University of Essex
Department of Computer Science

Final Group Project Report

Federated Learning for Activity Detection based on Wearable Devices and Smartphones

Team Name : Team 1

Supervisor : Sefki Kolozali

June 30, 2022
Colchester

Contents

1 Abbreviations

ML Machine Learning

FL Federated Learning

HAR Human Activity Recognition

CNN Convolutional Neural Network

FedAvg Federated Averaging

FedPer Federated Learning with Personalization Layers

FedMA Federated Matched Averaging

LSTM Long short-term memory

Abstract

Smartphones and wristbands, which contain powerful sensors, offer a convenient platform for creating and deploying mobile motion-based behavioral bio metrics. Although the studies that utilize these smart devices for signal-based bio metrics are few in terms of sensors and diverse physical activities to assess. In many related studies only the smartphone's sensors and less physical activities are analyzed. The notion of Internet of Things has been comprehensive due to its wide-ranging applications in healthcare. HAR (Human Activity recognition) techniques have a significant role in assessing day to day activities of humans such as investigation of activities , sports and healthcare. Human activity recognition can be used as an supportive technology when combined with trending technology like Internet of Things(IoT). Edge Computing which enables data computation to be conducted at the network's edge. The processes on the edge will involve training the data and updating the model parameters accordingly to make it more robust. Federated learning is a collective machine learning framework permitting devices from different resources with different data sets collaborating together to train a global model. In our proposed study we are developing an application i-Tracker with Machine Learning algorithms which are best suited for federated learning and edge computing. These components will take data from both smartwatch and smartphone sensors and eleven different daily activities are assessed. The everyday human activities that are selected to be perceived are walking, brushing teeth, drinking, writing, jogging, eating, sitting, standing, sleeping, downstairs and upstairs. The performance of our methods will be evaluated by comparing the accuracy of different models.

Keywords: Human Activity Recognition; Machine Learning; Deep Learning; IoT; MQTT, Federated Learning; Edge Computing; Message Queue

2 Introduction

Our lifestyle is influenced by technological advancement. This favors a sedentary lifestyle and makes us dependent too much on technology. This sedentary lifestyle and harmful eating habits increase the risk of depression, which impacts negatively on quality of life at a very young age. The internet of things allows us to be monitored remotely for health status and can support several fitness programs. It can work very efficiently with the help of activity tracking technology and sensors. Hence there is a rising of interest in monitoring and recognizing the activities attained by human beings using the Internet applications. Internet of Things (IoT) which introduced that our life becomes part of internet, has been extensively applied at different places such as mobile healthcare. For instance, healthcare with the help of mobile applications brings doctors and patients together and helps in healthcare management. In background a more practical factor of health application is human activity recognition which endeavors to record daily activities [6].

The activity recognition of human beings with the help of wrist bands and smart phones from different users using gyroscope accelerometer and from GPS of the smartphones which mainly targets human activities like Walking, Jogging, downstairs, upstairs , standing, typing, drinking, brushing teeth, sleeping and eating. However, these activities are simple but a thought-provoking classification task, because of the diversity and complexity of different activities, quality of data, and 1) inter-class variability (same activity may differ for every user), 2) inter-class similarity (diverse activities may direct similar shapes)[17]. Edge computing will help us here to transmit data among devices from the edge, near to where IoT applications are sited, instead of centralized servers. The edge client is generally the source-constraint device that the end user practices and it

is geographically close to the nearest edge server, who has profuse computing resources and high bandwidth interaction with end nodes. But in case an edge server needs additional computing power, it will associate it to the cloud server. Federated learning has been cited a lot in recent times. It is a collective machine learning framework permitting devices from different resources with different data sets to collaborate together to train a global model. This framework not only teams up with computational means from dissimilar devices, but also maintains privacy at that time [16].

This project proposes an IoT application i-Tracker for identifying health issues with the help of human activity recognition with the significance of components, edge computing, and federated learning to develop the IoT platform. This app enables the users to keep track of their physical activities and prevent a sedentary lifestyle. Temporal patterns of sedentary behaviors of a user are discovered from data collected from wearable wristband and smartphone sensors. The physical activity patterns recorded on wristband sensors are further used to maintain daily and monthly activity logs for a user. For project implementation, we are acquiring data from wristband and smartphone sensors. The activities will be recorded by each member of our project team. The application is designed to maintain the privacy of the data as well as get a higher efficiency IoT system. The smartphone application, i-Tracker, is used to display all the details about the user physical activities as well as monitor the health condition With i-Tracker, we aim to track the number of steps, distance, calories, and sleep quality of a user and predict the activity using Artificial Intelligence based Human activity Recognition models. We are mainly focusing on the activities Walking, Jogging, downstairs, upstairs , standing, typing, drinking, brushing teeth, sleeping, and eating .The user activity prediction will help the user maintain an activity log on a daily, weekly, and monthly basis. The application can log user activities even if they are offline, providing continuous tracking support.

3 Related Work

We have summarized a list of experiments for Human Activity Recognition work done recently

Bulbul et al. [2]	**Experiment Data 1 [**described in Figure. 2]
Decision Tree(20)	91.70%
Decision Tree(100)	94.40%
SVM	99.40%
k-NN(k=1)	97.10%
k-NN(k=3)	97.50%
AdaBoost	97.40%
Bagging	98.10%
Stacking	98.60%
Ronao et al [9]	**Experiment Data 1 [**described in Figure. 2**]**
Convnet (inverted pyramid archi) + MLP	94.79%
temporal Fast Fourier Transform + Convnet (J(L1) = 200)	95.75%
Benhaili et al [1]	**Experiment Data 2 [**described in Figure. 3**]**
stacked LSTM network	99.00%
Saurabh et al. [4]	**Experiment Data 3 [**described in Figure. 4**]**
CNN-GRU	96.54% on smartwatch and 90.44% for smartphone

Figure 1: Summary of Human Activity Recognition using Centralized Approaches

Bulbul et al. applied different machine learning classification approaches on dataset described in Table 2 with highest success rate of **98.6%** using Stacking classifier consisting of 30 k-NN classifiers [2]. Ronao et al. applied deep learning neural networks on dataset described in Table 2 by proposing a deep convolutional neural network (convnet), exploiting the inherent characteristics of activities and 1D time-series signals, at the same time providing a way to automatically and data-adaptively extract robust features from raw data [12]. Their experiments show that convnets were able to derive relevant and complex features with every additional layer, although difference of feature complexity level decreases with every additional layer. The model achieved an overall performance of **94.79%** on the test set with raw sensor data, and **95.75%** with additional information of temporal fast Fourier transform of the HAR data set.

Benhaili et al. in his study proposed a deep learning model that can work with raw data without any pre-processing on dataset described in Table 3 [1]. The human activities were recognized using

Human Activity Recognition Using Smartphones Data Set	
Devices	Waist-mounted smartphone with embedded inertial sensors
Number of Participants	30
Data Samples	10299
Number of Activities	6
Activities	WALKING, WALKING_UPSTAIRS, WALKING_DOWNSTAIRS, SITTING, STANDING, LAYING
Sensor Names	Accelerometer and Gyroscope
Sensor Data	Triaxial acceleration from the accelerometer from Smartphone Triaxial Angular velocity from the gyroscope from Smartphone
Sampling Rate	50 Hz

Figure 2: Experiment Data 1

stacked LSTM network. They were able to achieve an accuracy of **99%** with maximum confusion in Upstairs and Downstairs. Their model was lightweight and could be applied on edge devices.

WISDM Dataset	
Devices	Smartphone
Number of Participants	36
Data Samples	10,98,207
Number of Activities	6
Activities	Walking, Jogging, Downstairs, UpStairs, Sitting, Standing
Sensor Names	Accelerometer
Sensor Data	Triaxial acceleration from the accelerometer from Smartphone
Sampling Rate	20 Hz

Figure 3: Experiment Data 2

Saurabh et al. demonstrated usage of a novel hybrid deep learning model on dataset described in Table 4, CNN-GRU that combines convolutional and gated recurrent units for human activity recognition [5]. They classified the 18 activities into major three activities a) Ambulation Oriented Activities such as walking, jogging, etc. b) Hand Oriented Activities (General) such as writing, typing, etc. c) Hand Oriented Activities (Eating) such as eating chips, eating pasta, etc. The best model for the smartwatch dataset produced an accuracy of **96.54%** and **90.44%** for smartphone dataset.

WISDM Smartphone and Smartwatch Activity and Biometrics Dataset Data Set	
Devices	Smartphone and Smartwatch
Number of Participants	51
Data Samples	15630426
Number of Activities	18
Activities	Walking, Jogging, Stairs, Sitting, Standing, Typing, Brushing, Teeth, Eating, Soup,Eating Chips, Eating Pasta, Drinking from Cup, Eating Sandwich, Kicking (Soccer Ball), Playing Catch with Tennis Ball, Dribblinlg (Basketball), Writing ,Clapping , Folding Clothes
Sensor Names	accelerometer and gyroscope
Sensor Data	Triaxial acceleration from the accelerometer from Smartphone and Watch
Sampling Rate	20 Hz
Devices	Google Nexus 5/5x or Samsung Galaxy S5 Smartphone LG G Watch

Figure 4: Experiment Data 3

In paper[8] the author presents a survey on federated learning. It begins with a general introduction of FL, then it highlights challenges, recent implementation, review of some existing solutions and finally some future directions. In FL, end devices custom their local data to train a model which is mandatory by the server. Then end devices send updates rather than raw data to the server for collaboration. Federated learning is serving in mobile edge networks thus it enables collaborative training and optimization for mobile edge networks. The author also compares conventional cloud-centric training approach with federated learning at mobile edge network features. Advantages of FL over cloud based centric are:

- Highly efficient use of network bandwidth: A reduced amount of information is needed to be transmitted to the clouds. For example: Participating devices merely provide the updated

model parameters for aggregation, rather than delivering the raw data for processing. As a result, data communication costs are reduced dramatically, and backbone networks are relieved from stress.

- Privacy: As a result of the following statement, users' raw data does not need to be sent to the cloud. This improves user privacy and minimizes the likelihood of eavesdropping to some level, assuming that FL participants and servers are not malevolent. Indeed, with improved privacy, more users will be ready to participate in collaborative model training, allowing for the development of better inference models.

- Low Latency: ML (Machine Learning) models can be regularly trained and updated using FL. Meanwhile, real-time decisions, such as event detection, can be made locally at the edge nodes or end devices in the MEC paradigm. As a result, the latency is lower than when choices are made in the cloud and then sent to the end user.

FL has recently seen success in several applications because of the above benefits. The Federated Averaging algorithm, for example Federated Averaging Algorithm(FedAvg) which has been applied to Google's Gboard [15]. In future work author suggests some work need to do in Asynchronous FL and unlabeled data.

4 Methodology

4.1 System Interface

The Figure.5 illustrate the components to develop a Federated Learning System for Human Activity Recognition. The i-Tracker Mobile Application is the application for end users to track their activities on mobiles. The application will utilize machine learning algorithms trained using Federate Model Learning techniques for activity prediction. The *Sensor Data* component show the sensor data recording using Smart watch and Smartphone sensors. The sensor data is available on the mobile devices as well on the cloud Platform. The Offline Training for the Initial Model captures the Data collection, Feature extraction and best model selection for Human Activty recognition.

The two major components of the application are Edge Computing and Federated Learning as discussed in below sections:

4.1.1 Edge Computing

Edge Computing enables data computation to be conducted at the network's edge, i.e. the path between data sources and cloud data centers, on downstream data for cloud services and upstream data for IoT services. It aims to overcome the issues faced by IoT, such as strict latency, network capacity constraints, resource-constrained devices, uninterrupted services with intermittent connectivity, and security challenges, which the centralised cloud computing architecture cannot fully address [3]. A 2020 examination of Edge intelligence by Xu et al explores a network of connected systems and devices to gather, cache, process, and analyse data in close proximity to where it is captured [18]. In this solution, recognition and prediction are handled by edge servers and peer devices, or by using the edge-cloud collaboration paradigm with little or no data sent to the cloud. The components of edge computing includes edge caching, edge training and edge inference which are described in detail below:

- Edge Caching: In edge caching data generated by the smart watch are stored on edge i.e in close proximity to the users. The data will be used to train and retrain the model. The Data will be collected from 8 humans performing 11 different activities at different times of the day making a total of 88 different activity data.

- Edge Training: This will be a distributed learning procedure that learns the optimal values for all the weights and bias, or the hidden patterns based on the training set cached at the edge. The training will be collaborative meaning that multiple devices will cooperate to train a common model/algorithm. The data will be split into 70%, 20% and 10% for training, test and evaluation respectively. To predict human activities using the sensor data, we would make use of discriminative classification machine learning models such as Random forest, SVM.

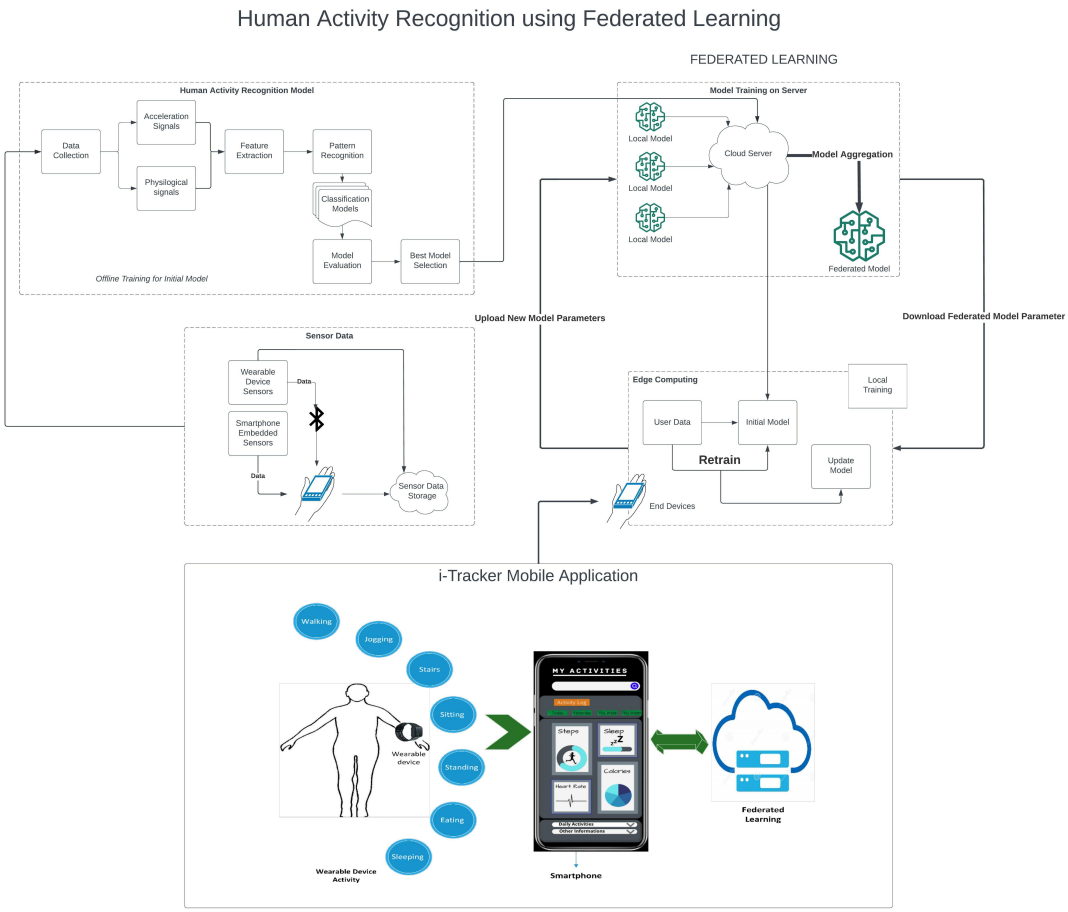

Figure 5: System Architecture

- Edge Inference: This entails applying our trained model/algorithm in a forward pass to compute the output on edge devices and servers. This is where our model will predict the activities of users based on the data inferred.

4.1.2 Federated Learning

Machine learning techniques have proliferated in Human Activity Recognition Human Activity Recognition (HAR) field which attempt to classify human activities in order to monitor user's health, discover activities patterns, and improve user's wellbeing. However, training these models using data collected from client's smartphone and smartwatch can lead to high communication cost and privacy contravention [14]. Federated Learning (FL) is a new decentralized learning paradigm based on sending the Machine Learning (ML) model to the users' devices for local training instead of sending their data to the centralized server, then the model parameters will be sent again to the server that uses some approach to aggregate all the clients' models to update the global ML model.

- Initialisation: At the beginning, the FL model will be set using initial parameters. Two of the edge devices are selected randomly for every round to receive the current global training model parameters. The selected edge devices will train the received model on each of their local dataset.

- Client procedure: At the client side, the local data is split to b minibatches. For each batch b, a one-step gradient will be calculated. The new weight parameter is obtained by differentiate the loss function with respect to the weight, the bias, and the learning rate. Then, the updated parameters will be sent back by the local models to the server where the aggregation process will be executed to get a new global model parameters.

- Server procedure: There are various FL algorithms can be applied to aggregate the local models such as:

1-Federated Learning with Personalization Layers (FedPer): this model splits the model into two parts: personalized layer and base layer. The base layers are aggregated to the server using transfer learning methodologies while the personalized layers have no communication with the server [4].
2-Federated Matched Averaging (FedMA): which use the layer-wise matching scheme. Layers in this approach are trained and averaging hidden parameters layer by layer. It is used for modern neural networks as Convolutional Neural Network (CNN) and Long short-term memory (LSTM) [13].
3-Federated Averaging (FedAvg): in this algorithm, the average of all of the new local parameters will be calculated by the server to set a new parameters for the global aggregated model.

For our project we are going to use the classical FL algorithm FedAvg to calculate the updated parameters generated by the clients which will be sent again to the new selected clients. The local models will be replaced with the new one which will used as the starting point for another round of training. The FL model will converge after several rounds of training and for each round, the local models can acquire new training data inputs. Also, devices may drop out or join the system at any time.

5 System Design

5.1 Harware Interfaces

Requirement specification	**Description**	**Risk Level**
Wearable device (E4 Empatica)	System shall have wearable device technology which takes input from user it possible to monitor human activity and behavior	High
Smartphones (Moto G50)	System shall have description for User Interface (UI)	High
PPG sensors	PPG is a non-invasive technology that uses a light source and a photodetector at the surface of skin to measure the volumetric variations of blood circulation.)	High
EDA sensors	EDA Sensor (GSR Sensor) Measures the constantly fluc- tuating changes in certain electrical properties of the skin	High
3-axis Accelerometer	A 3-axis accelerometer measures the accelerations that take place in relation to the 3 Cartesian coordinate axes.	High
Infrared Thermopile	Reads peripheral skin temperature	High
Internal real time clock	System shall have specification for UI input/output timing	High

5.2 Software Interfaces

Software used	Description	Version
JIRA	System shall have wearable device technology which takes input from user it possible to monitor human activity and behavior	8.2.x
Android	It is an open-source UI software development kit, used to develop a smartphone app compatible with our hardware device in easy and fast way.	13.0.0
GitLab	Used for code writing collaboration	Enterprise Edition 14.1
RabbitMQ	RabbitMQ is an open-source message-broker software that originally implemented the Advanced Message Queuing Protocol and has since been extended with a plug-in architecture to support Streaming Text Oriented Messaging Protocol, MQ Telemetry Transport, and other protocols	above 3.7
TensorFlow	Open-source framework used for applying machine learning on decentralized data.	above 0.23

5.3 Communication Interfaces

The rapid growth of IoT (Internet of Things) devices being connected to the internet and exchanging the data between each other and the server results in giving more attention to communication protocol choice. Power consumption, bandwidth usage, and planned IoT deployments of the project are fundamental factors in making the choice. MQTT, AMQP, CoAP, LoRa, and LoRaWAN are some of the most common communication protocols used in IoT applications.

Message Queuing Telemetry Transport (MQTT) was invented for oil and gas pipelines communication by two engineers Andy Stanford-Clark (IBM) and Arlen Nipper (Eurotech) in 1999.The following figure shows an overview of the history timeline of the MQTT protocol.

Figure 6: MQTT protocol history timeline

5.3.1 Protocol choice

MQTT is one of the most common protocols used in the IoT area as an underlying communication protocol. Moreover, it is used in smart automation systems like smart homes and platforms such as IBM Watson, Microsoft Azure, and Instagram platform. MQTT protocol is used in this project, the decision to use this protocol was based on the following reasons:

- MQTT is a lightweight protocol with fast response.
- Efficient usage of energy and data.
- Minimal resources requirements and Bi-directional communications.
- Handling unreliable networks by supporting persistent sessions.
- Supporting message reliability using Quality of service (QoS) feature which is responsible for ensuring message delivery.
- SSL/TLS is used for security and protecting data contents.
- Scalability due to publish/subscribe model characteristic.

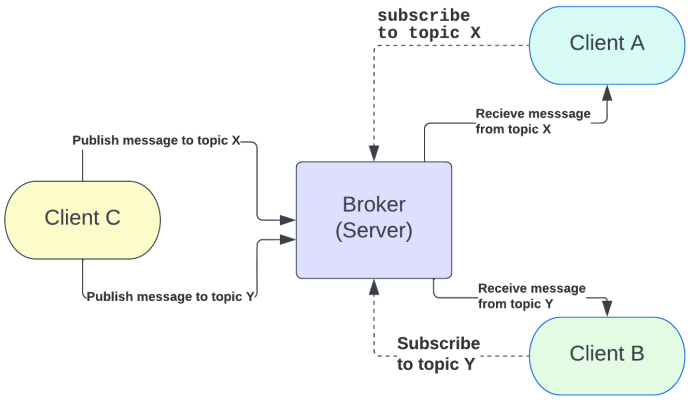

Figure 7: MQTT protocol architecture

5.3.2 Protocol architecture

- Message: the data which is sent from device to another across the network, it includes some parameters such as: Payload data which is the content, Quality of Service level, and Topic name. The following table shows different messages of MQTT protocol.
- Publisher: the client who sends a message with a specific topic to the broker.
- Subscriber: the client who subscribes to a certain topic on the broker to receive the messages.
- Broker (Server): It is the center of the system because it is responsible for all the process and messages across the network as handling the clients and their subscriptions, receiving the messages, filtering them according to Topics names, sending the correct message to the correct subscriber.

MQTT protocol is based on a publish/subscribe model for communication, and it runs on top of TCP/IP. Direct communication between publisher and subscribers is removed in publish/subscribe models and a third party, the broker, is responsible for filtering the messages and delivering each of them to the intended recipient. The messages are filtered and routed according to the topic, which is a hierarchically structured string that is contained in each message. MQTT decouples the publisher from the subscriber in several dimensions:

- Space decoupling: Publisher and receiver do not change their IP address or port number.
- Time decoupling: Publisher and receiver do not need to connect at the same time.
- Synchronization decoupling: During the process of publishing or receiving, other operations on both sides are not interrupted.

5.3.3 Features of MQTT protocol

- Quality of Service Levels (QoS): This feature is used to handle message delivery between the sender and the receiver; there are three QoS levels in MQTT:
  - At most once (0): the message is sent at most once and there is no guarantee of delivery. It is also called "fire and forget."
  - At least once (1): the message is delivered at least once to the receiver by waiting for the sender acknowledgment packet.
  - Exactly once (2): The highest level of the service; it guarantees that each message is delivered to the intended receiver only once.
- Message Retention: It is a normal message with a true value of retained flag. A major advantage of this feature is that new subscribers can get a message immediately after establishing the connection and subscribing to a topic containing retained message.
- Last Will message: MQTT is considered to deal with unreliable networks. This feature is used to notify clients about any ungraceful disconnection from other clients. This message is specified by each client when connecting to a broker. All Last will messages will be stored by the broker and will be used in the case of ungraceful disconnection only.
- Keep Alive: this function specifies the maximum time interval with no communication between a client and the broker. If the client does not send any message during this period, the broker will send a ping request message to make sure that the client is still reachable.
- Persistent sessions: These sessions save all the information of the client on the broker including the topics that have been subscribed to by the client. As a result, there is no need to resubscribe again in case of facing any connection interruptions.
- Authentication: MQTT provides the following to identify an MQTT client:
  - Client ID is associated with every client.
  - Username and password are used for restricting access to the broker or some topics.

5.4 Product Functions

The 'i-Tracker' mobile application is an activity tracker which aims to log physical activities of a user and provide continuous health monitoring support.

1. Activity Log Tracker: The user can track the real time activities of themselves using real time monitoring.
    - Walking
    - Jogging
    - Upstairs
    - Downstairs
    - Sitting
    - Standing
    - Brushing Teeth
    - Eating
    - Drinking
    - Sleeping
    - Writing
2. Activity History Logs: The user activities are logged and can be viewed on a daily, weekly and monthly basis.
3. Search Activity: A search option will be available to the user to retrieve the activities according to the activity type like Sleeping and Eating.
4. Activity Dashboards: The user can view donut charts to track the percentage of time spent on an average for a particular activity in each day of 24 hours.
5. Sedentary Lifestyle Detection: The user can view daily calorie consumption based on their activities. Additionally, the deskbound and sedentary activities can be tracked using Dashboard visualization charts and indicate unhealthy lifestyle. Ref Figure 8

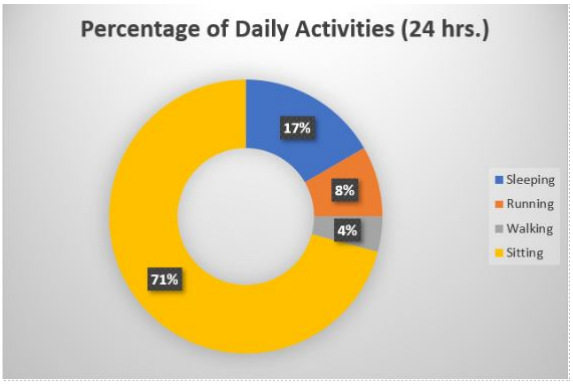

Figure 8: Sedentary Lifestyle Indicator

5.5 Functional Requirements

Title: User Activity Log on Edge device
Priority: High
Description: The functional requirements were achieved through the users' activities and the information retrieved through the sensors from the wearable device to smartphone. The system will have sensors that are put near to the body and measure aspects including temperature, movement, and pulse. Bluetooth Low Energy (BLE) protocol connectivity, which is the most widely used method of connecting wearable devices to a smartphone or home network. More sensors are required to detect physical or environmental variables, as well as the capacity to send data from many sensors to the outside world via wireless or other network technologies.

Title: Human Activity Recognition using AI
Priority: High
Description: The model will be trained to classify user activities using dataset getting into the smartphone.

Title: Efficient Application Development using Federated Learning
Priority: High
Description: By having minimal latency and connecting to the internet to analyse data, you can predict activity in real time. Based on user actions, the system will recognise new activities and create a dashboard.The system must assign the appropriate classification to each activity.

Title: User Data
Priority: High
Description: The smart device helps to collect relevant activity data for predictions as per user's permissions. The system must adhere to privacy regulations, not keep user data, and process large amounts of data as needed.

6 Implementation

6.1 Edge Device Setup

- Empatica E4: It is a wearable device that is also a single device, but one of the most multifunctional sensor data collection devices available. In this project we use one wearable device for data collection which is provided by University of Essex. The actual sensors in wearable devices are accelerometer, photoplethysmography (PPG) and electrodermal activity sensors, all composed for the study [11].
- Moto G50(Smartphone): To collect smart phone data, we use MotoG50 Android phone which is provided by University of Essex. On device sensors captured data include accelerometer x-axis, y-axis, and z-axis readings. These axes help to record horizontal/sideways movement of user(x-axis), downward/upward movement (y-axis), and forward/backward movement(z-axis)[10].

(a) Empatica E4

(b) MotoG50

Figure 9: Devices Used for sensor Data Collection

Dataset	8itracSens
Individual	8
Characteristics	Gender, Age, Id
Hz	32
Devices	Smart Watch, Smartphone
Sensors	Accelerometer, Gyroscopes
Location	Wrist, Left hand, Right hand
Activities	Walking, Upstairs, Downstairs, Jogging, Sitting, Standing, Sleeping, Brushing, Writing, Eating, Drinking

Figure 10: Summary of Dataset charaterstics used in experiments. The activities performed by different subjects.

6.2 itracker Initial HAR model

6.2.1 Data Collection

For Data Collection an 'itracker Recording' Android application was developed to collect real time data from Empatica E4 watch and Mobile sensors. The frequency of data collection from accelerometer was 32 Hz whereas frequency for BVP was 64 Hz. In the next step downsampling is performed for stable frequency. The collected data is stored in a SQLite database on the device. We collected dataset from 8 subjects as shown in 8itracSens. Each subject completes two or three activities from the list (Sitting, walking, brushing teeth, eating, drinking, downstairs, upstairs, writing, jogging, sleeping, and standing). Every subject's data collection begins with extremely basic information (age and gender) as described in Figure 10. Every subject was asked to wear Empatica E4 and hold MotoG50 in their hands to record data for 5 minutes for each selected activity. Empatica E4 and MotoG50 measure accelerometer data and blood volume pulse while activities performed by the subject. Each session lasted for 5 minutes. The sensor provides physiological information like heart rate. The app collected data from both devices and data was manually uploaded to back-end server for further analysis.

6.2.2 Data Preprocessing

- Summarization of dataset: Summarization of characteristics of datasets used in experiments is described in Figure 11. For the 8iTracsens dataset, all subjects performed all 11 activities. All dataset's sensors, such as the accelerometer and gyroscope, were chosen based on the evaluation situation. Participants wear wristwatches and carry cellphones while participating in activities, which are then recorded. The activity recording is split into two sections by the mobile replacement. For some activities, it is on the right hand, whereas for others, it is on the left hand.

Sl.No.	Features	Description
1	min	Minimum value
2	max	Maximum value
3	mean	Average value
4	median	Median value
5	std	Standard deviation
6	skew	Skewness
7	kurtosis	Kurtosis
8	diff	Difference between Minimum and Maximum value
9	IQR	Inter Quartile Range
10	pos_count	Number of positive counts
11	neg_count	Number of negative counts
12	above_mean	Number of samples above mean value
13	avg_abs_diff	Average of absolute deviation from central point
14	energy	Average of sum of squares of the values in a window

Figure 11: Selected features from Accelerometer, Gyroscope and BVP

- Downsampling and Segmentation: Before plotting and merging BVP signal it is downsampled to the same number of sampling frequencies. Because with wearable device E4 provides accelerometer frequency 32 Hz and for BVP it was 64 Hz. The BVP data is downsampled to get constant frequency of 32Hz. Following the separation of features and labels. Data is split into two categories: training and testing. The next step is to scale the data so that it fits into a given scale. /
- Selected Features: Selected features from Accelerometers, Gyroscope and BVP of Wrist band and Smartphone

6.3 Edge Computing

6.3.1 Implementation Strategy

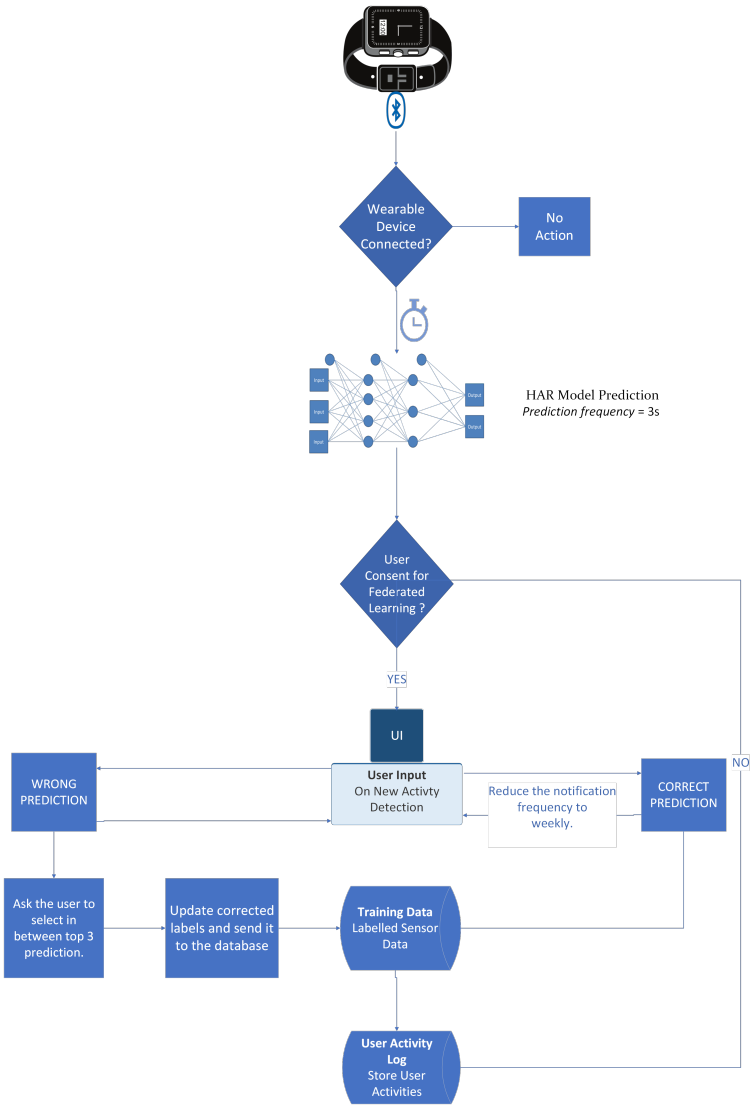

Figure 12: Edge Learning Strategy

Edge training plays a vital role to execute Federated Learning. Edge training or Edge Machine Learning gaining importance because of the rise of IoTs. To make a device smart we have to make sure that the device should learn itself overtime and this is where Edge Training comes in. Edge Training takes the real time data from the user and re-train itself when required to increase the accuracy. Edge device can still send the data to the cloud if required; however, we are keeping the concept of Federated Learning in mind to maintain the data privacy. We have trained the model on the device, thereafter, sending the parameters and weights to the server to average out weights.

In figure 12 and 13, we have designed the architecture of our Edge device training. To make prediction we have to make sure that the device is connected with the wearable from which we can fetch the real time data. Once the device is connected with the smartphone, it will start collecting the data with the sampling rate of 3s(32Hz 96 samples). Once the sample is collected it will be passed to our pre-trained LSTM deep learning model for prediction. Post prediction, we will be passing a prompt to the user for the consent whether the user wants to participate in the Federated Learning and help the model become smarter. In case user doesn't want to participate in Federated Learning, the prediction along with the time stamp will be stored in the 'User Activity Log' where user's previous activity is stored. If user wants to participate in the Federated Learning then user will be notified via notification about the prediction with the time stamp. In case of correct prediction the data will accumulate for edge training with the correct label. And will be passed to User Activity log. In case of wrong prediction the user will be shown 3 more activity option with

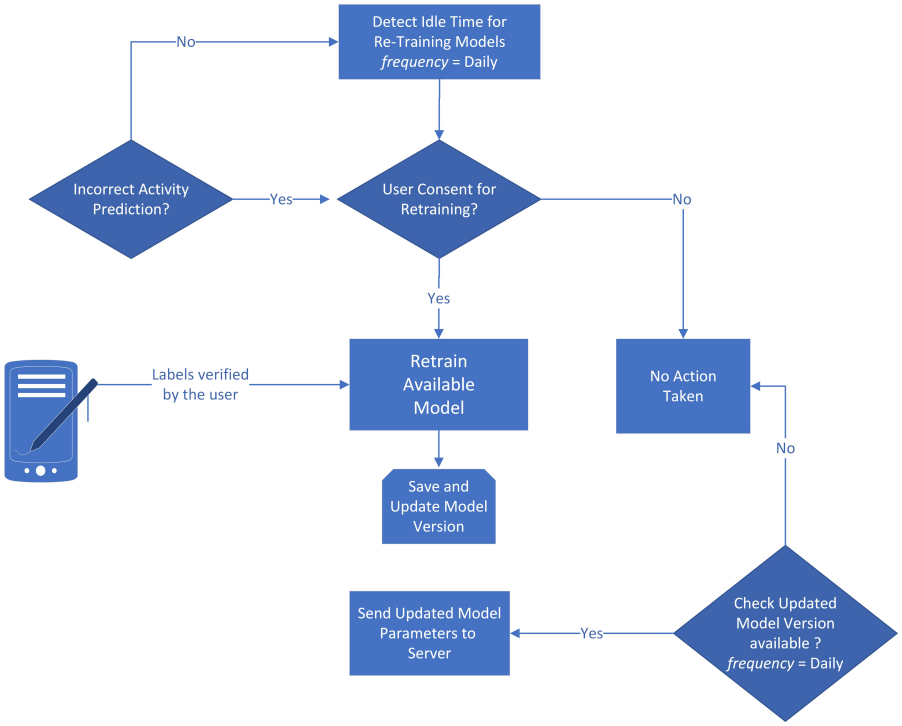

Figure 13: Edge Learning Strategy

the highest probability(Calculated by softmax layer). Once the user selected the correct label it will be aligned with the wrong label and send the data for re-training. Again activity with correct label be stored in the User Activity Log.

We have discussed the edge computing process; however, in figure 5 we have discussed the time of the training so that it does not interfere in the user experience. If our model is performing well then we will look for an idle time to tr-train the model; however, if out predictions are wrong then we will immediately ask the user for Edge training. If the user denies for immediate edge training then no action will be taken, whereas, if the user agrees for retraining the model then the model will take the corrected labels by the user and re-train itself. Once re-training is done model will save the updated version. Edge device will daily check if there is any updated version of the model present, in case there is any updated version present then the updated model's parameters will be sent to the service, otherwise no action will be taken.

6.4 Federated Learning

For this technique we ran a simulation of the data processes from training to prediction to make the final choice on which strategy to employ in our app. The detailed results and training graphs can be found at Section.7

6.4.1 Aggregation Strategies

To get the best results for prediction we compared the two :

- **Federated averaging (FedAvg)**
  For this simple aggregation technique the server selects a subset of clients from the population for the current iteration and provides them with the current global model. The edge devices optimise the loss on their local training data using SGD over several epochs to update the parameters of their local copy of the model. The local parameters are sent to the server at the end of the round, which aggregates them using a weighted average, and the aggregated parameters constitute the global model for the following round. We assume there are K clients over which the data is partitioned, with Pk the set of indexes of data points on client k, with

  $$n_k = |P_k| \tag{1}$$

  These clients update the parameters of their local copy of the model by optimizing the loss on their local training data using SGD for given epochs [9]:

  $$f(w) = \sum_{k=1}^{K} \frac{n_k}{n} F_k(w) \tag{2}$$

  where

  $$F_k(w) = \frac{1}{n_k} \sum_{i \in P_k} f_i(w) \tag{3}$$

  The local parameters are sent to the server at the end of the round, which aggregates them by performing a weighted average. The aggregated parameters define the global model for the next round.

- **Federated Approximation (FedProx)**
  This aggregation technique is a generalization of the FedAvg with some modifications to address variability of data and systems. Following the same steps the learning is performed in rounds and at each round, the server samples a set of clients and sends them the current global model. The difference in this method is that the clients optimize a regularized loss with a proximal term [7]. FedAvg is a particular case of FedProx with

  $$\mu = 0 \tag{4}$$

  Different from FedAvg, the clients optimize a regularized loss with a proximal term. In particular, the new function to minimize is

  $$F_k(w) + \frac{\mu}{2} ||w - w^t||^2 \tag{5}$$

6.4.2 Data Training Strategies

- **Population vs FedAvg Training**
  For this simulation we trained on data from 3 edge devices and used the last device to test and evaluate the model. We trained Logistic Regression, LSTM and Federated Average with LSTM modelling techniques to compare our results.

- **Personalized vs Fine Tuned Training**
  For this simulation we split from all edge devices in to 80% for training and 20% for testing as shown in the diagram below. We trained Logistic Regression, LSTM.

6.5 Advantages

- Latency : Even after developing high accuracy Machine Learning models, it is a challenge to deploy them in production environments. In general, these models are deployed in Cloud and inference is done using language based Client API. However, this approach suffers network latency, transporting data takes a lot of time in the process of prediction. In our suggested approach, the model are localized and are deployed in the mobile device itself thus mitigating the Network Latency for each prediction. We were able to deploy our initial Machine Learning model in our Edge device and obtain real time predictions for the selected 11 Activities.

- Model Optimization : The trained machine learning models are huge in size and usually result in higher device storage, slower execution and higher battery consumption. To mitigate these challenges we used TensorFlow Lite which is a Model Optimization Toolkit offered by google. The two major techniques applied by TensorflowLite are Quantization and Pruning. Quantization is a generic term describing technologies that reduce numerical precision of static parameters an execute operations in lower precision. Precision reduction makes the model smaller and lower precision execution make the model prediction faster. Pruning introduces sparsity in the connections and thus helps in reducing model size. We successfully trained our Machine Learning models and converted them in TF Lite version before deployment reducing the App size to 300 MB.

- Security : An efficient Model training requires training data, however, this imposes a huge risk to User data security and privacy. The centralized Machine learning approaches demand exchange of User data frequently and expose it during transmission or storage. To mitigate the challenge we successfully demonstrated a Federated Learning simulation which can be used as a decentralized learning approach in real word scenario.

  The sensor data collection in Empatica E4 watch stores the data on Cloud as a backup. We were successfully able to stop the data backup mechanism on Cloud and store the sensor data on the device itself.

- Low Power Consumption : To implement Federated Learning communication setup is requried between server and client, for power efficient communication we successfully setup Message Queues with MQTT protocol. The MQTT protocol was developed to be reliable in constrained conditions like satellite transmissions. The protocol can transfer data at low energy, low cost and low bandwidth over unstable connections. We were successful in sending model weights over message queues from an Android device.

  To enhance the performance features we have successfully developed a strategy to backup only the User Activity data and cleanse the sensor data after utilization for training.

- Reliability : We have implemented Message Queues which makes data persistent, and reduce the errors that happen when different parts of your system go offline. By decoupling components with message queues, the system is more fault tolerance.

7 Evaluation

For testing, we had labeled data for six users. Population model was trained using 4 users and tested on other 2 users. Personalize model was trained and tested on users individual datasets using the train and test split of 80% and 20% respectively. Randomised model was trained using random train and test splits of 80% and 20% respectively. We were able to achieve highest accuracy using personalized model of 95%. The maximum confusion was seen between activities which had overlapping patterns between primary and secondary activities for example sitting, eating and drinking. Additionally, the logistic regression was able to achieve highest accuracy in all three approaches due to feature engineering techniques applied to the data before training the model

Logistic Regression Confusion matrix on test data of population model, personalized model and randomized model is shown below 14, 15 and 16

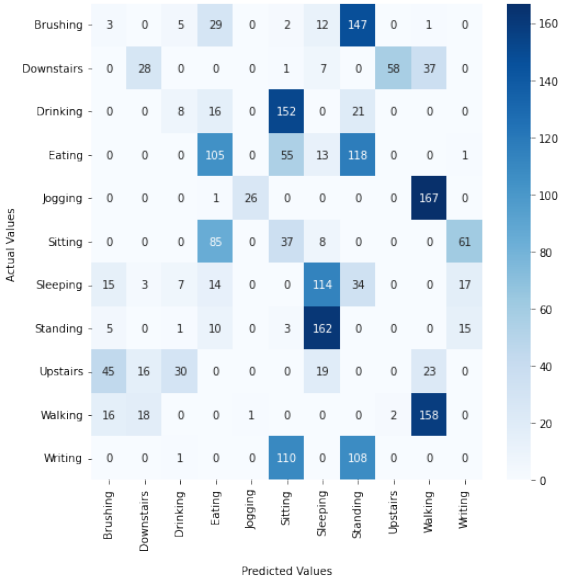

Figure 14: LR Confusion Matrix on test data of population model

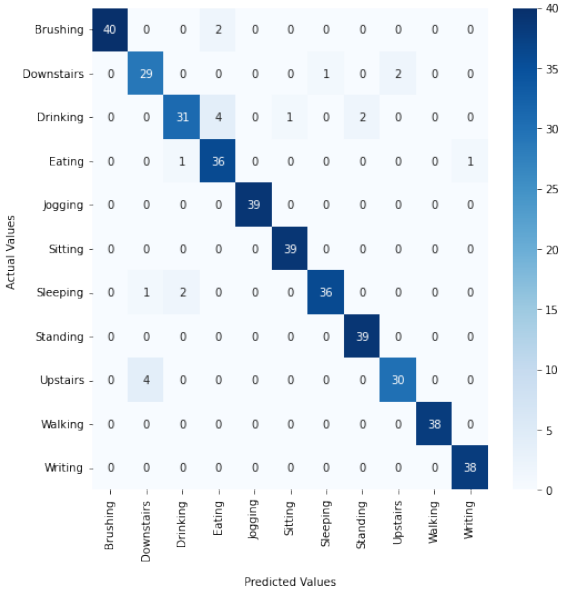

Figure 15: LR Confusion Matrix on test data of personalized model

LSTM Model Confusion matrix on test data of population model, personalized model and randomized model is shown below 17, 18 and 19. The highest accuracy for LSTM was achieved using randomized split model with 86%. Similar confusion matrix was observed for LSTM where maximum confusion was seen between activities which had overlapping patterns between primary and secondary activities for example sitting, eating and drinking.

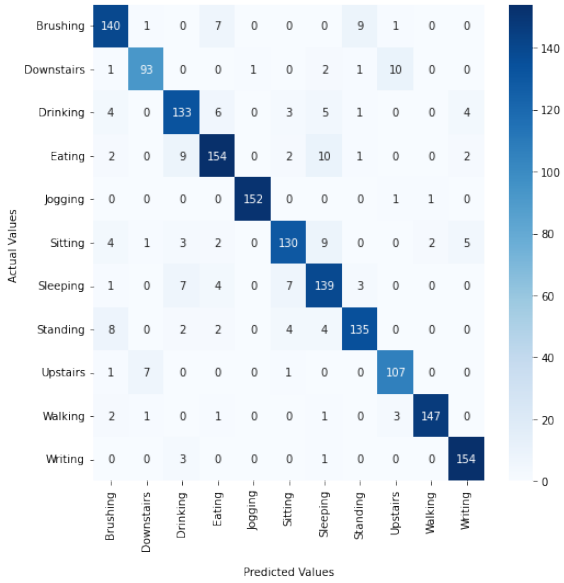

Figure 16: LR Confusion Matrix on test data of random model

Figure 17: LSTM Confusion Matrix on test data of population model

For FedAvg we were not able to achieve expected accuracy due to suspected anomalies and deficiencies in data. In general we observed FedAvg which is equivalent to Population model which did not perform well on our datasets.

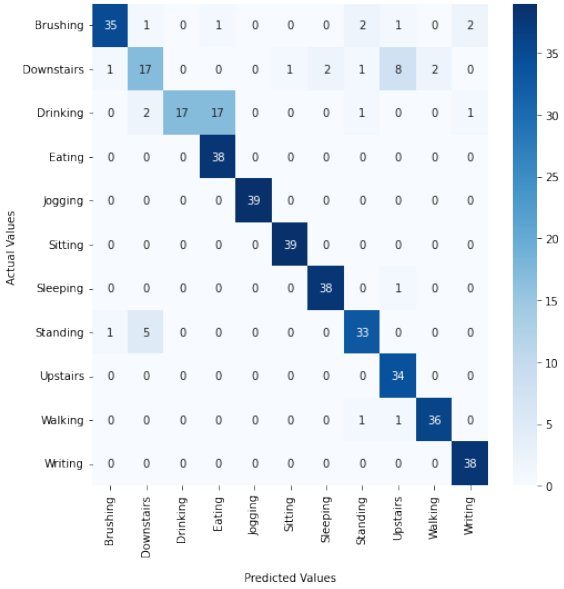

Figure 18: LSTM Confusion Matrix on test data of private model

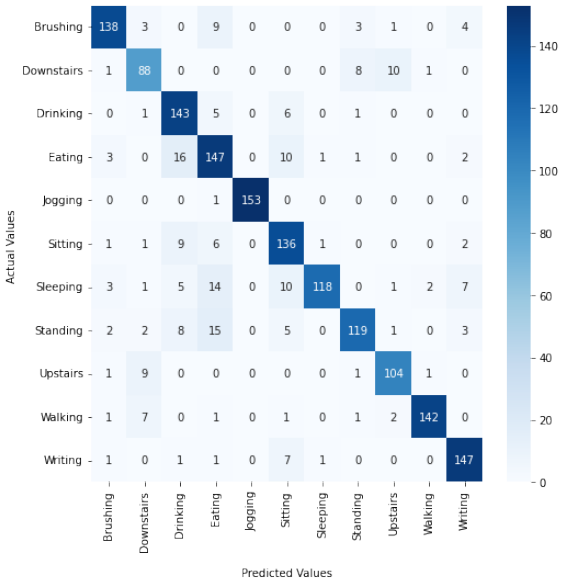

Figure 19: LSTM Confusion Matrix on test data of random model

Sl. No.	Model	Accuracy			
		Personalize Model	Random Split Model	Population Model	FedAvg
1	Logistic Regression	0.95 (Avg)	0.9	0.22	0.08
2	LSTM	0.83 (Avg)	0.86	0.35	0.09

Figure 20: Accuracy for Data Training Strategies

8 Project Management

A project management methodology is a set of principles and practices that guide you in organizing your projects to ensure their optimum performance where Project planning is a key component in the life cycle of of any major developmental venture.In addition ,The project plan gives chance to team members for comprehend the precise actions needed to deliver the finished project.Therefore, it is crucial that significant efforts be made to accurately identify the various activities, organize them in the proper order, and estimate the time required to complete each part of the final product.This strategy will helps to the whole project make successful by it simpler to forecast or predict any possible challenges.
In this project, we adopted the Evolutionary Model(agile) to manage our project which is a strong focus on change management.Agile projects differ from traditional processes in that they consist of a sequence of tasks that are created, carried out, and modified as needed. Through incremental, iterative work processes, teams may adjust to unpredictability by becoming agile.Being that we had to complete each section of the project before moving on to the next and that the requirements were essentially set in stone at the outset of development made it more appropriate to the process we were trying to design. we have followed the below steps to fulfill the targets.

- **Brainstorming Session:** we have taken brainstorming session on daily basis to determine a problem question and we are figure out how to deal with.The session based on clear and prompt participants to think of solutions, such as "How can we differentiate our project ?" or "How can we increase performance of application ?"as well as we have set boundaries for possible solutions for example solutions may have to be implemented within month or involve finding a new methodology.
- **JIRA Task assignment:**In this section, we have step-up JIRA project management for the task management of project which are create most basic workflow and status of completing work.There are below steps in the workflow To do and Done.
  - **Set due dates & reminders:**For the type of tasks we had set due dates and reminders.The assigned person will then be automatically notified when the task is due. In this group project admin become a watcher on the task/issue where he can send reminded as well (Figure 21).

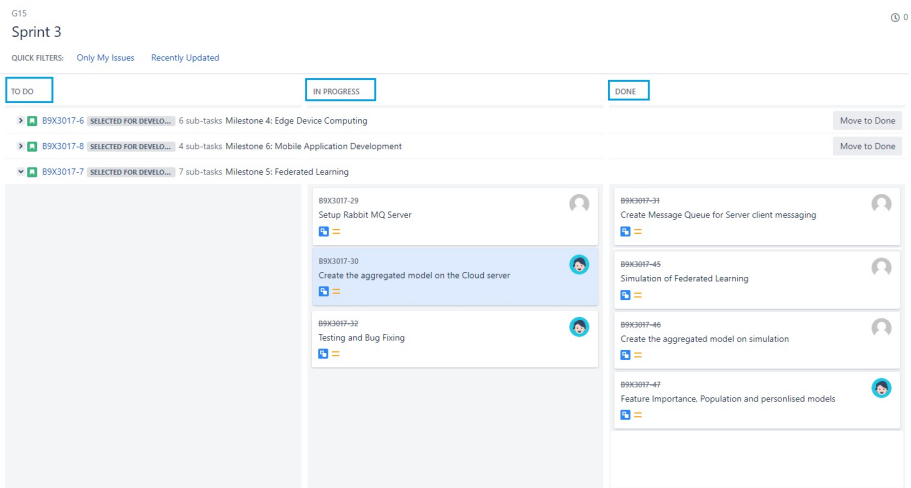

Figure 21: Jira Active Sprint: Activity Tracking

  - **Create sub-tasks:**If team leader assign task to team members where team member individually can create smaller sub task and assign them to different people.The task will not be able to move to Done, until all the sub-tasks are complete as well.
  - **Manage activity on the board:**We are using JIRA Work Management board to track incomplete tasks where we can add new ones and check what has been done.The board gives us a quick view of what is overdue (if we are using due dates) and who is assigned to tasks so admins can follow up.
  - **Tracking:**JIRA project management tool offer to track project progress Time-tracking capabilities enable admin to track the time each task takes (Figure 23). and Project completion in Figure 22.

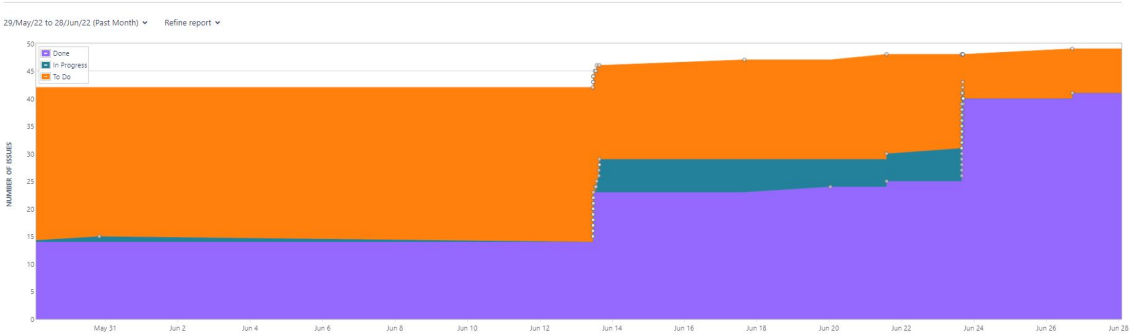

Figure 22: Cumulative Flow Diagram

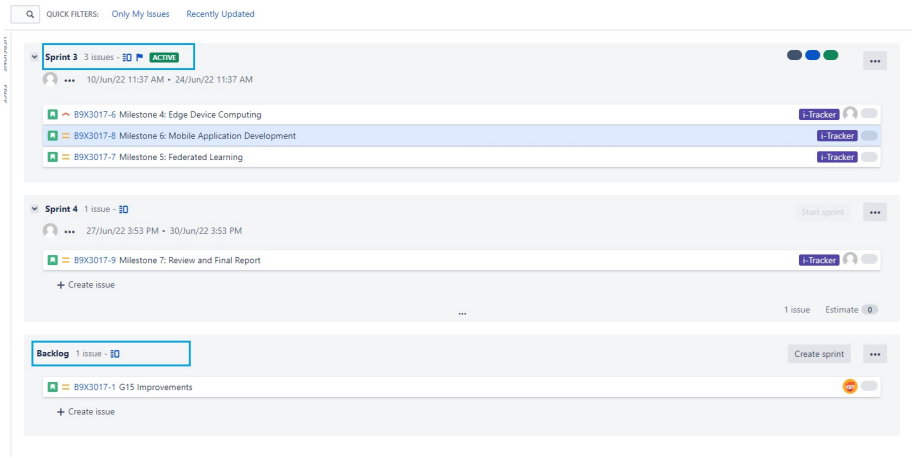

Figure 23: Jira Sprint Tracking

- **Customize issues:**Below (Figure 21) show the specifications for default issues in a task project Jira Work Management admins can customize these defaults. For example, admins can add other issue types to the project, or add new fields to this issue type.
- **Project completion:**After the completion of task/issue team member will get automatically notification mail to notified when the task is done.

- **Daily stand up :** In this session ,we had taken daily stand up team and discuss about the status of a project .Each participant are report on their successes as well suggest potential solutions and strategies where every participant takes only 2 minutes and explain their thought and help people who "think by talking" to organize their thoughts .During this session team member also written down bullet point in there notes because of limited the amount is provided to each member.

- **Knowledge transfer:**In this session,we regularly share our Knowledge with team to avoid knowledge loss.However In the group project, knowledge transfer is help us the process of sharing team' which was the best practices.It was also help us establish a central source of group project information where all team member can access up-to-date group project task as well Knowledge.

- **Retrospective meeting** :It is very important to begin Project Retrospective meeting with a high- scale overview of the project timeline, where any major events that took place.Summarizing the project as a whole reminds the team of what teams initially set out to do and how you were able to accomplish it as a team.the Agenda of Project Retrospective Meeting is valuable to share factual information about the projects execution and outcome in a completely objective manner.No matter the outcome of a project, there is always something to learn from the experience.In this session we are fostering a continuous learning environment for team.it is also making it clear that perfect execution isn't possible and that there is always room for growth and improvement.A lot of time and hard work went into this project, which needs to be recognized by giving thanks to everyone who made the execution possible.we celebrate the good and give acknowledgement to those or all who played a critical

role in the success of the project as well as encouraging project retrospective questions such as, “what should we continue doing?” or “what should we do again for future projects?” will harness a productive conversation with your group. It is a wonderful idea going ahead for your team members to keep logs or create notes to keep track of learning, what is doing well, and what may require some work as the project progresses since conducting project retrospective sessions was beneficial.

- **Testing**
  - **Unit level Testing** :We have done Unit testing where the smallest testable parts of an application tested by individually and independently scrutinized for proper operation.This testing methodology is done during the development process.
  - **Application level Testing**:We have also don Front-end or User Interface testing Back-end testing and server testing that verifies the application behavior.
  - **Integration level Testing:** Integration testing is done to test the modules/components when integrated to verify that they work as expected on server.we had check the combinations of different units, their interactions, the way subsystems unite into one common system, and code compliance with the requirements.

9 Testing of i-Tracker Application

Testing ensures that all functions of our application are working well, enabling the entire structure to cooperate and perform as planned.Test cases for our application are described below in form of tables. The prototype application testing is divided into four parts.

- User Interface Testing,
- Functional Testing,
- Non-Functional Testing,
- Model Testing.

User Interface testing User Interface testing involved testing of the first appearance of our application in front of the user. The main purpose of **Functional testing** involved testing every function of an application by providing adequate input and verifying the output against it.**Non-Functional** testing involved how an application runs on the mobile device and in addition to this its performance results. Further **Model Testing** is performed with the help of three models Logistics Regression (LR), LSTM and LSTM CNN to check accuracy on prediction of results.

- **User Interface Testing**: is a method for testing our application features before a user interacts with them. For User interface testing of our application, we perform UI test cases manually. UI test cases are displayed below in (figure 24).

Mobile App UI Testing				
Test Case_Id	**Test case**	**Test Step**	**Expected Result**	**Status**
IT_UI_001	Test overall color scheme and theme of the app on device	Navigate to home page and explore app	Overall color scheme of whole application is appropriate	Pass
IT_UI_002	Read the application content on the Samsung Galaxy device	Load of application content on mobile device	Mobile application is working on Samsung Galaxy device	Pass
IT_UI_003	Check all activities are in the checklist	Click on the dropdown list	All activities are there	Pass
IT_UI_004	Test the button sensitivity	Response time after clicking	When clicked on button it responds	Pass
IT_UI_005	Check if the page navigation and scrolling are working fine	Scroll application from top to bottom	Navigation and scrolling	Pass
IT_UI_006	Tests the overall responsiveness of the application on the device	Recording and Tracking of an activity	Recorded data is there and making predictions.	Pass

Figure 24: Test Cases for User Interface

- **Functional Testing**: This type of testing is performed to check if all application features are working according to pre-decided requirements. Like while recording of an activity data is stored from all three sensors (ACC, BVP, Gyroscopes). Functional testing Test cases are displayed below in (figure 25).

Functional Testing				
Test Case_ID	**Test Case**	**Test step**	**Expected result**	**Status**
IT_FT_001	Check the correctness of mandatory fields	Skip mandatory field and try to submit	Page is not submitting until all fields are complete	Pass
IT_FT_002	Verify that the application receives data	Record any activity and store data	Data is Recorded. App is making Predictions	Pass
IT_FT_003	Check data is stored from all three sensors (ACC, BVP, Gyroscopes)	Start record of an activity, press stop and go to recorded sensors data	Data details are present there from all three sensors	Pass
IT_FT_004	Check for error messages	Give Input of activity which is not in list for example talking	Application will not work for activity which is not listed	Pass
IT_FT_005	Check activity prediction in real time	Choose activity, press start and after some time press stop.	Making Predictions	Pass

Figure 25: Test Cases of Functional Testing

- **Non-Functional Testing**:The success of our application depends on the performance of our application. It involves how our application runs on the mobile device and its performance. Non Functional test cases are displayed above in figure 26

Non-Functional Testing			
Test Case_Id	**Test Case**	**Expected Results**	**Status**
IT_PT_001	Real Time Activity: Check performance of an application while doing predictions	Prediction should not take more than 3 seconds	Pass
IT_PT_002	Battery Consumption: The app should be lightweight and do not drain much battery while use of an application.	After 3 hours of continuous use, the result shows that the app uses 0.03(ah) of 1500(mAh) of mobile battery	Pass
IT_PT_003	Training Time	Edge computing should not exceed the expected training time	Pass
IT_PT_004	Latency	Communication between federated learning and edge device should not exceed the expected delay time	Pass

Figure 26: Test Cases of Non-Functional Testing

- **Model Testing** In the model testing scenarios models are testing on the dataset which is collected by us during experiment. The main aim of performing model testing is to check raw data for missing values and outliers. The Model testing tables and results are displayed below.

- **Model Testing 1** : A logistic regression model analyses the relationship between one or more existing independent factors to predict a dependent data variable. Implemented models not only help to evaluate data but also provides exceptionally good accuracy. Testing Results of Model Testing Test case 1 are displayed below in (figure 27).

- **Testing Objective**: To check any missing value and outliers. To obtain the best model accuracy.

Model Testing 1			
Test Case_Id	**Test Case**	**Evaluation Metrics**	**Result**
IT_MT_001	Logistic Regression (calculates the likelihood of an event occurring) and improve accuracy by adding Grid search and Feature Engineering	Accuracy, Precision, Recall, F1 score	Pass

Figure 27: Model Testing Case 1

- **Model Testing 2** : A LSTM (Long Short-Term Memory) model is a recurrent neural network that can learn order dependence in sequence prediction challenges. It makes predictions for time series data. Test results for Model Testing 2 are displayed below in (figure 28).

- **Testing Objective**: System is impervious to fluctuations and system parameters are trainable.

Model Testing 2			
Test Case_Id	**Test Case**	**Evaluation Metric**	**Results**
IT_MT_002	LSTM (for classification and make prediction on time-series data)	Accuracy and F1 score	Pass

Figure 28: Model Testing Case 2

10 Future Work

As a future work, we would like to implement better testing strategies for Non Functional Requirements of the proposed Federated Learning system like Power consumption, Latency and Security. Also we would like to enhance the performance of FedAvg strategy and include Fine Tuning strategy for comparison with Personalized models. We would like to enhance our Mobile application for better product functions and User Friendly interface.

11 Conclusions

To conclude, we aimed to develop a mobile application based on Human Activity Recognition using energy-efficient Internet of Things (IoT) platform. The mobile application enabled the user to identify, track and monitor their activities using their Smartphones and wearable devices. We delved into the importance and advantages of the proposed mobile application and Federated Learning approach. We were able to compare and analyze the accuracy of different model learning techniques like Population, Personalized and Randomized models. We applied Feature Engineering Techniques to achieve higher accuracy and build light models. We were able to develop an efficient and reliable communication interface for Federated Learning. Logistic Regression achieved a maximum of 95% accuracy for personalized model and LSTM achieved a 90% for Randomized model. We compared our model results with existing research work and found the proposed models were evaluated on either watch or mobile data and used random split to train the models and achieved good results with combination of Convolution and LSTM layers as opposed to our only LSTM layer model. As a future work we would collect more data before performing Federated Learning and avoid collection of data in controlled environment. We would also like to build ConvNet models and evaluate the accuracy. The main focus of the document is to create a methodology to develop a Federated Learning system and evaluate their efficiency to build a robust end user application. We also discussed some important open research issues in activity recognition and hope to stimulate further research in this important research and development area.

References

[1] Z. Benhaili, Y. Abouqora, Y. Balouki, and L. Moumoun. Basic activity recognition from wearable sensors using a lightweight deep neural network. *Journal of ICT Standardization*, pages 241–260, 2022.

[2] E. Bulbul, A. Cetin, and I. A. Dogru. Human activity recognition using smartphones. In *2018 2nd international symposium on multidisciplinary studies and innovative technologies (ismsit)*, pages 1–6. IEEE, 2018.

[3] M. Chiang and T. Zhang. Fog and iot: An overview of research opportunities. *IEEE Internet of Things Journal*, 3(6):854–864, 2016.

[4] S. Ek, F. Portet, P. Lalanda, and G. Vega. Evaluation of federated learning aggregation algorithms: Application to human activity recognition. UbiComp-ISWC '20, New York, NY, USA, 2020. Association for Computing Machinery.

[5] S. Gupta. Deep learning based human activity recognition (har) using wearable sensor data. *International Journal of Information Management Data Insights*, 1(2):100046, 2021.

[6] F. Hernandez, L. F. D. Suárez, J. Villamizar, and M. Altuve. Human activity recognition on smartphones using a bidirectional lstm network. *2019 XXII Symposium on Image, Signal Processing and Artificial Vision (STSIVA)*, pages 1–5, 2019.

[7] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2:429–450, 2020.

[8] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, and C. Miao. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys and Tutorials*, 22(3):2031–2063, 2020.

[9] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[10] M. Milenkoski, K. Trivodaliev, S. Kalajdziski, M. Jovanov, and B. R. Stojkoska. Real time human activity recognition on smartphones using lstm networks. In *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 1126–1131, 2018.

[11] N. Milstein and I. Gordon. Validating measures of electrodermal activity and heart rate variability derived from the empatica e4 utilized in research settings that involve interactive dyadic states. *Frontiers in Behavioral Neuroscience*, 14, 2020.

[12] C. A. Ronao and S.-B. Cho. Human activity recognition with smartphone sensors using deep learning neural networks. *Expert systems with applications*, 59:235–244, 2016.

[13] E. Sannara, F. Portet, P. Lalanda, and V. German. A federated learning aggregation algorithm for pervasive computing: Evaluation and comparison. In *2021 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 1–10. IEEE, 2021.

[14] K. Sozinov, V. Vlassov, and S. Girdzijauskas. Human activity recognition using federated learning. In *2018 IEEE Intl Conf on Parallel Distributed Processing with Applications, Ubiquitous Computing Communications, Big Data Cloud Computing, Social Computing Networking, Sustainable Computing Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom)*, pages 1103–1111, 2018.

[15] G. Wang, C. X. Dang, and Z. Zhou. Measure contribution of participants in federated learning. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 2597–2604, 2019.

[16] Q. Wu, K. He, and X. Chen. Personalized federated learning for intelligent iot applications: A cloud-edge based framework. *IEEE Open Journal of the Computer Society*, 1:35–44, 2020.

[17] Q. Xia, W. Ye, Z. Tao, J. Wu, and Q. Li. A survey of federated learning for edge computing: Research problems and solutions. *High-Confidence Computing*, 1(1):100008, 2021.

[18] D. Xu, T. Li, Y. Li, X. Su, S. Tarkoma, T. Jiang, J. Crowcroft, and P. Hui. Edge intelligence: Architectures, challenges, and applications. *arXiv preprint arXiv:2003.12172*, 2020.