A Deep Learning approach to smartphone based Human Activity Recognition with CNN and LSTM models

(This project report has been submitted in partial fulfillment of the requirements for the degree of Bachelor of Science in Electrical and Electronic Engineering)

Submitted By

Exam Roll: 29960

Reg No: 2016614965

Session: 2016-17

Exam Roll: 29901

Reg No: 2016615018

Session: 2016-17

DEPT. OF ELECTRICAL AND ELECTRONIC ENGINEERING UNIVERSITY OF DHAKA

February 7, 2022

Abstract

This project investigated the use of a combination of Deep Learning models in Human Activity Recognition (HAR) using smartphone sensor data. It works with 3 datasets on Human Activity Recognition (HAR) based on sensors, namely UCI Dataset, PAMAP2 Dataset and ARS DLR Dataset. The sensors used in this project were accelerometer and gyroscope sensors present in the Inertial Measurement Unit (IMU) of smartphones and in case of PAMAP2, heartrate sensor and magnetometer sensor were also included. Various approaches like windowing, filtering, normalization etc. were used to preprocess the datasets to make them fit for use with deep learning models. As some datasets had imbalanced data for classes, certain augmentation methods were used to produce synthetic data to balance out the datasets. The datasets contained both static and dynamic activities and in case of PAMAP2, complex activities like ironing and vacuum cleaning. Three combinations of CNN and LSTM layers were selected for this project and trained on the datasets aforementioned. Finally, comparison was done on the performance of these models and it was found that our designed model was better at generalization and accuracy overall.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Motivation and Goal:

With the advancement of technology, machine learning methods and deep learning methods are being applied in numerous fields. Among these, Human activity recognition (HAR) is considered to be one of the most significant topics in the active research field. Human activity recognition intends to monitor, recognize a person's activity based on a series of observations and the surrounding environment. In easy words, HAR technology is used in recognizing the daily activities of humans, from simple activities like standing, sitting, walking upstairs, running to complex activities like cooking while standing, watching TV while sitting, or lying [12].

Recognition of human activity, due to the easy availability of wearable devices and sensors at low cost has become an integral part of everyday human life and is widely used in some domains like health controls - adult surveillance, disease prevention, rehabilitation, and in smart cities – for monitoring. In addition, HAR is used in security concerns as monitoring solutions for each activity as well as anonymous crowd detection. In addition, wearable and internal sensors combined with embedded systems are used in sports activities [13, 14].

The purpose of this research is to create a model that can detect daily and simple human behaviors in real time. Our objective is to detect events in the ARS DLR

dataset in real time utilizing two types of sensor data to keep things simple (3-axis accelerometer and 3-axis gyroscope). Furthermore, we wish to compare the performance of our model to that of a model published in the study [15] that has shown promising results on a variety of standard datasets. We compared a multi-sensor dataset called PAMAP2 to a single-sensor dataset called UCI.

1.2 Approaches to human activity recognition

To recognize human activities, various methods can be used. Approaches may vary in the type of data collection and data processing to the acquisition methods. Depending on the collection of work data, HAR can be divided into two types:

- Computer Vision based
- Sensor based

1.2.1 Computer Vision based approach

In a computer vision-based approach, images and videos are collected employing optical sensors like cameras, CCTVs, and then captured images or videos are analyzed. Vision-based data being affordable and collectible hardly with any trouble, a myriad of research has been conducted on vision-based HAR [16]. On the contrary, contact-based or wearable sensor-based approach sometimes requires sophisticated equipment, and also it has to be affordable in terms of cost, correct size, and user's acceptability. Moreover, vision-based systems will not require the user to wear devices uncomfortable to them in different parts of the body. In vision based approach mainly three stages are followed which include – 1. detection (first stage), which determines the part of the body to recognize or follow (methods like using skin color, shape, pixel values, etc. are used), 2. Tracking (second stage), where links between successive images are provided (methods like feature tracking, contour tracking, optimal estimation, etc. are used) and 3. Classification, which is the final stage, and where different machine learning and deep learning algorithms are used to finally recognize the activity [17].

1.2.2 Sensor based approach

As mentioned, human activity recognition can be carried out in a sensor-based approach. Sensors of different kinds are used in data acquisition for the recognition to be performed. Sensors can be integrated into a device or can be used separately. Based on the platforms used, sensors can be:

- Wearable Sensors
- Smartphone Sensors

Wearable Sensors: These forms of sensors are used for only one particular purpose/function. These sensors can only perform tasks for which these were designed. Usually, these sensors are integrated into a device used just for a given task only. Wearable sensors can further be classified into more types – Inertial sensors which include accelerometer, gyroscope, magnetometer, Physical health sensors which include heart rate sensors (HR), skin temperature, oxygen saturation (SPO2), etc., Environmental sensors which include, temperature, barometer, humidity, light sensors, etc. Sometimes these wearable sensors are not utilized in applications, for example in human activity recognition due to size, price, and acceptability to carry by the user [18].

Smartphone Sensors: Since smartphones, as well as smartwatches, are easy to carry and use, they are used widely to collect data. Today smartphones and smartwatches are embedded with sensors like accelerometers, gyroscopes, barometers, GPS, temperature, etc. Smartphones and watches are now also used in other fields alongside human activity recognition, like, health monitoring, monitoring sports activities, etc. [14, 18].

1.3 Challenges in sensor based human activity recognition

In conducting human activity recognition, over the years, a variety of challenges have been faced. Challenges vary from type of sensors to datasets to be used to methods or models and many more.

1.3.1 Appropriate Sensors and Selecting Sensor Types

First of all, the problem arises and much challenge is faced in choosing a suitable sensor or set of sensors as well as the type of sensors that will be used to collect raw data.

Challenges in Wearable Sensors:

Sensor-based human activity recognition eliminates the drawbacks of vision-based systems, where data has to be collected through external sensing methods like cameras. When collecting data utilizing a camera or CCTV the user or the subject has to stay within a particular range, which limits the reliability. Furthermore, data taken by cameras are influenced by the background, daylight, weather, etc. and thus, sensor-based systems are preferable which eliminates all of these drawbacks [19, 20].

Now the challenge arises in choosing the suitable type of sensors. Sensors can be separate wearable sensors or sensors used in smartphones. Each has its trade-off. A wearable sensor is used for only a particular purpose whereas smartphones have multiple sensors embedded in them. The first challenge in wearable sensors is to select which sensor to use, that is whether to use only one sensor or multiple sensors at a time. If only one sensor is used the data processing part becomes trouble-free. But the recognition accuracy and thus the performance might not be satisfactory. On the other hand, with the use of multiple sensors, the accuracy and performance of the system will be more acceptable and better. Moreover, one sensor-based system can be smart for recognizing straightforward activities like walking, sitting, standing to take a seat, etc., though not ensured to good accuracy, however, can face several challenges in sleuthing complicated activities and cannot provide high accuracy. Hence, a multi-sensor-based system is desirable over one detector. Multiple sensors like measuring instruments, gyroscopes, meters, etc. are used to collect information and acknowledge complicated activities [1, 21].

Though multi-sensor-based wearable sensors have shown high accuracy and good performance, challenges are also faced in this method. First, if to be mentioned, users may feel uncomfortable in wearing or carrying these sensors while data is being collected. And it is problematic to carry multiple sensors and data becomes

difficult to process. Besides, in unfavorable environments, hindrances are faced while calibrating and collecting data [14, 22]. To overcome these challenges smartphones and smartwatches have been preferred in recent years. Especially with the advancement of technology, powerful smartphones and smartwatches have been introduced, which have high computation capacities along with powerful sensors. Smartphones are used in our daily lives and it is not only a communicating device but also a good sensor platform and sensors are calibrated easily as well as data can be transferred easily using Bluetooth, Wi-Fi, etc. [3, 23].

Challenges in Smartphone sensors:

Challenges are faced in using smartphones, more precisely in collecting sensor data. Moreover, another problem with smartphones is that the rapid loss of battery energy. The most challenge that's assumed is that the variation of the situation of smartphones. In quite a number of papers like [3, 24, 25, 26, 27], this problem has been discussed and also solutions were proposed. Unlike wearable sensors, smartphones aren't kept in a fixed position and it's not reasonable to try to do so either. Smartphones are often kept in shirt or pant pockets, bags and different pockets in bags, etc., and also these are kept in either right or left alongside various orientations. Signals will vary in these conditions which makes it arduous to recognize the particular activity. If activity recognition is done using smartphone data while keeping it in only a particular position, orientation, or place, low accuracy, and wrong recognition will result. And, it is the greatest challenge to make the recognition system with smartphones a position/location independent system. And also data collected from smartphones may contain a lot of noise. Hence, research has been conducted and is still being conducted focusing on such problems with the location of smartphones [3, 23, 24].

1.3.2 Choosing suitable methods

Using Hand Crafted Features:

Apart from the above-mentioned challenges, one of the biggest challenges is the method to choose for recognition i.e. whether to use a hand-crafted featured model

or a deep learning model. Whether a hand-crafted featured model or deep learning model is used, selecting and finding appropriate features for the recognition system is the most significant task in human activity recognition. Feature extraction is more challenging as different activities can have similar characteristics and hence it becomes strenuous to obtain unique features for each activity [28, 29]. Hand-crafted featured models require features to be selected manually, knowledge equivalent to an expert is required. Moreover, hand-crafted features might not work for other similar applications and also will face many difficulties while recognizing complex activities [2, 30]. Furthermore, hand-crafted featured human activity recognition systems and most importantly labeled data from sensors are vital for good performance. Otherwise, expected accuracy and performance will not be obtained. Additionally, to collect such data and hence prepare proper datasets sophisticated infrastructure might be required leading to more cost and time consumption. Also, many datasets have labeled data indeed but those labels might be assigned to data having information from secondary activities. And, training errors might occur in machine learning algorithms due to such data and degrade the performance, especially while a class or activity is being assigned to a data segment [31, 32].

Deep Learning:

In order to tackle the challenges and drawbacks of hand-crafted features, deep learning models are used in recent years. In deep learning models, necessary features are automatically extracted from raw sensor data. Unlike popular machine learning algorithms such as KNN (K nearest neighbor), SVM (Support Vector Machine), etc. deep learning models don't need carefully engineered hand-crafted features as they can develop the most efficient features from raw data [33]. Deep learning models contain multiple layers and the path deepens and hence the name deep learning [13]. Deep learning methods are being used in human activity recognition(HAR) to automatically extract useful features from raw sensor data using multiple layers of abstraction. And it is applied in HAR concerning the movements of human beings hierarchically [32]. Numerous deep learning models using deep learning networks like CNN (Convolutional Neural Network), LSTM (Long Short Term Memory), Bi-LSTM (Bidirectional Long Short Term Memory), RNN (Recurrent Neural Network), etc. have been introduced in a large number of papers.

For example, in CHIH-TA YEN et al. a deep learning model using CNN has been proposed [13], in Xiaokang Zhou et al. LSTM based model [1] and in [32] RNN based model has been proposed to overcome previous challenges and obtain better performances.

1.3.3 Challenges in Deep learning method

The implementation of deep learning models doesn't mean that the system will not have a lack of performance and efficiency. Challenges are faced even when using deep learning models. As we know deep learning models have numerous layers, these require a large number of initializations and parameter tuning. All of these escalates computational costs, time and require powerful processing units for fast computations. Moreover, due to such large computations, low processing devices and low energy mobile devices aren't suitable [33]. Hence, pre-processing and dimensionality reduction are significant in the recognition system. Dimensionality reduction can lessen the computational complexity mentioned above. Now, one of the greatest challenges is the pre-processing of the data to achieve optimum performance. For this, various processing techniques such as normalization, standardization, etc. have been used and need further experimentation to find suitable accuracies and performances which is the challenge. Moreover, challenges remain in issues with hyper-parameters like learning rate optimization, kernel filter size, reduction of data size, etc. [34]. To improve the performance, the data augmentation method is used to generate more training examples from the existing small dataset and to reduce and sometimes prevent overfitting. For data augmentation techniques like - permutation of location with sensor events, arbitrary rotation, etc. are used. This method is still a challenge for implementing motion sensors' (accelerometer, gyroscope) data for improving performance [10].

1.3.4 Appropriate Datasets

Apart from all the challenges that have been mentioned above, challenges that cannot be denied, remain in using proper datasets. Custom datasets can be prepared by collecting data using wearable sensors, smartwatches, or smartphones. But preparing custom datasets is challenging because the raw data needs to be pre-processed properly for the deep learning layers to extract useful features and perform recognition. Since sometimes many challenges are faced in preparing custom datasets, publicly available datasets can be found and used for human activity recognition. In KAIXUAN CHEN et al. [28] information and also challenges about publicly available datasets have been given. For example, the most common three datasets used are UCI HAR, WISDM Activity Prediction, OPPORTUNITY, and their challenges that for the first and third one have been mentioned to multimodality of the data whereas for the second dataset, the challenge is said to fix class imbalance [7].

1.4 Research Objectives

This research aims to investigate the use of deep learning in the recognition of human activity (HAR). Convolutional Neural Networks (CNN) and Long-Short Term Memory (LSTM) models are widely used in time-series classification and they are worth investigating for use in Human Activity Recognition. The combination of these two models has a lot of potential since they compensate for each other's shortcomings. The most effective approach to combining the two models was investigated in the project. Overfitting for smaller datasets is a typical issue with deep learning models. Because Human Activity Recognition (HAR) datasets are often smaller, this study also looks at the impact of augmentation approaches on enhancing accuracy and precision.

1.5 Project Overview

In Chapter 1 “Introduction”, a brief introduction of Human Activity Recognition has been provided along with the concepts and motivation of this project. A brief description of various methods in this field has also been given. Furthermore, different challenges that are faced while performing research in this field have been discussed. Lastly, the objectives of our research are reviewed in short.

In Chapter 2 “Related Works”, we have discussed other works related to this field. Various methods in sensor-based HAR of previous works along with their accuracies have been overviewed. Moreover, we have also made a comparison between methods of previous works along with their accuracy along with problems in respective works.

In Chapter 3 “Theoretical Overview”, the theoretical overview of this work was discussed. Sensors and datasets utilized in the study were described in detail. The theoretical features of the project’s models were then explained. Finally, the augmentation procedures and data pre-processing were discussed.

In Chapter 4 “Methodology”, the research methodology of the project was discussed in detail. The preprocessing and augmentation techniques were explained further with the datasets used in the project. Description of the models and their training parameters were added after that.

In Chapter 5 “Result and Analysis”, Results obtained from training the models were shown and analyzed. Comparison between the performance of the models was given. Finally, insights obtained from comparison of the results obtained were discussed.

In Chapter 6 “Conclusion and Future Works”, conclusion has been drawn and scope of improvement in the future in this research has been discussed.

CHAPTER 2

RELATED WORKS

Sensor-based Human Activity Recognition has been the subject of several studies throughout the years. New approaches, models, and even devices have been presented in the studies to increase performance and flexibility in a myriad of postures, settings, and devices. A microcontroller-based gadget worn around the waist is utilized by Yen et al. [13] to identify six fundamental human activities (walking, sitting, standing, lying, going upstairs, and going downstairs). An inertial sensor with a microprocessor, a three-axis accelerometer, and a three-axis gyroscope comprised the wearable gadget. They implemented a recognition technique that included signal acquisition, signal normalization, and a feature learning method, with the feature learning method being focused on a 1D Convolutional Network that can automatically extract features and perform classification from raw data. They employed both a publicly available dataset (UCI-HAR) and their own raw data. On training samples, the technique and model employed in the study produced an accuracy of 98.93% and 97.19% on UCI-HAR and recorded data, respectively, and 95.99% and 93.77% on test samples.

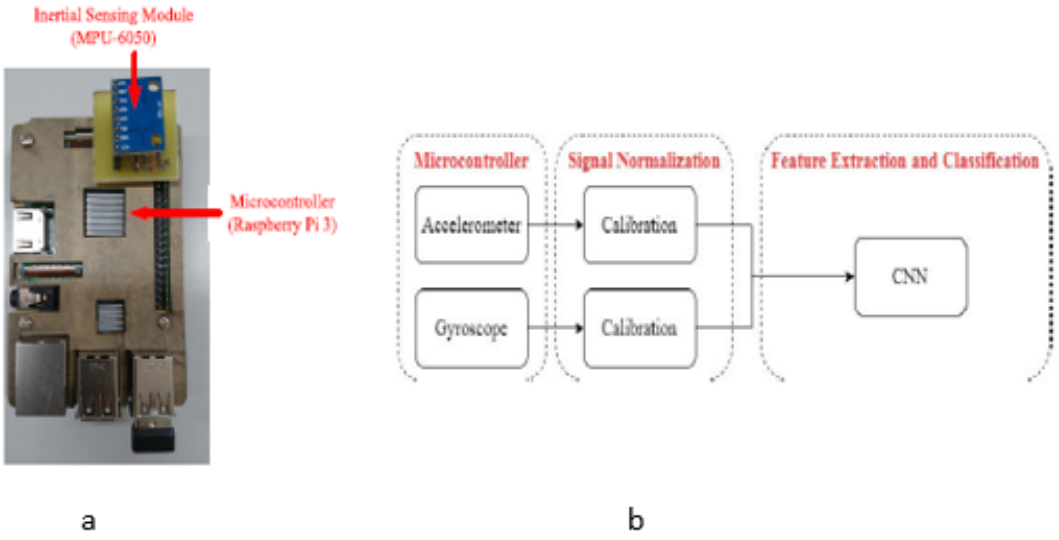


Figure 2.1: (a) Structure of the hardware used in the paper; (b) The proposed algorithm

In K. Xia et al. [15], a deep learning model consisting of LSTM and CNN is proposed where raw data collected from mobile sensors were fed. The proposed architecture in the paper also contained GAP (Global Average Pooling) layer in order to reduce model parameters and a batch normalization layer after it to speed up the convergence and training process. The model was evaluated on three public datasets (UCI-HAR, WISDM, and OPPORTUNITY) where overall accuracy for UCI-HAR, WISDM, and OPPORTUNITY were 95.78%, 95.85%, and 92.63% respectively.

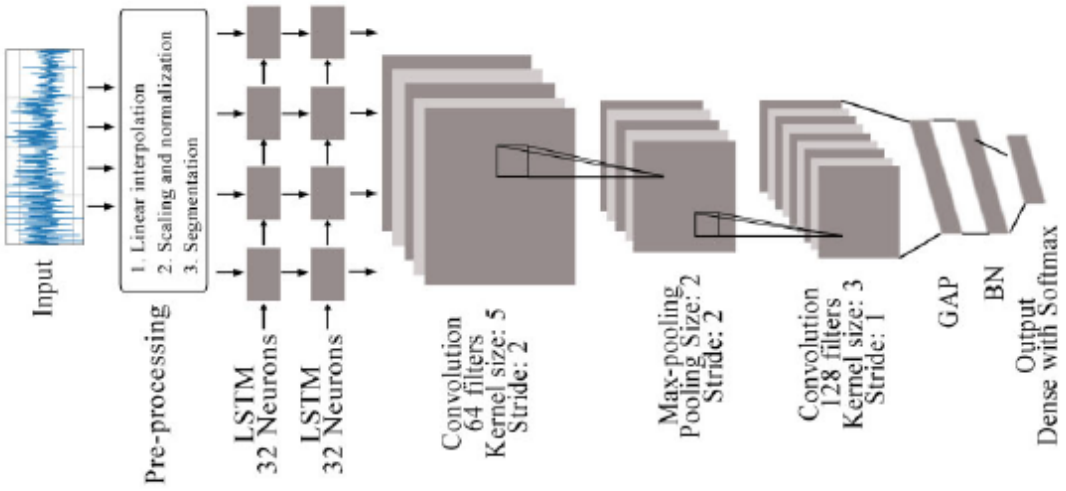


Figure 2.2: Frame diagram of the proposed model

One of the most popular fields in which HAR is applied in health care. With the advancement of body sensors, this technology is being applied in health care. Moreover, due to the ongoing popularity and advancement in the Internet of Things (IoT), sensor-based HAR has become a hot topic in smart healthcare which can escalate rehabilitation for elderly and weak people [1].

In Md. Zia Uddin [20], an approach using body sensor and CNN model (deep Convolutional Neural Network) was proposed for smart healthcare. In the research signals from body sensors like ECG, magnetometer, accelerometer and gyroscope were used and analyzed to extract suitable features using methods like Gaussian kernel-based PCA (Principal Component Analysis) and Z-score normalization. Based on the extracted features the proposed CNN model was trained and the entire approach was applied not only on raw data collected but also on

the mHealth public dataset (which contained recordings of 10 subjects for 12 activities) and compared with other typical approaches. Their approach resulted in an average accuracy of 93.90% whereas approaches that they compared with – ordinary ANN (Artificial Neural Network) with an average accuracy of 87.99%, DBN (Deep Belief Network) with an average accuracy of 90.01%.

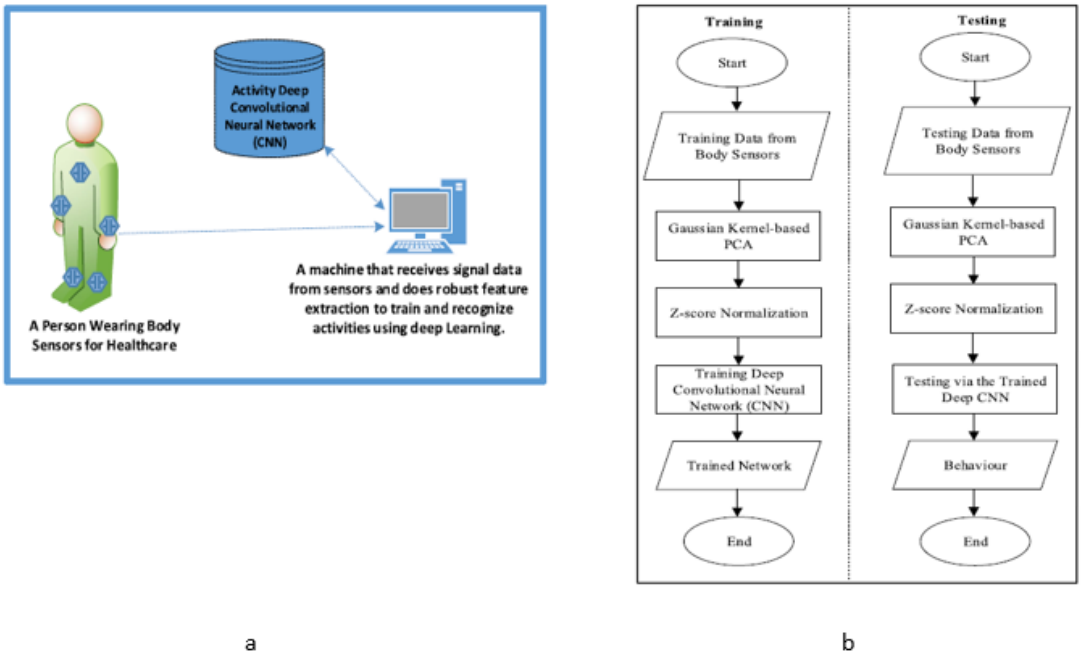


Figure 2.3: Figure - 2.3: (a) Schematic setup for the recognition system; (b) Flow chart for the proposed approach

In Xiaokang Zhou et al. [1], a semi-supervised deep learning framework for HAR in an IoT environment was introduced. The deep learning model was developed in order to improve the efficiency of HAR performance and deal with feebly labeled sensor data. In order to apply auto labeling, a method or scheme which is a Deep Q Network (DQN) was designed and it was developed based on a distance-based reward rule and along with it, an LSTM based architecture was implemented to train and perform the recognition. An evaluation was done on finding accuracies on different positions of sensors on the body and results were compared with other approaches like typical DNN (Deep learning network), typical machine learning approaches like Random Forest (RF), and Support Vector Machine (SVM). The proposed method gave higher accuracies on different body positions than the other

methods. Experimental results obtained from fusing different features from different sensors were evaluated based on performance metrics like Recall, Precision, and F1-score, where the proposed method gave the best performance.

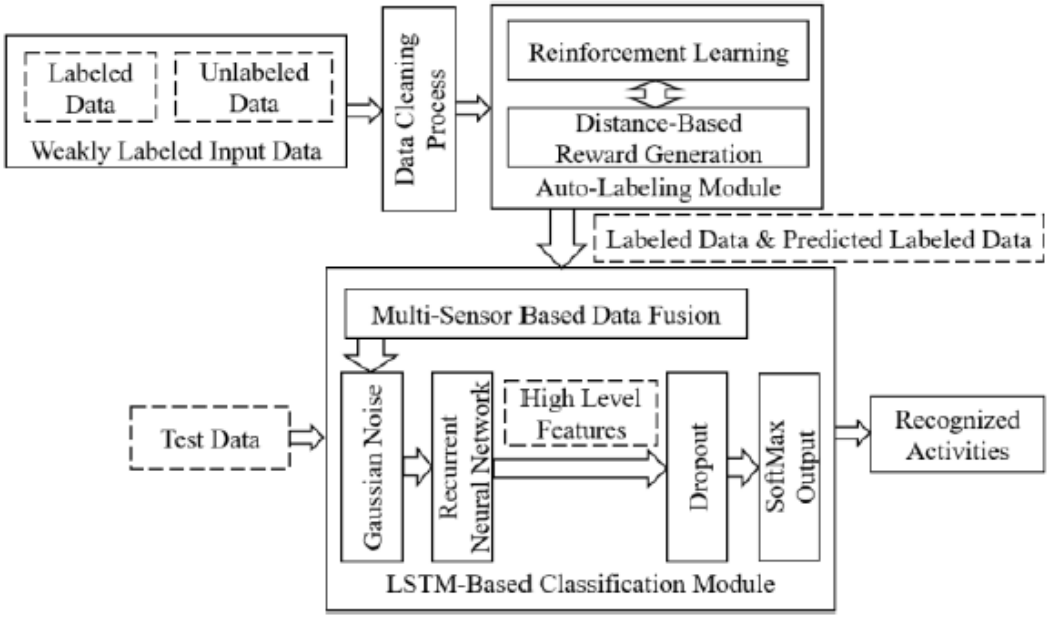


Figure 2.4: The proposed semi – supervised deep learning framework in [1].

Indeed, there has been much advancement in wearable sensors for which the overall performances of HAR have increased with the introduction of many frameworks, methods, or models after conducting research, but the problem with wearable sensors, as mentioned many times, is that its requirement of additional sensing components along with user discomfort. Hence, with the technological advancement of smartphones, smartwatches, and thus sensors in these devices, smartphones are used in the field of HAR widely nowadays. A myriad of researches has been and is also being conducted using smartphone data due to their availability and being widely used by everyone in their daily lives.

In C.A. Ronao [2], a robust deep learning framework with a deep convolutional neural network (convnet) has been proposed to improve the performances of HAR using smartphone data. The convnet automatically extracts useful features and the temporal–dependency of local time-series data is eliminated by the convolution

layer while the small input translations are eliminated by the pooling layer used in the model. Dataset was prepared collecting raw data with smartphones, from 30 subjects performing six daily activities with the smartphone in their pocket. In this research, they used 7352 samples (21 subjects) for training and 2947 samples (9 subjects) for testing purposes. The model after training and validation results were compared with other state-of-the-art methods. The proposed model combined with a multilayer perceptron classifier (MLP) gave an accuracy of 94.79% on test samples and using temporal Fast Fourier transform (tFFT) on the data the proposed convnet gave a slightly increased accuracy by approximately 1% which was 95.75%.

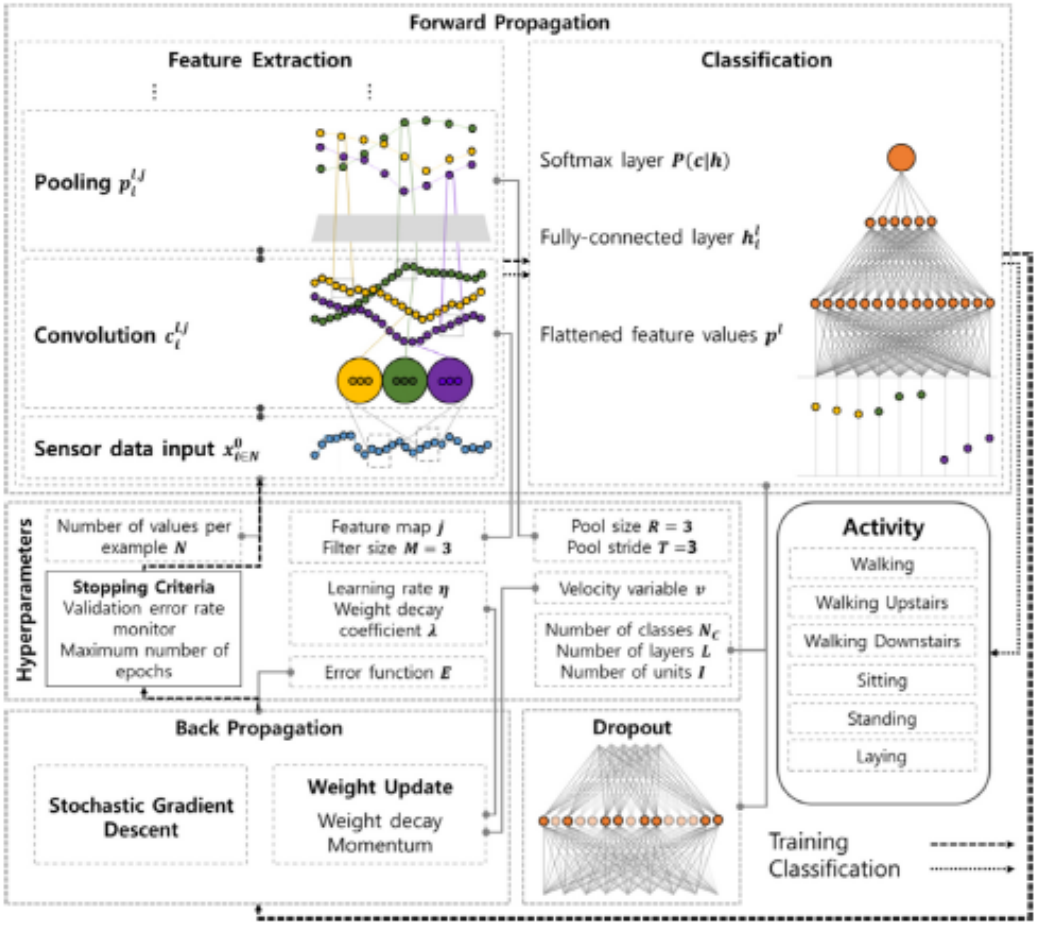


Figure 2.5: The overview of the model proposed in the paper [2].

In the above-mentioned paper (C.A. Ronao [2]), smartphone data were used for

HAR. But smartphone has some issues to be dealt with, which includes the variation in orientation, positions/ locations and these will vary the signals for the same activity which will degrade the overall performance and the reliability of HAR.

Zhenghua Chen et al. [3], has worked on the problem that occurs due to orientation issue in smartphones. A HAR system was introduced in the paper-based on coordinate transformation and principal component analysis (CT-PCA) and online support vector machine (OSVM). PCA eliminates the problem due to orientation and improves the accuracy of the system and the model showed effective performance on different placements and subjects separately. Again, to overcome the degradation of performance over different placements overall, an online independent SVM algorithm was used. Results were obtained for many orientations and compared with other state of art methods, where the proposed method outperformed all of those in every orientation and subject. For instance, for the orientations in Figure 2.6 (a), (b), and (c) accuracies achieved by the proposed method were 96.22%, 94.89%, 93.56% respectively, and a total average of 94.89%.

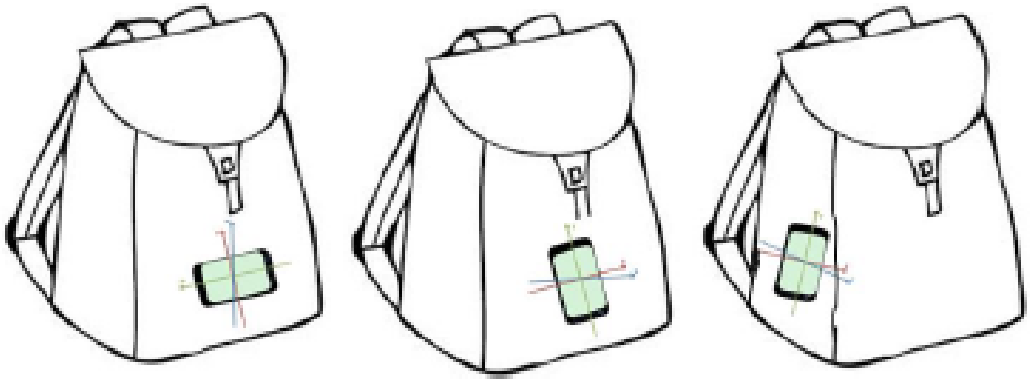


Figure 2.6: One set of orientation experimented in Zhenghua Chen et al. [3]

In Rong Yang [4], another method for dealing with smartphone location was introduced which is known as PACP (Parameters Adjustment Corresponding to smartphone Position). In this method, according to the paper, features were extracted from raw data of accelerometer and gyroscope and used to recognize the position of the smartphone first. Then the sensor data were adjusted and thus necessary features were extracted and train the model to recognize the activities. In the research, they collected data by building an iOS app, and data were

collected from 10 volunteers performing 5 activities (walking, standing, running, going upstairs, and going downstairs) keeping the phone in 4 different positions (bag, trouser pocket, coat pocket, and hand). Results obtained by the proposed PACP method were compared with another method described in Mario et al. [35]. The proposed PACP method gave better accuracy of 91.27% whereas in Mario et al. the accuracy was 87.24%.

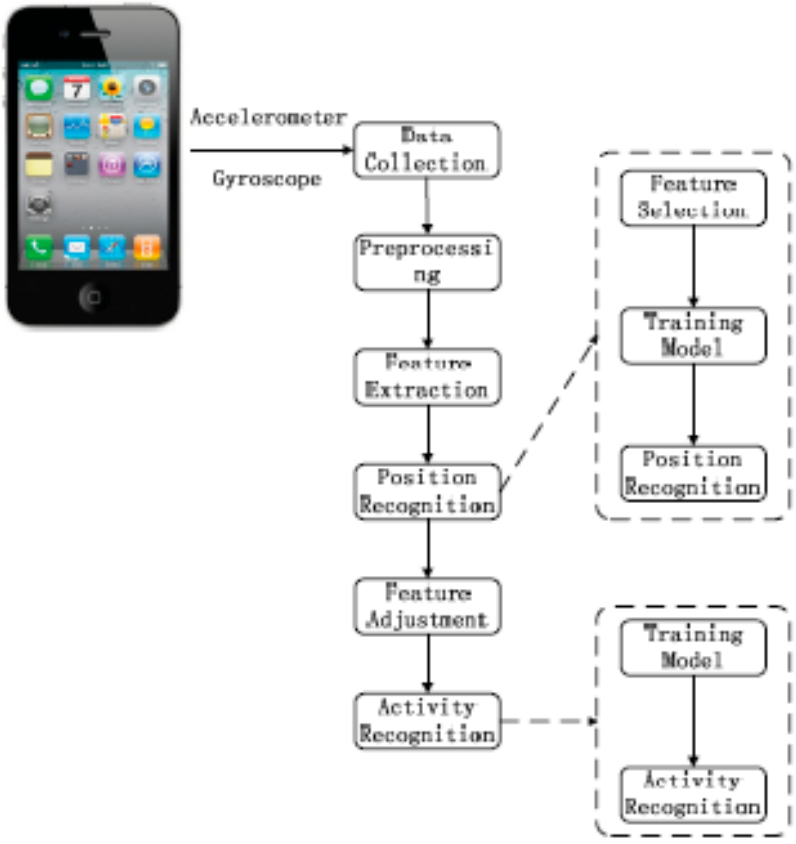


Figure 2.7: An Overview of the proposed PACP method in Rong Yang [4].

In Masud Ahmed et al. [36], a Position and Orientation Independent Deep Ensemble Network (POIDEN) was introduced to improve the performance of HAR system irrespective of smartphone orientation and position/location. The method was developed and tested for complex activities like sitting on a bike or train or subway. The proposed architecture uses an LSTM network to select an optimum classifier for activity recognition. For the purpose of classifier selection i.e. for the

LSTM network, they used an intermediate feature set (IFS), while for the classification purposes statistical classifier feature set (SCFS) was used. Moreover, in order to deal with the orientation characteristics, the rotational characteristics of the accelerometer and magnetometer were transformed from local coordinate to earth coordinate, and additionally jerk features, position-independent features were used along with parameter adjustment. In the paper, SHL (Sussex – Huawei Locomotion) dataset was used where there were 59 days' equivalent training data, 3 days of validation data, and 20 days of test data, varying smartphone positions from 4 places (torso, hip, hand, and bag). The result achieved from the architecture was evaluated against the dataset. Since the activities were very complex the overall F1 score came out to be 70.57% and accuracy to be 67.5%.

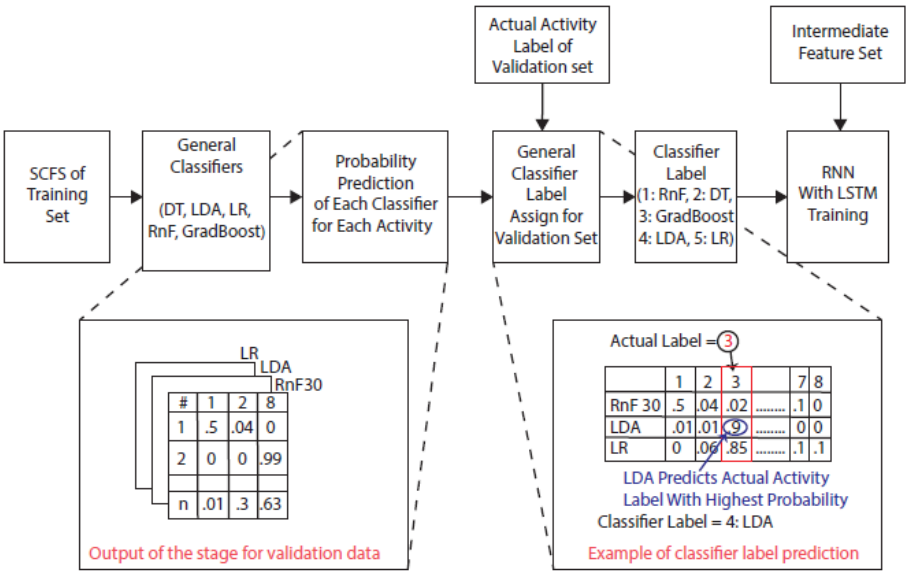


Figure 2.8: The proposed POIDEN architecture.

CHAPTER 3

THEORETICAL OVERVIEW

The project's purpose is to be able to categorize human actions using smartphone sensors. In today's world, every smartphone has an integrated IMU (Inertial Measurement Unit). Some basic activities can be detected using data from the IMU sensors if the smartphone is in the appropriate posture. Data from an accelerometer and a gyroscope were utilized in this experiment to anticipate human activities such as walking, sitting, standing, climbing, and so on. In terms of smartphone orientation, the accelerometer, gyroscope and magnetometer sensors each offer data for three axes (X, Y, and Z). Heartrate sensor just provides the instantaneous value of the heart rate.

3.1 Sensor Description

Accelerometer: The accelerometer measures the phone's acceleration in relation to its reference frame. The measurement is made on three axes: X, Y, and Z, which are defined by the alignment of the smartphone IMU. The basic reference frame is oriented with the X-axis pointing to the right side of the front face, the Y-axis pointing up, and the Z-axis pointing outward from the front face. The measurement is given in unit ms-2. The sampling frequency and precision vary depending on the smartphone, ranging from 20 to 200 Hz [12]. The data used in this project was resampled to 50 Hz for the ease of usage.

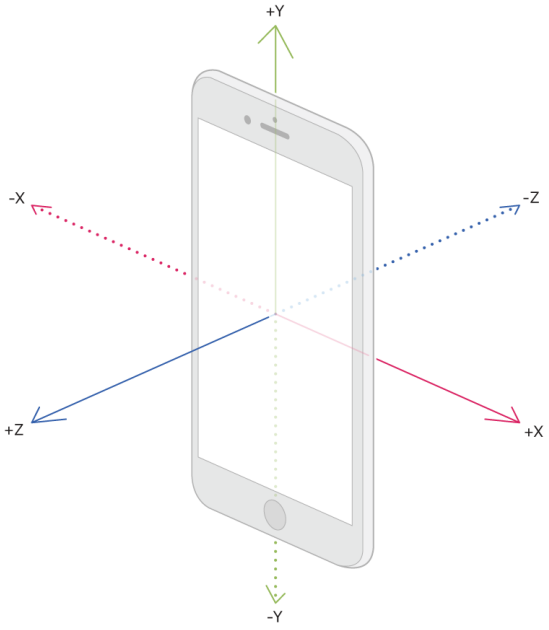


Figure 3.1: Smartphone Coordinate axes.

Gyroscope: The rotational velocity of the gadget around the three coordinate axes is returned by this sensor. The gyroscope sensor uses the same reference frame as the accelerometer. The sensor's data is measured in degrees per second (°/sec). The sampling frequency of this sensor varies from 20 to 200 Hz depending on the smartphone utilized, and data from this sensor was resampled to 50 Hz for use in this project [14].

Magnetometer: This sensor measures the strength and at times the direction of magnetic fields surrounding the device. It is used by a smartphone for orientation calculation and finding direction. The output is usually measured in micro-Tesla (μT). The sampling frequency is the same as the other sensors between 20-200 Hz [14].

3.2 Dataset Description

For this project, datasets containing activities of daily life (ADL) were used. The datasets used are namely:

- UCI HAPT Dataset
- PAMAP2 DATASET
- ARS DLR Dataset

A brief overview of the three datasets has been given in Table-3.1

Table 3.1: Overview of three datasets.

Dataset	Activities	User	Total Samples	Sampling Rate
UCI HAPT	12	30	815614	50Hz
PAMAP2	18	9	1940872	100Hz
ARS DLR	7	16	991638	50Hz

UCI HAPT Dataset:

The tests were conducted on a group of 30 volunteers ranging in age from 19 to 48 years old. They performed an operating procedure consisting of six basic activities that included three static (standing, sitting, and lying) and three dynamic (moving) postures (walking, walking downstairs, and walking upstairs). Postural transitions between the static postures were also included in the study. Stand-to-sit, sit-to-stand, sit-to-lie, lie-to-sit, stand-to-lie, and lie-to-stand are examples of these positions. During the study, all of the volunteers wore a smartphone around their waist (Samsung Galaxy S II). Using the device's inbuilt accelerometer and gyroscope, 3-axial linear acceleration and 3-axial angular velocity were recorded at a constant rate of 50Hz. The experiments were videotaped so that the data could be manually labeled. This project took advantage of the data from inertial sensors. All transition data were renamed transition and grouped into a single class. Walking, walking downstairs, walking upstairs, standing, sitting, lying, and

transitions were the total number of activity classes. Both accelerometer and gyroscope data are included in this dataset [37]. The distribution of activities in the dataset is shown in Table 3.3 where it can be seen that percentage of postural transitions is far less than the rest of the activities.

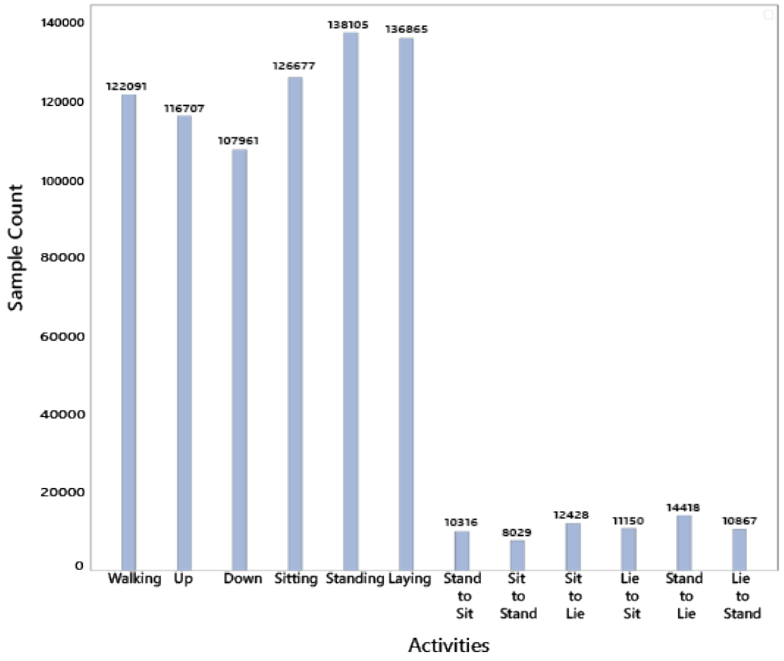


Figure 3.2: Sample distribution of activities in UCI Dataset.

PAMAP2 Dataset:

This dataset contains information of 18 different physical activities performed by 9 subjects. Data were collected from Inertial Measurement Units (IMUs) placed in three locations of the body namely hand, chest and ankle. Each IMU contains two 3-axis accelerometers, 1 three-axis gyroscope and a magnetometer. Other sensors used were heartbeat sensor and a temperature sensor for each location of the IMU. Each person had to follow a 12-step protocol that included 12 separate tasks. These recordings are organized by subject in the folder Protocol. In addition, some of the participants engaged in a few optional activities. These recordings are organized by subject in the folder Optional.

In this work, we only considered the Protocol activities. Also, out of the 2 accelerometers, the one with better resolution and range scale was used. Hear rate sensor was included but temperature sensor data were not used. The sampling

rate of the IMU sensors in the dataset is 100Hz. For this dataset a 5 second window was selected to detect activity. A total of 12 activities were considered for detection. This dataset is quite imbalanced and augmentation was used to improve accuracy [38].

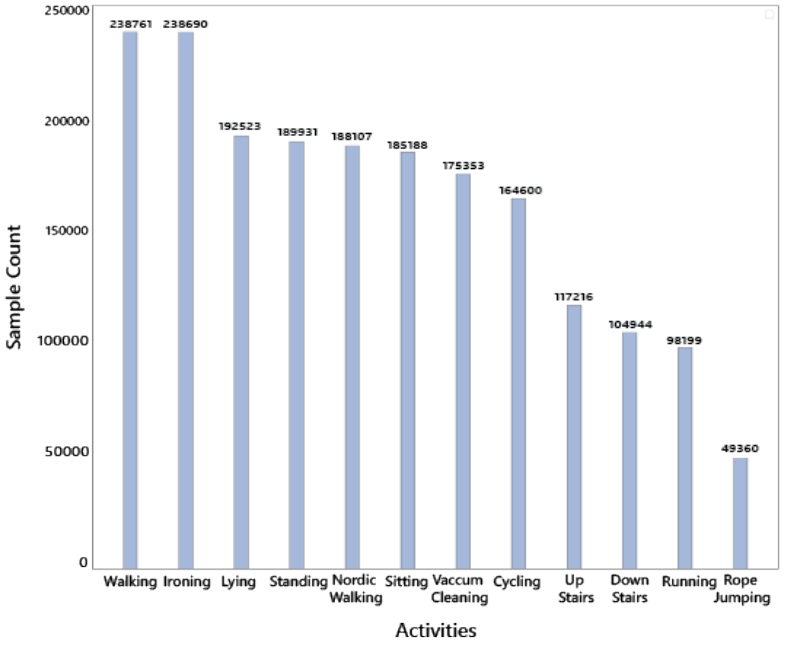


Figure 3.3: Sample distribution of activities in PAMAP2 Dataset.

ARS DLR Dataset:

An observer personally annotated the data set utilized to accomplish our results, which was gathered from 16 male and female individuals aged 23 to 50. It has around 4.5 hours of annotated actions, including Sitting, Standing, Walking, Running, Jumping, Falling, and Lying. Accelerometer and Gyroscope sensors were used to detected the activities. Four activities are very low in this dataset namely lying, running, jumping and falling. Augmentation techniques were applied to balance the dataset [39].

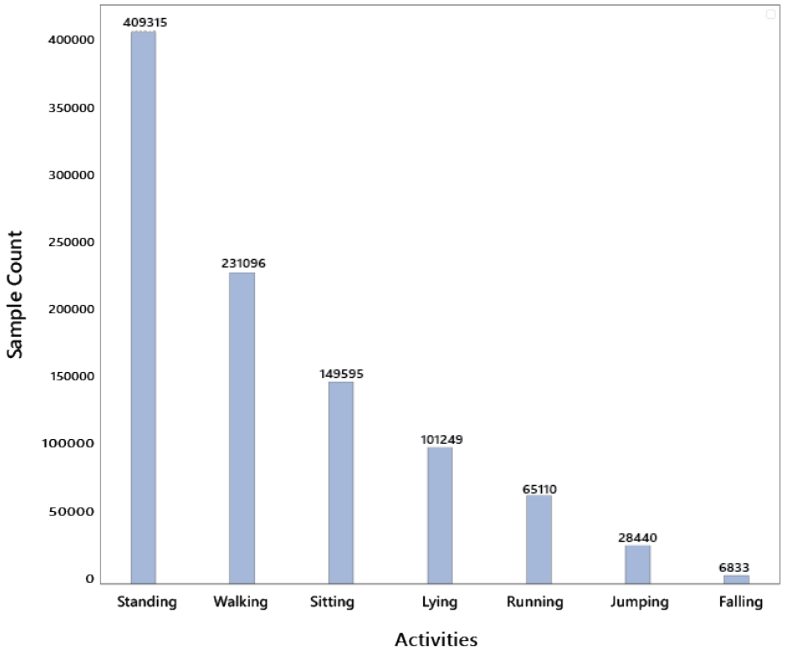


Figure 3.4: Sample distribution of activities in ARS DLR Dataset.

3.3 Classification Models

Human Activity Recognition models can be divided into two categories: predetermined features that are predefined before training and deep learning models that define features while training. In this project, the deep learning method was used. This project's models are primarily a combination of two sorts of models:

- Convolutional Neural Network (CNN)
- Long Short-Term Memory (LSTM)

Convolutional Neural Networks:

Convolutional Neural Networks (CNN) are excellent at detecting important salient features of a signal. The deeper the model's convolution layer, the more local salience of signals that may characterize each signal class is obtained. The signals are represented at a high level by the high-level layers [40]. Let, $x_i^0 = [x_1, x_2, x_3, \ldots, x_N]$ be the accelerometer and gyroscope sensor data input vector where N is the window length. Then the output of the l^{th} layer is given by the equation,

$$c_i^{l,j} = \sigma\left(b_j^l + \sum_{m=1}^{M} w_m^{l,j} x_{i+m-1}^{l-1,j}\right) \tag{3.1}$$

CNN layers frequently employ pooling layers. These layers can extract a summary of the preceding layer's output and represent it with a smaller vector, lowering the complexity of subsequent levels' computations. Average pooling and max-pooling are two types of pooling layers. Considering $C_i^{l,j}$ to be the output of the previous layer of the max-pooling layer, the output of the layer will be given by the equation

$$p_i^{l,j} = \max_{r \in R}\left(c_{i \times T+r}^{l,j}\right) \tag{3.2}$$

where R is the pooling size and T being the pooling stride. To classify activity signals, a softmax layer can be added at the end of a stack of CNN and pooling layers. The output of the softmax layer is given by,

$$P(c \mid p) = \underset{c \in C}{\operatorname{argmax}} \frac{\exp\left(p^{L-1} w^L + b^L\right)}{\sum_{k=1}^{N_c} \exp\left(p^{L-1} w_k\right)} \tag{3.3}$$

where c is the activity class, L is the last layer index, and N_c is the total number of activity classes [2]. In [5] CNN model is used to detect transportation mode from smartphone sensor data. Figure-3.3 shows the proposed model structure in [5].

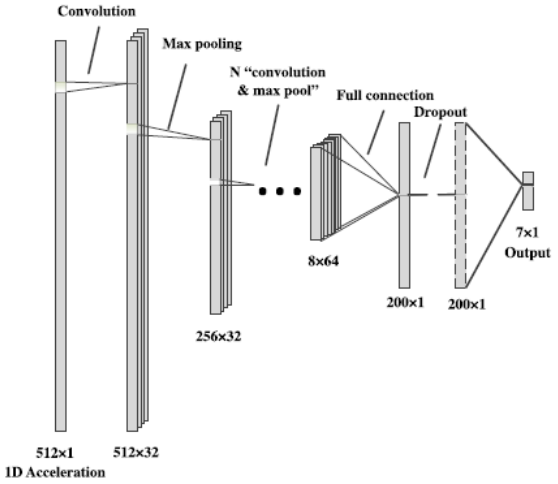


Figure 3.5: CNN model used in [5]

LSTM Models:

CNNs are good at recognizing salient features from signals, but they lose temporal features in the process, as previously stated. However, temporal information may be critical in distinguishing between different types of activities. Recurrent Neural Networks (RNN) can extract temporal properties from time-series data in a sequential manner, such as sensor data from a smartphone. Vanishing and exploding gradient problems are common in general RNN models. By integrating Long Short-Term Memory (LSTM) and RNN, these issues can be avoided [41]. An RNN layer consists of 3 layers namely, input layer, hidden layer, and output layer. Let, the input set be x = [x_0, x_1,x_2, x_3, ..., x_t, x_{t+1}, ...], hidden set be h = [h_0, h_1, h_2, h_3, ..., h_t, h_{t+1}, ...], output layer be y = [y_0, y_1, y_2, y_3, ..., y_t, y_{t+1}, ...] and U,V,W denote weight metrics from the input layer to the hidden layer, from the hidden layer to the hidden layer, and from the hidden layer to the output layer, respectively. Then the outputs of the hidden layer and output layer can be defined as,

$$h_i = \begin{cases} g\left(U_{x_i} + b_i^h\right) & i = 0 \\ g\left(U_{x_i} + W h_{i-1} + b_i^h\right) & i = 1, 2, \ldots \end{cases} \tag{3.4}$$

$$y_i = g\left(V h_i + b_i^y\right) \quad i = 0, 1, \ldots \tag{3.5}$$

where h_i is the output of the hidden layer, y_i is the output of the output layer and g is the activation function [6].

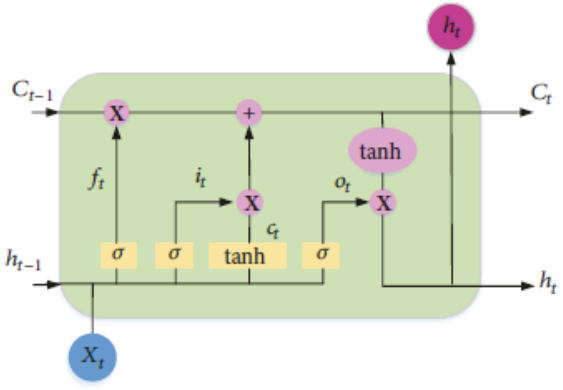


Figure 3.6: Structure of an LSTM cell [6]

The transmission of state information in an RNN or one-directional LSTM layer is one way, from front to back. Bidirectional LSTM is used to solve this problem and increase the availability of information. To facilitate transmission of information in both ways two layers are implemented, forward and backward layer, which have no interaction between them until the final output layer. Output for bidirectional LSTM can be defined by the following equations and the structure is given in the Figure 3.5 [7].

$$\overrightarrow{\text{forward}}_t = f\left(U_f X_t + W_f \overrightarrow{h_{t-1}} + \vec{b}\right) \tag{3.6}$$

$$\overleftarrow{\text{backward}}_t = f\left(U_b X_t + W_b \overleftarrow{h_{t+1}} + \overleftarrow{b}\right) \tag{3.7}$$

$$\text{output}_t = g\left(J\left[\overrightarrow{\text{forward}}_t V_f + \vec{d}; \overleftarrow{\text{backward}}_t V_b + \overleftarrow{d}\right] + c\right) \tag{3.8}$$

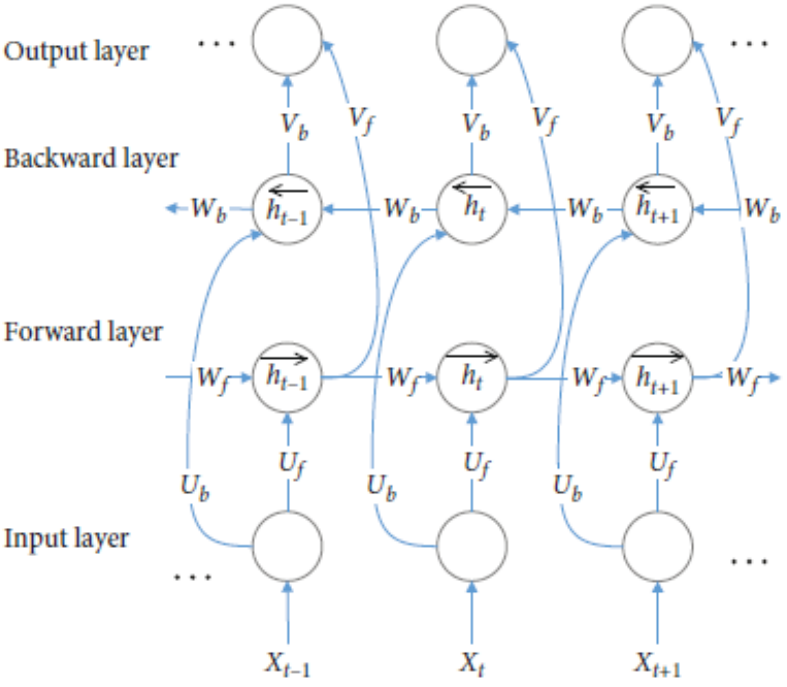


Figure 3.7: Structure of Bidirectional LSTM [7]

LSTM models are used to classify human activities in [6] ,[7] and [42].

CNN and LSTM Hybrid Models:

In the detection of human activities from sensor signals both salient and temporal relations are required to be identified. Although CNN models are great at obtaining salient features from signals, temporal features are ignored. Again, LSTM models are great at identifying temporal features from signals. So, using a hybrid model of CNN and LSTM layers may be better at classifying activity signals.

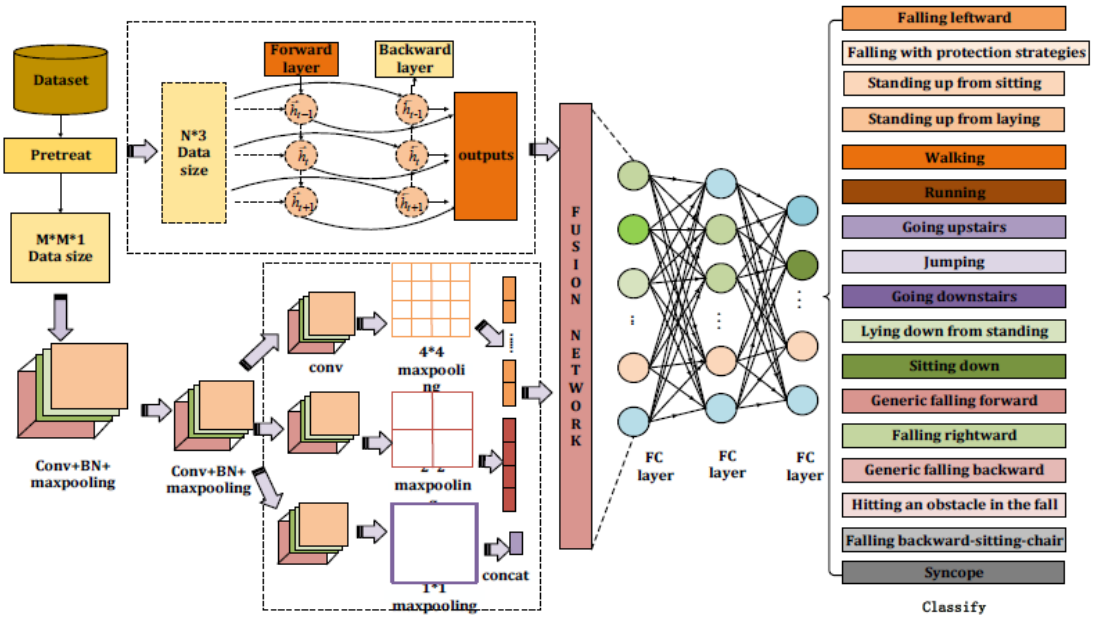


Figure 3.8: Fusion Network model of CNN and LSTM in [8].

Different hybrid models using CNN and LSTM layers can be created. In [8] features detected from CNN and LSTM model are combined in a single layer and classification is done based on that layer.

In [9] and [43] CNN layers were applied to the input signal first followed by the LSTM layer to define the feature layer that is finally used for classification.

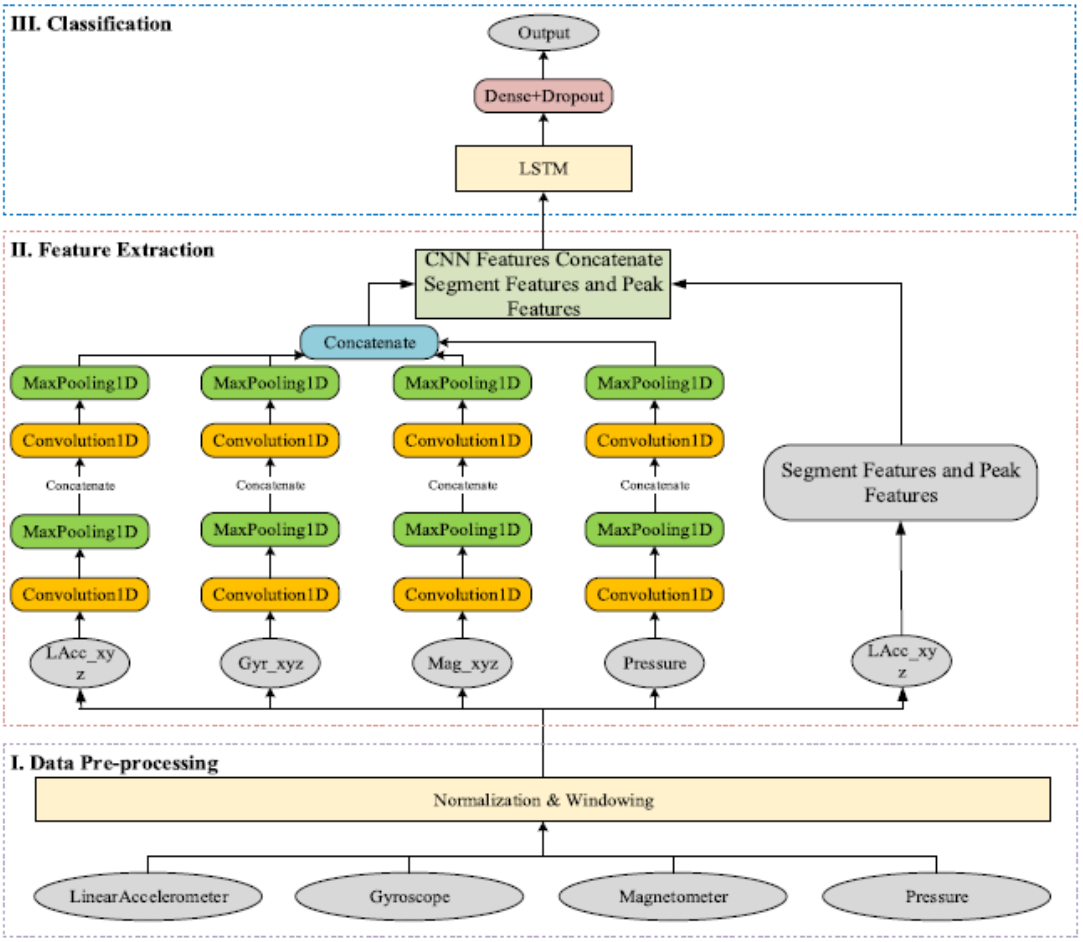


Figure 3.9: A Hybrid model of CNN and LSTM for activity recognition [9].

3.4 Data Augmentation

Deep learning techniques have achieved benchmark results and proved to be excellent in classification in many different fields. But in order to achieve such results, deep learning models require lots of input training data which may not be available for many fields.

In the case of human activity datasets based on sensor data, data scarcity is a problem for some datasets. Again, for some datasets class imbalance is a problem as some activities have more data as the activities are easy to perform [44]. Data Augmentation can solve these problems and improve the performance of deep

learning models, enabling them to generalize better. Terry T. Um et al. [10] demonstrates some data augmentation techniques for time-series sensor data for Parkinson's disease detection using Convolutional Neural Networks.

Some methods used to augment data are namely permutation, time warping, scaling, magnitude warping, cropping, and rotation. Permutation is used to randomly change the temporal location of different incidents within the data window. Time warping slightly distorts time steps between data points and changes the temporal location of events. Jittering introduces random noise in the data. Rotation is used to rotate a signal around an axis to avoid bias towards a certain orientation.

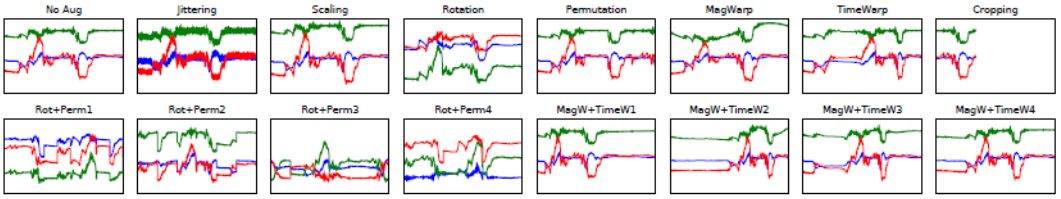


Figure 3.10: Various data augmentation methods and corresponding outputs obtained in [10].

3.5 Global Reference Frame

Sensor data measured by the Inertial Measurement Unit (IMU) of a smartphone is with respect to the reference frame of the IMU or the smartphone. As a result, changing the orientation of the smartphone will lead to different sensor data. But in most sensor datasets, data was collected for a specific orientation hence a specific reference frame of IMU. A model trained on such data will not perform on sensor data from other orientations and will have a bias to the specific orientation on which it was trained. This problem can be solved by converting the data from a specific reference frame to a global reference frame.

In [4] acceleration data from the accelerometer were separated into acceleration towards the gravity and acceleration towards the horizontal direction. Then features were calculated from the separated data.

Henprasertttae et al. [11] presents a transformation matrix to convert acceleration value from local reference frame of IMU to global reference frame. Here, at first, the downward direction is found from averaging values of the dynamic part of the acceleration data. Let, w be the mean of the dynamic portion. The forward axis can be calculated from the projection of data onto the plane normal to w. The projected data on the plane can be found by subtracting the acceleration data along the vertical axis along w from the original data. It can be calculated using the given equation,

$$x'_t = x_t - \left(x_t^T w\right) w \tag{3.9}$$

where x′ is the removed acceleration signals along the vertical axis and x is the original acceleration data. Eigen-decomposition was performed next on the covariance matrix of the projected data using the following equation,

$$C = \frac{1}{T} \sum_{t=1}^{T} \left(x'_t - \mu'\right) \left(x'_t - \mu'\right)^T \tag{3.10}$$

where μ' is the mean of the projected data, calculated as follows

$$\mu' = \frac{1}{T} \sum_{t=1}^{T} x'_t \tag{3.11}$$

Lastly, the sideward axis can be found by considering the cross product between the vertical and the forward axes

$$\mathrm{v} = \mathrm{u} \times \mu \tag{3.12}$$

Then, the transformation matrix T is,

$$T = \begin{bmatrix} u_x & u_y & u_z \\ v_x & v_y & v_z \\ w_x & w_y & w_z \end{bmatrix} \tag{3.13}$$

Figure 3.9 shows transformed signal after reference frame transformation,

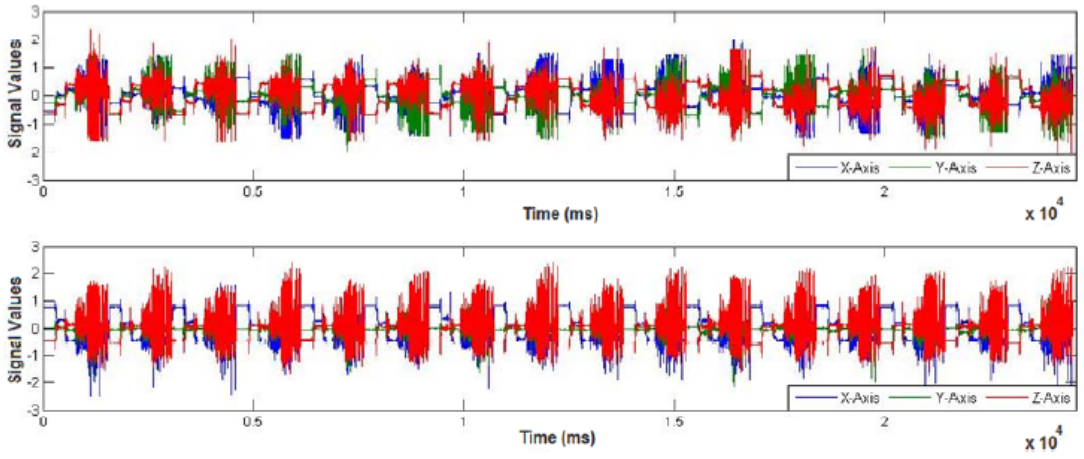


Figure 3.11: Transformation output from local reference frame to global reference frame where signal in the upper graph represents before transforming through the matrix and the lower part represent the transformed signal [11].

CHAPTER 4

METHODOLOGY

Deep learning-based Human Activity Recognition was attempted in this project. Multiple deep learning models were created and trained on various public datasets on HAR which were pre-processed using various techniques. Many datasets are available on smartphone sensor-based Human Activity Recognition (HAR). But some datasets are considered to be standard ones for testing a model. The datasets mentioned in the previous chapter were used in this project. The datasets provide raw sensor data collected from a smartphone from various positions. Some further pre-processing was done on the data before using it to train a deep learning model.

4.1 Data Pre-processing

Data generated by smartphone sensors are not fit to be used directly for activity recognition as they contain various unwanted signals. Also, some pre-processing is required to make the data better for use in deep learning models through various techniques or methods.

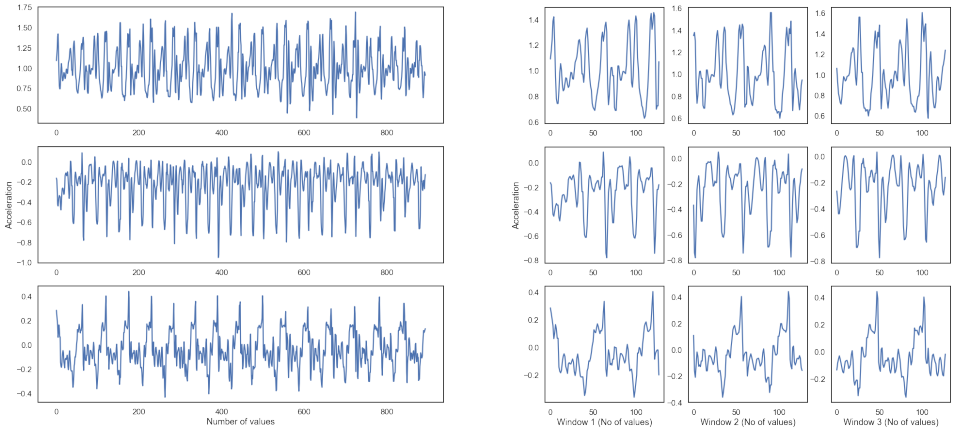
4.1.1 Windowing

In most datasets, continuous sensor data is given for a long period of time. Using data of such long periods is not practical. Using 2-5 seconds of sensor data is

enough to identify simple activities. So first the continuous data are divided into windows of a specified length. This process is called segmentation or windowing. Data is divided into groups that contain similar information about the activity. It is done in such a way that all windows contain enough properties for activity detection [14].

A sliding window can be defined as a sequence of values X = [x_1, x_2, x_3, ..., x_n] where x is the whole data and the nth value of the sequence is signified by x_n. Let X' = [x_p, x_{p+1}, ..., x_{p+w-1}] be a segment where w is the size of the window and p is any position [45].

Windows can also be both overlapping or non-overlapping. In overlapping windows, two adjacent windows have some common values between them whereas in non-overlapping windows two adjacent windows have no common values. In this project windows of 2.56 seconds were used with 50% overlapping between adjacent windows.



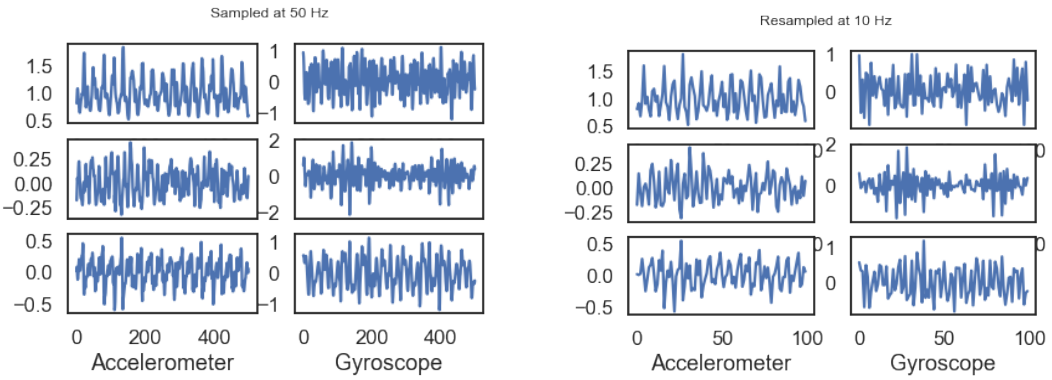
(a) (b)

Figure 4.1: :(a) Signal before passing through the sliding window;(b) Signal in each window for accelerometer

4.1.2 Resampling

Different sensors don't usually generate data at the same rate. So, same number of values for the sensors signify the different amount of time which can cause an error

in windowing. Again, sometimes the sampling rate can be too high or low for use. These problems can be solved by resampling the sensor data using interpolation. Resampling can contain the shape of the signal with a fewer number of values.



(a) (b)

Figure 4.2: (a) Signal Sampled at 50Hz; (b) Signal resampled at 10Hz

4.1.3 Filtering

Sensor data need to be filtered at times for removing undesired signals from the actual signal. Also, some specific desired signals can be extracted from the signal using filtering. Gravity may be required to be separated from the accelerometer data at times to get the body acceleration only. In this case, it can be done easily by passing the original accelerometer data through a Butterworth-lowpass filter with a 0.3Hz cutoff.

4.2 Reference Frame Change

Data received from smartphone sensors are measured with respect to the Inertial Measurement Unit's (IMU) reference frame. So, smartphone orientation has an impact on the data received from the sensors. Converting data from this local reference frame to a global reference frame can solve this problem.Effect of orientation can be removed by taking the horizontal and vertical components of the

accelerometer data. The vertical axis points toward the center of the Earth (like gravity) and the other axis is perpendicular to it [4].

A three-dimensional global reference frame can be established with respect to the direction of gravity. The three axes being the vertical axis, forward axis, and sideward axis. The vertical axis can be determined from the gravity direction. For simplicity, the local reference frame data can be separated into vertical and horizontal components where the horizontal component can be found by subtracting the vertical component from the total data.

4.3 Normalization

Input without normalization may cause training bias and cause problems in model training. Normalization of data results in faster convergence of weights in deep learning models. In this project, mean and variance (standard deviation) normalization on the z-score scale with a standard deviation of 0.5 was used. Such a small standard deviation is often useful in deep learning. Normalized data can be calculated by

$$x^* = \frac{x - \mu}{\sigma} \tag{4.1}$$

where x is data, μ is the mean and σ is the standard deviation [46].

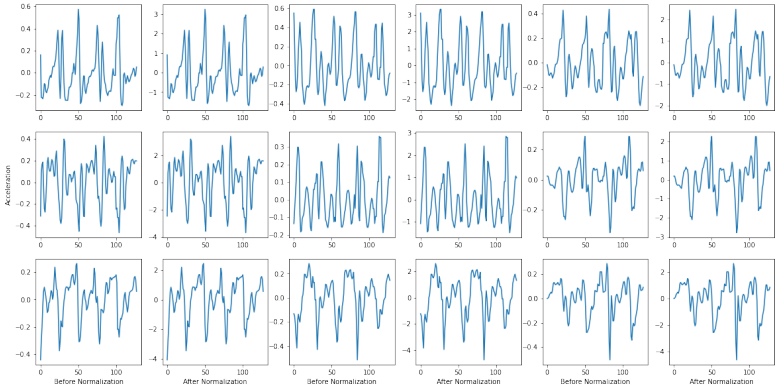


Figure 4.3: Accelerometer data before and after normalization for three axes (X, Y, Z from left to right).

4.4 Augmentation

Deep Learning models usually perform well when trained on a large amount of data. But sensor data are not as readily available as image datasets and they are smaller in size as well.Although Convolutional Neural Networks have shown promise in this field, they are not up to the mark in generalization when trained on small datasets [47]. Data class imbalance is also a hindrance in the case of CNN models. It is when certain classes have much more data than others that creates a bias towards that class.

Synthetic data generation using augmentation techniques can solve both these problems. Small alterations in image datasets are acceptable as they are likely to occur in real life. But such alterations are difficult to create intuitively for time series sensor data [10].

Augmentation was done in such a way that the class with the lowest sample count has at least half the number of samples of the class with the highest sample count. The augmentation techniques used in the project are permutation, rotation, and jittering/scaling.
Rotation: The sensor orientation bias can be removed by randomly rotating the signal data along a random axis. The most common technique used to rotate data is based on quaternions.

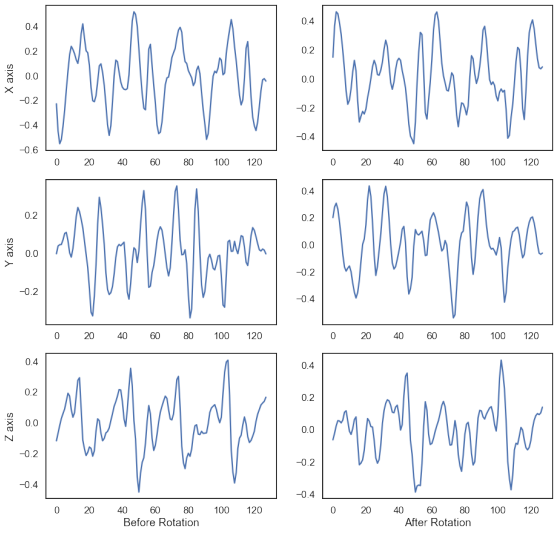


Figure 4.4: Accelerometer data along three axes before and after rotation technique.

Permutation: A window can be divided into a number of sub-sections and randomly shuffled to change temporal locations of certain occurrences within the window.

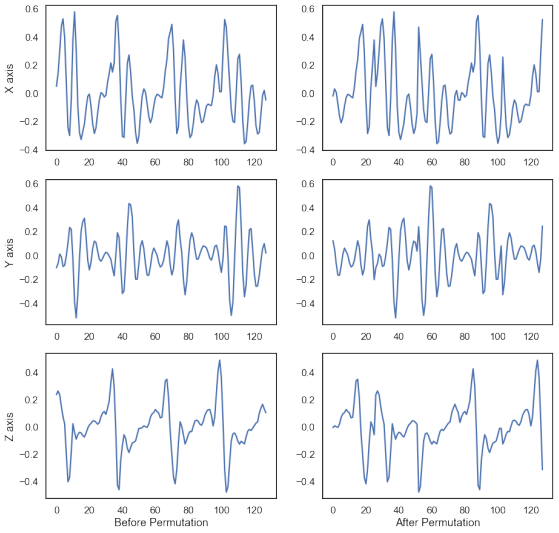


Figure 4.5: Accelerometer data along three axes before and after permutation technique.

Jittering: Some random noise can be added to a window to change the temporal properties of the window which helps the model to generalize better.

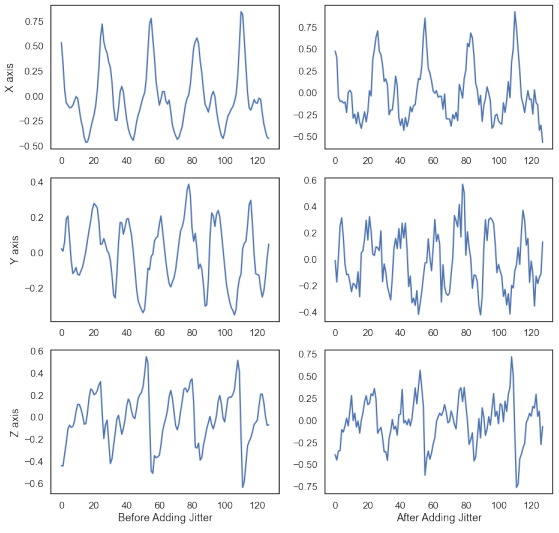


Figure 4.6: Accelerometer data along three axes before and after jittering technique.

Lastly, all three are combined to create the final augmented window.

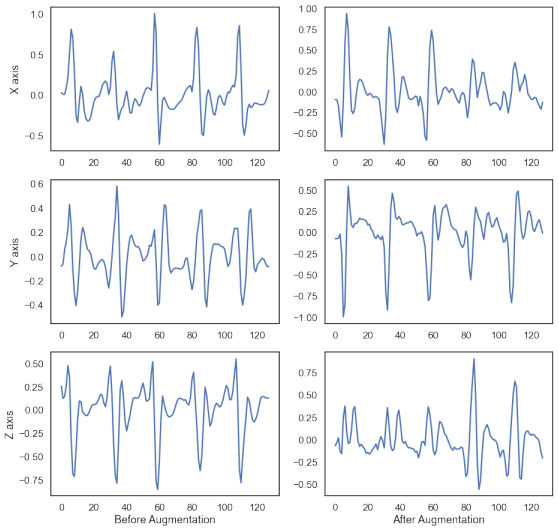


Figure 4.7: Accelerometer data along three axes before and after applying all the three augmentation techniques combined.

4.5 Application of Augmentation in Datasets:

It was found in our previous work that augmentation doesn't increase accuracy in UCI HAPT Dataset as the major activities are quite balanced but the transition activities are very low in amount and cannot generate proper number of augmented data to balance the dataset. So, augmentation was applied on the other two dataset namely PAMAP2 and ARS DLR Dataset.

PAMAP2 Dataset:

As shown before, 4 activities running, jumping, ascending stairs and descending stairs had smaller sample count in this dataset. So, augmentation was used to increase the training data.

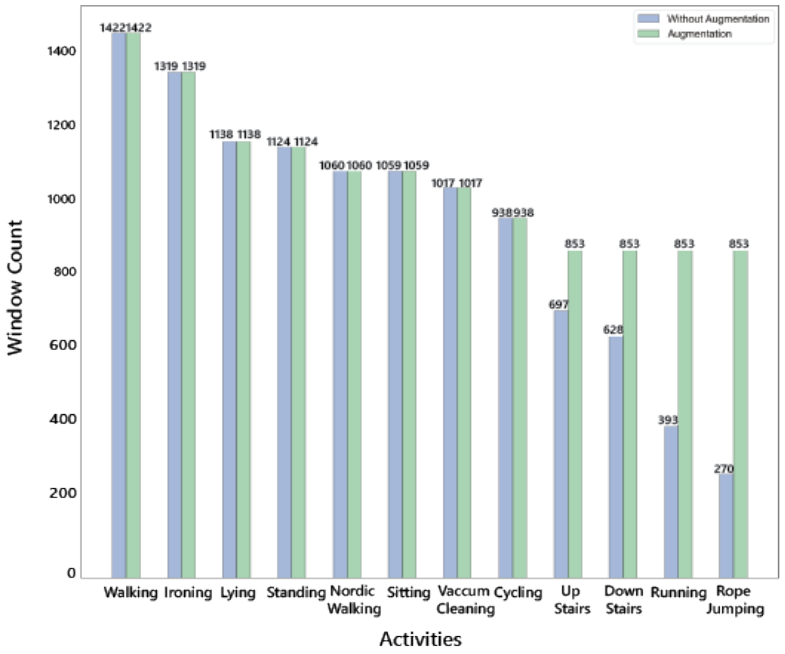


Figure 4.8: Distribution of training data of PAMAP2 Dataset before and after Augmentation.

ARS DLR Dataset:

The dataset is very imbalanced with sample count of running, jumping, falling and lying being very low. Augmentation was applied on data of all the activities to balance the dataset. Standing still had a lot more window count but the other activities were fairly balanced.

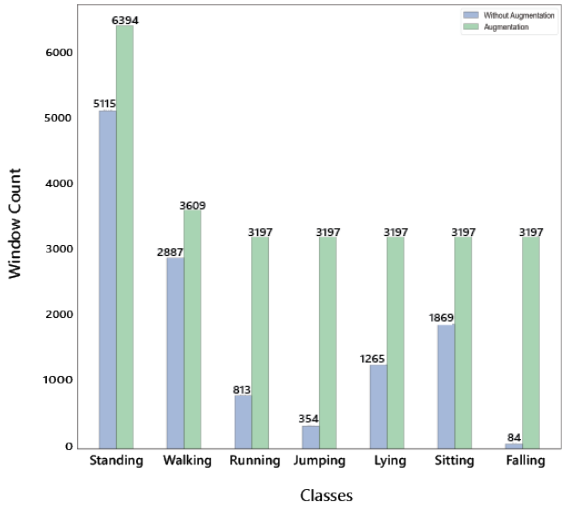


Figure 4.9: Distribution of training data of ARS DLR Dataset before and after Augmentation.

4.6 Training and Testing Data

Dataset needs to be split into training and test dataset to be used for training and testing purposes consecutively. The training dataset is also further divided into training and validation datasets. Train-Test split can be done based on a lot of things.

UCI: In the UCI dataset, data from the first 21 subjects were used for training, data from 3 subjects were used for validation and the rest was used for testing.

PAMAP2: Out of 9 subjects, 8 were taken as subject109 had insufficient data. Subject 1,2,5,6 and 8 had data for all the activities. Data from subject 1,3,4,5,7 and 8 were used for training and validation by performing a train-test split with 7:3 ratio. Data from Subject 2 and 6 were used for testing purpose. Augmentation was then performed on the training set to balance the dataset.

ARS DLR Dataset: Data from all the users were divided into 8:2 ratio to form training and testing datasets. The training set was further split in 7:3 ratio to generate training and validation datasets. Augmentation was then performed on the training set to balance the dataset.

For CNN, the shape of input data used was (batch size, time step, channels) and in case of LSTM layer input shape is (time step, no of features). Here, batch size depends on the total sample number in the dataset, time step depends on window size and no of channels is total number of sensor channels. The array shape of data is (sample no,128,9) for UCI Dataset, (smaple no, 500, 28) for PAMAP2 dataset and (sample no, 128, 6) for ARS DLR Dataset. The 9 axes for UCI dataset being, body acceleration without gravity (3), gyroscope (3), and acceleration with gravity (3). For PAMAP2 dataset, 9 channels for each IMU and heartrate sensor channel with a total of 28 channels. Lastly for ARS DLR dataset, accelerometer data (3) and gyroscope (3) make a total of 6 channels.

Table 4.1: Distribution of train, test and validation data in the datasets.

Dataset	Training		Validation		Test	
	Sample No	Percentage	Sample No	Percentage	Sample No	Percentage
UCI	8618	70.95	1356	11.16	2172	17.88
PAMAP2	7746	51.18	3319	21.89	4091	26.99
ARS DLR	11285	62.54	4571	26.8	4585	10.64

4.7 Models

Our previous work showed, combination of LSTM and CNN layers worked better if LSTM layer was used before CNN layers. For this project, two more models were created for comparison with the LSTM_CNN model found in the work [15].

4.7.1 Model 1

Adding LSTM layers before CNN layers has been shown to achieve good accuracy as well as better generalization. Also, having fewer layers in the model reduces complexity and overfitting. The model from [15] was chosen as the first model. This model will be referred to as the "LSTM_CNN" model.

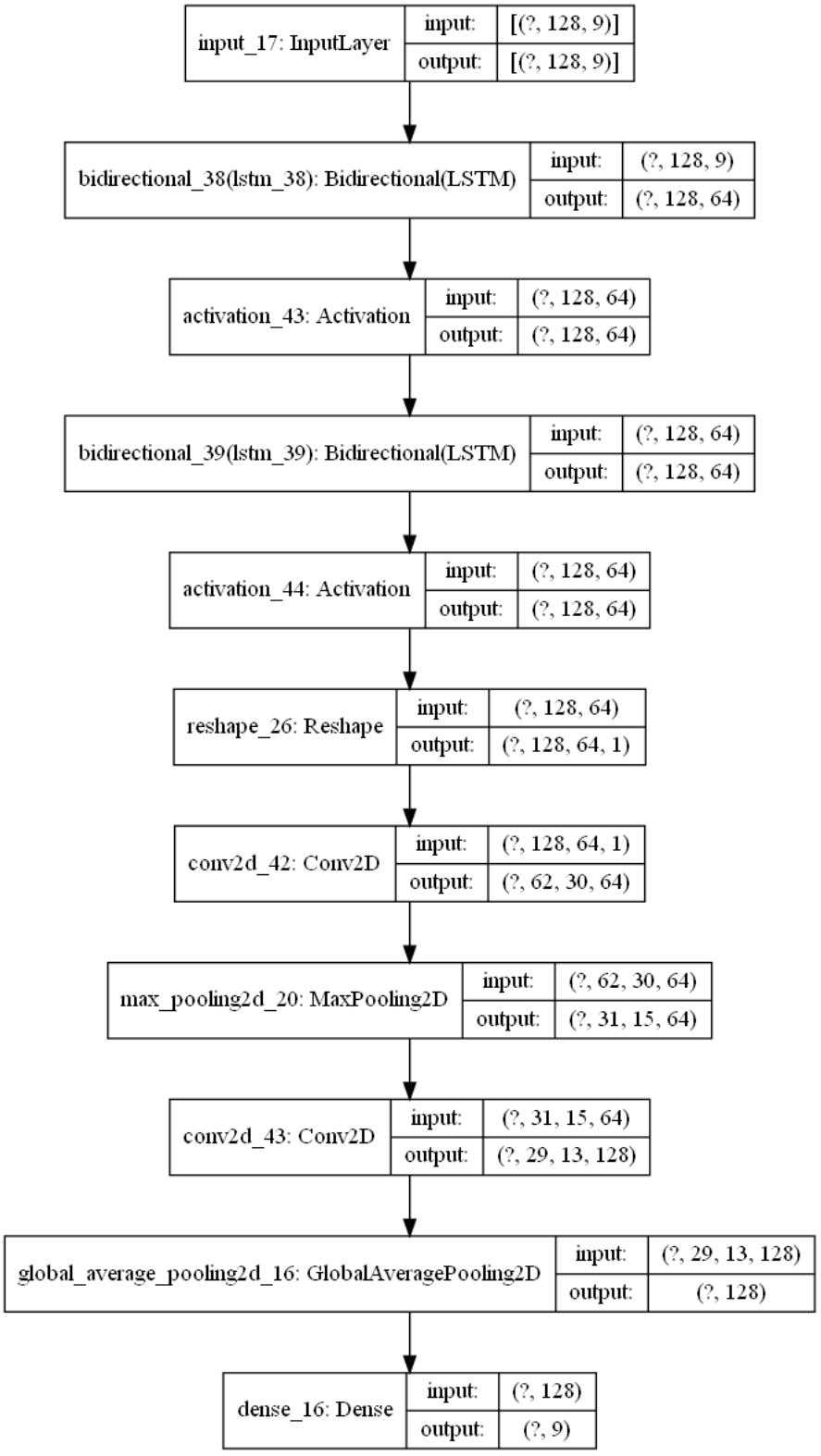


Figure 4.10: Structure of LSTM_CNN Model.

4.7.2 Model 2

In order to investigate using the CNN and LSTM layers parallelly, this model was designed where the input is applied to two models each made of LSTM layers and CNN layers consecutively. This model will be referred to as "LSTM_CNN_PARALLEL".

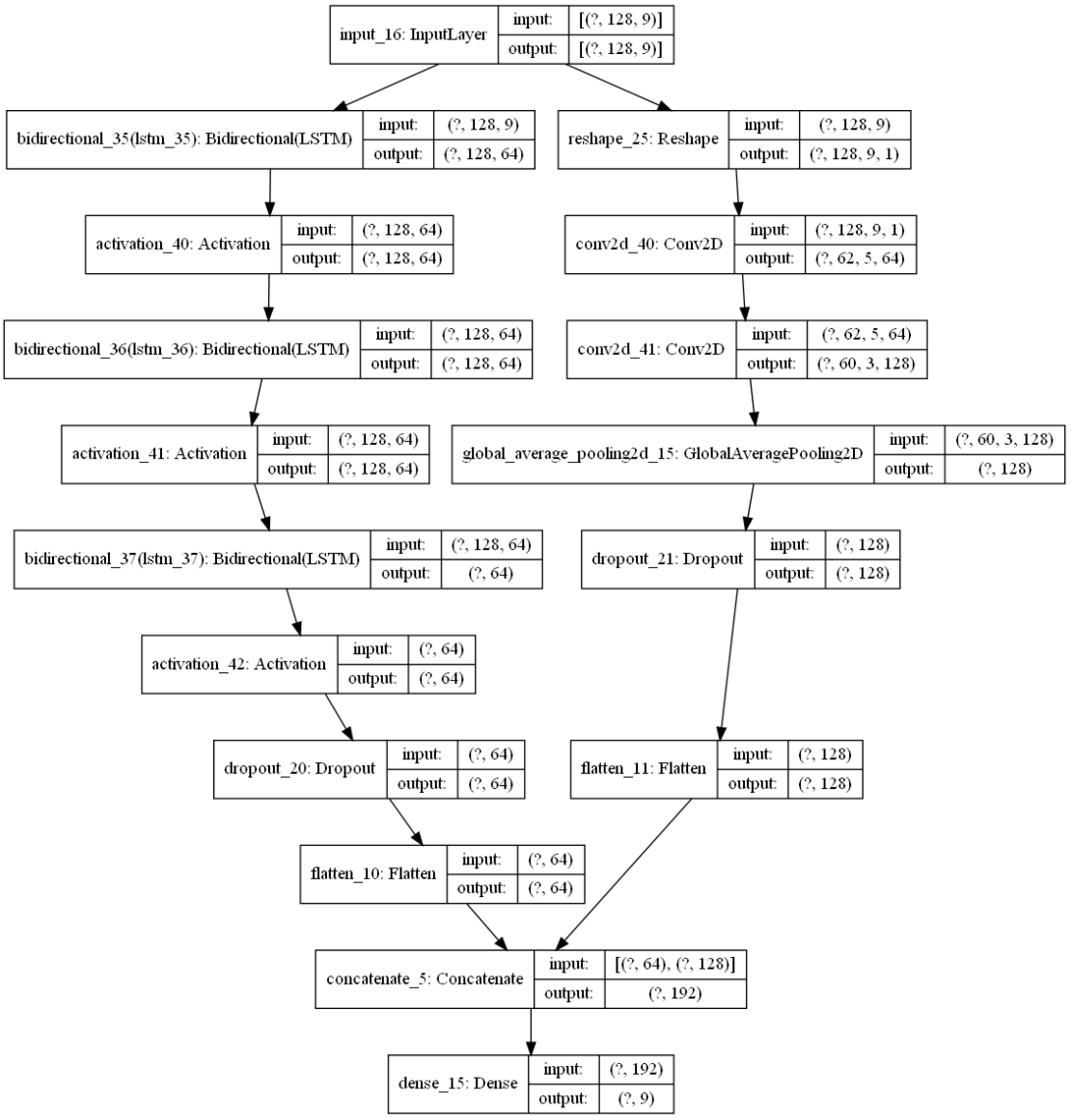


Figure 4.11: Structure of LSTM_CNN_PARALLEL model.

4.7.3 Model 3

In this model, at first the input is passed through an LSTM and CNN layer block. The calculated features then pass through another LSTM-CNN block to finally produce the outputs. This model is referred to as "LSTM_CNN_SANDWICH".

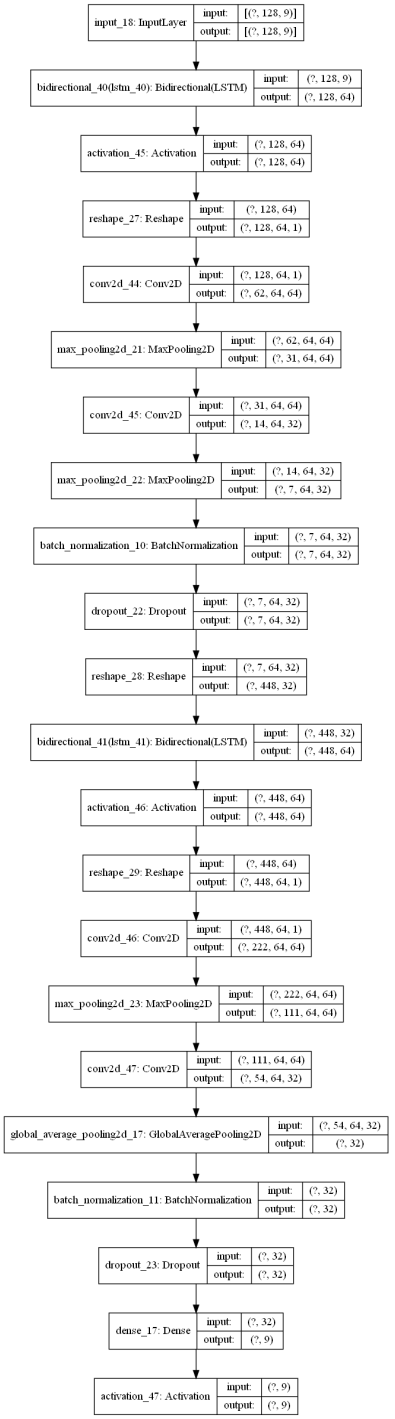


Figure 4.12: Structure of LSTM_CNN_SANDWICH model

4.8 Model Training

Following model hyper-parameters were chosen for training all the models:
Optimizer: RMSprop,Learning rate: 0.001,
Epoch: 100, Earlystopping: Enabled with patience = 30,
Batch Size: 32

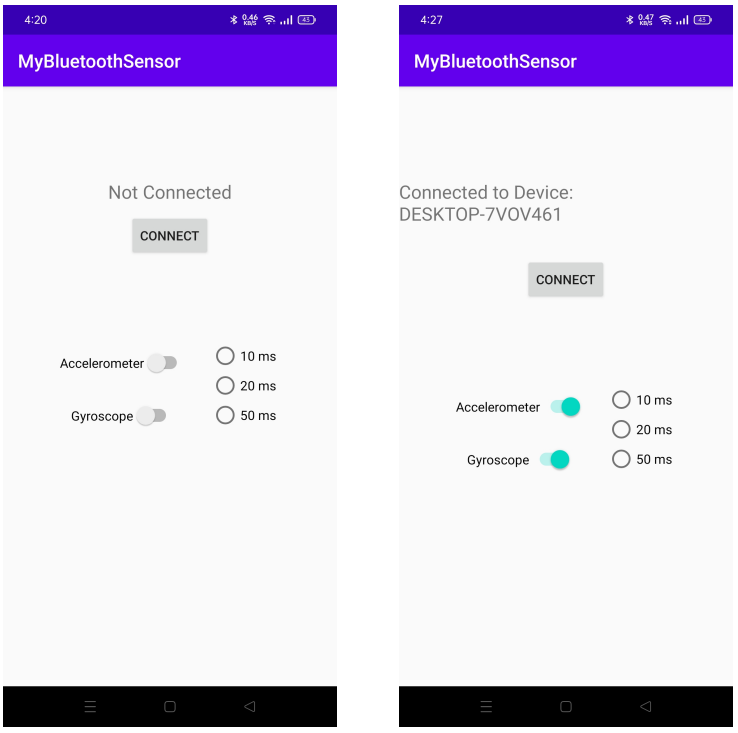
4.9 Realtime Prediction:

A system was developed to predict activity from smartphone sensor data in real time. The system was built using python and consists of 3 major parts:

1. Sensor data streaming
2. Data processing and prediction
3. Graphical User Interface (GUI)

4.9.1 Sensor data streaming:

In order to generate predictions, smartphone sensor data must be transferred to a computer. Data connection between a PC and a smartphone may be accomplished in a variety of ways. Bluetooth was chosen as the means of communication for this project. To send sensor data through Bluetooth, an app was created in Android Studio. On the PC, a python application was created to continually accept input and keep a temporal frame of a given duration using the pybluez module. This data is stored in a data buffer in the program.



(a) (b)

Figure 4.13: (a) App UI when disconnected (b) App UI when connected

4.9.2 Data processing and prediction

Accelerometer and gyroscope data received from the smartphone are processed with the following steps:

1. The program checks if it has enough data to make prediction.
2. If yes, it takes the required amount of data from the data buffer.
3. The sensor data are resampled to match the training dataset sampling rate.
4. The sensor data are stacked together and windowed.
5. The windows are normalized with the dataset mean and standard deviation.

Prediction on the data is done by the following steps:

1. Model settings are loaded at the beginning of the program. Multiple models can be used if necessary.
2. Predictions are made by the models on the windows.
3. Voting is done on the results and the most predicted activity is selected.
4. Confidence score is calculated by averaging the confidence scores of all the predictions of that activity.

4.9.3 Graphical User Interface (GUI)

A Graphical User Interface (GUI) was built using python library PyQt5. This library is the python version of a C++ library called Qt that has tools to make UIs for software. The GUI has 2 windows, 1 for data collection and recording and the other one for real time prediction of streamed sensor data.

Data Collection Window:

This window is used for collecting our own data. Clicking the start button creates two files named acc.csv and gyro.csv and saves the streamed data in the files.

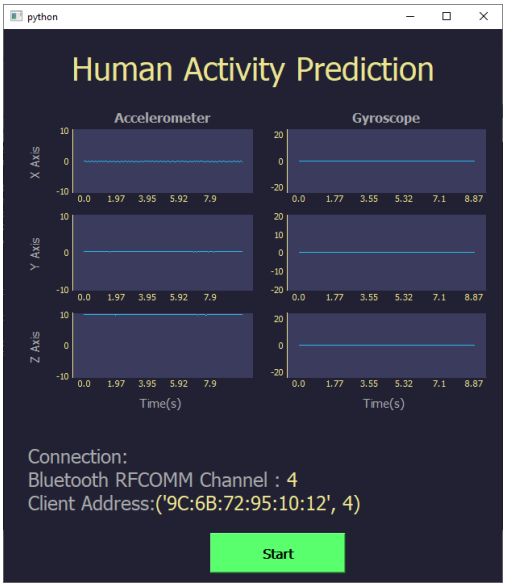
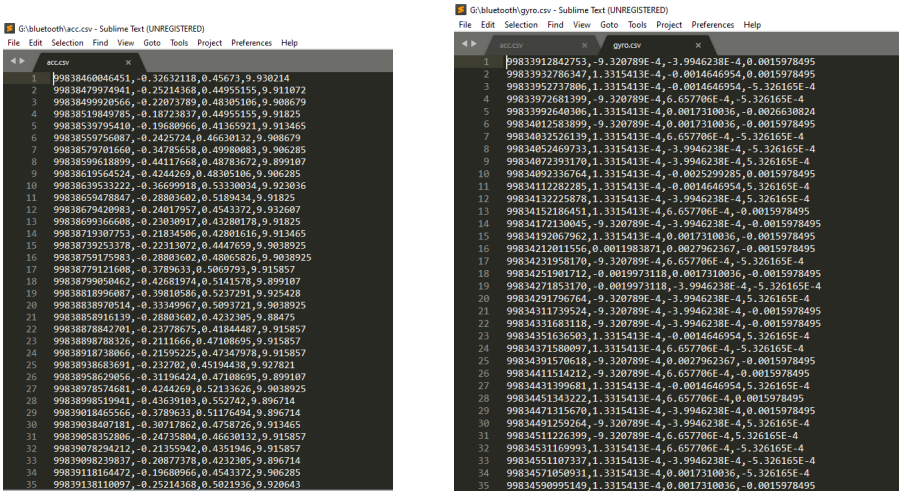


Figure 4.14: Graphical User Interface for data recording to files.



(a) (b)

Figure 4.15: (a) Sample of accelerometer data files of collected sensor data. (b) Sample of gyroscope data files of collected sensor data.

Data Prediction Window:

This UI window is used for showing the real time predictions made on the received data. It has several components:

1. Graph Widget: This widget is responsible for showing graphs of the received signal data from the sensors.

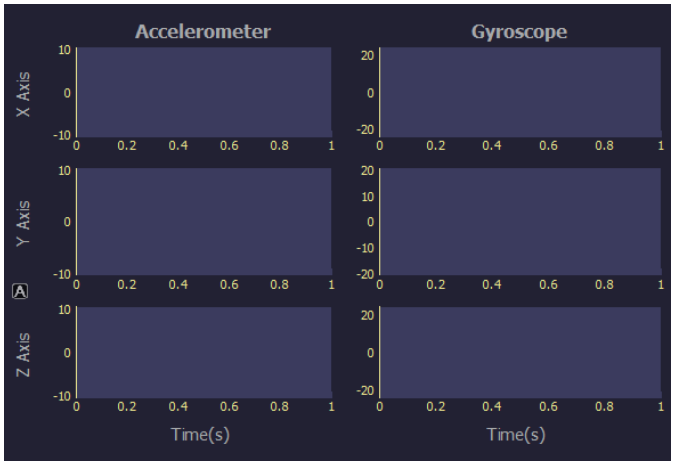


Figure 4.16: PyQt5 widget to show graph of signals.

2. Confidence Chart Widget: This shows the confidence score of the activities for a certain prediction

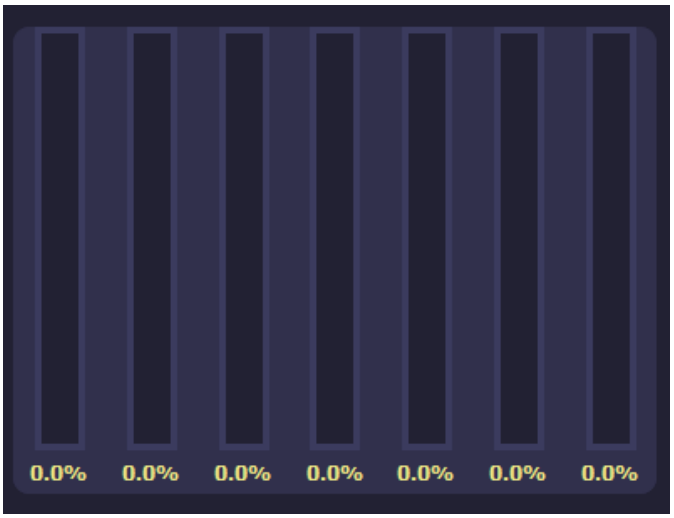


Figure 4.17: PyQt5 widget to show confidence of prediction.

3. Prediction Widget: This widget shows the predicted activity.

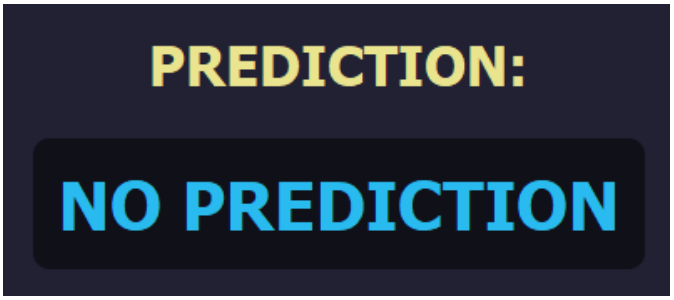
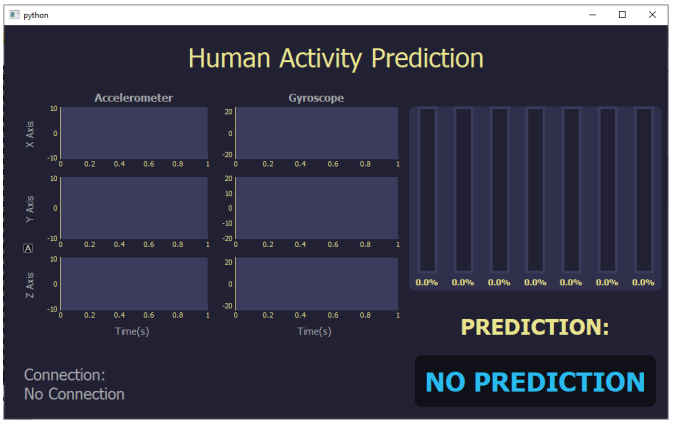


Figure 4.18: PyQt5 widget to show predicted activity.

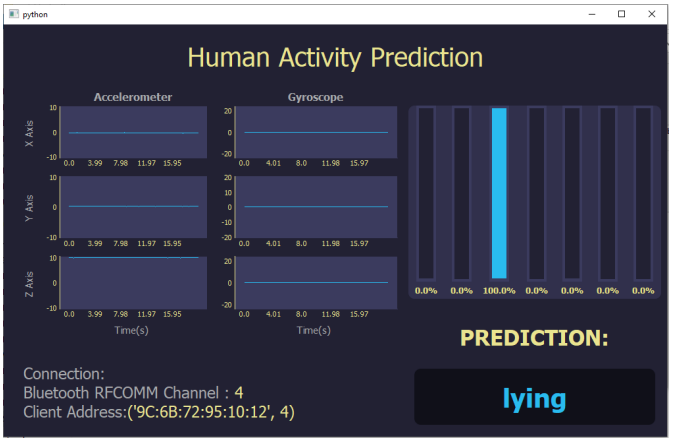
4. Connection Widget: This shows the connection status between the smartphone and the computer.

Figure 4.19: PyQt5 widget to show connection status.

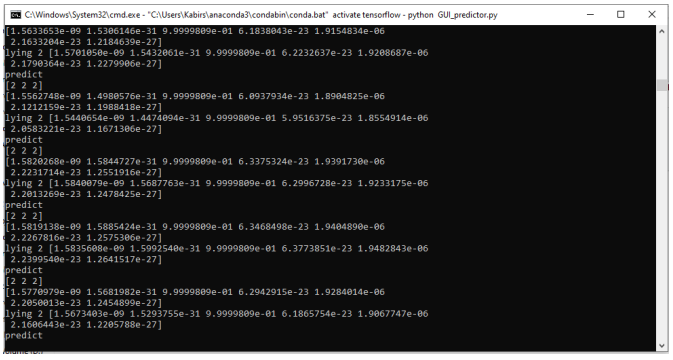
the entire GUI is shown below:



(a)



(b)



(c)

Figure 4.20: (a) GUI when disconnected (b) GUI when connected to smartphone (c) Console output of the python program in command prompt.

4.9.4 Model training for Realtime

Out of the 3 datasets used in the project, PAMAP2 has the most activities but it has some complicated activities and uses much more sensor channels than the others, which makes it unfavorable for practical application in this case. So, ARS DLR dataset was used for training models for real time detection. All 3 models were trained on this dataset and performance was compared.

CHAPTER 5

RESULT AND ANALYSIS

In this chapter, results from the trained models has been shown and analyzed to figure out the impact of the processing techniques and efficiency of the models. Results from the 3 trained models have been compared to determine which one is superior in terms of accuracy and efficiency.

5.1 Evaluation Metrics

In order to analyze and compare performance of models, some factors or criteria need to be established. Any prediction made by a model will fall within any of the 4 types below:
True Positive (TP): When a model predicts a class to which the input actually belongs.
True Negative (TN): When a model predicts the input doesn't belong to a class to which it actually doesn't belong.
False Positive (FP): When a model predicts a class to which the input doesn't belong.
False Negative (FN): When a model predicts input doesn't belong to a class to which it actually belongs.

Based on the number of occurrences of these outcomes, some parameters or metrics are defined which can provide information about a model's performance.

Accuracy: This is the most common metric used to measure model performance. Number of correct predictions made per prediction is accuracy. Accuracy can be defined by,

$$\text{Accuracy } = \frac{TP + TN}{TP + FP + TN + FN} \tag{5.1}$$

Accuracy is not always an accurate measure of model performance, especially in case of imbalanced class.

Precision: Percentage of positive instances out of the total predicted positive instances.

$$\text{Precision } = \frac{TP}{TP + FP} \tag{5.2}$$

Recall/Sensitivity: This is the ratio of true positives over total number of positive cases. This term shows how accurately a class is predicted by a model.

$$\text{Recall } = \frac{TP}{TP + FN} \tag{5.3}$$

Specificity: This is the ratio of true negatives over total number of negative cases.

$$\text{Specificity } = \frac{TN}{TN + FP} \tag{5.4}$$

F1 score: This is the harmonic mean of precision and recall. Higher the F1 score, the better the model performs.

$$F1 = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}} = \frac{2 * \text{ Precision } * \text{ Recall}}{\text{Precision } + \text{ Recall}} \tag{5.5}$$

Confusion Matrix: It is a matrix used to visualize comparison between predicted class and actual class.

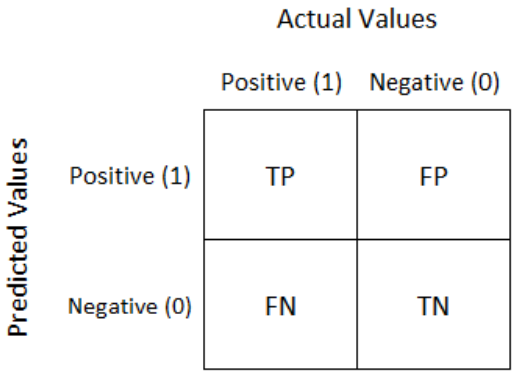


Figure 5.1: Various data augmentation methods and corresponding outputs obtained in [10].

5.2 Bias Variance Tradeoff:

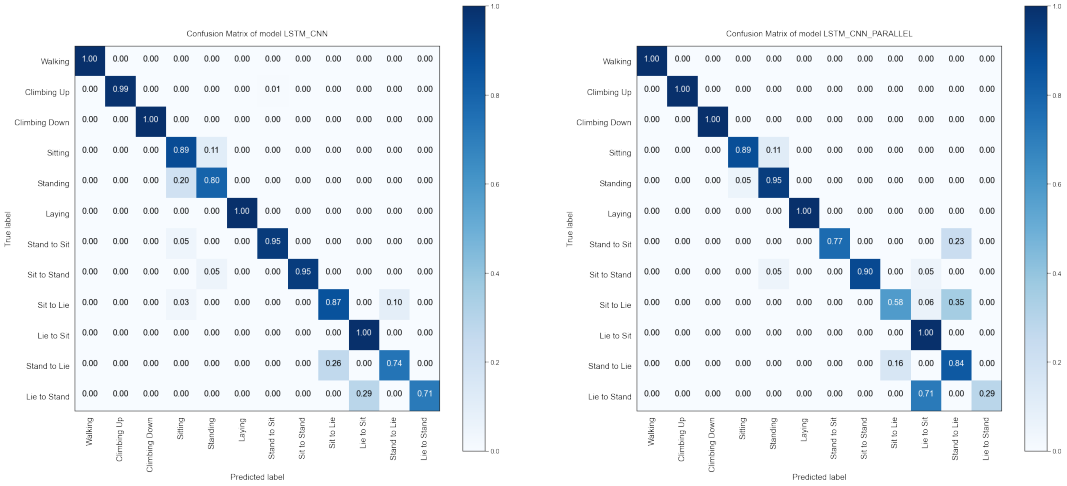
Bias is the difference between actual value and predicted value. High bias signifies that the model pays less attention to training data diversities and oversimplifies predictions. Variance is how much the estimate of model would change for different data. High variance suggests that model is not generalizing well. The ideal outcome from a model is to achieve both low bias and low variance.

Underfitting: A model is underfitted when it tends to oversimply and fails to recognize patterns in data. Usually, high bias and low variance are indications of underfitted model.
Overfitting: An overfitted model learns noise of the training data and considers it during predictions. The model then fails to predict correctly on new data as the noise is random and different for different data. Low bias and high variance are indications of overfitted model.
Relation between bias and variance in models is that decreasing one results in increase of the other. So, low bias results in high variance and vice versa. This also means there is an optimal point where bias and variance are low enough that a model is not underfitted or overfitted.Deep learning models have a tendency to overfit especially when trained on smaller datasets. Overfitting problem can be solved to some extent by reducing parameters of the model, adding dropout layers, increasing size of dataset and cross-validation.

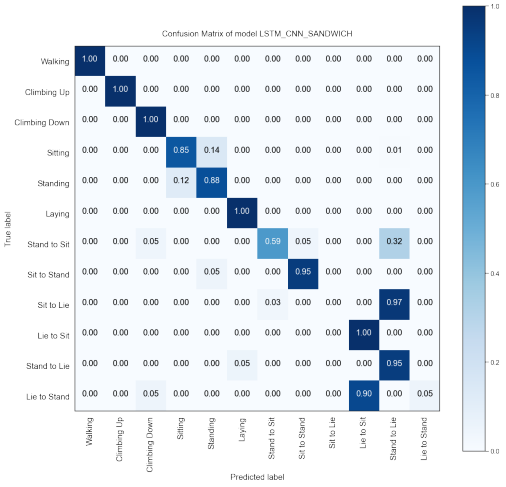
5.3 Performance Analysis

5.3.1 UCI HAPT Dataset

5.3.1.1 Confusion Matrices



(a) (b)

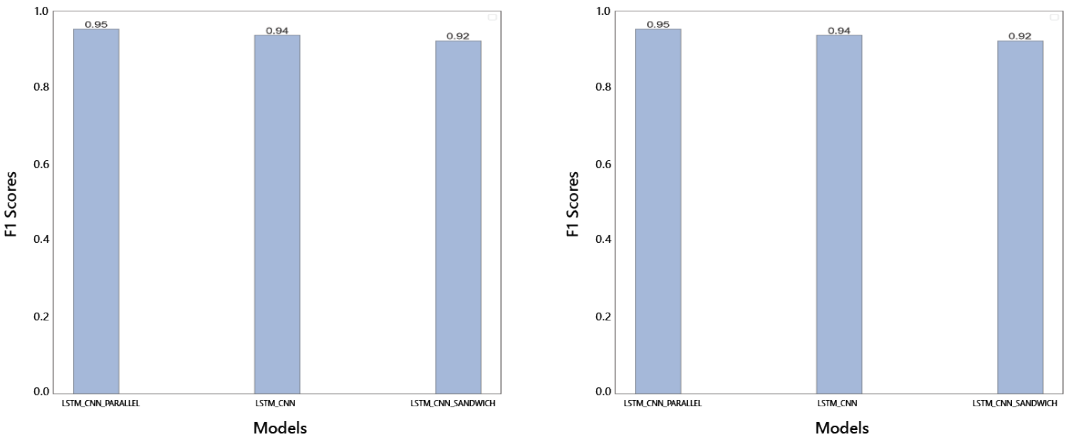


(c)

Figure 5.2: (a) Confusion matrix of model LSTM_CNN on UCI Dataset. (b) Confusion matrix of model LSTM_CNN_PARALLEL on UCI Dataset. (c) Confusion matrix of model LSTM_CNN_SANDWICH on UCI Dataset.

It can be seen the models have some problem in detecting transition activities properly as those activities were very low on count. The major activities were detected properly by all models.

5.3.1.2 Accuracy and F1 Scores



(a) (b)

Figure 5.3: (a) Accuracy of models on UCI Dataset. (b) F1 Score of models on UCI Dataset.

Our model LSTM_CNN_PARALLEL performed better than the other two models on both accuracy and F1 score.

5.3.1.3 Training Curves

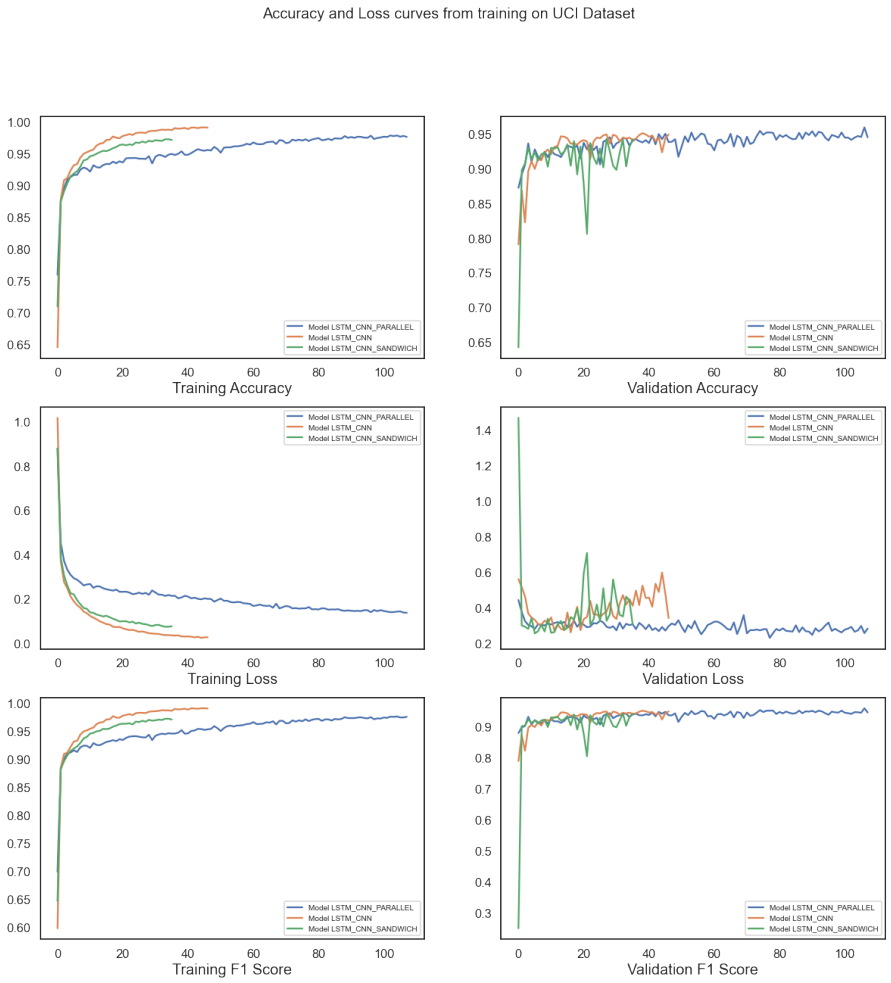
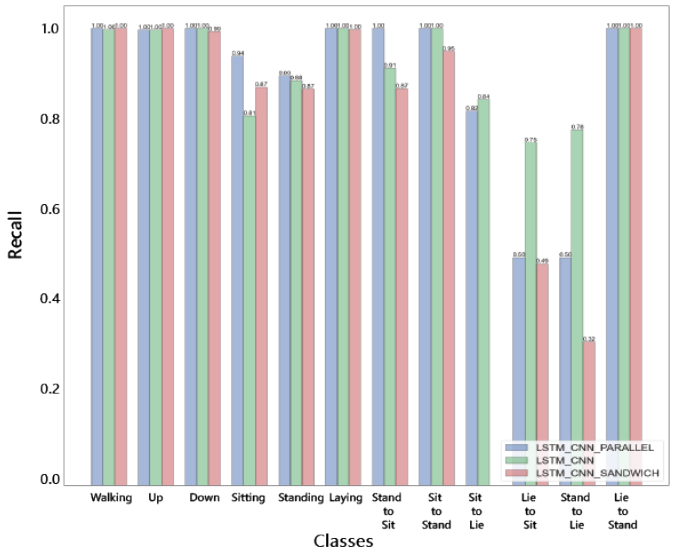
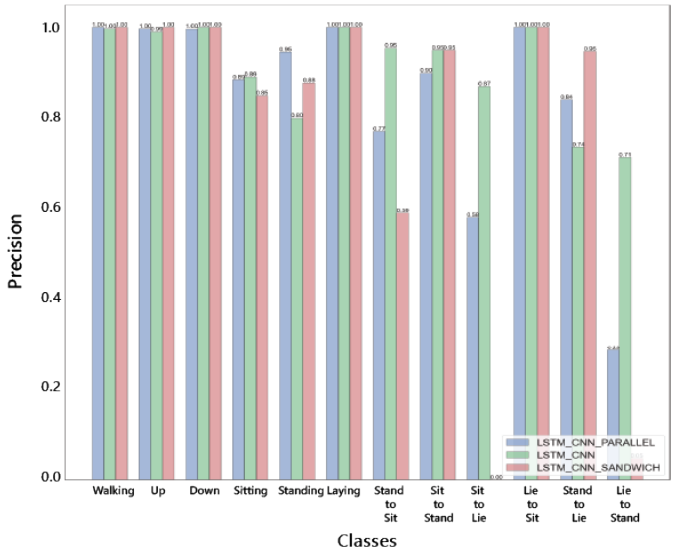


Figure 5.4: Training curves of models on UCI Dataset.

5.3.1.4 Recall and Precision



(a)

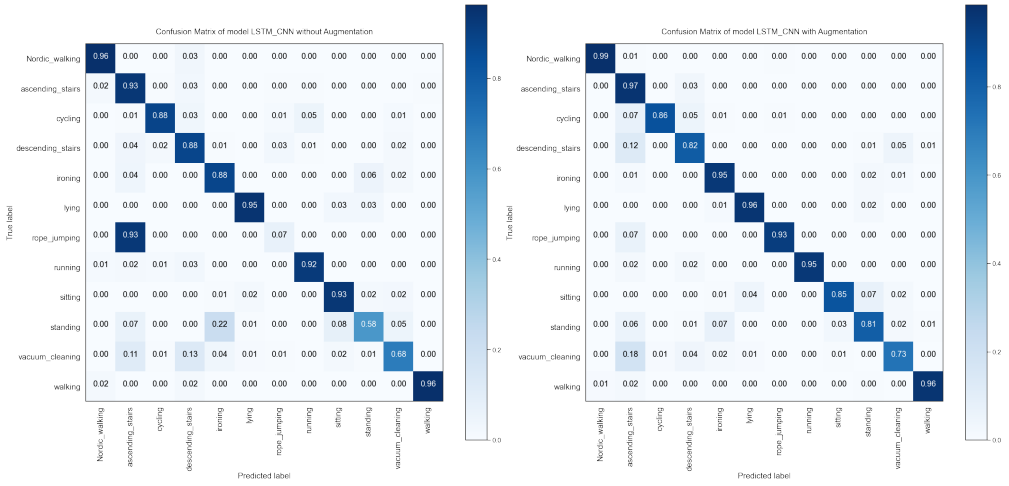


(b)

Figure 5.5: (a) Recall of models on UCI Dataset. (b) Precision of models on UCI Dataset.

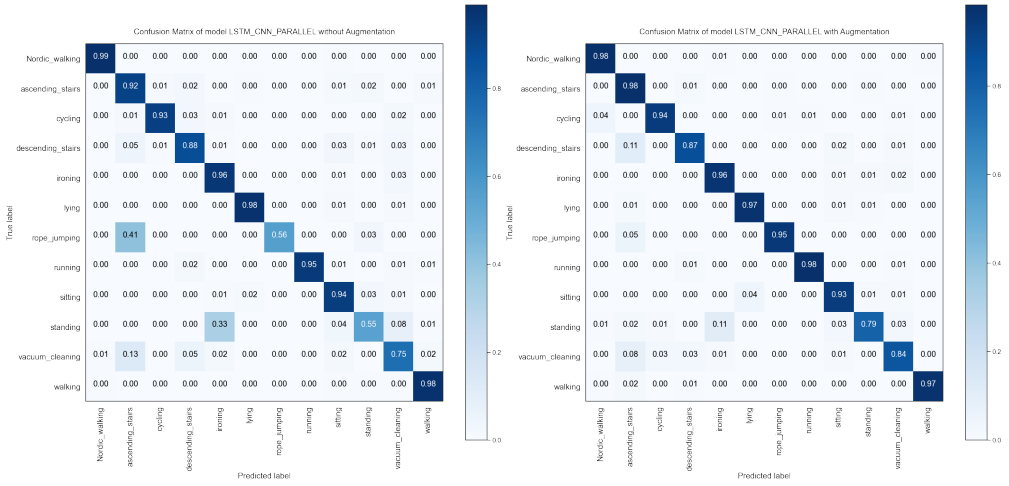
5.3.2 PAMAP2 Dataset

5.3.2.1 Confusion Matrices



(a) (b)

Figure 5.6: (a) Confusion matrix of model LSTM_CNN on PAMAP2 Dataset without Augmentation. (b) Confusion matrix of model LSTM_CNN on PAMAP2 Dataset with Augmentation.



(a) (b)

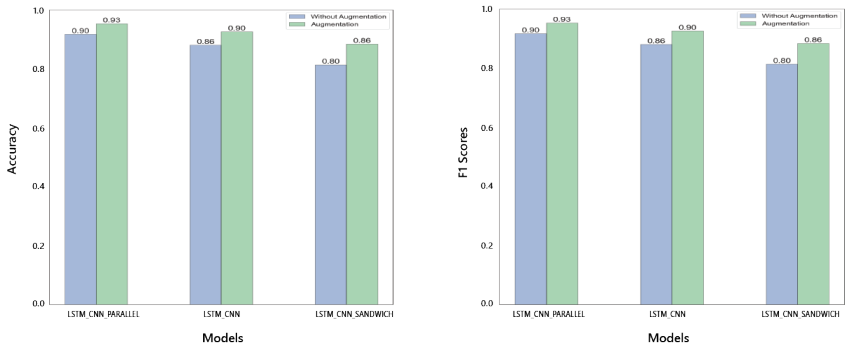
Figure 5.7: (a) Confusion matrix of model LSTM_CNN_PARALLEL on PAMAP2 Dataset without Augmentation. (b) Confusion matrix of model LSTM_CNN_PARALLEL on PAMAP2 Dataset with Augmentation.

(a) (b)

Figure 5.8: (a) Confusion matrix of model LSTM_CNN_SANDWICH on PAMAP2 Dataset without Augmentation. (b) Confusion matrix of model LSTM_CNN_SANDWICH on PAMAP2 Dataset with Augmentation.

It can be clearly observed from the confusion matrices that confusion among activities were reduced by a considerable amount after application of the augmentation techniques, especially on activities with low sample count such as rope jumping.

5.3.2.2 Accuracy and F1 Scores



(a) (b)

Figure 5.9: (a) Accuracy of models on UCI Dataset. (b) F1 Score of models on UCI Dataset.

Accuracy and F1 scores improved significantly in all cases after application of augmentation techniques. Furthermore, our model LSTM_CNN_PARALLEL performed better than the other two models in all cases.

5.3.2.3 Training Curves

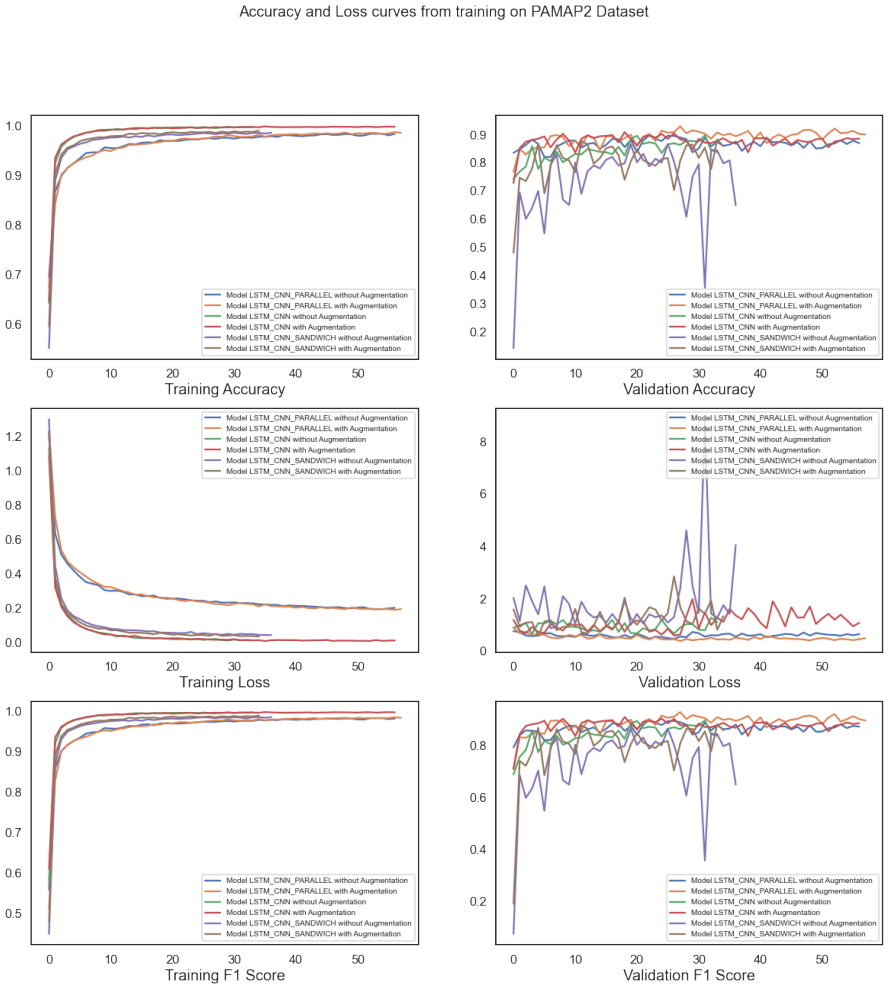
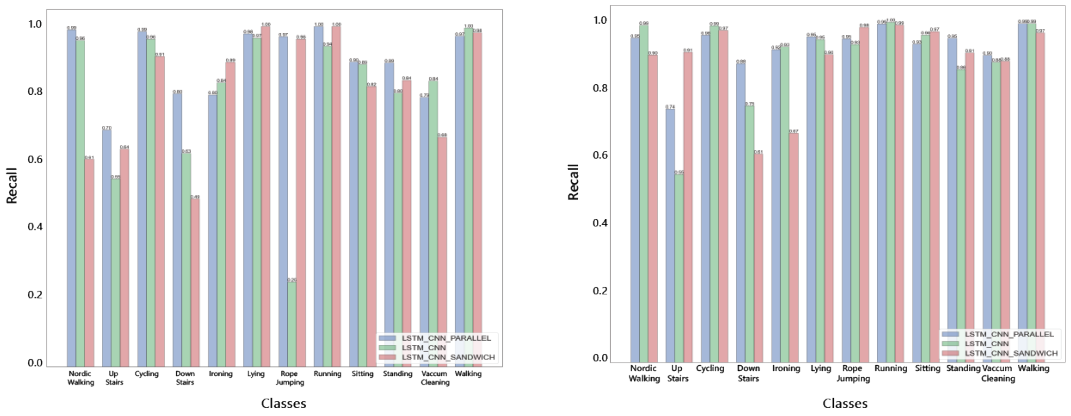


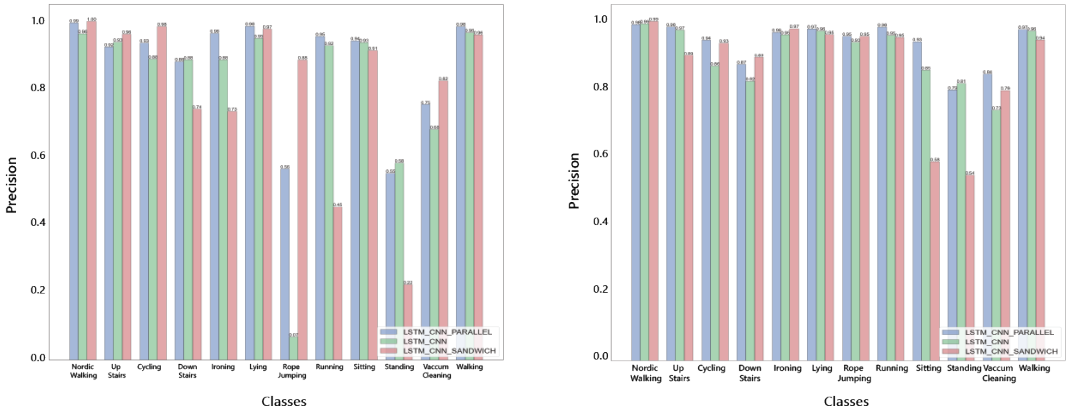
Figure 5.10: Training curves of models on PAMAP2 Dataset.

5.3.2.4 Recall and Precision



(a) (b)

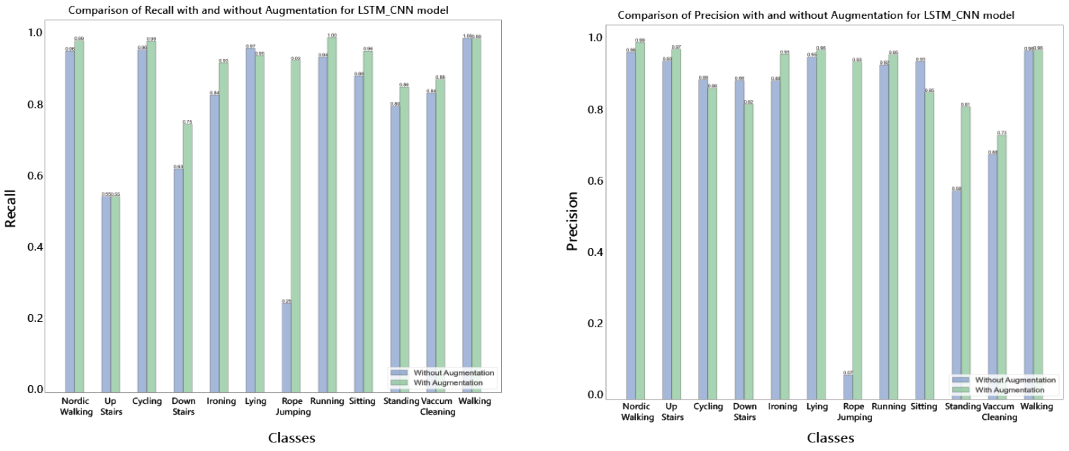
Figure 5.11: (a) Recall of models on PAMAP2 Dataset without Augmentation. (b) Recall of models on PAMAP2 Dataset with Augmentation.



(a) (b)

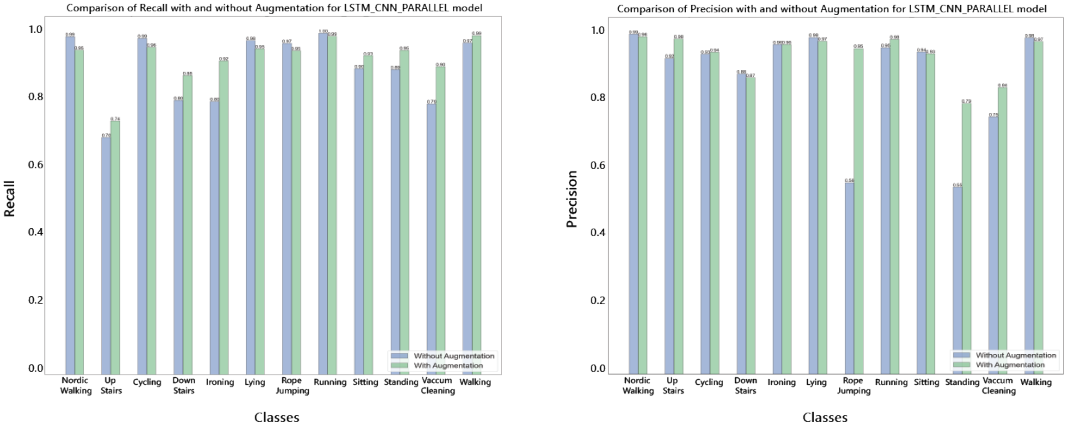
Figure 5.12: (a) Precision of models on PAMAP2 Dataset without Augmentation. (b) Precision of models on PAMAP2 Dataset with Augmentation.

Comparison between augmentation and without Augmentation:



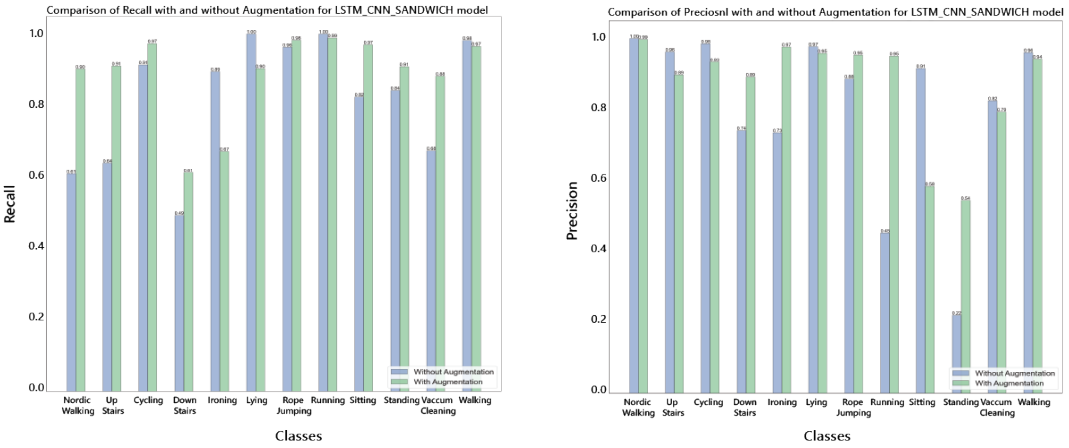
(a) (b)

Figure 5.13: (a) Recall of model LSTM_CNN on PAMAP2 Dataset with and without Augmentation. (b) Precision of model LSTM_CNN on PAMAP2 Dataset with and without Augmentation.



(a) (b)

Figure 5.14: (a) Recall of model LSTM_CNN_PARALLEL on PAMAP2 Dataset with and without Augmentation. (b) Precision of model LSTM_CNN_PARALLEL on PAMAP2 Dataset with and without Augmentation.



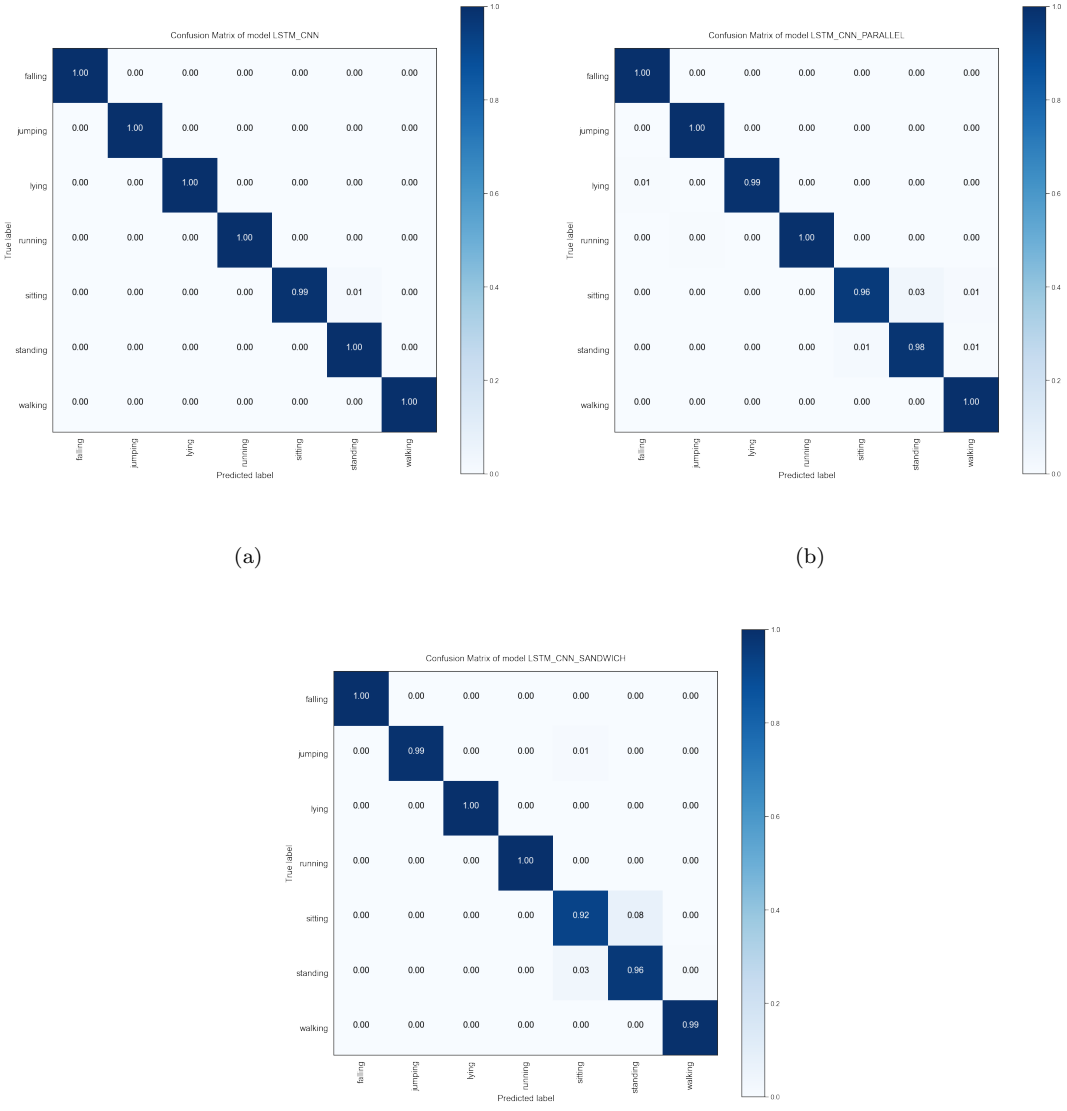
(a) (b)

Figure 5.15: (a) Recall of model LSTM_CNN_SANDWICH on PAMAP2 Dataset with and without Augmentation. (b) Precision of model LSTM_CNN_SANDWICH on PAMAP2 Dataset with and without Augmentation.

Again, it is apparent that recall and precision for low sample count activities improved significantly with augmentation.

5.3.3 ARS DLR Dataset

5.3.3.1 Confusion Matrices

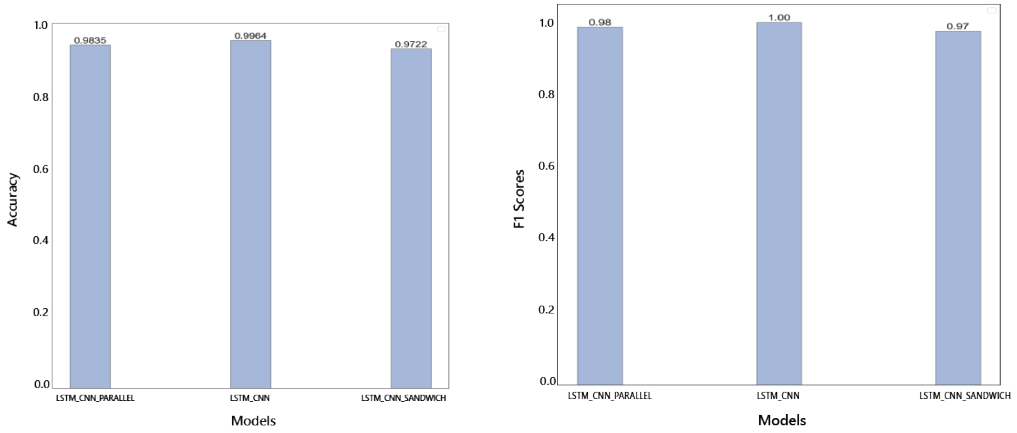


(a) (b)

(c)

Figure 5.16: (a) Confusion matrix of model LSTM_CNN on ARS DLR Dataset. (b) Confusion matrix of model LSTM_CNN_PARALLEL on ARS DLR Dataset. (c) Confusion matrix of model LSTM_CNN_SANDWICH on ARS DLR Dataset.

5.3.3.2 Accuracy and F1 Scores



(a) (b)

Figure 5.17: (a) Accuracy of models on ARS DLR Dataset. (b) F1 Score of models on ARS DLR Dataset.

5.3.3.3 Training Curves

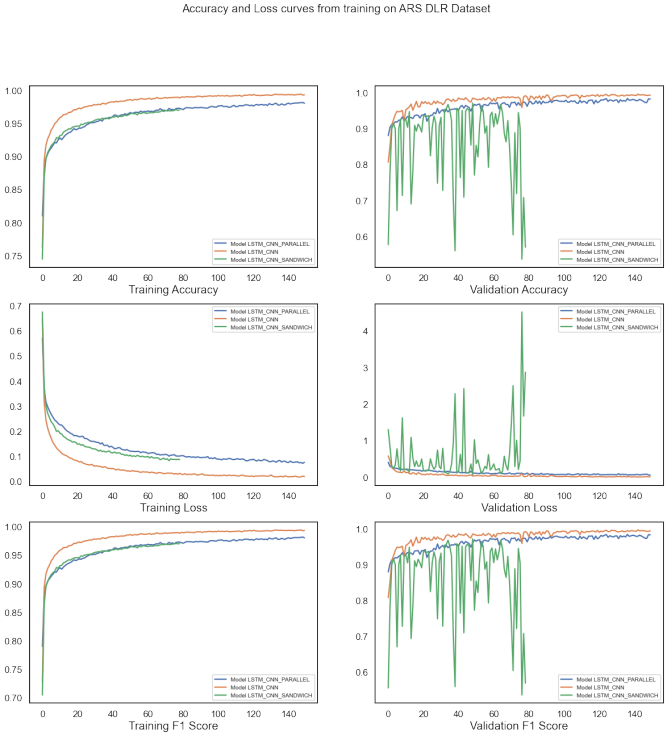
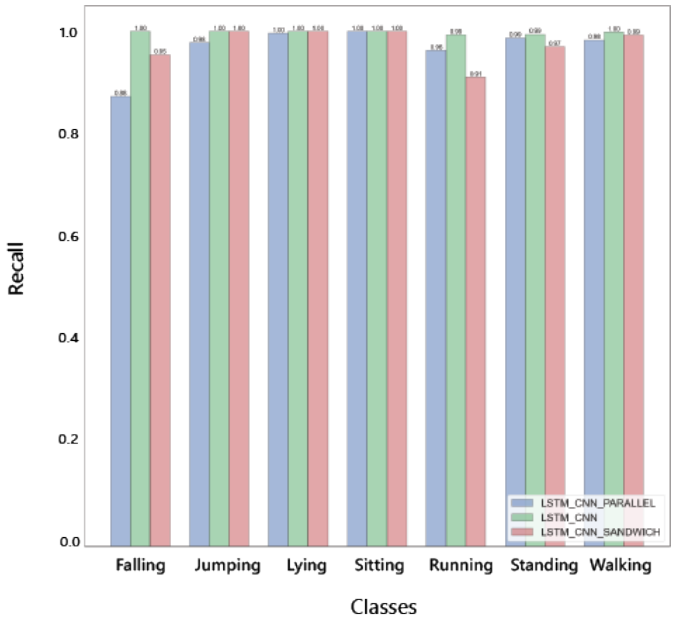
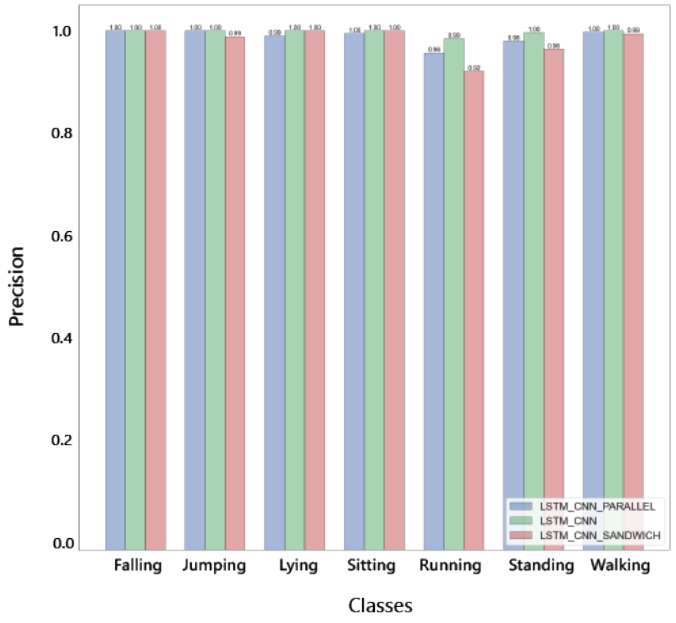


Figure 5.18: Training curves of models on ARS DLR Dataset.

5.3.3.4 Recall and Precision



(a)



(b)

Figure 5.19: (a) Recall of models on ARS DLR Dataset. (b) Precision of models on ARS DLR Dataset.

5.3.4 Overall Accuracy

Table 5.1: Overall performance comparison

Model Name	Accuracy on Dataset (%)			
	UCI	PAMAP2		ARS DLR
		No Augmentation	Augmentation	
LSTM_CNN_PARALLEL	95.26	90	93	98.35
LSTM_CNN	93.65	86	90	99.64
LSTM_CNN_SANDWICH	92.45	80	86	97.22

Model LSTM_CNN_PARALLEL outperforms the other two models in almost every scenario except the ARS DLR Dataset where it has done fairly well. It is also further established that Augmentation techniques help models in generalization and performing better.

CHAPTER 6

CONCLUSION AND FUTURE WORKS

Goal of this project was to investigate the performance of our models against a standard model and implement a real time prediction system. Based on the previous work, two models were designed to compare performance with a standard model from [15] that is referred to as LSTM_CNN in this work. It is clearly observable from the results that LSTM_CNN_PARALLEL has outperformed the model LSTM_CNN in [15] using the same hyperparameters.

Another objective of the project was to reduce overfitting problem of the models by generating synthetic data. Data augmentation in case of sensor signals is difficult as it is not intuitively apparent. Augmentation in this work was done based on techniques used in other time-series signals. Results showed significant improvement in the unbalanced dataset PAMAP2 with such augmentation techniques. It can be deduced from the results achieved that overfitting was reduced to a certain level by the augmentation and accuracy was increased.

Also, the real time implementation was satisfactory as the system was able to detect most of the activities fairly well in real time scenario.

More research can be done to better the models further with more hyper-parameter tuning and introduction of newer models or layers. Further work on the number of LSTM layers, filter sizes and other hyperparameters of the LSTM_CNN_PARALLEL may further improve it's accuracy. Data from sensors in multiple body-position can be included to remove sensor position dependence of model. Effects of inclusion

of more sensors like magnetometer, proximity, GPS etc. can also be investigated. Further improvement can be achieved by tuning the augmentation factors better for datasets. Recently, Generative Adversarial Networks (GAN)s are being used to generate synthetic data for sensor signals. In [48],[49] and [50], Generative Adversarial Network (GAN)s were developed to generate sensor signals of Human Activity. Application of these models in reduction of overfitting in existing deep learning models can also be investigated.

BIBLIOGRAPHY

[1] X. Zhou, W. Liang, K. I.-K. Wang, H. Wang, L. T. Yang, and Q. Jin, "Deep-learning-enhanced human activity recognition for internet of healthcare things," IEEE Internet of Things Journal, vol. 7, no. 7, pp. 6429–6438, 2020.

[2] C. Ronao and S.-B. Cho, "Human activity recognition with smartphone sensors using deep learning neural networks," Expert Systems with Applications, vol. 59, 04 2016.

[3] Z. Chen, Q. Zhu, C. Yeng, and L. Zhang, "Robust human activity recognition using smartphone sensors via ct-pca and online svm," IEEE Transactions on Industrial Informatics, vol. PP, pp. 1–1, 06 2017.

[4] R. Yang and B. Wang, "Pacp: A position-independent activity recognition method using smartphone sensors," Information, vol. 7, p. 72, 12 2016.

[5] X. Liang, Y. Zhang, G. Wang, and S. Xu, "A deep learning model for transportation mode detection based on smartphone sensing data," IEEE Transactions on Intelligent Transportation Systems, vol. 21, no. 12, pp. 5223–5235, 2020.

[6] Y. Zhao, R. Yang, G. Chevalier, and M. Gong, "Deep residual bidir-lstm for human activity recognition using wearable sensors," ArXiv, vol. abs/1708.08989, 2017.

[7] H. Zhao, C. Hou, H. Alrobassy, and X. Zeng, "Recognition of transportation state by smartphone sensors using deep bi-lstm neural network," Journal of Computer Networks and Communications, vol. 2019, pp. 1–11, 01 2019.

[8] J. Wang, Q. Long, PiRahc, K. Liu, and Y. Xie, "Human action recognition on cellphone using compositional bidir-lstm-cnn networks," 01 2019.

[9] Y. Qin, H. Luo, F. Zhao, C. Wang, J. Wang, and Y. Zhang, "Toward transportation mode recognition using deep convolutional and long short-term memory recurrent neural networks," IEEE Access, vol. 7, pp. 142 353–142 367, 2019.

[10] T. Um, F. Pfister, D. C. Pichler, S. Endo, M. Lang, S. Hirche, U. Fietzek, and D. Kulic, "Data augmentation of wearable sensor data for parkinson's disease monitoring using convolutional neural networks," 06 2017.

[11] A. Henpraserttae, S. Thiemjarus, and S. Marukatat, "Accurate activity recognition using a mobile phone regardless of device orientation and location," 05 2011, pp. 41–46.

[12] M. Razzaq, I. Cleland, C. Nugent, and S. Lee, "Semimput: Bridging semantic imputation with deep learning for complex human activity recognition," Sensors, vol. 20, p. 23, 05 2020.

[13] C.-T. Yen, J.-X. Liao, and Y.-K. Huang, "Human daily activity recognition performed using wearable inertial sensors combined with deep learning algorithms," IEEE Access, vol. 8, pp. 174 105–174 114, 2020.

[14] W. Sousa Lima, E. Souto, K. El-Khatib, R. Jalali, and J. Gama, "Human activity recognition using inertial sensors in a smartphone: An overview," Sensors, vol. 19, p. 3213, 07 2019.

[15] K. Xia, J. Huang, and H. Wang, "Lstm-cnn architecture for human activity recognition," IEEE Access, vol. 8, pp. 56 855–56 866, 2020.

[16] L. M. Dang, K. Min, H. Wang, M. Piran, H. Lee, and H. Moon, "Sensor-based and vision-based human activity recognition: A comprehensive survey," Pattern Recognition, vol. 108, 07 2020.

[17] B. Romaissa, B. Nini, M. Sabokrou, and A. Hadid, "Vision-based human activity recognition: a survey," Multimedia Tools and Applications, vol. 79, 11 2020.

[18] H. Yu, S. Cang, and Y. Wang, "A review of sensor selection, sensor devices and sensor deployment for wearable sensor-based human activity recognition systems," 01 2016, pp. 250–257.

[19] S. Mukhopadhyay, H. Ghayvat, J. Liu, and X. Gui, "Wellness sensors networks: A proposal and implementation for smart home to assisted living," IEEE Sensors Journal, vol. 15, pp. 1–1, 12 2015.

[20] M. Z. Uddin and M. M. Hassan, "Activity recognition for cognitive assistance using body sensors data and deep convolutional neural network," IEEE Sensors Journal, vol. 19, pp. 8413–8419, 2019.

[21] J. Qi, P. Yang, M. Hanneghan, S. Tang, and B. Zhou, "A hybrid hierarchical framework for gym physical activity recognition and measurement using wearable sensors," IEEE Internet of Things Journal, vol. 6, no. 2, pp. 1384–1393, 2019.

[22] L. Gou, D. Peng, X. Chen, L. Wu, and Q. Tang, "A self-calibration method for angular displacement sensor working in harsh environments," IEEE Sensors Journal, vol. 19, no. 8, pp. 3033–3040, 2019.

[23] S. Wan, L. Qi, X. Xu, C. Tong, and Z. Gu, "Deep learning models for real-time human activity recognition with smartphones," Mobile Networks and Applications, vol. 25, pp. 743–755, 2020.

[24] A. Das Antar, M. Ahmed, and M. A. R. Ahad, "Challenges in sensor-based human activity recognition and a comparative analysis of benchmark datasets: A review," 05 2019, pp. 134–139.

[25] J. Yang, H. Zou, H. Jiang, and L. Xie, "Fine-grained adaptive location-independent activity recognition using commodity wifi," 2018 IEEE Wireless Communications and Networking Conference (WCNC), pp. 1–6, 2018.

[26] T. Hur, J. Bang, D. Kim, O. Banos, and S. Lee, "Smartphone location-independent physical activity recognition based on transportation natural vibration analysis," Sensors, vol. 17, p. 931, 04 2017.

[27] B. Almaslukh, A. Artoli, and J. Al-Muhtadi, "A robust deep learning approach for position-independent smartphone-based human activity recognition," Sensors, 11 2018.

[28] K. Chen, D. Zhang, L. Yao, B. Guo, Z. Yu, and Y. Liu, "Deep learning for sensor-based human activity recognition: Overview, challenges and opportunities," 01 2020.

[29] A. Bulling, U. Blanke, and B. Schiele, "A tutorial on human activity recognition using body-worn inertial sensors," ACM Computing Surveys, vol. 46, 01 2013.

[30] E. Zdravevski, P. Lameski, V. Trajkovik, A. Kulakov, I. Chorbev, R. Goleva, N. Pombo, and N. Garcia, "Improving activity recognition accuracy in ambient assisted living systems by automated feature engineering," IEEE Access, vol. PP, pp. 1–1, 03 2017.

[31] S.-M. Lee, S. M. Yoon, and H. Cho, "Human activity recognition from accelerometer data using convolutional neural network," in 2017 IEEE International Conference on Big Data and Smart Computing (BigComp), 2017, pp. 131–134.

[32] M. Munoz-Organero, "Outlier detection in wearable sensor data for human activity recognition (har) based on drnns," IEEE Access, vol. 7, pp. 74 422–74 436, 2019.

[33] H. F. Nweke, Y. W. Teh, M. A. Al-garadi, and U. R. Alo, "Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: State of the art and research challenges," Expert Systems with Applications, vol. 105, pp. 233–261, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417418302136

[34] F. J. Ordóñez and D. Roggen, "Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition," Sensors, vol. 16, no. 1, 2016. [Online]. Available: https://www.mdpi.com/1424-8220/16/1/115

[35] F. Miao, Y. He, J. Liu, Y. Li, and I. Ayoola, "Identifying typical physical activity on smartphone with varying positions and orientations," Biomedical engineering online, vol. 14, p. 32, 04 2015.

[36] M. Ahmed, A. Das Antar, T. Hossain, S. Inoue, and M. A. R. Ahad, "Poiden: position and orientation independent deep ensemble network for the classification of locomotion and transportation modes," 09 2019, pp. 674–679.

[37] J.-L. Reyes-Ortiz, L. Oneto, A. Samà Monsonís, X. Parra, and D. Anguita, "Transition-aware human activity recognition using smartphones," Neurocomputing, vol. 171, 08 2015.

[38] A. Reiss and D. Stricker, "Introducing a new benchmarked dataset for activity monitoring," in 2012 16th International Symposium on Wearable Computers, 2012, pp. 108–109.

[39] D. D. B. Ahmed and D.-I. S. Kaiser, Human Activity Recognition with Inertial Sensors, Last accessed on January 2022. [Online]. Available: https://www.dlr.de/kn/en/desktopdefault.aspx/tabid-8500/14564_read-36508/

[40] J. B. Yang, M. N. Nguyen, P. P. San, X. L. Li, and S. Krishnaswamy, "Deep convolutional neural networks on multichannel time series for human activity recognition." AAAI Press, 2015.

[41] P. Agarwal and M. Alam, "A lightweight deep learning model for human activity recognition on edge devices," ArXiv, vol. abs/1909.12917, 2019.

[42] H. Tan, N. Aung, J. Tian, M. Chua, and Y. Ou Yang, "Time series classification using a modified lstm approach from accelerometer-based data: A comparative study for gait cycle detection," Gait Posture, vol. 74, 09 2019.

[43] M. M. H. Shuvo, N. Ahmed, K. Nouduri, and K. Palaniappan, "A hybrid approach for human activity recognition with support vector machine and 1d convolutional neural network," 10 2020.

[44] E. Sansano, R. Montoliu, and O. Belmonte Fernández, "A study of deep neural networks for human activity recognition," Computational Intelligence, vol. 36, 03 2020.

[45] D. Cook and N. Krishnan, Activity learning: Discovering, recognizing, and predicting human behavior from sensor data, 02 2015.

[46] S. Wiesler, A. Richard, R. Schlüter, and H. Ney, "Mean-normalized stochastic gradient for large-scale deep learning," 05 2014, pp. 180–184.

[47] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Data augmentation using synthetic data for time series classification with deep residual networks," 08 2018.

[48] G. Ramponi, P. Protopapas, M. Brambilla, and R. Janssen, "T-cgan: Conditional generative adversarial network for data augmentation in noisy time series with irregular sampling," 11 2018.

[49] E. Soleimani and E. Nazerfard, "Cross-subject transfer learning in human activity recognition systems using generative adversarial networks," Neurocomputing, vol. 426, 11 2020.

[50] J. Wang, Y. Chen, Y. Gu, Y. Xiao, and H. Pan, "Sensorygans: An effective generative adversarial framework for sensor-based human activity recognition," in 2018 International Joint Conference on Neural Networks (IJCNN), 2018, pp. 1–8.