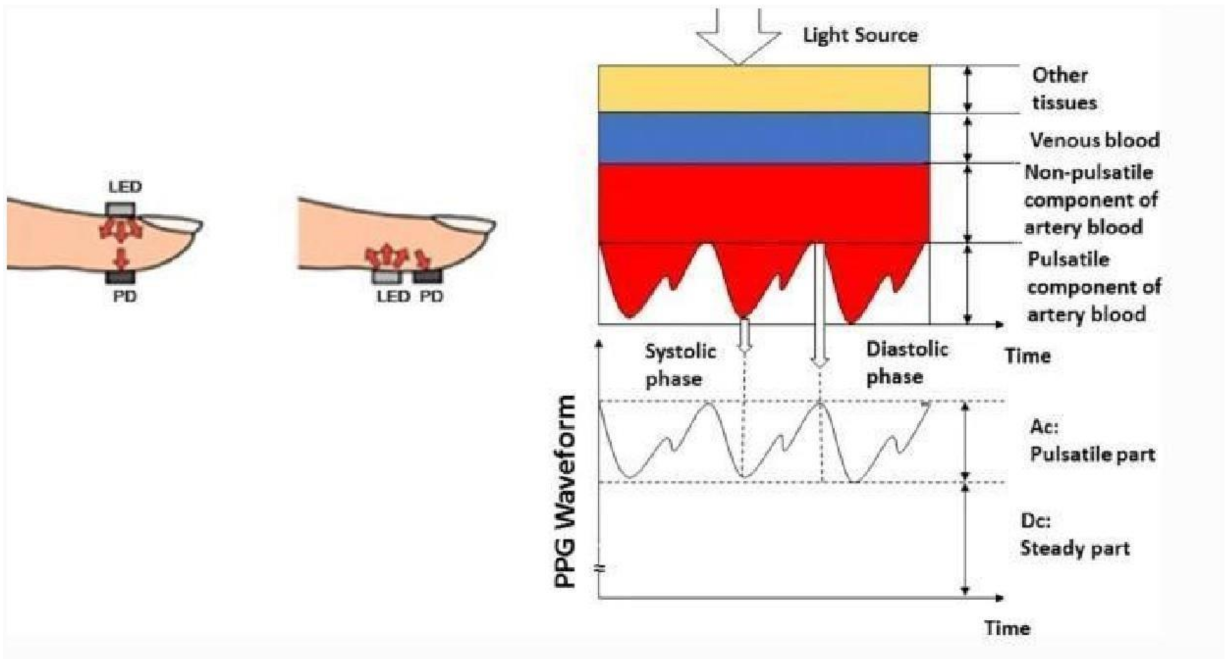
This article proposes a Convolutional Neural Network-based model to estimate the heart rate, which reduced the mean absolute error by about 31% over the traditional methods.

2.1.2 A Machine Learning Approach for Heart Rate Estimation from PPG Signal using Random Forest Regression Algorithm

In this article, the heart rate is estimated from wearables that take in the PPG signals. They used Random Forests Regression for training the model, with various feature engineering following what is called a multimodel machine learning approach (MMMLA). The dataset they used consisted of 12 subjects (male, with ages ranging from 18 to 35). Simultaneous PPG signals, acceleration signals, and a single-channel ECG signal were recorded for each subject at the time of physical exercise. They were able to obtain a mean absolute error of about 1.11 beats per minute (BPM) with this research.

2.1.3 Existing system (Elaborate)

PPG is an emerging optical technology and is non-invasive in nature. It is also cheap as it uses LED as a light source and photodetector (PD) as a receiver to measure volumetric changes in blood [9]. PPG is of either transmittance or reflectance type. In the transmittance type, the LED used as the light source is placed opposite to PD while LED and PD are on the same side in the reflectance type PPG. Figure 1a shows both types where LED and PD are on the same and opposite sides of the finger for transmittance and reflectance type PPG, respectively.



When light from LED is incident on living tissue, it is mainly absorbed by tissues and arteries. Figure 1b shows light absorption from the light source (LED) to the receiver (PD) and the resultant PPG waveform which represents a quasiperiodic signal. It is clear from Fig. 1b that the waveform is composed mainly of non-pulsatile or DC components and pulsatile or AC components. The DC component corresponds to the average or steady blood volume of both arterial and venous blood. The AC component refers to changes in blood volume during the systole and diastole phases. Changes in blood volume can be used to calculate the average heart rate (HR) which is one of the major applications of PPG among others mentioned above. However, PPG has one distinct advantage over other HR calculation modalities like phonocardiogram (PCG), electrocardiography (ECG), etc. as it does not require any specific technique to attach sensors at pre-defined positions in the body. In fact, PPG can be collected from the earlobe, finger, or wrist. This ease of use has made PPG attractive to calculate HR during exercise and other physical activities.

2.2 Proposal(Techniques and algorithms applicable)

The proposed system is comprised of three different machine learning models

- Random Forest

- Convolutional Neural Network
- Recurrent Neural Network

The dataset that we considered for the training of the models is PPG-DaLiA. This dataset has been a near-perfect dataset with near real-life environments while recording and with various activities. An in-depth description of the dataset will be done later in the report.

2.3 Applications:

Heart rate estimation machines are mainly used to determine the exercise intensity of training sessions or races. Compared with other indications of excessive intensity, HR is easy to monitor, is relatively cheap, and can be used in most situations. Keeping track of your heart rate can give you insight into your fitness level, heart health, and emotional health.

- Medical field
- Sports
- Healthcare at home

3. System Design:

3.1 Technological Review:

Machine Learning:

Machine learning (**ML**) is the study of computer algorithms that can improve automatically through experience and by the use of data. It is seen as a part of artificial intelligence. Machine learning algorithms build a model based on sample data, known as training data, in order to make predictions or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications, such as in medicine, email filtering, speech recognition, and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks.

A subset of machine learning is closely related to computational statistics, which focuses on making predictions using computers; but not all machine learning is statistical learning. The study of mathematical optimization delivers methods, theory, and application domains to the field of machine learning. Data mining is a related field of study, focusing on exploratory data analysis through unsupervised learning. Some implementations of machine learning use data and neural networks in a way that mimics the working of a biological brain. In its application across business problems, machine learning is also referred to as predictive analytics.

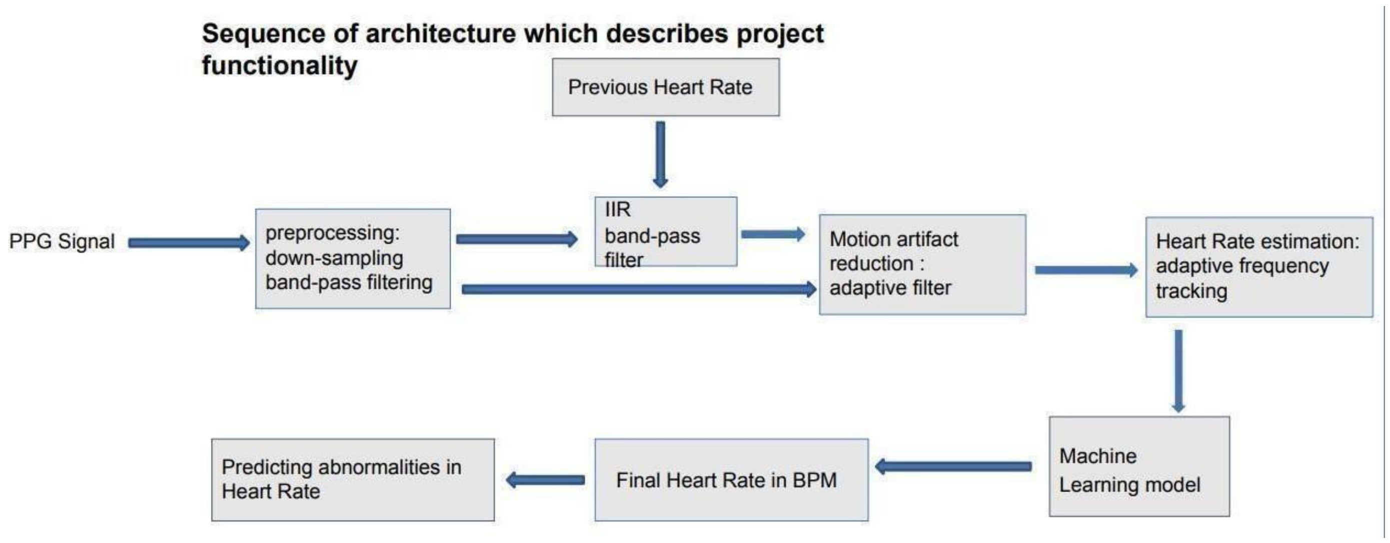
IoT (INTERNET OF THINGS):

The Internet of things describes the network of physical objects—“things”—that are embedded with sensors, software, and other technologies for the purpose of connecting and exchanging data with other devices and systems over the Internet.

- **Access to low-cost, low-power sensor technology.** Affordable and reliable sensors are making IoT technology possible for more manufacturers.
- **Cloud computing platforms.** The increase in the availability of cloud platforms enables both businesses and consumers to access the infrastructure they need to scale up without actually having to manage it all.

- **Machine learning and analytics.** With advances in machine learning and analytics, along with access to varied and vast amounts of data stored in the cloud, businesses can gather insights faster and more easily. The emergence of these allied technologies continues to push the boundaries of IoT and the data produced by IoT also feeds these technologies.

3.2 Sequence Diagram:


Sequence of architecture which describes project functionality
Previous Heart Rate
PPG Signal
preprocessing: down-sampling band-pass filtering
IIR band-pass filter
Motion artifact reduction : adaptive filter
Heart Rate estimation: adaptive frequency tracking
Machine Learning model
Final Heart Rate in BPM
Predicting abnormalities in Heart Rate

3.3 Model Description:

3.3.1 Random Forest Regression:

Before understanding what Random Forests are, we have to have a basic understanding of

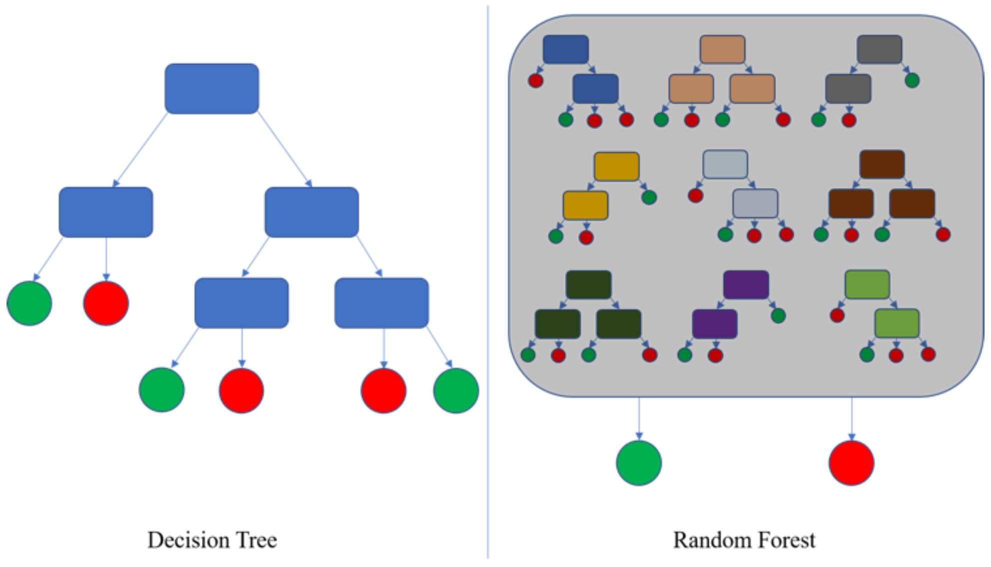
Decision Trees.

Decision Trees: A decision tree builds regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with **decision nodes** and **leaf nodes**. A decision node has two or more branches, each representing values for the attribute tested. The leaf node represents a decision on the numerical target.

The topmost decision node in a tree corresponds to the best predictor called the **root node**. The core algorithm for building decision trees is called **ID3** by J. R. Quinlan and employs a top-down, greedy search through the space of possible branches with no backtracking. The ID3 algorithm can be used to construct a decision tree for regression by replacing Information Gain with Standard Deviation Reduction.

Random Forests: Random forests are an ensemble learning method for classification, regression, and other tasks that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes or mean/average prediction of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set. Random forests generally outperform decision trees.

Fig-3: Decision Tree Vs Random Forests (Illustration)



3.3.2 convolutional neural network:

A convolutional neural network consists of an input and an output layer, as well as multiple hidden layers. The hidden layers of a CNN typically consist of a series of convolutional layers that convolve with multiplication or other dot product. The activation function is commonly a ReLU layer and is subsequently followed by additional convolutions such as pooling layers, fully connected layers, and normalization layers referred to as hidden layers because their inputs and outputs are masked by the activation function and final convolution.

Let’s have a look at different convolutional layers.

- **Conv Layer**: CONV layer will compute the output of neurons that are connected to local regions in the input, each computing a dot product between their weights and a small region they are connected to in the input volume.

- **Pooling Layers:** Pooling layers are used to reduce the dimensions of the feature maps. Thus, it reduces the number of parameters learned and the amount of computation performed in the network. There are 3 types of Pooling layers.

- **Max Pooling:** Max pooling is a pooling operation that selects the maximum element from the region of the feature map covered by the filter.

- **Average Pooling:** Average pooling computes the average of the elements present in the region of the feature map covered by the filter.

- **Global Pooling:** Global pooling reduces each channel in the feature map to a single value.

We have used some of the above-mentioned layers that are available in ‘Convolutional Layers’ from Python’s ‘Keras’ library, in our model. The model can be visualized.

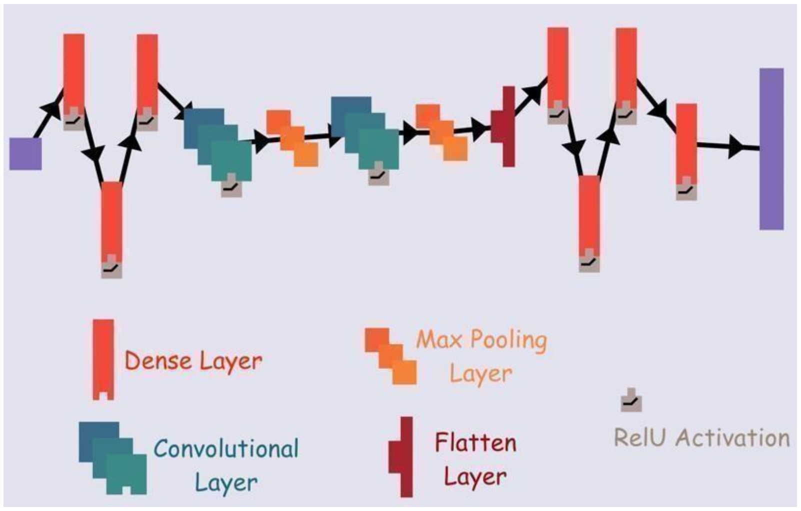


Fig-4: Illustration of different connected layers of the CNN model.

3.3.3 Recurrent Neural Networks:

A Recurrent neural network (RNN) is a class of artificial **neural networks** where connections between nodes form a directed graph along a temporal sequence. This allows it to exhibit temporal dynamic behavior. There are many well-known RNN layers of which we have a look at the LSTM layer.

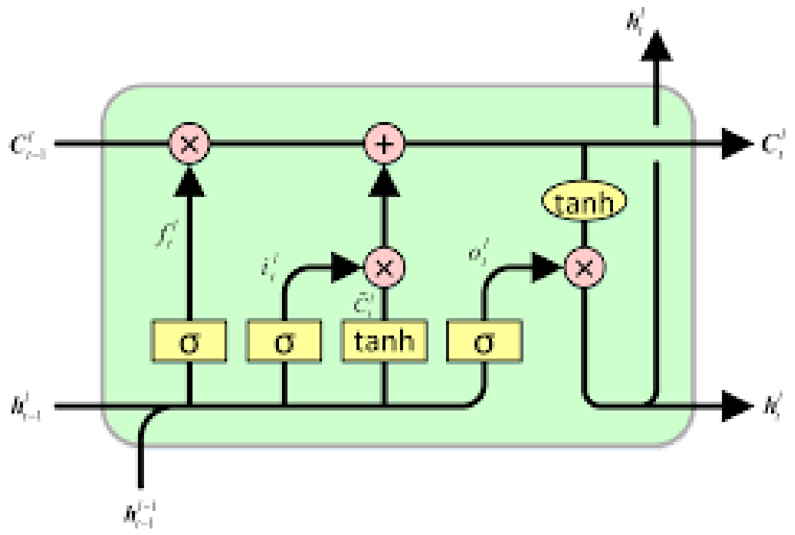


Fig-5:A high-level illustration of the LSTM Layer.

LSTM: Which stands for Long Short-Term Memory, are a type of recurrent neural network capable of learning order dependence in sequence prediction problems. This can be used when there is a dependence of the previous values on the current prediction. We make use of this layer along with various convolutional layers available in the Keras module to train the model. The model can be visualized as shown below.

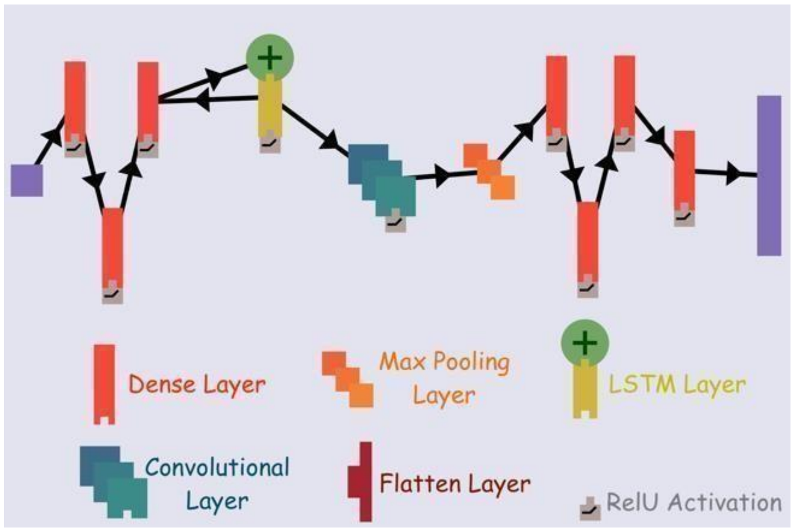
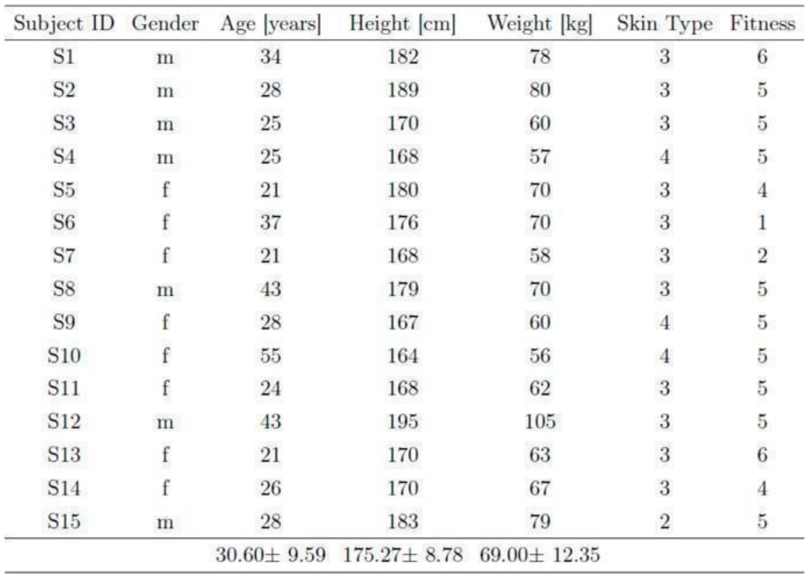


Fig-6: Illustration of different connected layers of the RNN model.

4. Implementation:

4.1 Collecting the dataset:

The dataset we use is the PPG-DaLiA dataset, which stands for “PPG dataset for motion compensation and heart rate estimation in Daily Life Activities”. This dataset includes in total more than 36 hours of data, recorded from 15 subjects.



Subject ID	Gender	Age [years]	Height [cm]	Weight [kg]	Skin Type	Fitness
S1	m	34	182	78	3	6
S2	m	28	189	80	3	5
S3	m	25	170	60	3	5
S4	m	25	168	57	4	5
S5	f	21	180	70	3	4
S6	f	37	176	70	3	1
S7	f	21	168	58	3	2
S8	m	43	179	70	3	5
S9	f	28	167	60	4	5
S10	f	55	164	56	4	5
S11	f	24	168	62	3	5
S12	m	43	195	105	3	5
S13	f	21	170	63	3	6
S14	f	26	170	67	3	4
S15	m	28	183	79	2	5
		30.60± 9.59	175.27± 8.78	69.00± 12.35		

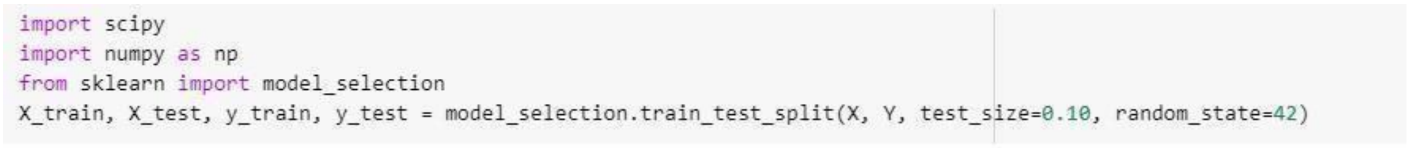
This PPG-DaLiA dataset is collected from the website of “ **UCI Machine Learning Repository** “.

4.1.1 Google Colab setup:

Navigate to the Google Colab platform and log in using the credentials. Create a new notebook and change the runtime type to GPU. Mount the google drive using the following code snippet after checking if the Nvidia GPU is enabled. Colab supports both GPU and TPU instances, which makes it a perfect tool for deep learning and data analytics enthusiasts because of computational limitations on local machines as access to faster GPUs and TPUs means we spend less time waiting while our code is running

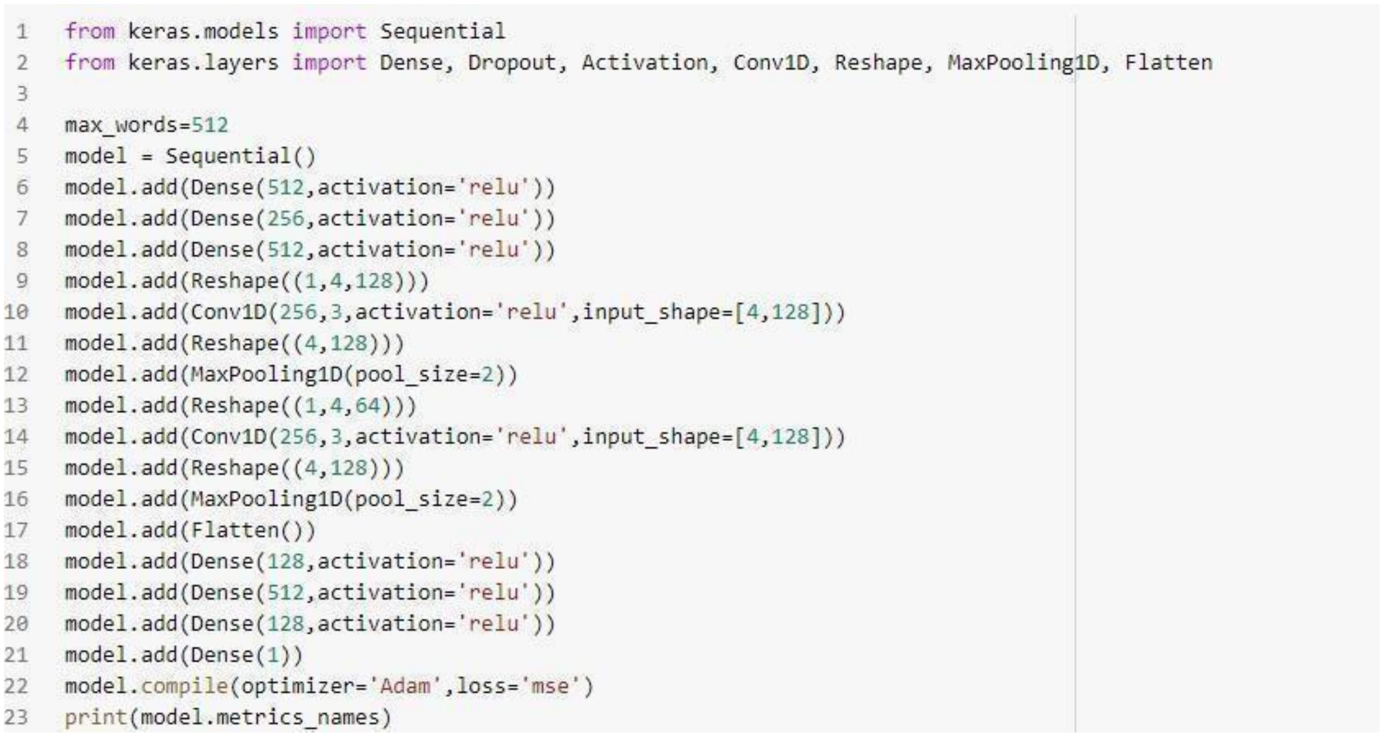
4.2 IMPLEMENTATION OF EACH MODULE:

The dataset is split into training and testing data samples in the ration of 9:1. The input and output data are spilt into train and test sets using sklearn's train-test
split.



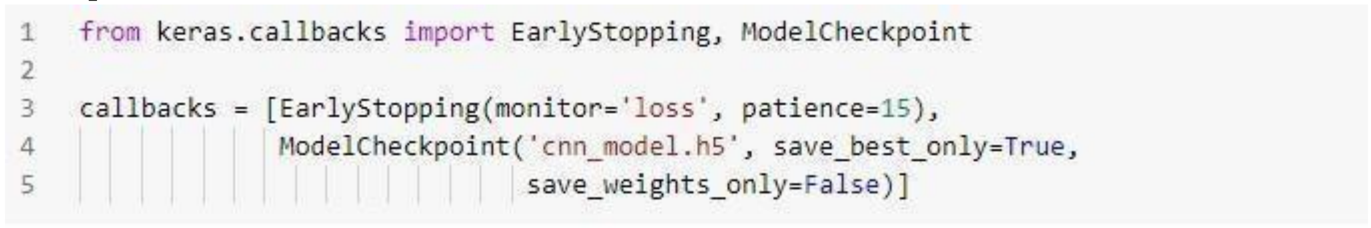
```
import scipy
import numpy as np
from sklearn import model_selection
X_train, X_test, y_train, y_test = model_selection.train_test_split(X, Y, test_size=0.10, random_state=42)
```

The CNN model has 16 layers. The most important layer is the Conv1D layer as the PPG-DaLiA dataset completely consists of 1-dimensional information. The total data accounted for 64697 samples and each input sample is of size 512.



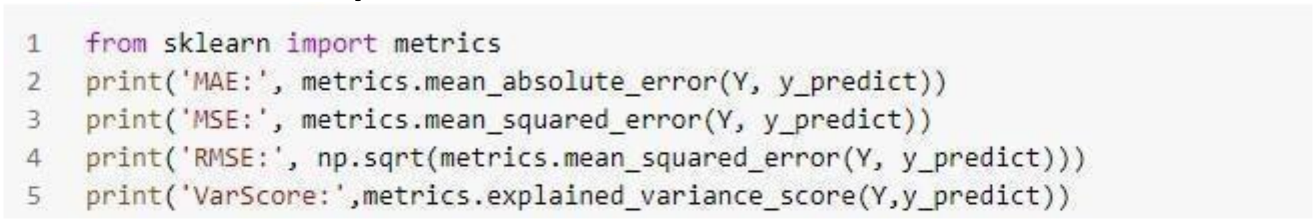
```
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation, Conv1D, Reshape, MaxPooling1D, Flatten

max_words=512
model = Sequential()
model.add(Dense(512,activation='relu'))
model.add(Dense(256,activation='relu'))
model.add(Dense(512,activation='relu'))
model.add(Reshape((1,4,128)))
model.add(Conv1D(256,3,activation='relu',input_shape=[4,128]))
model.add(Reshape((4,128)))
model.add(MaxPooling1D(pool_size=2))
model.add(Reshape((1,4,64)))
model.add(Conv1D(256,3,activation='relu',input_shape=[4,128]))
model.add(Reshape((4,128)))
model.add(MaxPooling1D(pool_size=2))
model.add(Flatten())
model.add(Dense(128,activation='relu'))
model.add(Dense(512,activation='relu'))
model.add(Dense(128,activation='relu'))
model.add(Dense(1))
model.compile(optimizer='Adam',loss='mse')
print(model.metrics_names)
```

Also, used callbacks with early stopping with the patience of 15 and Model checkpoint to save the model.



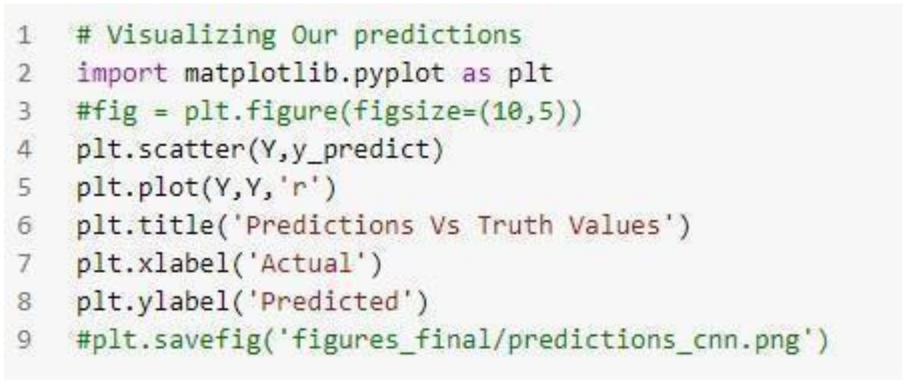
```python
from keras.callbacks import EarlyStopping, ModelCheckpoint

callbacks = [EarlyStopping(monitor='loss', patience=15),
             ModelCheckpoint('cnn_model.h5', save_best_only=True,
                             save_weights_only=False)]
```

Also, estimated accuracy metrics.



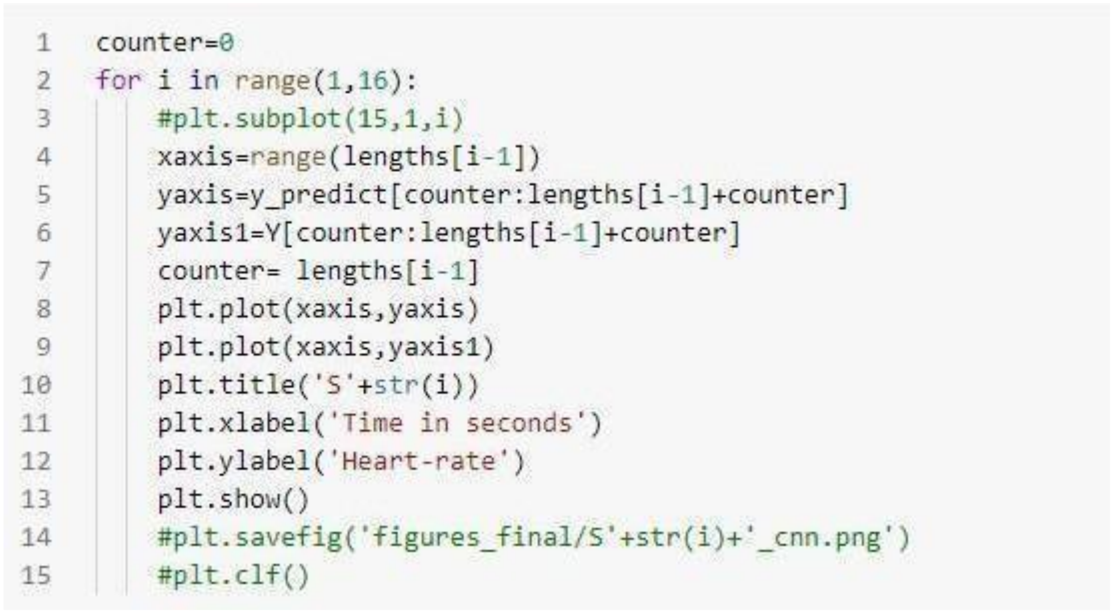
```python
from sklearn import metrics
print('MAE:', metrics.mean_absolute_error(Y, y_predict))
print('MSE:', metrics.mean_squared_error(Y, y_predict))
print('RMSE:', np.sqrt(metrics.mean_squared_error(Y, y_predict)))
print('VarScore:',metrics.explained_variance_score(Y,y_predict))
```

plotted a graph to see how far away the predictions are from the ground truth.



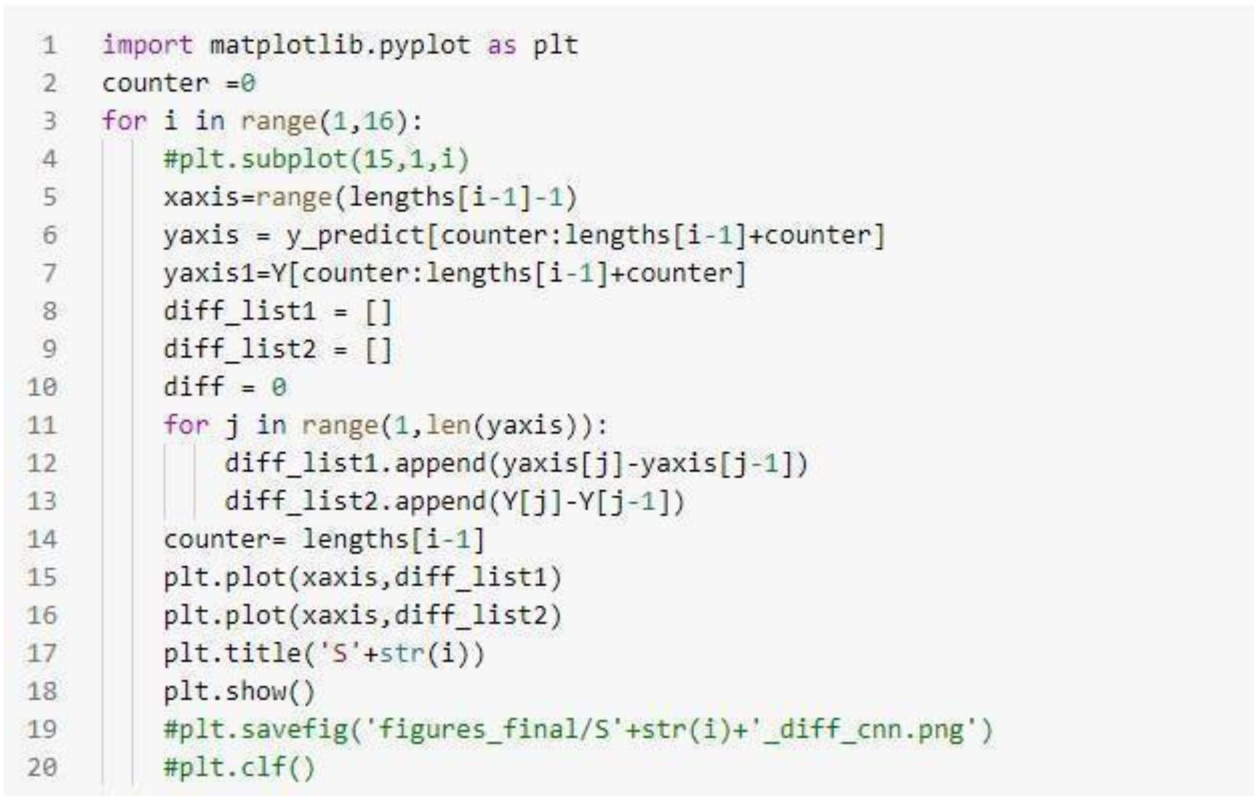
```python
# Visualizing Our predictions
import matplotlib.pyplot as plt
#fig = plt.figure(figsize=(10,5))
plt.scatter(Y,y_predict)
plt.plot(Y,Y,'r')
plt.title('Predictions Vs Truth Values')
plt.xlabel('Actual')
plt.ylabel('Predicted')
#plt.savefig('figures_final/predictions_cnn.png')
```

Predicted heart rate vs ground truth for all the individuals.



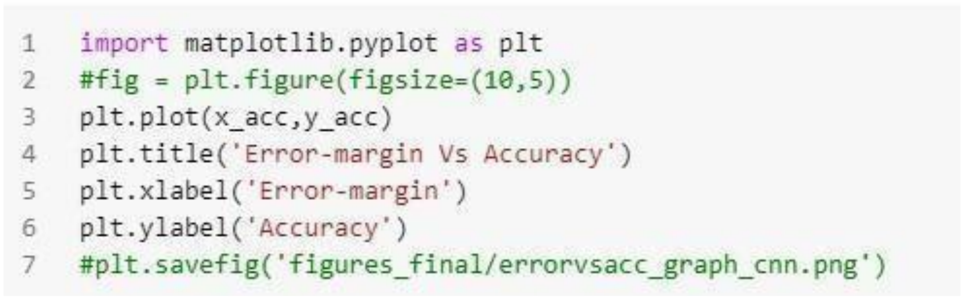
```python
counter=0
for i in range(1,16):
    #plt.subplot(15,1,i)
    xaxis=range(lengths[i-1])
    yaxis=y_predict[counter:lengths[i-1]+counter]
    yaxis1=Y[counter:lengths[i-1]+counter]
    counter= lengths[i-1]
    plt.plot(xaxis,yaxis)
    plt.plot(xaxis,yaxis1)
    plt.title('S'+str(i))
    plt.xlabel('Time in seconds')
    plt.ylabel('Heart-rate')
    plt.show()
    #plt.savefig('figures_final/S'+str(i)+'_cnn.png')
    #plt.clf()
```

Comparing how fast the heart rate is changing by getting the difference between adjacent heart rates for both predicted and ground truth heart rates.



```python
import matplotlib.pyplot as plt
counter =0
for i in range(1,16):
    #plt.subplot(15,1,i)
    xaxis=range(lengths[i-1]-1)
    yaxis = y_predict[counter:lengths[i-1]+counter]
    yaxis1=Y[counter:lengths[i-1]+counter]
    diff_list1 = []
    diff_list2 = []
    diff = 0
    for j in range(1,len(yaxis)):
        diff_list1.append(yaxis[j]-yaxis[j-1])
        diff_list2.append(Y[j]-Y[j-1])
    counter= lengths[i-1]
    plt.plot(xaxis,diff_list1)
    plt.plot(xaxis,diff_list2)
    plt.title('S'+str(i))
    plt.show()
    #plt.savefig('figures_final/S'+str(i)+'_diff_cnn.png')
    #plt.clf()
```

Plotted error margin vs accuracy over threshold values from 1 through 10.



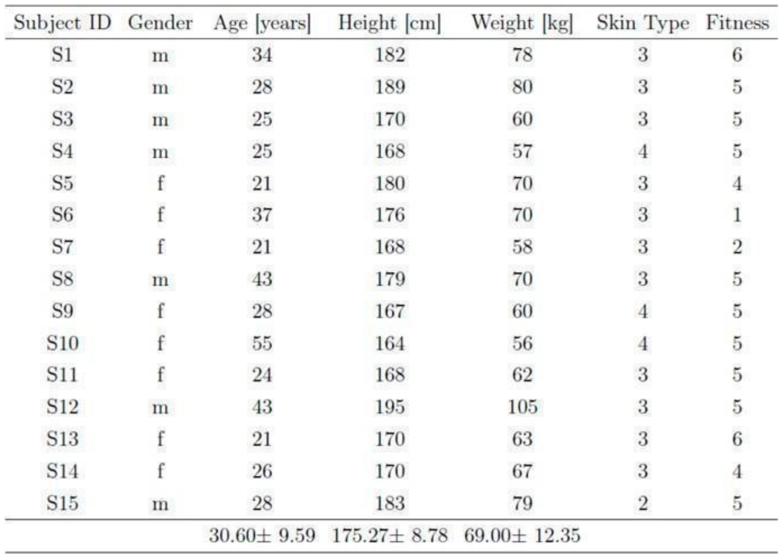
```python
import matplotlib.pyplot as plt
#fig = plt.figure(figsize=(10,5))
plt.plot(x_acc,y_acc)
plt.title('Error-margin Vs Accuracy')
plt.xlabel('Error-margin')
plt.ylabel('Accuracy')
#plt.savefig('figures_final/errorvsacc_graph_cnn.png')
```

5. Testing and results:

5.1 Dataset:

The dataset we use is the PPG-DaLiA dataset, which stands for “PPG dataset for motion compensation and heart rate estimation in Daily Life Activities”. This dataset includes in total more than 36 hours of data, recorded from 15 subjects. The data collection protocol includes eight different types of activities (which are typically performed in daily life) and transition periods in between. Subjects performed these activities under natural, close to real-life conditions. This dataset can serve as a benchmarking dataset, which enables to evaluate and further explore the generalization capabilities of PPG-based heart rate estimation approaches.

The dataset consists of 15 folders, S1 through S15, one for each subject. The details of all the subjects are given below.



Subject ID	Gender	Age [years]	Height [cm]	Weight [kg]	Skin Type	Fitness
S1	m	34	182	78	3	6
S2	m	28	189	80	3	5
S3	m	25	170	60	3	5
S4	m	25	168	57	4	5
S5	f	21	180	70	3	4
S6	f	37	176	70	3	1
S7	f	21	168	58	3	2
S8	m	43	179	70	3	5
S9	f	28	167	60	4	5
S10	f	55	164	56	4	5
S11	f	24	168	62	3	5
S12	m	43	195	105	3	5
S13	f	21	170	63	3	6
S14	f	26	170	67	3	4
S15	m	28	183	79	2	5
		30.60± 9.59	175.27± 8.78	69.00± 12.35		

overview of the subjects participating in the data collection

Each folder consists of 5 files,

- SX_activity.csv
- SX_E4.zip
- SX_quest.csv
- SX_RespiBan.h5
- SX.pkl”.

All the files other than SX.pkl contain all the raw data. We’re only interested in SX.pkl, which contains all the synchronized data required for training. Once we unpickle the data, we get a JSON object which looks like the following

```
SX {
'rpeaks': ,
'signal':
{
'chest': {
'ACC': ,
'ECG': ,
'EMG': ,
'EDA': ,
'Temp': ,
'Resp':
},
'wrist': {
'ACC': ,
'BVP': ,
'EDA': ,
'TEMP':
        }
},
'label': ,
'activity': ,

'questionnaire':
{ 'WEIGHT': ,
'Gender': ,
'AGE': ,
'HEIGHT': ,
'SKIN': ,
'SPORT':
},
'subject':
}
```

The data consists of two types of signals, chest signals, and wrist signals, obtained from two different devices RespiBan [7] and Empatica E4 [8] devices respectively. Some important data that’s important for our training are as follows,

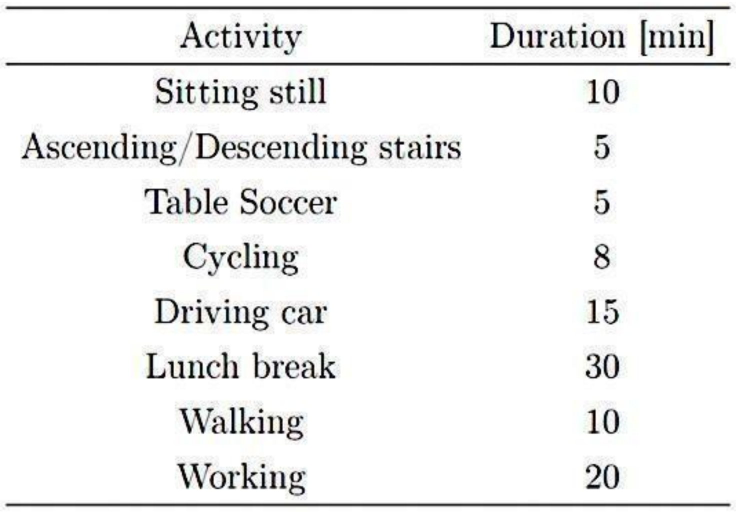
- SX -> ‘signal’ -> ‘wrist’ -> ‘BVP’: The synchronized PPG signals sampled at 64 Hz.
- SX -> ‘signal’ -> ‘chest’ -> ‘ECG’: The synchronized ECG signals sampled at 64 Hz.
- SX -> ‘label’: The heart rate ground truth for the 8-second segments, shifted with 2-seconds (obtained from ECG)
- SX -> ‘activity’: The activities the subject was performing when the signals were recorded.

We use “SX -> ‘signal’ -> ‘wrist’ -> ‘BVP’” PPG signals for training. These signals are sampled at 64 Hz, in simple terms, there are 64 values in a second. The heart rate has been obtained for 8-second segments, shifted with 2-seconds. This implies that our input data will be a 512-sized array and the number of entries would be ((length of PPG signals/ (2*64)) - 3).

“SX -> ‘label’” includes the heart rates which would be output data.

“SX -> ‘activities’” can be used later for comparing accuracies across different activities.

The following are the different activities and their durations throughout recording the signals,



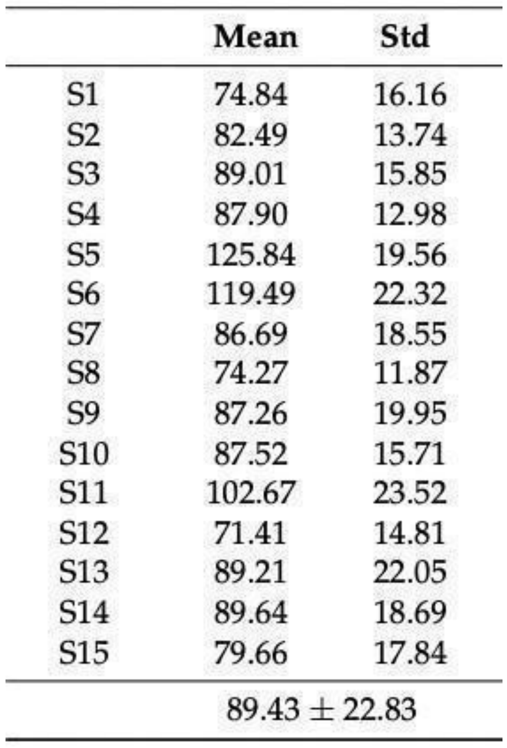
Activity	Duration [min]
Sitting still	10
Ascending/Descending stairs	5
Table Soccer	5
Cycling	8
Driving car	15
Lunch break	30
Walking	10
Working	20

The heart rate distribution across different ranges is given below,

Heart rate	**No of Samples**
0-40	0
40-60	3746
60-80	21585
80-100	22374
100-120	9884
120-140	4878
140-160	1679
160-180	512
180-200	39

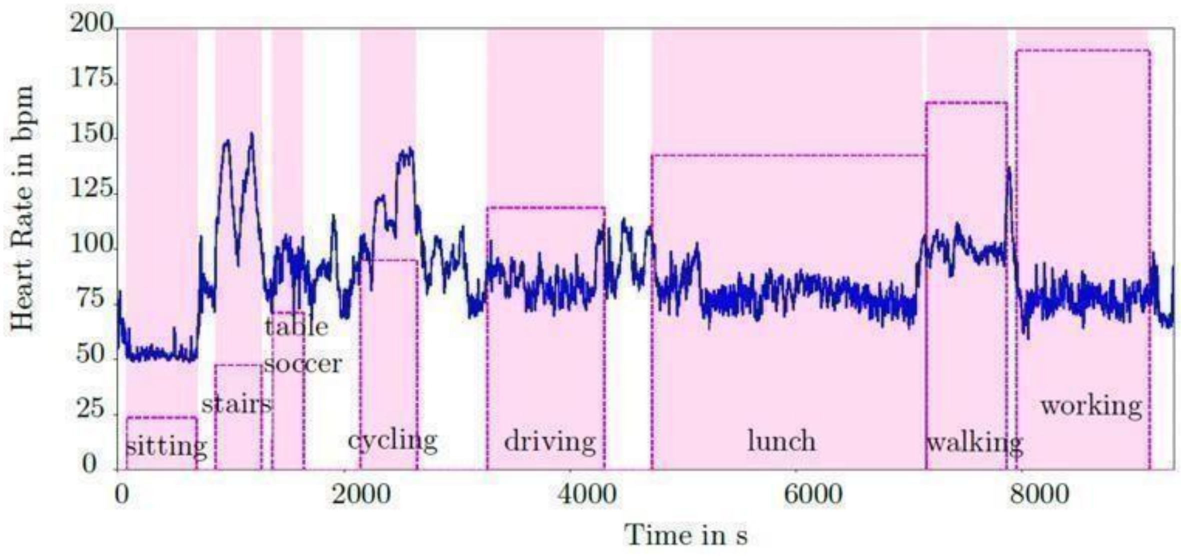
The mean of heart rates of different subjects is given below,

The heart rate and different activity durations for the subject ‘S7’ have been plotted below (the white spaces indicate transition-duration),



	Mean	Std
S1	74.84	16.16
S2	82.49	13.74
S3	89.01	15.85
S4	87.90	12.98
S5	125.84	19.56
S6	119.49	22.32
S7	86.69	18.55
S8	74.27	11.87
S9	87.26	19.95
S10	87.52	15.71
S11	102.67	23.52
S12	71.41	14.81
S13	89.21	22.05
S14	89.64	18.69
S15	79.66	17.84
	89.43 ± 22.83	

Fig-7:



The dataset has to be pre-processed to match our desired data. As we discussed earlier, the PPG signals are sampled at 64 Hz and the sliding window is 8 seconds with a 2-second shift. To obtain the input training data, we can follow the following steps,

- The first 512 (64*8) values of the signal correspond to the first heart rate.
- From here on, we shift by 2 seconds i.e., 128 (64*2) values, and move our sliding window forward and this corresponds to the next heart rate.
- The above process is done till we reach the end of the signal. Our input shape would be
(512,) and we try to predict the heart rate based on this 512-sized vector.
- All these 512 values will serve as features for our model. The output of the models is the heart rate of the person that changes every 2 seconds.
- Note that age, gender, height, fitness, etc., are not being considered in training the model, as the model has to be a standalone system where the heart rate is estimated based on PPG alone.

5.2 Results:

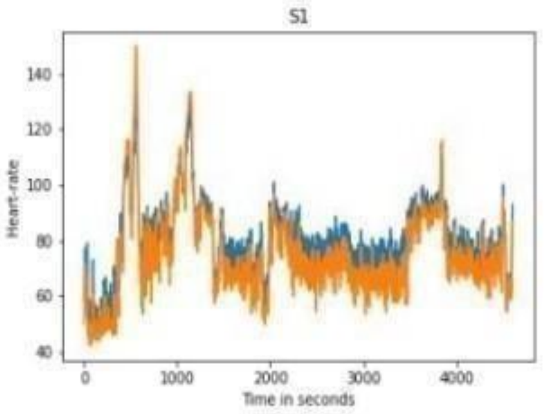
5.2.1 Random Forest Regression:

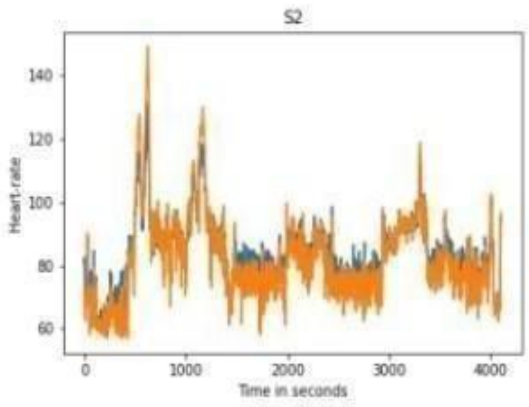
We used 'RandomForestRegressor' from Scikit Learn's ensemble learning module in Python.
The number of trees in the regressor has been tested with a number starting from 10 and finalized at 300 trees. The model has been trained over the preprocessed dataset. The input shape of the data is (64697,512), with each vector with size 512 and the total number of entries accounting for 64697 combining the data from all the 15 subjects.

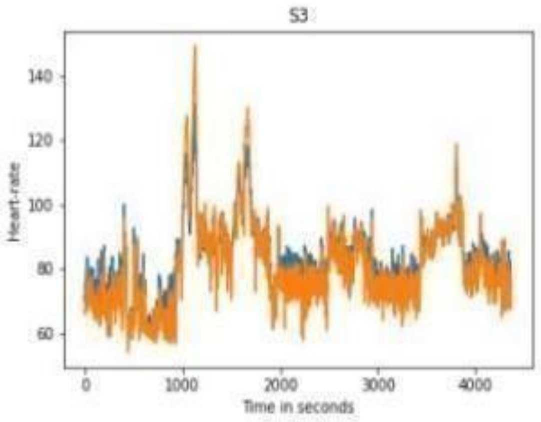
The accuracy metrics for the trained model are stated below,

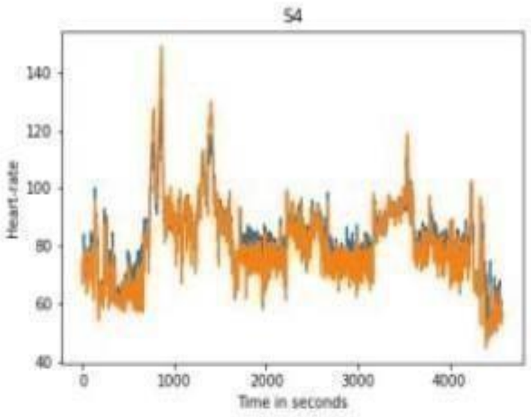
- Mean Absolute Error (MAE): 4.07
- Mean Squared Error (MSE): 34.87
- Root-Mean Squared Error (RMSE): 5.90
- Variance: 0.93

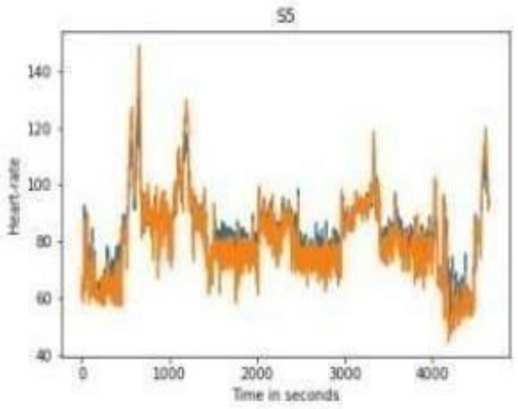
Let us see how the trained model predicted the heart rate for the 15 subjects individually.

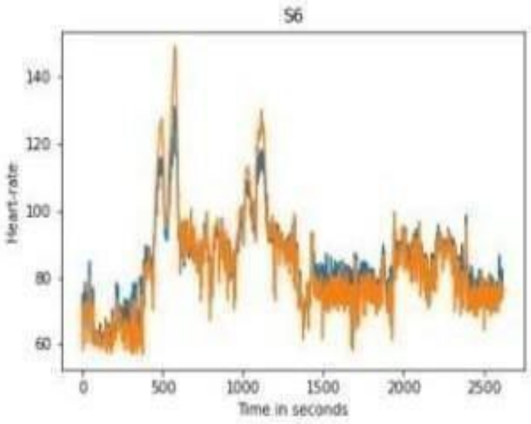

S1
Heart-rate
Time in seconds

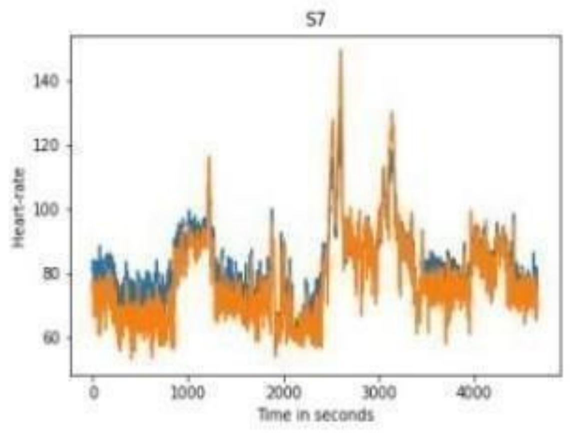

S2
Heart-rate
Time in seconds

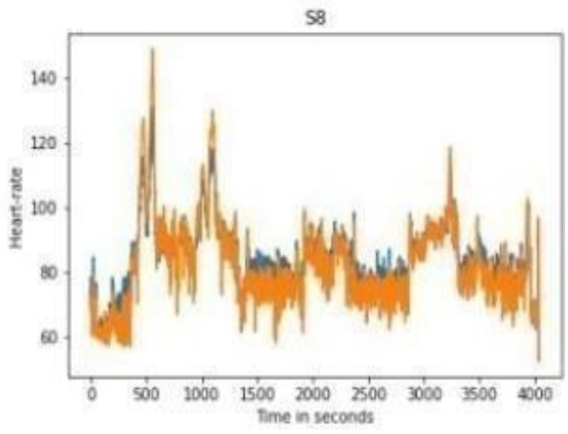

S3
Heart-rate
Time in seconds

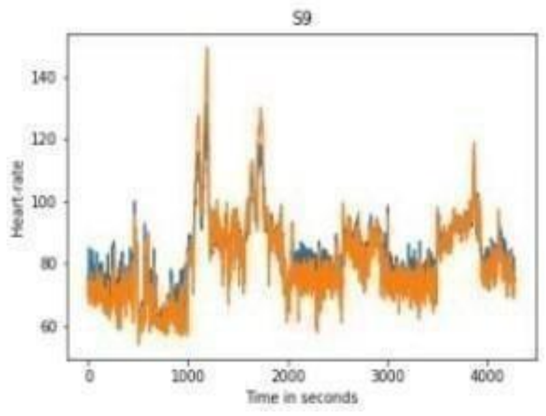

S4
Heart-rate
Time in seconds

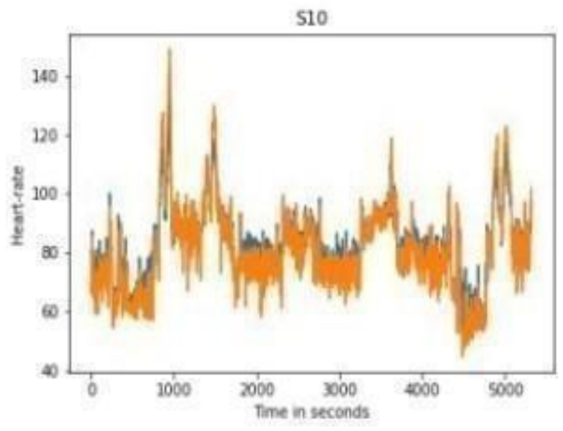

S5
Heart-rate
Time in seconds

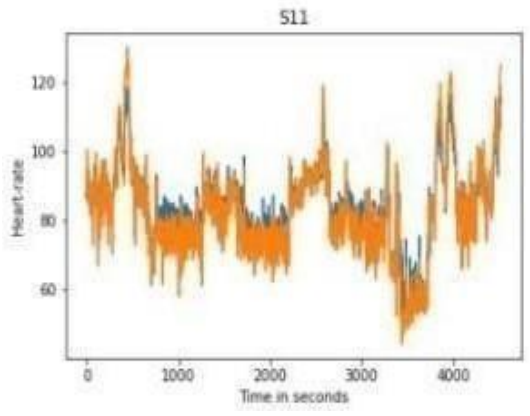

S6
Heart-rate
Time in seconds

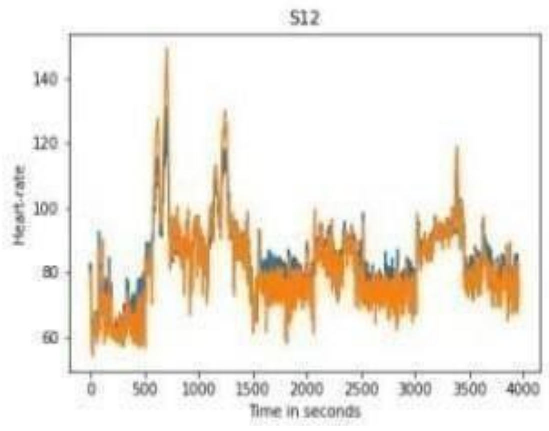

S7
Heart-rate
Time in seconds
60
80
100
120
140
0
1000
2000
3000
4000


S8
Heart-rate
Time in seconds
60
80
100
120
140
0
500
1000
1500
2000
2500
3000
3500
4000


S9
Heart-rate
Time in seconds
60
80
100
120
140
0
1000
2000
3000
4000


S10
Heart-rate
Time in seconds
40
60
80
100
120
140
0
1000
2000
3000
4000
5000


S11
Heart-rate
Time in seconds
60
80
100
120
0
1000
2000
3000
4000


S12
Heart-rate
Time in seconds
60
80
100
120
140
0
500
1000
1500
2000
2500
3000
3500
4000

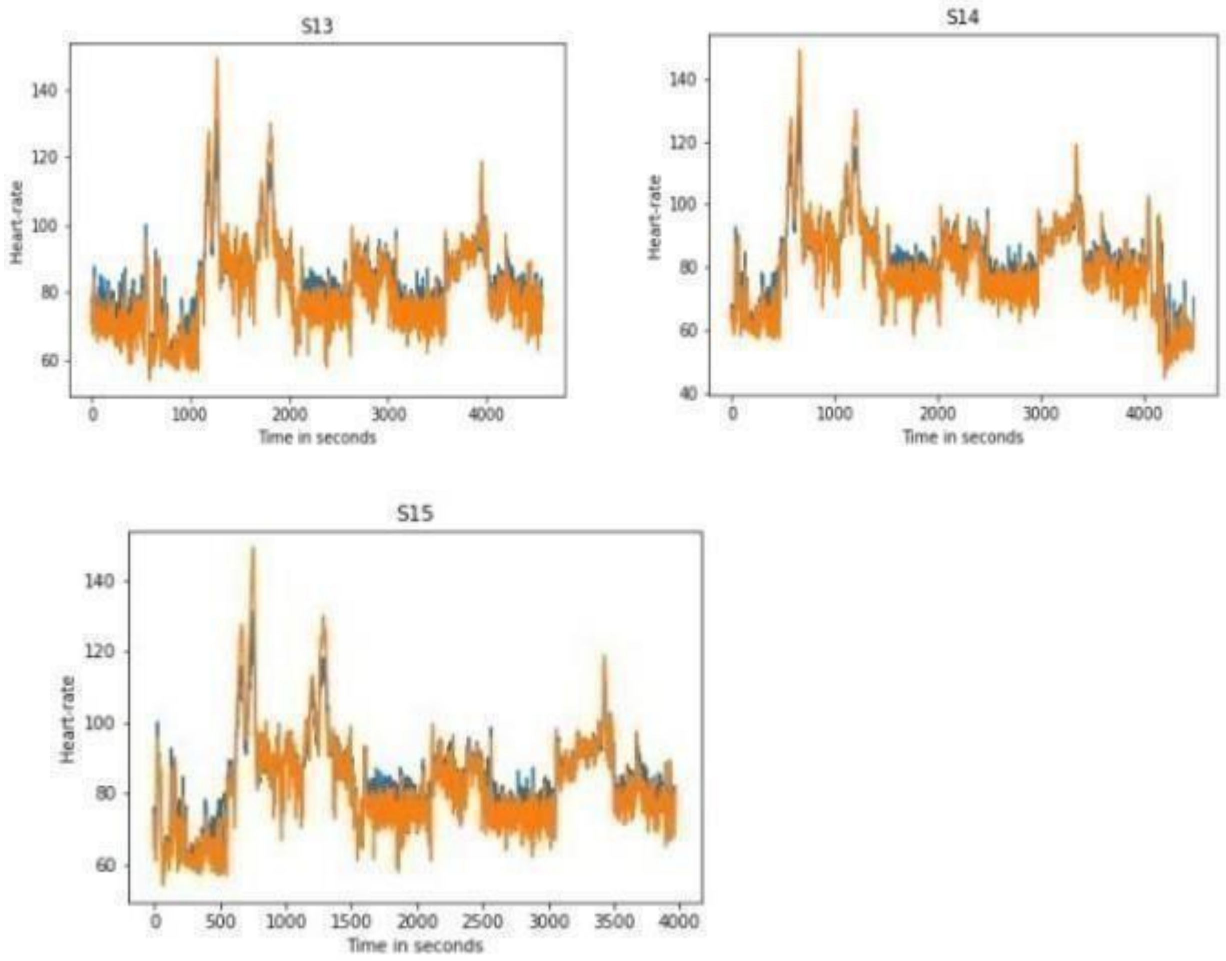


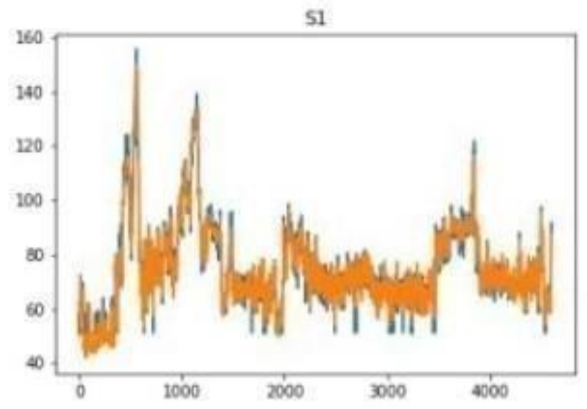
Fig-8 predicted heart rate (RFR)

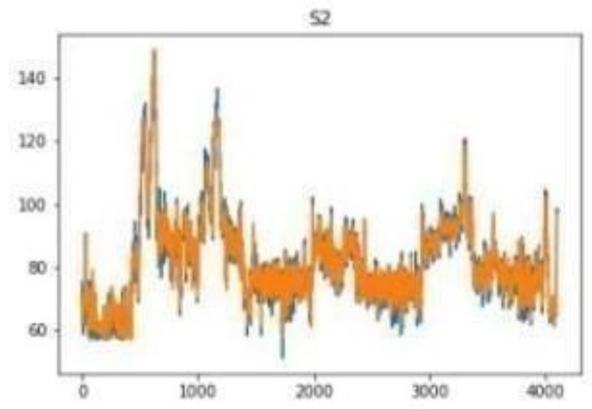
5.2.2 Convolutional Neural Networks:

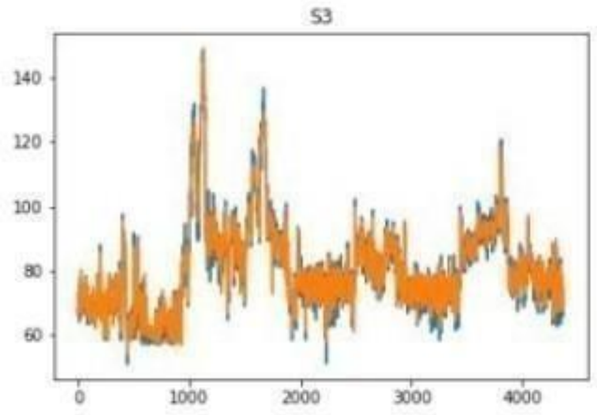
The accuracy metrics for the trained model are stated below,

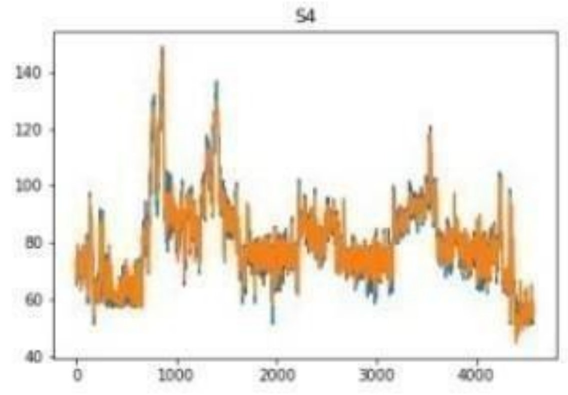
- Mean Absolute Error (MAE): 1.87
- Mean Squared Error (MSE): 6.55
- Root-Mean Squared Error (RMSE): 2.56
- Variance: 0.98

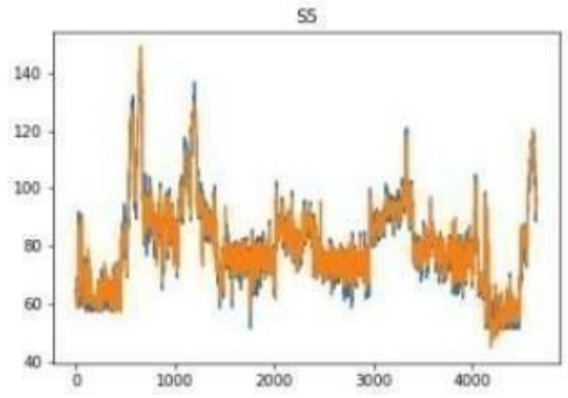
Let us see how the trained model predicted the heart rate for the 15 subjects individually.

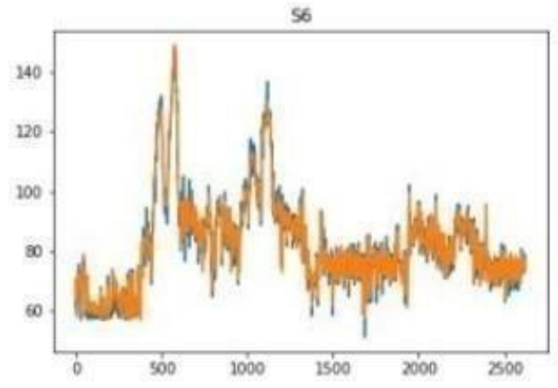

S1
160
140
120
100
80
60
40
0
1000
2000
3000
4000

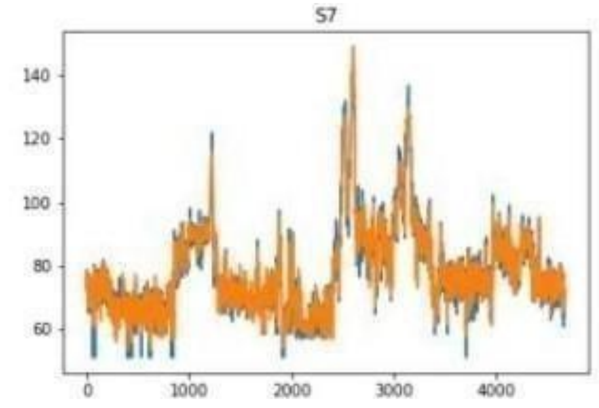

S2
140
120
100
80
60
0
1000
2000
3000
4000

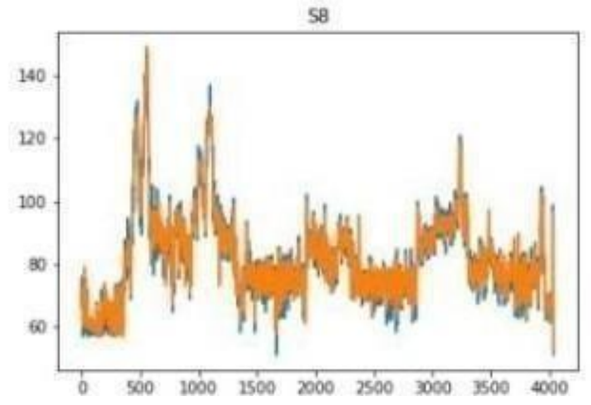

S3
140
120
100
80
60
0
1000
2000
3000
4000

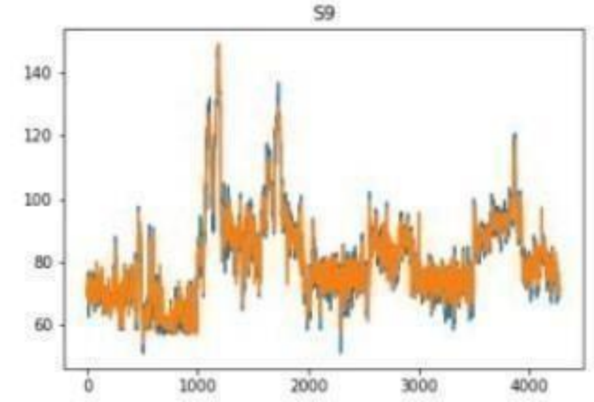

S4
140
120
100
80
60
40
0
1000
2000
3000
4000

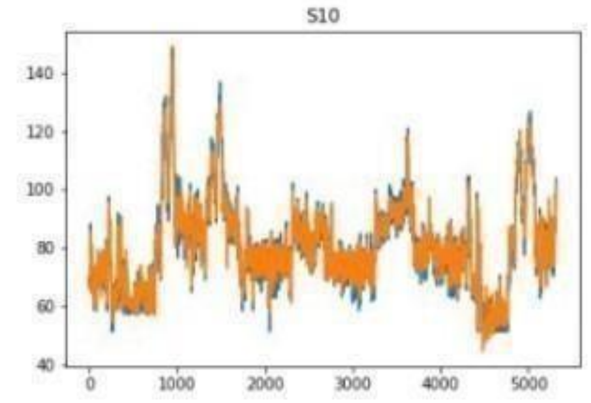

S5
140
120
100
80
60
40
0
1000
2000
3000
4000

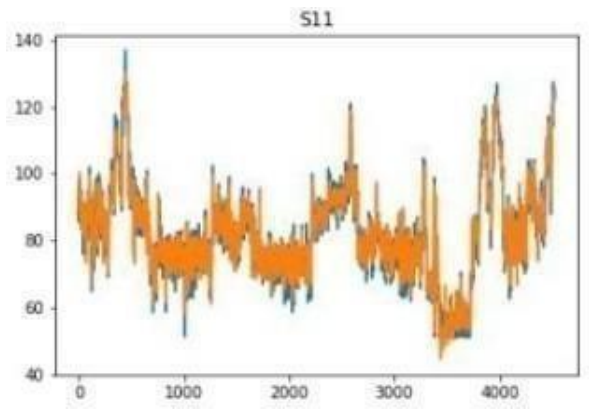

S6
140
120
100
80
60
0
500
1000
1500
2000
2500

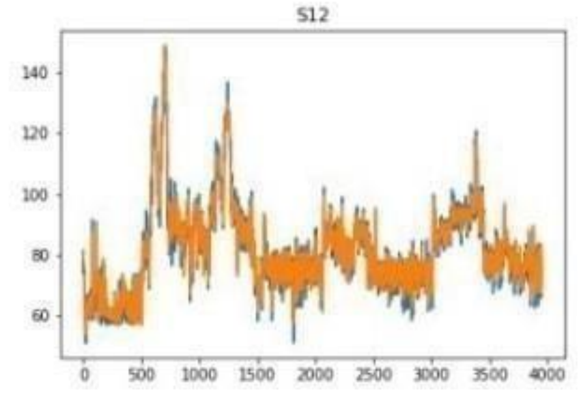

S7


S8


S9


S10


S11


S12

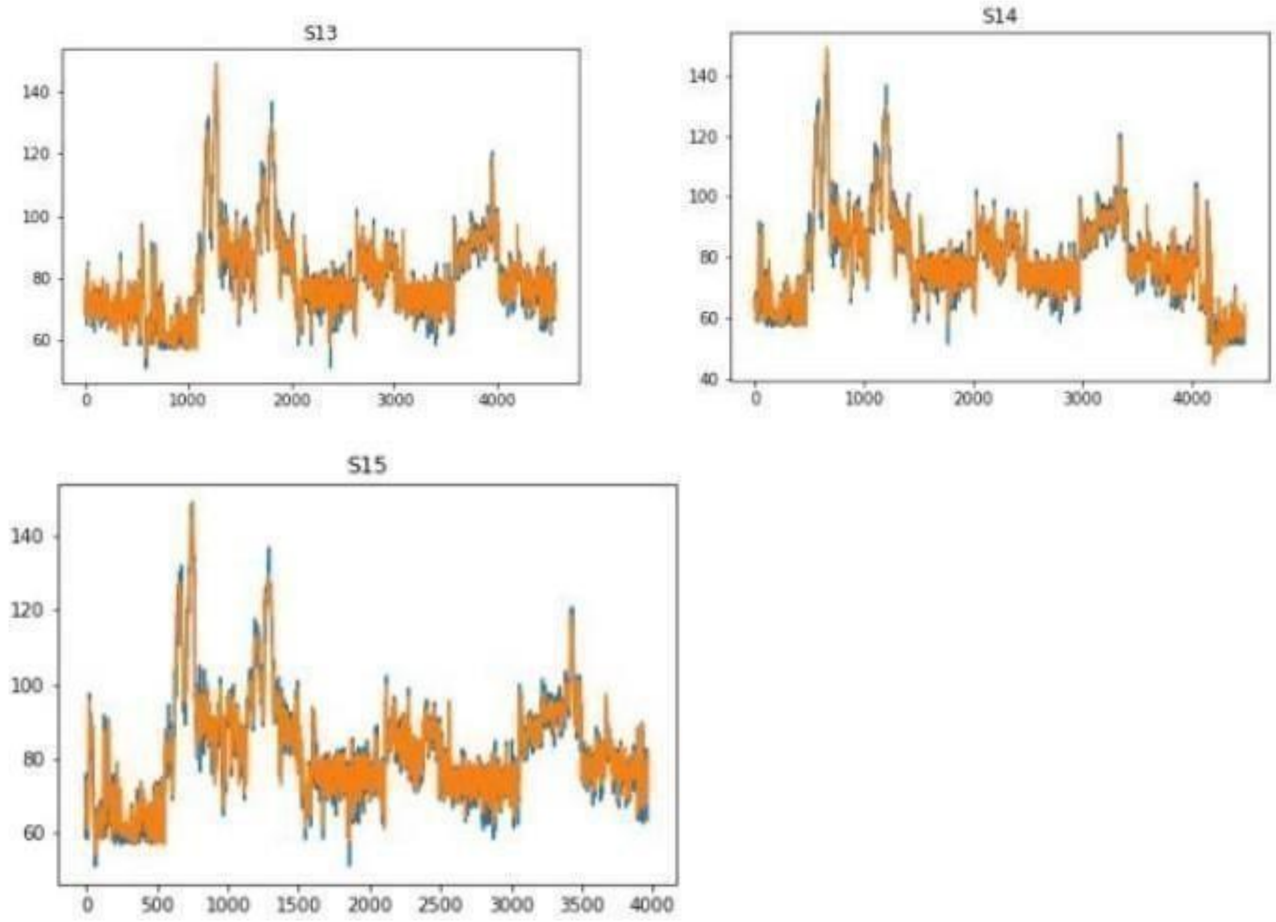


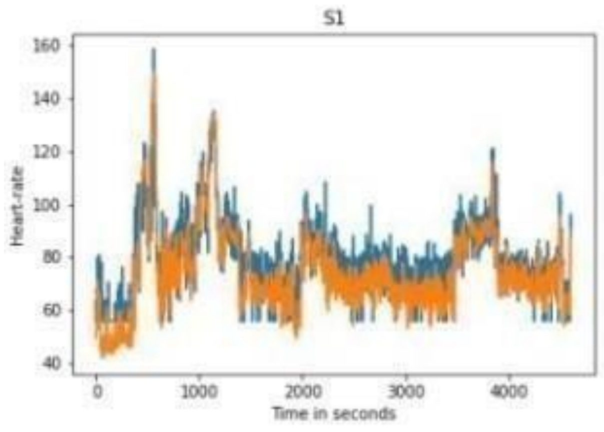
fig-9 predicted heart rate (CNN)

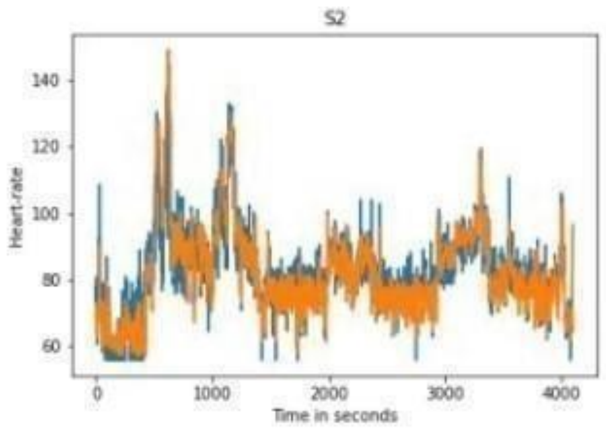
5.2.2 **Recurrent Neural Networks:**

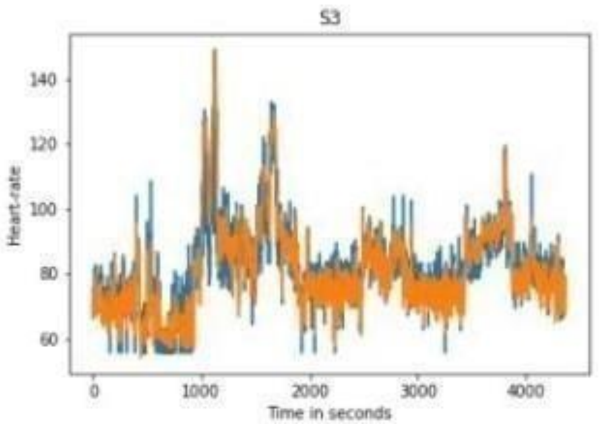
The accuracy metrics for the trained model are stated below,

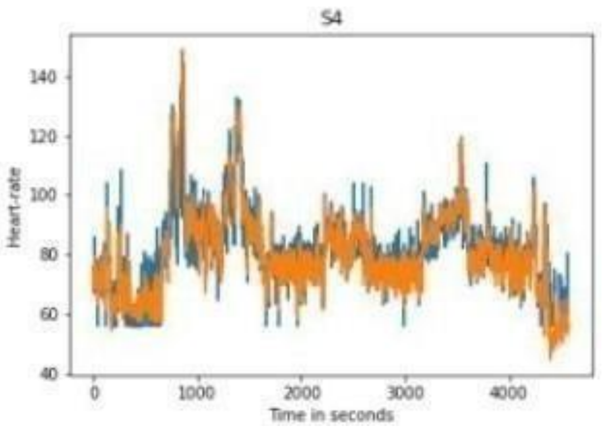
- Mean Absolute Error (MAE): 3.81
- Mean Squared Error (MSE): 28.52
- Root-Mean Squared Error (RMSE): 5.34
- Variance: 0.94

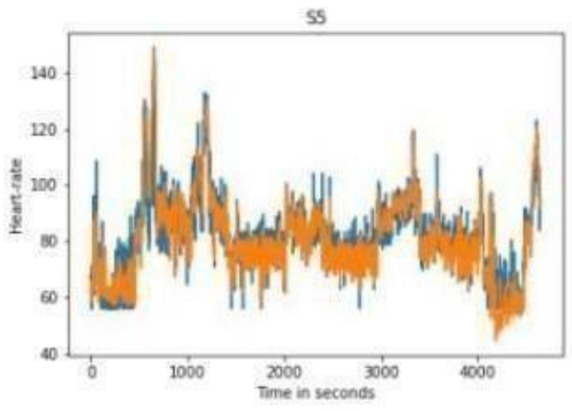
Let us see how the trained model predicted the heart rate for the 15 subjects individually.

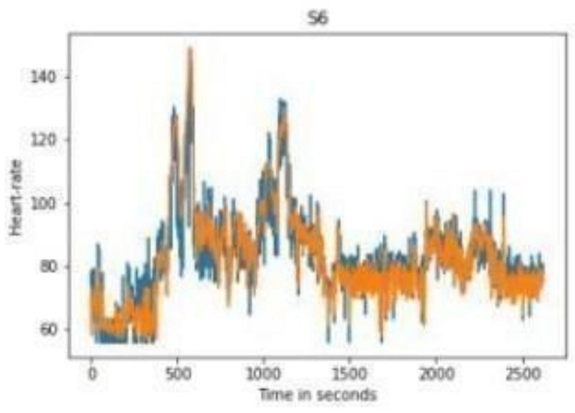

S1
Heart-rate
Time in seconds
160
140
120
100
80
60
40
0
1000
2000
3000
4000

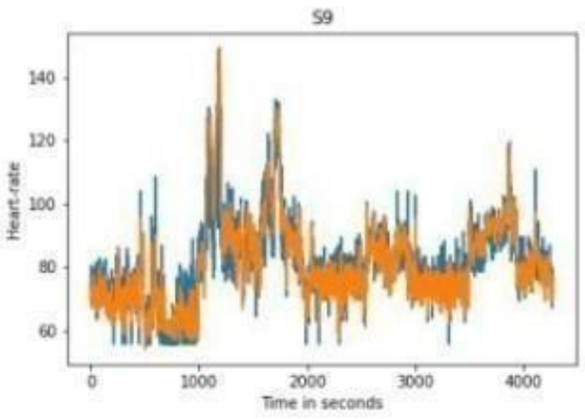

S2
Heart-rate
Time in seconds
140
120
100
80
60
0
1000
2000
3000
4000

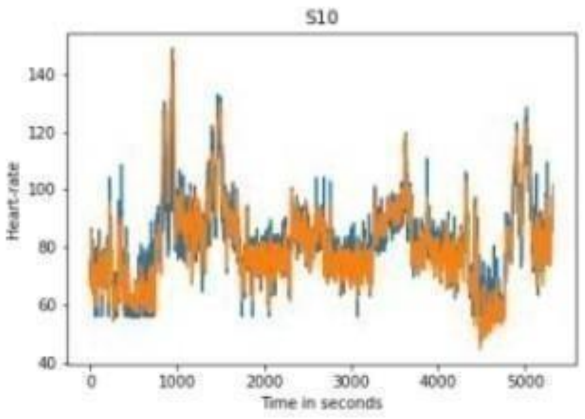

S3
Heart-rate
Time in seconds
140
120
100
80
60
0
1000
2000
3000
4000

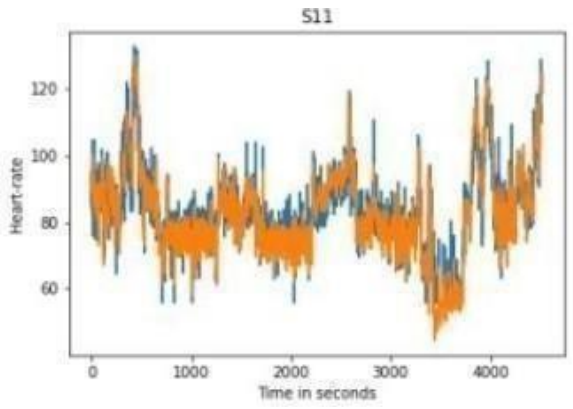

S4
Heart-rate
Time in seconds
140
120
100
80
60
40
0
1000
2000
3000
4000

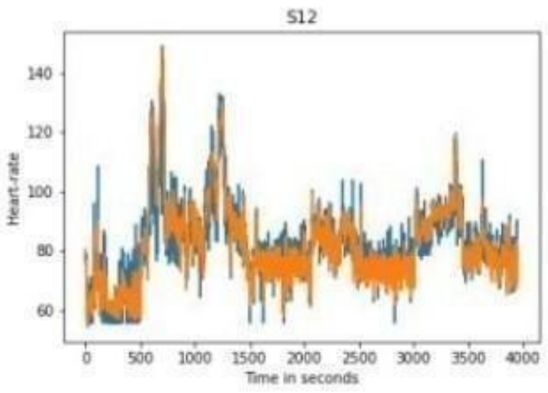

S5
Heart-rate
Time in seconds
140
120
100
80
60
40
0
1000
2000
3000
4000


S6
Heart-rate
Time in seconds
140
120
100
80
60
0
500
1000
1500
2000
2500


S9
Heart-rate
Time in seconds


S10
Heart-rate
Time in seconds


S11
Heart-rate
Time in seconds


S12
Heart-rate
Time in seconds

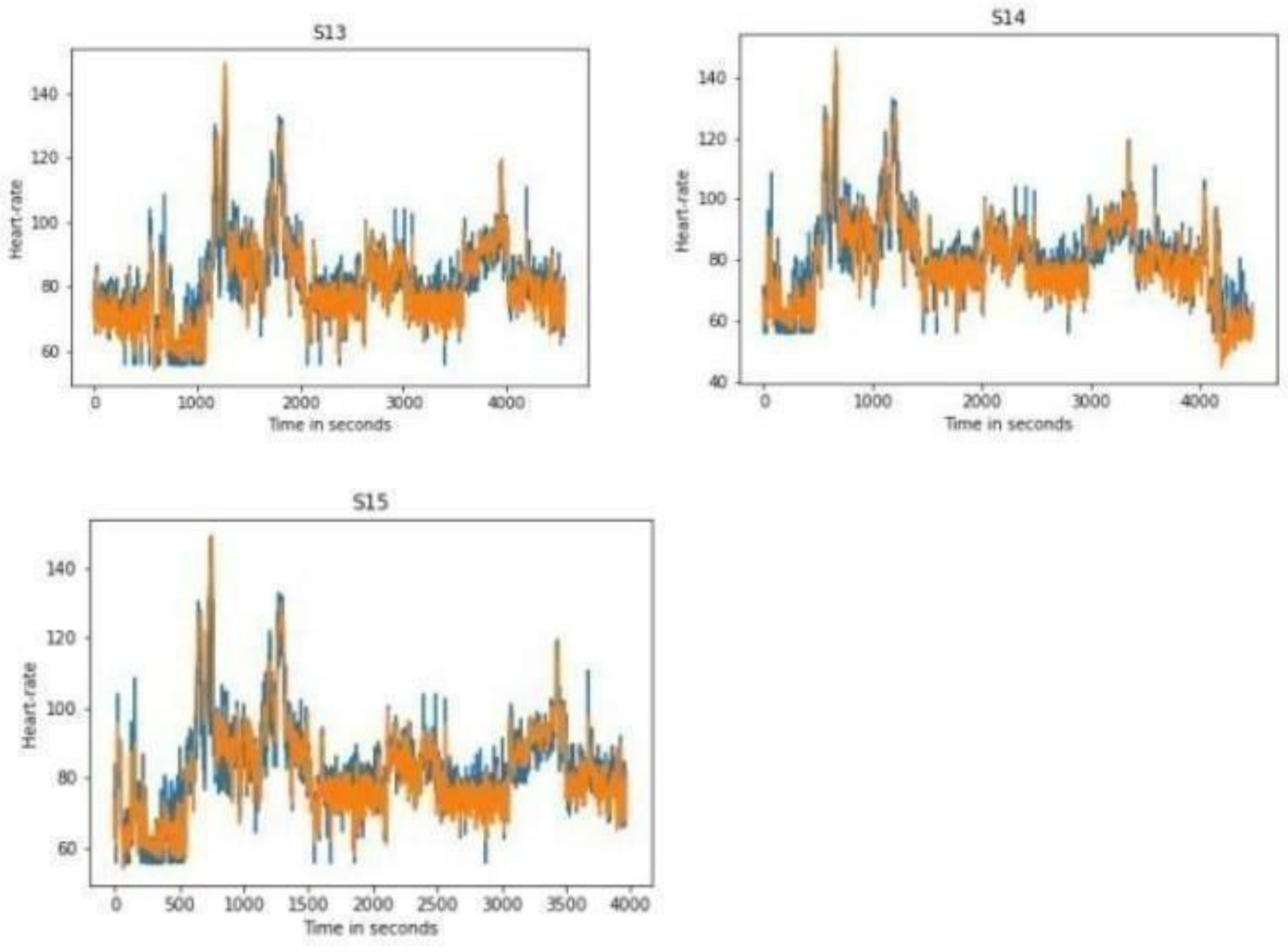


Fig-10 predicted heart rate(RNN)

5.3 Overview of metrics for the models:

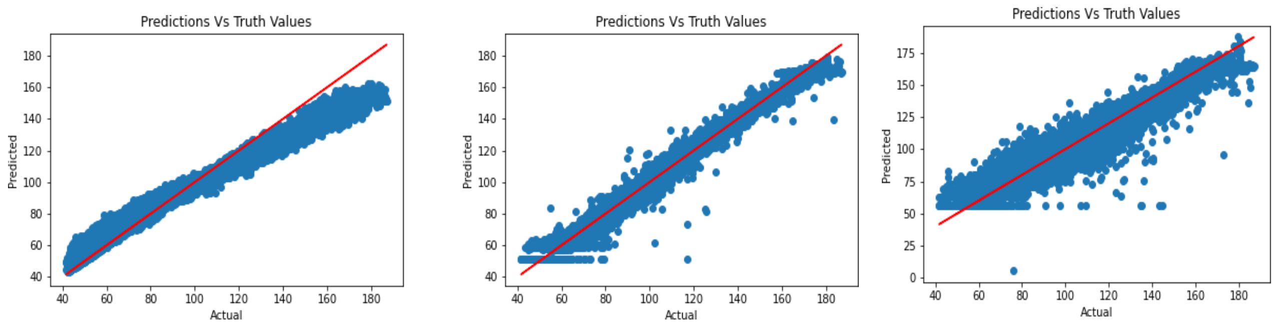
We’ll try to look at the results of all the three models and understand where the respective models are lagging and leading. Looking at the accuracy metrics we see that the CNN model seems to perform a better job, but looking at the predictions for different individuals by the three models, all three models are doing a fair job. We expected the RNN model to do better prediction than the CNN model because of the sequential nature of the data, but the results show the contrary. The reason lies in the training feature set of the data. For the CNN model, we have considered the last 6 seconds data as well for the predictions, instead of the mere 2-second data that the heart rate corresponds to. This helped us to get an RNN type of functionality in the training and gave us a lead over the other two models. RNN model would have performed better if there were more lstm layers in the model and sequential training would have helped. The random Forests Regression model was also performing well looking at the predictions for all the individuals. The increase in the number of trees gave a significant improvement in the beginning but wasn't helping much later.

5.4 comparison of results:

5.4.1 Predictions Vs Ground Truth comparison:

Let us now look at the predictions each model made side by side and how far the predictions are away from the ground truth.

Fig 11 -Predictions Vs Ground Truth for all the three models trained. (Random Forests Regression, Convolutional Neural Networks, Recurrent Neural)



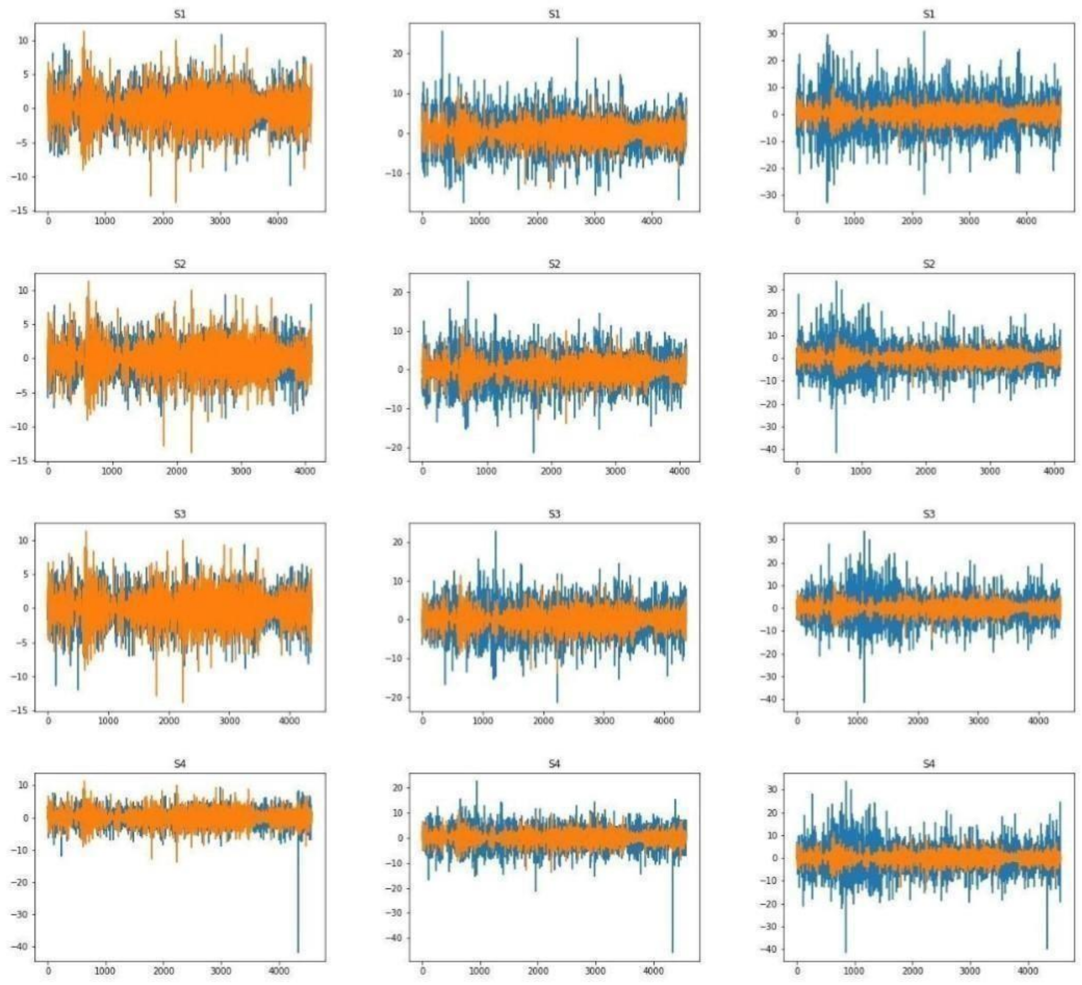
5.4.2 Heart Rate change over time comparison:

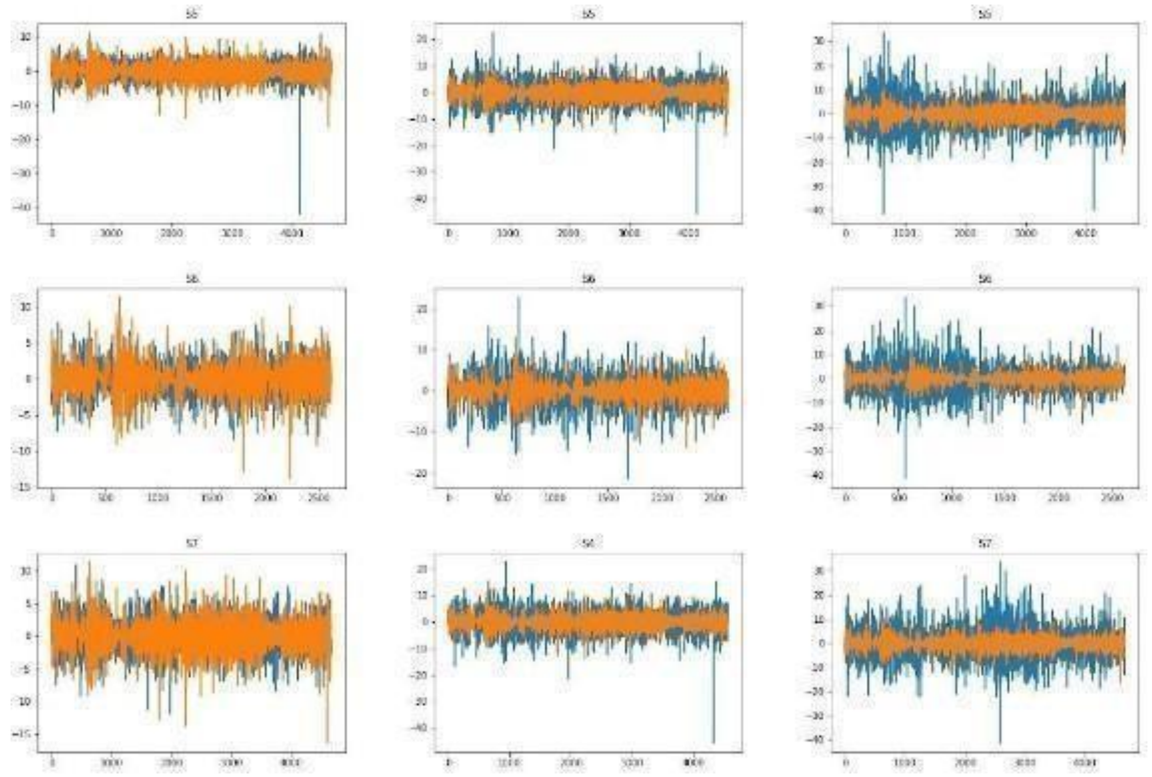
We can see that the CNN Model could predict the heart rate more closely to the ground truth for most cases, but there are some abnormal cases where the expected heart rates were very far away. The same is the case for the RNN model as well, predictions are a bit more off than the CNN model, but the abnormalities are still present. These abnormalities are reduced with Random Forest Model, but most predicted heart rates are far from the true value.

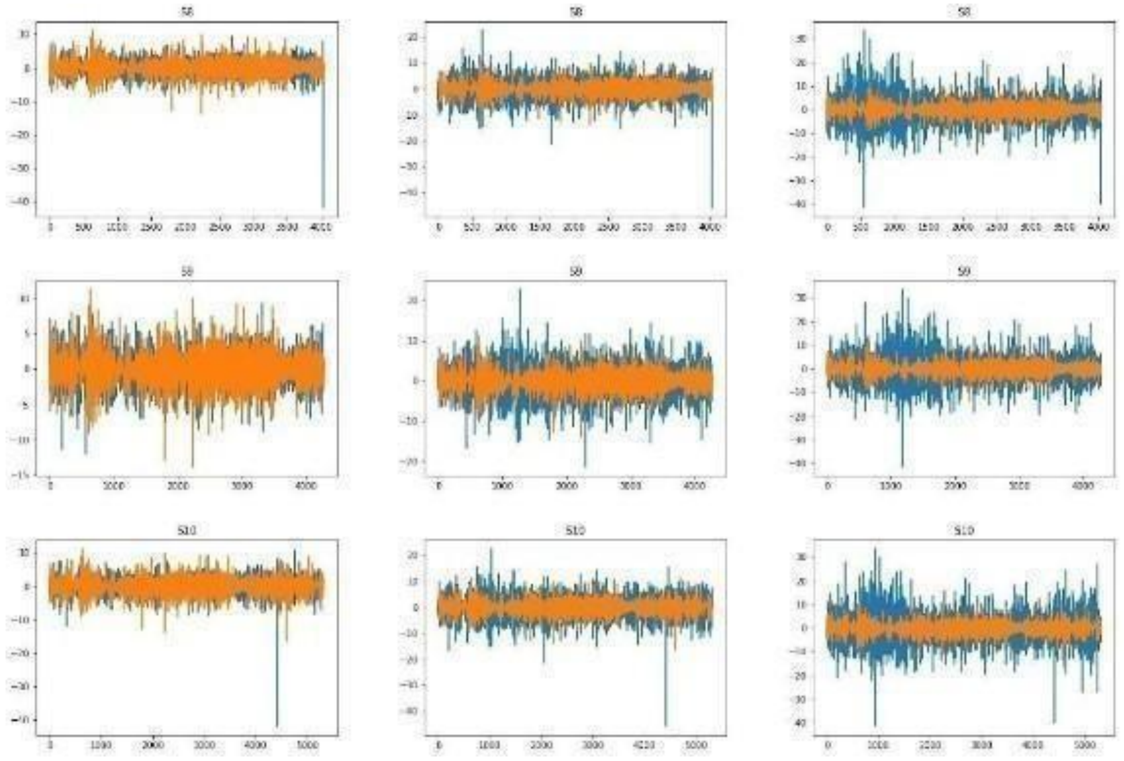
Let us now see how rapidly the heart rate changes as time passes for different models here. This can be obtained by observing the difference between the adjacent heart rates and comparing them side-by-side with the ground truth differences. The higher the peaks, the higher the change in heart rate.

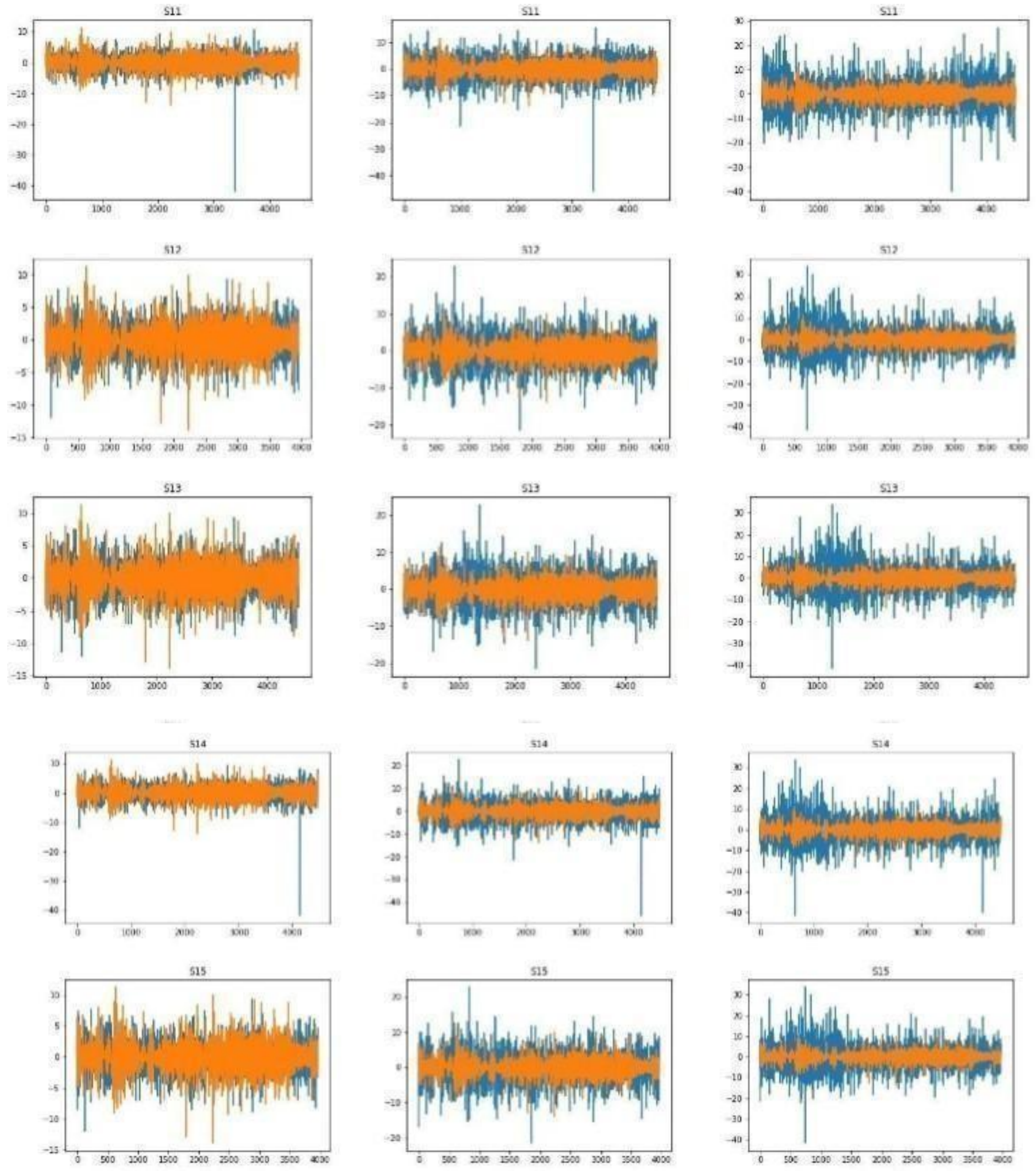
Note that the heart rates are calculated for a 2-second window, which means the heart rate changes every 2 seconds, and the peaks can be as high as ±20 bpm.

Fig-12 predictions of subjects




S5
S5
S5
S6
S6
S6
S7
S6
S7


S8
S8
S8
S8
S9
S9
S10
S10
S10



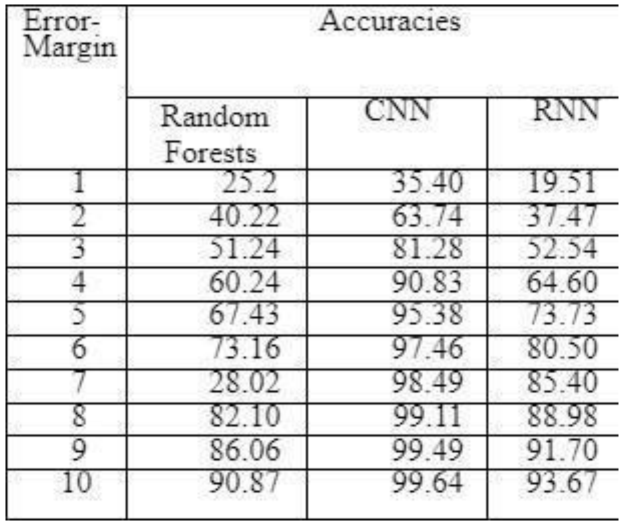
Figures showing predictions over all the subjects (Random Forests Regression, Convolutional Neural Networks, Recurrent Neural Networks – from left to right) (Blue – Predictions, Orange – Ground Truth)

Looking at the above graphs for different individuals across different models, the following observations can be made,

- Random Forests Model has lower peaks, implying no rapid change in heart rate predicted. This is desirable, but in some cases, it is even less than the ground truth peaks, which is not desirable.
- CNN Model has consistently higher peaks than ground truth, but always less than 20, which is the threshold, except in some cases; (like in S5, S11, etc.,) where there is a more extended peak at a point, which is also the case with Random Forests.
- RNN Model has pretty more extended peaks consistently over 30 and far away from ground truth difference, which is not desirable.

5.4.3 Error-margin Vs Accuracy comparison:

Let us now look at the accuracies for different error margins for all three models and compare them side by side.



Error-Margin	Accuracies		
	Random Forests	CNN	RNN
1	25.2	35.40	19.51
2	40.22	63.74	37.47
3	51.24	81.28	52.54
4	60.24	90.83	64.60
5	67.43	95.38	73.73
6	73.16	97.46	80.50
7	28.02	98.49	85.40
8	82.10	99.11	88.98
9	86.06	99.49	91.70
10	90.87	99.64	93.67

Fig - 13 Accuracies for different error margins for the model's

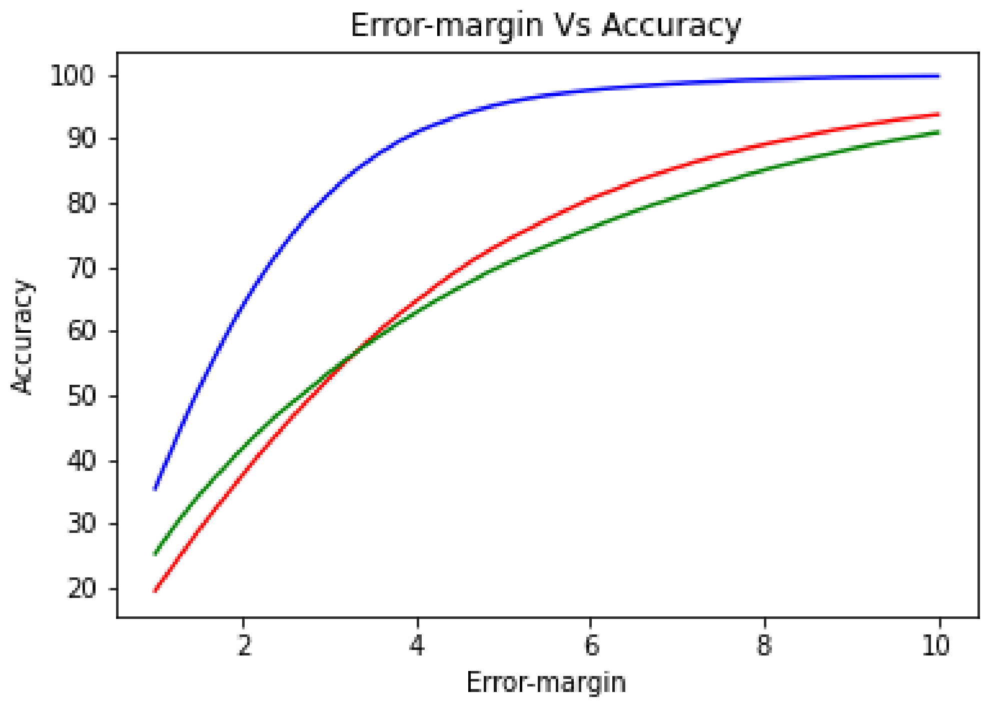


Fig - 14 Error-margin Vs accuracy (Blue – CNN, Red – RNN, Green – Random Forest)

The CNN model performs a better job at estimating the Heart rate than the other two models, with over 95% accuracy for an error margin of 5. Let us now look at the accuracies for different activities over various error margins for the three models.

5.4.4 Error-margin Vs Accuracy for different activities comparison:

Let us now look at the accuracies for different error margins for different activities for all three models and compare them side by side.

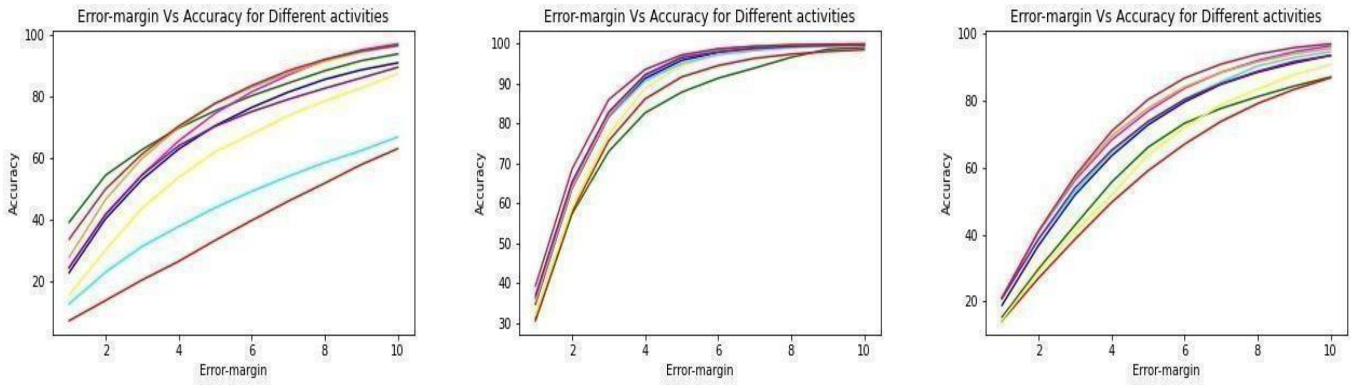


Fig - 15 Error-margin Vs accuracy for different activities (Random Forests Regression, Convolutional Neural Networks, Recurrent Neural Networks – from left to right) (blue - Transition Period, green – Sitting still, red – Ascending/ descending stairs, yellow – Table soccer, cyan - Cycling, magenta –
Driving a car, orange - Lunch, dark violet - Walking, crimson – Working)

Accuracy is lagging for some of the specific activities in Random Forests regression, especially for activities working and cycling. For the CNN model, some activities initially lagged but got ahead with an increased error margin. For the RNN model, table soccer activity weakened a bit more than other activities and got better later with an error margin increase.

This way of looking at the predictions paves the way for training a whole new classification model wherein the heart rates are estimated in terms of ranges like we estimate the heart rate to be in a particular range. This becomes a classification problem altogether where heart rates are estimated to be in a specific predetermined range. This ensures that the heart rate is well between 40-200, which is the case for our data. For some of the predictions, we are getting heart rates as low as ten and as high as 220, resulting in catastrophic results. The classification model can be used as a verifier for our regression model to be even more precise at estimating the heart rate and predicting the heart rate well in the range. Different classification models were trained and used for testing and improving the regression models in some extreme cases. We'll look at them and how they can be used to verify and improve the model.

5.4.5 Classification Models for Verification:

We try to train a classification model where the output classes will be the heart rate range. Here we are considering a range of 5 to obtain a good accuracy for an error margin of 5. Let us see the distribution of the data over different fields.

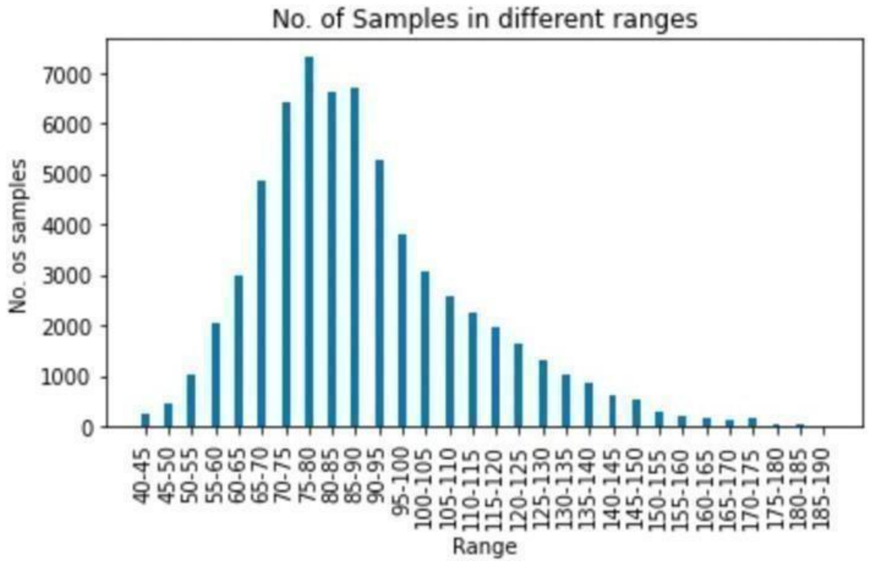


Fig - 16 Sample distribution over various ranges

These 30 ranges will be our output classes, and a classification model can be trained to predict the heart rate in one of these ranges. Two classification models have been trained in this regard.

- Random Forests Classifier
- CNN Classification Model

1. **Random Forests Classifier:**
A Random Forests Classifier model has been trained with the 30 classes and was used to verify and improve the model. It gave an accuracy of about 90% on the train set and about 87% on the test set.

2. **Convolutional Neural Networks Classification:**
A CNN classification model has been trained on the data with these 30 classes. The same neural network structure used for regression has been used with different activations and final layers. Even this model gave an accuracy of about 90% on both train and test sets.

How can these models help in verification and improving the model performance?

- Let us say we have a heart rate far away from the previous heart rate (2 sec earlier), like more than 20bpm away, and then this classification model can be used to find the range this heart rate falls under and have the mean as the heart rate.

- With the classification model, we won't be getting any heart rates that are not in our range and can quickly get rid of abnormal heart rates.

- We can predict the heart rate just with the help of this classification model, although this is only possible using a CNN model. For the CNN model, we get a vector of 30 (in this case), giving out the probability of the heart rate falling in the specific range. The maximum of this is the range which it falls under. These probabilities can be used to predict the heart rate. If the likelihood that it falls under this range is 100%, then the heart rate can be expected as the median of this range. Anything less than that can be thought of as normal distribution, and the heart rate can be greater than or less than the median, depending on the probabilities of the adjacent ranges.

- Even if this classification model predicts a faraway heart rate, we can have a pseudo heart rate within the permissible 20 bpm range to reduce the abnormality in heart rate. A simple notification shows this abnormality which can be monitored and intimate with the user in case of repeated irregularities.

6. Conclusion:

Continuous Heart Rate monitoring is of utmost importance, as many long-term diseases can be predicted way before by continuously monitoring the heart rate. In addition, abnormal Heart Rate is a direct consequence of Hypertension, a silent killer that can result in various major health problems if ignored.

With significant advancements being made every year, heart rate monitoring has been made easy with the help of wearables. For example, an Apple watch event showed that continuous heart rate monitoring helped many people go and visit a doctor before the situation goes out of hand after seeing abnormal conditions on their watch [10].

The relation between PPG and Heart Rate can be exploited and used to monitor Heart rate non-invasively continuously. With Machine Learning taking new leaps every year, it has been made possible to estimate Heart Rate using a Machine Learning model trained on PPG signals. Machine Learning models ensure that we can estimate Heart rate even on the new data.

We have seen three different Machine Learning models trained on the data and their respective accuracy metrics. The models can still be improved by tweaking various model parameters. This is a safety-critical system, and even the smallest of most negligible errors are to be taken care of. This can be assured with intensive formal testing. Nevertheless, the models performed considerably better than the previous work on the chosen dataset.

With wearables becoming more essential these days, more and more companies are coming forward with wearables. The global wearable healthcare devices market will reach USD 46.6 billion by 2025 from USD 18.4 billion in 2020, at a CAGR of 20.5% from 2020 to 2025. This shows that wearables can be easily affordable for many people, and continuous health monitoring can be ensured. With more people using wearables, there’s a need for better estimation of Heart Rate with the help of PPG signals.

Health monitoring is of prime importance. People generally ignore regular health check-ups. With the help of non-invasive techniques for health monitoring using wearables, this can be made more accessible for people to track their health metrics and prevent many major health problems.

7 . References:

Deep PPG: Large-Scale Heart rate Estimation with Convolutional Neural Networks by Attila Reiss, Ina Indlefoker, Phillip Schmidt, Kristof Van Laerhoven. https://www.mdpi.com/1424- 8220/19/14/3079

[1] PPG-DaLiA Dataset. https://archive.ics.uci.edu/ml/datasets/PPG-DaLiA

[2] WESAD

https://archive.ics.uci.edu/ml/datasets/WESAD+%28Wearable+Stress+and+Affect+Detection%29

[3] A Machine Learning Approach for Heart Rate Estimation from PPG Signal using Random Forest Regression Algorithm by Shikder Shafiul Bashar.

https://www.researchgate.net/publication/332726633_A_Machine_Learning_Approach_for_Heart_Rate_Estimation_from_PPG_Signal_using_Random_Forest_Regression_Algorithm

[4] Cup, I.S.P. Heart Rate Monitoring During Physical Exercise using Wrist-Type Photoplethysmographic (PPG) Signals. 2015. https://sites.google.com/site/researchbyzhang/ieeespcup2015

[5] Robust heart rate estimation using wrist-based PPG signals in the presence of intense physical activities by Chengzhi Zong, Roozbeh Jafari. https://pubmed.ncbi.nlm.nih.gov/26738168/

[6] Noise-Robust Heart Rate Estimation Algorithm from Photoplethysmography Signal with Low Computational Complexity. https://www.hindawi.com/journals/jhe/2019/6283279/

[7] RespiBAN Professional. http://www.biosignalsplux.com/en/respiban-professional

[8] Empatica. E4 Wristband. https://www.empatica.com/research/e4/

[9] Heart Rate Tracking using Wrist-Type Photoplethysmographic (PPG) Signals during Physical Exercise with Simultaneous Accelerometry by Mahdi Boloursaz, Ehsan Asadi, Mohsen Eskandari, Shahrzad Kiani. https://arxiv.org/pdf/1504.04785.pdf

[10] Apple Watch Real Stories https://www.youtube.com/watch?v=0tqB4jnCxqA

[11] IEEE – Accurate Heart Rate Monitoring During Physical Exercises Using PPG. https://ieeexplore.ieee.org/document/7867772

[12] Random Forests, Convolutional Neural Networks, Recurrent Neural Networks modules in Python.

[13] How fast can the heart rate increase or decrease per sec - https://www.researchgate.net/post/How_fast_can_the_heart_rate_increase_decrease_bpm_sec

8. APPENDIX

A: source/pseudo code

Convolutional Neural Network (CNN): cnn_model.ipynb

```
import pickle
import numpy as np
from math import floor
from tqdm import tqdm
lengths=[]
activities = []
ppg =  []
lines = []
X = []
for i in tqdm(range(1,16)):
   st ='S'+str(i)+'/'+'S'+str(i)+'.pkl'
   with open(st, 'rb') as f:
       temp = pickle.load(f, encoding='latin1')
   ppg = temp['signal']['wrist']['BVP'].tolist()
   qppg =[]
   lengths.append(int(len(ppg)/128)-3)
   ac = temp['activity']
   tempac =[]
   for l in range(int(floor(ac.shape[0]/8))-3):
       activities.append(max(ac[l*8:(l+4)*8]))
   #print(len(activities),len(temp['label']))
   for p in range(int(len(ppg)/128)-3):
       t = np.zeros(512)
       k = 0
       for j in range(p*128,(4+p)*128):
           t[k] = ppg[j][0]
           k = k+1
       qppg.append(t)
   X.extend(qppg)
   lines.extend(temp['label'])
   #activities.extend(temp['activity'].tolist())
   #print(i)
import numpy as np
X=np.array(X)
Y = []
for i in lines:
   t = np.zeros(1)
   t[0]=i
   Y.append(t)
Y = np.array(Y)
print(X.shape,Y.shape)
import scipy
import numpy as np
from sklearn import model_selection
X_train, X_test, y_train, y_test = model_selection.train_test_split(X, Y,
test_size=0.10, random_state=42)
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation, Conv1D, Reshape,
MaxPooling1D, Flatten
```

```
max_words=512
model = Sequential()
model.add(Dense(512,activation='relu'))
model.add(Dense(256,activation='relu'))
model.add(Dense(512,activation='relu'))
model.add(Reshape((1,4,128)))
model.add(Conv1D(256,3,activation='relu',input_shape=[4,128]))
model.add(Reshape((4,128)))
model.add(MaxPooling1D(pool_size=2))
model.add(Reshape((1,4,64)))
model.add(Conv1D(256,3,activation='relu',input_shape=[4,128]))
model.add(Reshape((4,128)))
model.add(MaxPooling1D(pool_size=2))
model.add(Flatten())
model.add(Dense(128,activation='relu'))
model.add(Dense(512,activation='relu'))
model.add(Dense(128,activation='relu'))
model.add(Dense(1))
model.compile(optimizer='Adam',loss='mse')
print(model.metrics_names)
from keras.callbacks import EarlyStopping, ModelCheckpoint

callbacks = [EarlyStopping(monitor='loss', patience=15),
             ModelCheckpoint('cnn_model.h5', save_best_only=True,
                             save_weights_only=False)]
batch_size = 64
epochs = 300
```

```
history = model.fit(X_train, y_train, batch_size=batch_size,
epochs=epochs,callbacks=callbacks,validation_data=(X_test,y_test),
verbose=1)
score = model.evaluate(X_test, y_test, batch_size=batch_size, verbose=1)
import keras
loaded=keras.models.load_model('cnn_model.h5')
y_pred = loaded.predict(X)
y_predict=[]
for i in range(y_pred.shape[0]):
    y_predict.append(y_pred[i][0][0])
from sklearn import metrics
print('MAE:', metrics.mean_absolute_error(Y, y_predict))
print('MSE:', metrics.mean_squared_error(Y, y_predict))
print('RMSE:', np.sqrt(metrics.mean_squared_error(Y, y_predict)))
print('VarScore:',metrics.explained_variance_score(Y,y_predict))
# Visualizing Our predictions
import matplotlib.pyplot as plt
#fig = plt.figure(figsize=(10,5))
plt.scatter(Y,y_predict)
plt.plot(Y,Y,'r')
plt.title('Predictions Vs Truth Values')
plt.xlabel('Actual')
plt.ylabel('Predicted')
#plt.savefig('figures_final/predictions_cnn.png')
counter=0
for i in range(1,16):
    #plt.subplot(15,1,i)
    xaxis=range(lengths[i-1])
    yaxis=y_predict[counter:lengths[i-1]+counter]
    yaxis1=Y[counter:lengths[i-1]+counter]
    counter= lengths[i-1]
    plt.plot(xaxis,yaxis)
    plt.plot(xaxis,yaxis1)
    plt.title('S'+str(i))
    plt.xlabel('Time in seconds')
    plt.ylabel('Heart-rate')
    plt.show()
    #plt.savefig('figures_final/S'+str(i)+'_cnn.png')
    #plt.clf()
import matplotlib.pyplot as plt
counter =0
for i in range(1,16):
    #plt.subplot(15,1,i)
    xaxis=range(lengths[i-1]-1)
    yaxis = y_predict[counter:lengths[i-1]+counter]
    yaxis1=Y[counter:lengths[i-1]+counter]
    diff_list1 = []
    diff_list2 = []
    diff = 0
    for j in range(1,len(yaxis)):
        diff_list1.append(yaxis[j]-yaxis[j-1])
        diff_list2.append(Y[j]-Y[j-1])
    counter= lengths[i-1]
    plt.plot(xaxis,diff_list1)
    plt.plot(xaxis,diff_list2)
    plt.title('S'+str(i))
    plt.show()
    #plt.savefig('figures_final/S'+str(i)+'_diff_cnn.png')
    #plt.clf()
y_acc = []
x_acc = []
threshold = 1
while threshold < 10:
    counter =0
    for i in range(len(y_predict)):
        if abs(Y[i][0]-y_predict[i])<threshold:
            counter +=1
    y_acc.append((counter/y_pred.shape[0])*100)
    x_acc.append(threshold)
    threshold += 0.1
import matplotlib.pyplot as plt
#fig = plt.figure(figsize=(10,5))
plt.plot(x_acc,y_acc)
plt.title('Error-margin Vs Accuracy')
plt.xlabel('Error-margin')
plt.ylabel('Accuracy')
#plt.savefig('figures_final/errorvsacc_graph_cnn.png')
acts = [[0 for i in range(10)] for j in range(9)]

for threshold in range(1,11):
    counter = 0
    positives = [0,0,0,0,0,0,0,0,0]
    totals = [0,0,0,0,0,0,0,0,0]
    for i in range(y_pred.shape[0]):
        if abs(Y[i][0]-y_pred[i])<threshold:
            positives[int(activities[i])]+=1
        totals[int(activities[i])]+=1
    for i in range(9):
        acts[i][threshold-1]=positives[i]/totals[i]
colr =['b','g','r','yellow','cyan','magenta','orange','darkviolet','crimson']
for i in range(9):
    plt.plot(range(1,11),[j*100 for j in acts[i]],color=colr[i])
plt.xlabel("Error-margin")
plt.ylabel("Accuracy")
plt.title("Error-margin Vs Accuracy for Different activities")
#plt.savefig('figures_final/act_cnn.png')
```

Random Forest Regression: random_forests.ipynb

```
from sklearn.ensemble import RandomForestRegressor
regressor = RandomForestRegressor(n_estimators=300,random_state=0)
regressor.fit(X_train,y_train)
import joblib
joblib.dump(regressor,"random_forests_model")
regressor = joblib.load('random_forests_model')
y_predict = regressor.predict(X)
from sklearn import metrics
print('MAE:', metrics.mean_absolute_error(Y, y_predict))
print('MSE:', metrics.mean_squared_error(Y, y_predict))
print('RMSE:', np.sqrt(metrics.mean_squared_error(Y, y_predict)))
print('VarScore:',metrics.explained_variance_score(Y,y_predict))
import matplotlib.pyplot as plt
plt.scatter(Y,y_predict)
plt.plot(Y,Y,color='red')
plt.title('Predictions Vs Truth Values')
plt.xlabel('Actual')
plt.ylabel('Predicted')
#plt.savefig('figures_final/predictions_rf.png')
counter=0
for i in range(1,16):
    #plt.subplot(15,1,i)
    xaxis=range(lengths[i-1])
    yaxis=y_predict[counter:lengths[i-1]+counter]
    yaxis1=Y[counter:lengths[i-1]+counter]
    counter= lengths[i-1]
    plt.plot(xaxis,yaxis)
    plt.plot(xaxis,yaxis1)
    plt.title('S'+str(i))
    plt.xlabel('Time in seconds')
    plt.ylabel('Heart-rate')
    plt.show()
    #plt.savefig('figures_final/S'+str(i)+'_rf.png')
    #plt.clf()
import matplotlib.pyplot as plt
counter =0
for i in range(1,16):
    #plt.subplot(15,1,i)
    xaxis=range(lengths[i-1]-1)
    yaxis = y_predict[counter:lengths[i-1]+counter]
    yaxis1=Y[counter:lengths[i-1]+counter]
    diff_list1 = []
    diff_list2 = []
    diff = 0
    for j in range(1,len(yaxis)):
        diff_list1.append(yaxis[j]-yaxis[j-1])
        diff_list2.append(Y[j]-Y[j-1])
    counter= lengths[i-1]
    plt.plot(xaxis,diff_list1)
    plt.plot(xaxis,diff_list2)
    plt.title('S'+str(i))
    plt.show()
    #plt.savefig('figures_final/S'+str(i)+'_diff_cnn.png')
    #plt.clf()
y_acc = []
x_acc = []
threshold = 1
while threshold < 10:
   counter =0
   for i in range(len(y_predict)):
       if abs(Y[i]-y_predict[i])<threshold:
           counter +=1
   y_acc.append((counter/len(y_predict))*100)
   x_acc.append(threshold)
   threshold += 0.1
import matplotlib.pyplot as plt
#fig = plt.figure(figsize=(10,5))
plt.plot(x_acc,y_acc)
plt.title('Error-margin Vs Accuracy')
plt.xlabel('Error-margin')
plt.ylabel('Accuracy')
#plt.savefig('figures_final/errorvsacc_graph_rf.png')
acts = [[0 for i in range(10)] for j in range(9)]

for threshold in range(1,11):
   counter = 0
   positives = [0,0,0,0,0,0,0,0,0]
   totals = [0,0,0,0,0,0,0,0,0]
   for i in range(y_predict.shape[0]):
       if abs(Y[i]-y_predict[i])<threshold:
           positives[int(activities[i])]+=1
       totals[int(activities[i])]+=1
   for i in range(9):
       acts[i][threshold-1]=positives[i]/totals[i]
colr =['b','g','r','yellow','cyan','magenta','orange','darkviolet','crimson']
for i in range(9):
   plt.plot(range(1,11),[j*100 for j in acts[i]],color=colr[i])
plt.xlabel("Error-margin")
plt.ylabel("Accuracy")
plt.title("Error-margin Vs Accuracy for Different activities")
#plt.savefig('figures_final/act_rf.png')
```

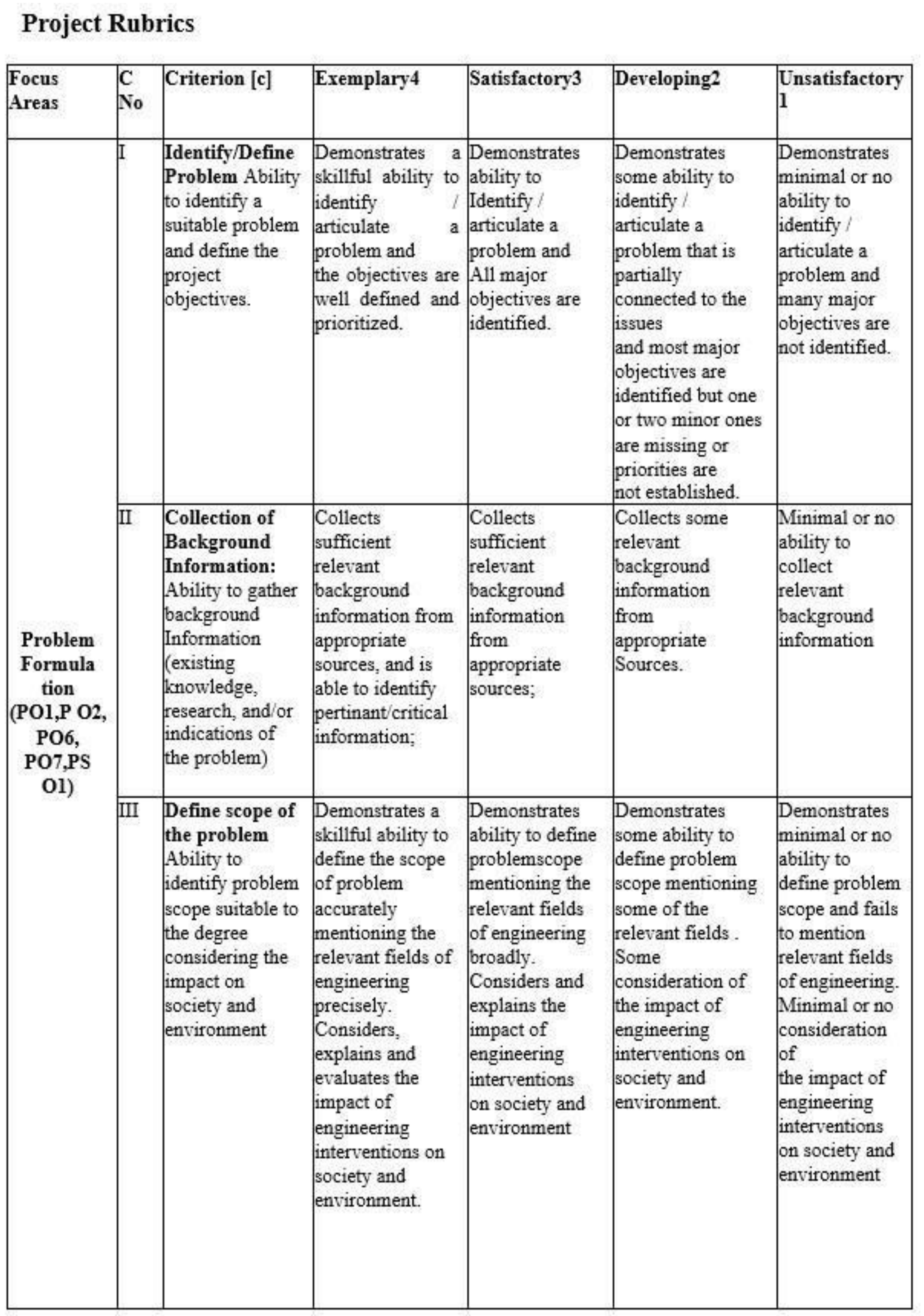
Recurrent Neural Network: rnn_model.ipynb

```
from keras.models import Sequential
from keras.layers import Dense, Activation, Conv1D, Reshape, Masking,
LSTM, MaxPooling1D, Flatten
```

```
max_words=512
model = Sequential()
model.add(Dense(512,activation='relu'))
model.add(Dense(256,activation='relu'))
model.add(Dense(512,activation='relu'))
model.add(Reshape((1,512)))
model.add(Masking(mask_value=0.0))
model.add(LSTM(512, return_sequences=False, activation = 'relu' ,
        dropout=0.1, recurrent_dropout=0.1))
model.add(Reshape((1,4,128)))
model.add(Conv1D(256,3,activation='relu',input_shape=[4,128]))
model.add(Reshape((1,4,128)))
model.add(MaxPooling1D(pool_size=2))
model.add(Flatten())
model.add(Dense(128,activation='relu'))

model.add(Dense(512,activation='relu'))
model.add(Dense(128,activation='relu'))
model.add(Dense(1))
model.compile(optimizer='Adam',loss='mse')
print(model.metrics_names)
from keras.callbacks import EarlyStopping, ModelCheckpoint

callbacks = [EarlyStopping(monitor='loss', patience=15),
       ModelCheckpoint('rnn_model.h5', save_best_only=True,
               save_weights_only=False)]
batch_size = 64
epochs = 300
```

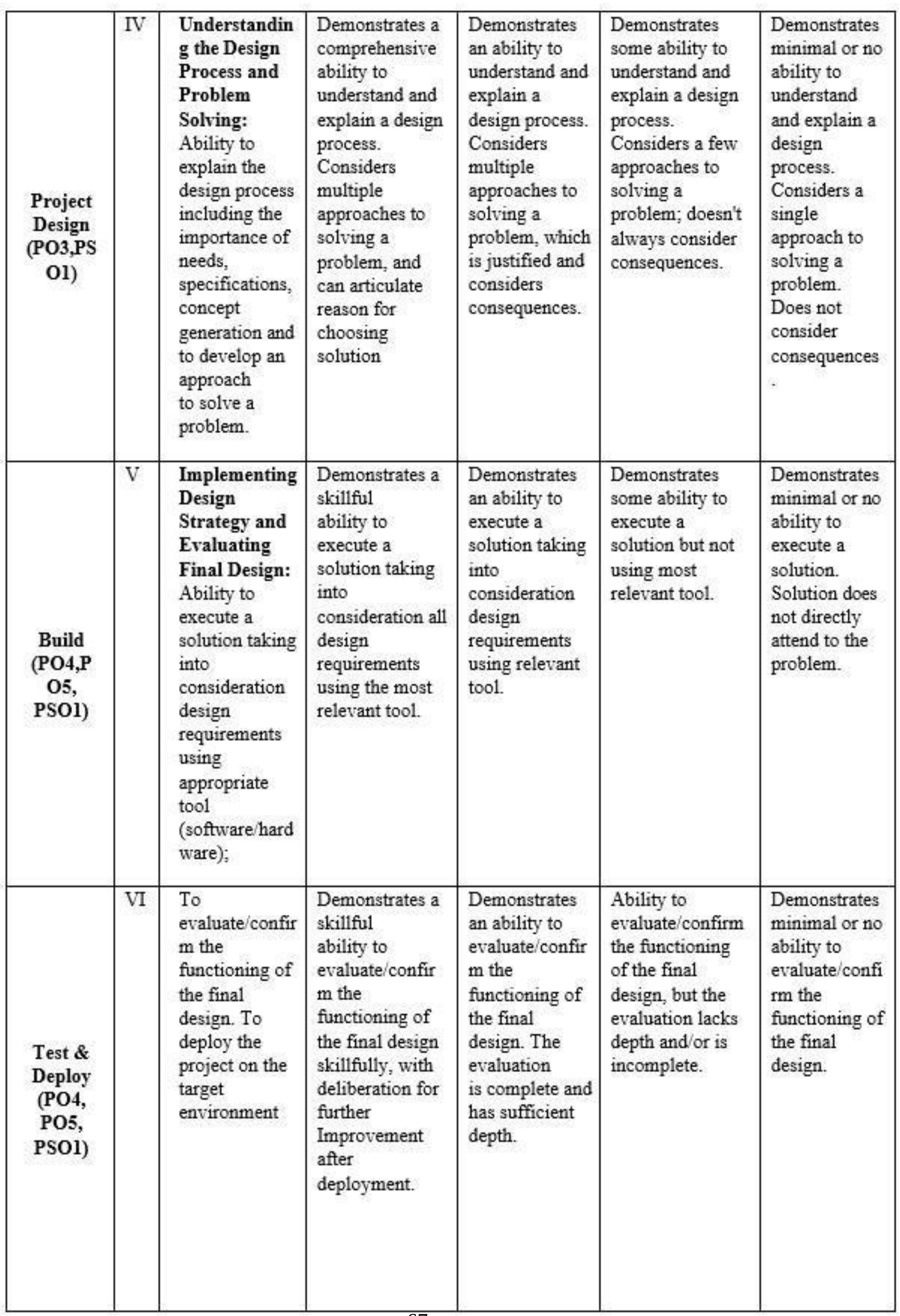
```
history = model.fit(X_train, y_train, batch_size=batch_size,
epochs=epochs,callbacks=callbacks,validation_data=(X_test,y_test),
verbose=1)
score = model.evaluate(X_test, y_test, batch_size=batch_size, verbose=1)
import keras
loaded=keras.models.load_model('rnn_model.h5')
y_pred = loaded.predict(X)
y_predict=[]
for i in range(y_pred.shape[0]):
    y_predict.append(y_pred[i][0][0])
from sklearn import metrics
print('MAE:', metrics.mean_absolute_error(Y, y_predict))
print('MSE:', metrics.mean_squared_error(Y, y_predict))
print('RMSE:', np.sqrt(metrics.mean_squared_error(Y, y_predict)))
print('VarScore:',metrics.explained_variance_score(Y,y_predict))
# Visualizing Our predictions
import matplotlib.pyplot as plt
#fig = plt.figure(figsize=(10,5))
plt.scatter(Y,y_predict)
plt.plot(Y,Y,'r')
plt.title('Predictions Vs Truth Values')
plt.xlabel('Actual')
plt.ylabel('Predicted')
#plt.savefig('figures_final/predictions_rnn.png')
counter=0
for i in range(1,16):
    #plt.subplot(15,1,i)
    xaxis=range(lengths[i-1])
    yaxis=y_predict[counter:lengths[i-1]+counter]
    yaxis1=Y[counter:lengths[i-1]+counter]
    counter= lengths[i-1]
    plt.plot(xaxis,yaxis)
    plt.plot(xaxis,yaxis1)
    plt.title('S'+str(i))
    plt.xlabel('Time in seconds')
    plt.ylabel('Heart-rate')
    plt.show()
    #plt.savefig('figures_final/S'+str(i)+'_rnn.png')
    #plt.clf()
import matplotlib.pyplot as plt
counter =0
for i in range(1,16):
    #plt.subplot(15,1,i)
    xaxis=range(lengths[i-1]-1)
    yaxis = y_predict[counter:lengths[i-1]+counter]
    yaxis1=Y[counter:lengths[i-1]+counter]
    diff_list1 = []
    diff_list2 = []
    diff = 0
    for j in range(1,len(yaxis)):
        diff_list1.append(yaxis[j]-yaxis[j-1])
        diff_list2.append(Y[j]-Y[j-1])
    counter= lengths[i-1]
    plt.plot(xaxis,diff_list1)
    plt.plot(xaxis,diff_list2)
    plt.title('S'+str(i))
    plt.show()
    #plt.savefig('figures_final/S'+str(i)+'_diff_rnn.png')
    #plt.clf()
y_acc = []
x_acc = []
threshold = 1
while threshold < 10:
   counter =0
   for i in range(len(y_predict)):
       if abs(Y[i][0]-y_predict[i])<threshold:
           counter +=1
   y_acc.append((counter/y_pred.shape[0])*100)
   x_acc.append(threshold)
   threshold += 0.1
import matplotlib.pyplot as plt
#fig = plt.figure(figsize=(10,5))
plt.plot(x_acc,y_acc)
plt.title('Error-margin Vs Accuracy')
plt.xlabel('Error-margin')
plt.ylabel('Accuracy')
#plt.savefig('figures_final/errorvsacc_graph_rnn.png')
acts = [[0 for i in range(10)] for j in range(9)]

for threshold in range(1,11):
   counter = 0
   positives = [0,0,0,0,0,0,0,0,0]
   totals = [0,0,0,0,0,0,0,0,0]
```

```
    for i in range(y_pred.shape[0]):
        if abs(Y[i][0]-y_pred[i])<threshold:
            positives[int(activities[i])]+=1
        totals[int(activities[i])]+=1
    for i in range(9):
        acts[i][threshold-1]=positives[i]/totals[i]
colr =['b','g','r','yellow','cyan','magenta','orange','darkviolet','crimson']
for i in range(9):
    plt.plot(range(1,11),[j*100 for j in acts[i]],color=colr[i])
plt.xlabel("Error-margin")
plt.ylabel("Accuracy")
plt.title("Error-margin Vs Accuracy for Different activities")
#plt.savefig('figures_final/act_rnn.png')
```



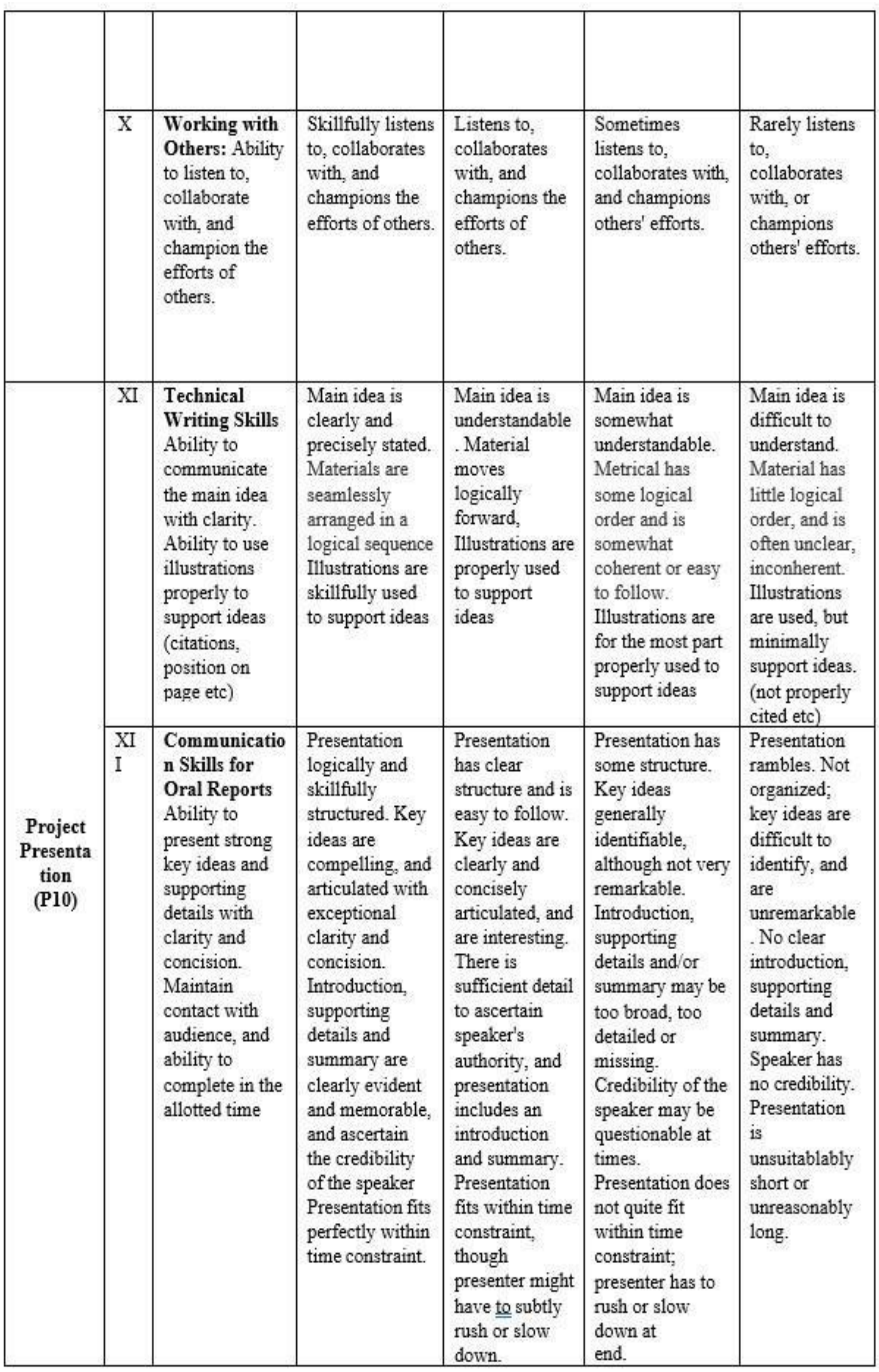
Project Rubrics

Focus Areas	C No	Criterion [c]	Exemplary4	Satisfactory3	Developing2	Unsatisfactory 1
Problem Formula tion (PO1,P O2, PO6, PO7,PS O1)	I	**Identify/Define Problem** Ability to identify a suitable problem and define the project objectives.	Demonstrates a skillful ability to identify / articulate a problem and the objectives are well defined and prioritized.	Demonstrates ability to Identify / articulate a problem and All major objectives are identified.	Demonstrates some ability to identify / articulate a problem that is partially connected to the issues and most major objectives are identified but one or two minor ones are missing or priorities are not established.	Demonstrates minimal or no ability to identify / articulate a problem and many major objectives are not identified.
	II	**Collection of Background Information:** Ability to gather background Information (existing knowledge, research, and/or indications of the problem)	Collects sufficient relevant background information from appropriate sources, and is able to identify pertinant/critical information;	Collects sufficient relevant background information from appropriate sources;	Collects some relevant background information from appropriate Sources.	Minimal or no ability to collect relevant background information
	III	**Define scope of the problem** Ability to identify problem scope suitable to the degree considering the impact on society and environment	Demonstrates a skillful ability to define the scope of problem accurately mentioning the relevant fields of engineering precisely. Considers, explains and evaluates the impact of engineering interventions on society and environment.	Demonstrates ability to define problemscope mentioning the relevant fields of engineering broadly. Considers and explains the impact of engineering interventions on society and environment	Demonstrates some ability to define problem scope mentioning some of the relevant fields . Some consideration of the impact of engineering interventions on society and environment.	Demonstrates minimal or no ability to define problem scope and fails to mention relevant fields of engineering. Minimal or no consideration of the impact of engineering interventions on society and environment

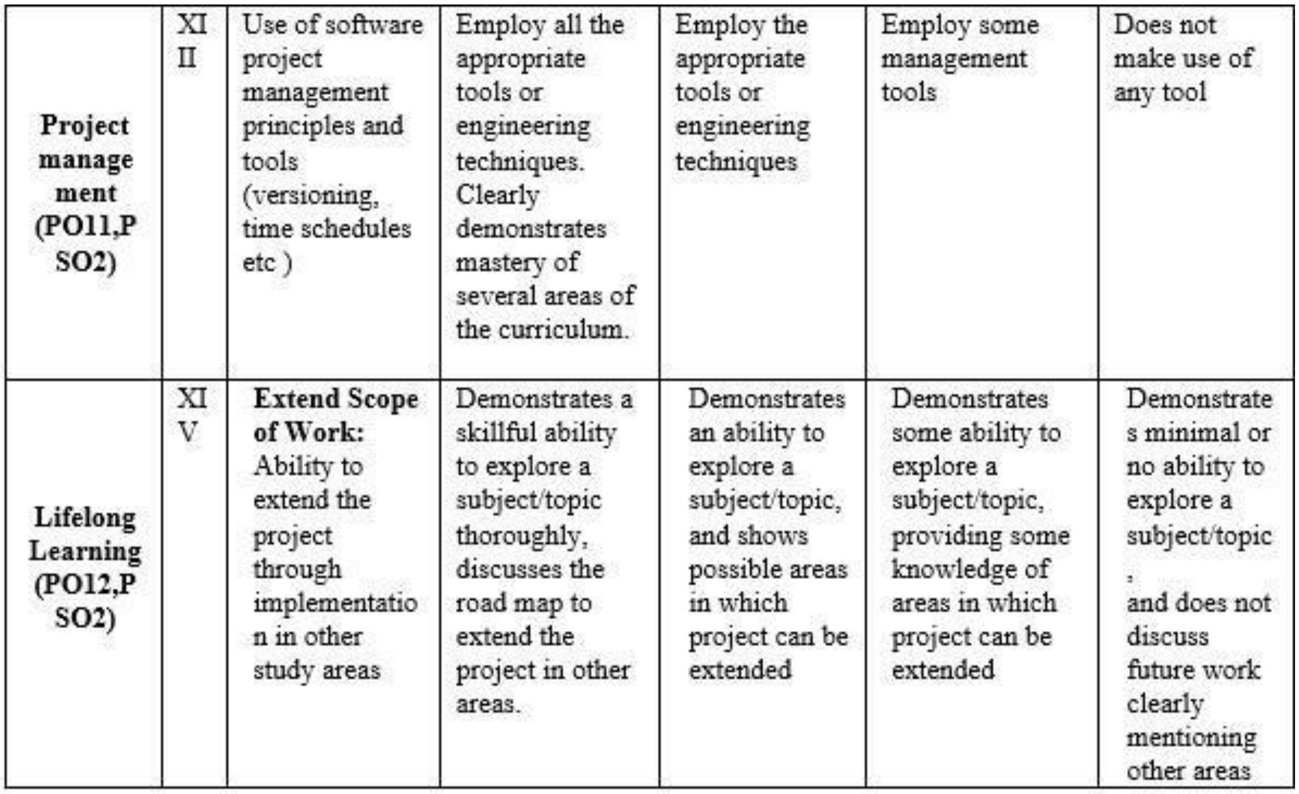


Project Design (PO3,PSO1)	IV	**Understanding the Design Process and Problem Solving:** Ability to explain the design process including the importance of needs, specifications, concept generation and to develop an approach to solve a problem.	Demonstrates a comprehensive ability to understand and explain a design process. Considers multiple approaches to solving a problem, and can articulate reason for choosing solution	Demonstrates an ability to understand and explain a design process. Considers multiple approaches to solving a problem, which is justified and considers consequences.	Demonstrates some ability to understand and explain a design process. Considers a few approaches to solving a problem; doesn't always consider consequences.	Demonstrates minimal or no ability to understand and explain a design process. Considers a single approach to solving a problem. Does not consider consequences.
Build (PO4,PO5, PSO1)	V	**Implementing Design Strategy and Evaluating Final Design:** Ability to execute a solution taking into consideration design requirements using appropriate tool (software/hardware);	Demonstrates a skillful ability to execute a solution taking into consideration all design requirements using the most relevant tool.	Demonstrates an ability to execute a solution taking into consideration design requirements using relevant tool.	Demonstrates some ability to execute a solution but not using most relevant tool.	Demonstrates minimal or no ability to execute a solution. Solution does not directly attend to the problem.
Test & Deploy (PO4, PO5, PSO1)	VI	To evaluate/confirm the functioning of the final design. To deploy the project on the target environment	Demonstrates a skillful ability to evaluate/confirm the functioning of the final design skillfully, with deliberation for further Improvement after deployment.	Demonstrates an ability to evaluate/confirm the functioning of the final design. The evaluation is complete and has sufficient depth.	Ability to evaluate/confirm the functioning of the final design, but the evaluation lacks depth and/or is incomplete.	Demonstrates minimal or no ability to evaluate/confirm the functioning of the final design.

Ethical responsibility (PO8)	VII	**Proper Use of Others' Work:** Ability to recognize, understand and apply proper ethical use of intellectual property, copyrighted materials, and research.	Always recognizes and applies proper ethical use of intellectual property, copyrighted materials, and others' research.	Recognizes and applies proper ethical use of intellectual property, copyrighted materials, and others' research.	Some recognition and application of proper ethical use of intellectual property, copyrighted materials, and others' research.	Minimal or no recognition and/or application of proper ethical use of intellectual property, Copyrighted materials, or others' research.
Team Skills (PO9)	VIII	**Individual Work Contributions and Time Management:** Ability to carry out individual Responsibilities and manage time (estimate, prioritize, establish deadlines/ milestones, follow timeline, plan for contingencies, adapt to change).	Designated jobs are accomplished by deadline; completed work is carefully and meticulously prepared and meets all requirements.	Designated jobs are accomplished by deadline; completed work meets requirements.	Designated jobs are accomplished by deadline; completed work meets most requirements.	Some Designated jobs are accomplished by deadline; completed work meets some requirements.
	IX	**Leadership Skills**: Ability to lead a team. (i) Mentors and accepts mentoring from others. (ii) Demonstrates capacity for initiative while respecting others' roles. (iii) Facilitates others' involvment. (iv) Evaluates team Effectiveness and plans for improvements	Exemplifies leadership skills.	Demonstrates leadership skills.	Demonstrates some leadership skills at times.	Demonstrates minimal or no leadership skills.

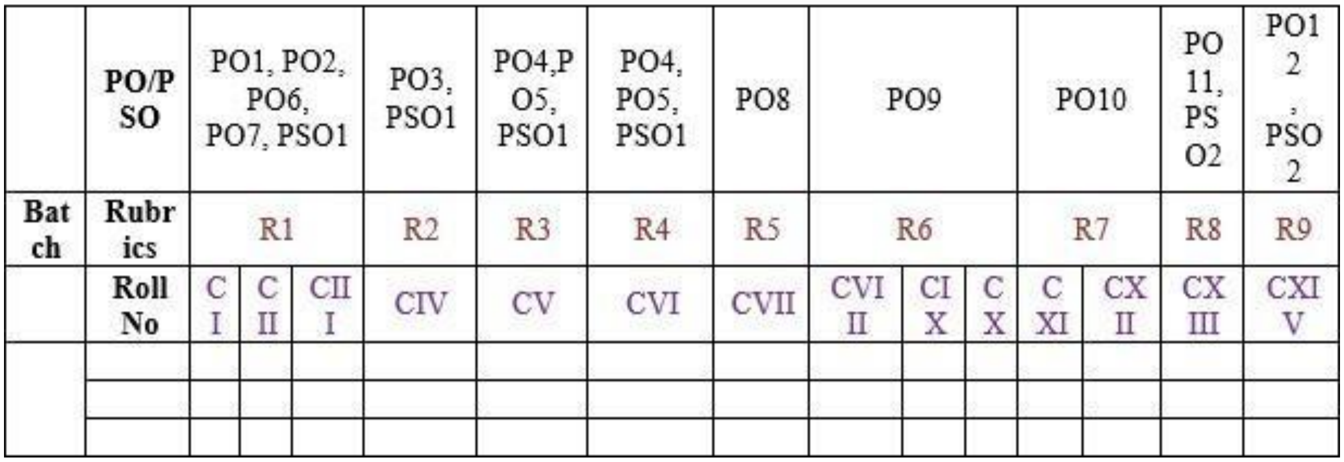


	X	**Working with Others:** Ability to listen to, collaborate with, and champion the efforts of others.	Skillfully listens to, collaborates with, and champions the efforts of others.	Listens to, collaborates with, and champions the efforts of others.	Sometimes listens to, collaborates with, and champions others' efforts.	Rarely listens to, collaborates with, or champions others' efforts.
Project Presentation (P10)	XI	**Technical Writing Skills** Ability to communicate the main idea with clarity. Ability to use illustrations properly to support ideas (citations, position on page etc)	Main idea is clearly and precisely stated. Materials are seamlessly arranged in a logical sequence Illustrations are skillfully used to support ideas	Main idea is understandable . Material moves logically forward, Illustrations are properly used to support ideas	Main idea is somewhat understandable. Metrical has some logical order and is somewhat coherent or easy to follow. Illustrations are for the most part properly used to support ideas	Main idea is difficult to understand. Material has little logical order, and is often unclear, inconherent. Illustrations are used, but minimally support ideas. (not properly cited etc)
	XII	**Communication Skills for Oral Reports** Ability to present strong key ideas and supporting details with clarity and concision. Maintain contact with audience, and ability to complete in the allotted time	Presentation logically and skillfully structured. Key ideas are compelling, and articulated with exceptional clarity and concision. Introduction, supporting details and summary are clearly evident and memorable, and ascertain the credibility of the speaker Presentation fits perfectly within time constraint.	Presentation has clear structure and is easy to follow. Key ideas are clearly and concisely articulated, and are interesting. There is sufficient detail to ascertain speaker's authority, and presentation includes an introduction and summary. Presentation fits within time constraint, though presenter might have to subtly rush or slow down.	Presentation has some structure. Key ideas generally identifiable, although not very remarkable. Introduction, supporting details and/or summary may be too broad, too detailed or missing. Credibility of the speaker may be questionable at times. Presentation does not quite fit within time constraint; presenter has to rush or slow down at end.	Presentation rambles. Not organized; key ideas are difficult to identify, and are unremarkable . No clear introduction, supporting details and summary. Speaker has no credibility. Presentation is unsuitablably short or unreasonably long.



Project management (PO11,PSO2)	XII	Use of software project management principles and tools (versioning, time schedules etc)	Employ all the appropriate tools or engineering techniques. Clearly demonstrates mastery of several areas of the curriculum.	Employ the appropriate tools or engineering techniques	Employ some management tools	Does not make use of any tool
Lifelong Learning (PO12,PSO2)	XIV	**Extend Scope of Work:** Ability to extend the project through implementation in other study areas	Demonstrates a skillful ability to explore a subject/topic thoroughly, discusses the road map to extend the project in other areas.	Demonstrates an ability to explore a subject/topic, and shows possible areas in which project can be extended	Demonstrates some ability to explore a subject/topic, providing some knowledge of areas in which project can be extended	Demonstrates minimal or no ability to explore a subject/topic , and does not discuss future work clearly mentioning other areas

Rubrics Evaluation:



	PO/PSO	PO1, PO2, PO6, PO7, PSO1			PO3, PSO1	PO4,PO5, PSO1	PO4, PO5, PSO1	PO8	PO9			PO10		PO 11, PSO2	PO12 , PSO2
Batch	**Rubrics**	R1			R2	R3	R4	R5	R6			R7		R8	R9
	Roll No	CI	CII	CIII	CIV	CV	CVI	CVII	CVIII	CIX	CX	CXI	CXII	CXIII	CXIV