



MSA UNIVERSITY
جامعة أكتوبر للعلوم الحديثة والآداب



UNIVERSITY of
GREENWICH

October University for Modern

Sciences and Art

Faculty of Computer Science

Graduation Project

**Gym Exercises Classifying
Using Deep Learning**

Supervisor: Dr. Ahmed farouk

Name: Ahmad Sameh

ID: 181631

Abstract

Research on activity recognition has historically focused on distinguishing between different activities, i.e., predicting which action will be done next "a certain period of time during which an activity was carried out the skill with which an activity is carried out, the 'how (good)' ", despite the fact that it has the ability to give relevant information for a wide range of applications, has gotten very little attention to yet. Specifically, we define quality of execution and study three areas of qualitative activity recognition: describing accurate execution, identifying execution problems, and delivering feedback to the user on these aspects of qualitative activity recognition. We use the challenge of qualitatively evaluating and delivering feedback on weight lifting workouts as an example to demonstrate our method. In two user trials, we investigate the effectiveness of a sensor- and a model-based method to qualitative activity identification, respectively. Our findings demonstrate the possibility of model-based evaluation as well as the beneficial influence of real-time user input on the overall quality of execution in practice.

In a consensus statement released by the British Association of Sport and Exercise Sciences, it was shown that physical exercise may lower the risk of coronary heart disease (CHD), obesity (OBE), type 2 diabetes (T2D), and other chronic illnesses [6]. Recent research projected that boosting people's cardio-respiratory fitness might save at least 16 percent of all fatalities [6], which is significant. An excellent strategy to improve cardio-respiratory fitness is to engage in regular muscle-strengthening activities on a consistent basis. Such activities are suggested even for healthy individuals since they have been found to decrease blood pressure, enhance glucose metabolism, and reduce the chance of developing cardiovascular disease [6].

Table of Contents

Table of Figures	VI
Chapter 1: Introduction	1
1.1 Introduction	2
1.2 Problem statement	4
1.3 Objective	5
1.4 Motivation	5
1.5 Thesis layout	6
Chapter 2: Background and Literature Review	7
2.1 Background	8
2.2 Previous Work	14
2.2.1 Research 1: Recognition and Repetition Counting for Complex Physical Exercises with Deep Learning	14
2.2.1.1 Strategy & Structure	14
2.2.1.2 Data	15
2.2.1.3 Method Evaluation	17
2.2.1.4 Results Evaluation	18
2.2.2 Research 2:	19
2.2.2.1 Strategy & Structure	19
2.2.2.2 Data	20
2.2.2.3 Method Evaluation	22
2.2.2.4 Results Evaluation	22
2.2.3 Research 3:	24
2.2.3.1 Strategy & Structure	24
2.2.3.2 Data	25

2.2.3.3 Method Evaluation	26
Chapter 3: Material and Methods.....	28
3.1 Materials.....	29
3.1.1 Data	29
3.1.2 Tools.....	29
3.1.3 Environment.....	29
3.2 Methods.....	29
3.2.1 System architecture Overview	30
Chapter 4: System Implementation.....	31
4.1 System Development	32
4.2.1 System Overview	35
4.2.2 TensorBoard.....	36
4.3 System Running	40
4.3.1 Reading and Preprocessing the Dataset	40
Chapter 5: Results and Evaluation	45
5.1 Testing Methodology	46
5.2 Results	46
5.2.1 Best Results Cases	46
5.2.3 acceptable Results Cases.....	54
5.2.3 Worst Results Cases.....	62
5.2.6 Limitations	67
5.3 Evaluation	68
5.3.1 Accuracy Evaluation	68
5.3.2 Time Performance.....	70
Chapter 6: Conclusion and Future Work	71

6.1 Conclusion 72

6.3 Future Work 72

References 73

Table of Figures

Figure 1 An example of sensor-based activity identification utilizing traditional pattern recognition techniques [13].....	2
Figure 2 move sensor [11]	3
Figure 3 sensor worn on chest [11]	4
Figure 4 Confusion matrix of exercise recognition using all sensors from both smartwatches. [9]5	
Figure 5 System pipeline [9]	15
Figure 6 Execution of four typical CrossFit exercises [9]	16
Figure 7 Neural network architecture for recognition and counting [9]	18
Figure 8 position of sensors [13].....	19
Figure 9 squats [11].....	20
Figure 10 Acceleration for an exercise set recorded on the chest. [11]	21
Figure 11 The CNN architecture utilized for recognizing workout types [13].....	22
Figure 12 Accuracy (a) and loss (b) over training and validation data [11]	23
Figure 13 sensing setup [12]	25
Figure 14 Repetition counted using joint position coordinates. At the end of each iteration, we verify the total range [12].	26
Figure 15 Averaged over all individuals and normalized across ground truth rows [12].....	27
Figure 16 system architecture	30
Figure 17 trainers records	33
Figure 18 classes records	34
Figure 19 system structure	35
Figure 20 first section of tensor board	36
Figure 21 second section of tensor board.....	37
Figure 22 third section of tensor board	38
Figure 23 forth section of tensor board	39
Figure 24 Last section of tensor board.....	40
Figure 25 data before any processes	40
Figure 26 Created X_train.....	41
Figure 27 Created y_train	42

Figure 28 Dataset Features.....	42
Figure 29 Epoch learning rate	43
Figure 30 Epoch Loss	44
Figure 31 Epoch Accuracy.....	44
Figure 32 model accuracy best case.....	47
Figure 33 model loss best case.....	48
Figure 34 model summary best case	48
Figure 35 confusion matrix between true and predicted results	49
Figure 36 model flow best case.....	51
Figure 37 confusion matrix for the second-best result.....	52
Figure 38 second-best result model accuracy	52
Figure 39 second best-result model loss	53
Figure 40 confusion matrix for the acceptable case.....	54
Figure 41 model accuracy for the acceptable case.....	55
Figure 42 model loss for the acceptable case.....	56
Figure 43 model accuracy acceptable case	57
Figure 44 model loss acceptable case	58
Figure 45 model summary acceptable case.....	59
Figure 46 confusion matrix for the acceptable case.....	59
Figure 47 model flow acceptable case	61
Figure 48 model accuracy worst case	62
Figure 49 model loss worst case	63
Figure 50 model summary worst case.....	63
Figure 51 confusion matrix worst case	64
Figure 52 model flow worst case	66
Figure 53 y_train and predicted y value.....	67
Figure 54 best case model accuracy.....	69
Figure 55 best case model loss.....	70

Table of Tables

Table 1: <i>List of all exercises with respective code and equipment</i>	15
Table 2: Statistics for the collected data.....	16
Table 3: Comparison of 5-fold cross validation test accuracy for different combinations of sensors.....	17
Table 4: classification report for acceptable case.....	44
Table 5: Classification report for second best-result.....	48
Table 6: Classification report for the acceptable case.....	51
Table 7: classification report for acceptable case.....	54
Table 8: classification report for worst case.....	60
Table 9: First Summary of the LSTM Models with their Accuracy.....	63

Chapter 1: Introduction

1.1 Introduction

In this part we state an overview of gym exercises. Exercises and Activities carried out inside, usually with the aid of equipment. A Gym is a big place with Equipment for training the body and gaining strength, or it's a club where you can do exercises and improve health. Home behavior analysis, video surveillance, gait analysis, as well as gesture recognition are examples of successful human activity recognition applications. HAR is classified into two types: video-based and sensor-based [13]. To perform a self-tracking of physical and behavioral information, such as step counts per day, sleep patterns, or statistics of conducted sporting activities, wearable devices, such as smartwatches or smartphones, are the most common tools for performing this type of activity gathering [4]. HAR may be thought of as a normal pattern recognition (PR) issue. By incorporating machine learning algorithms such as decision trees, support vector machines, naïve bayes, and hidden Markov models into traditional PR approaches, conventional PR approaches have made tremendous progress on HAR [13].

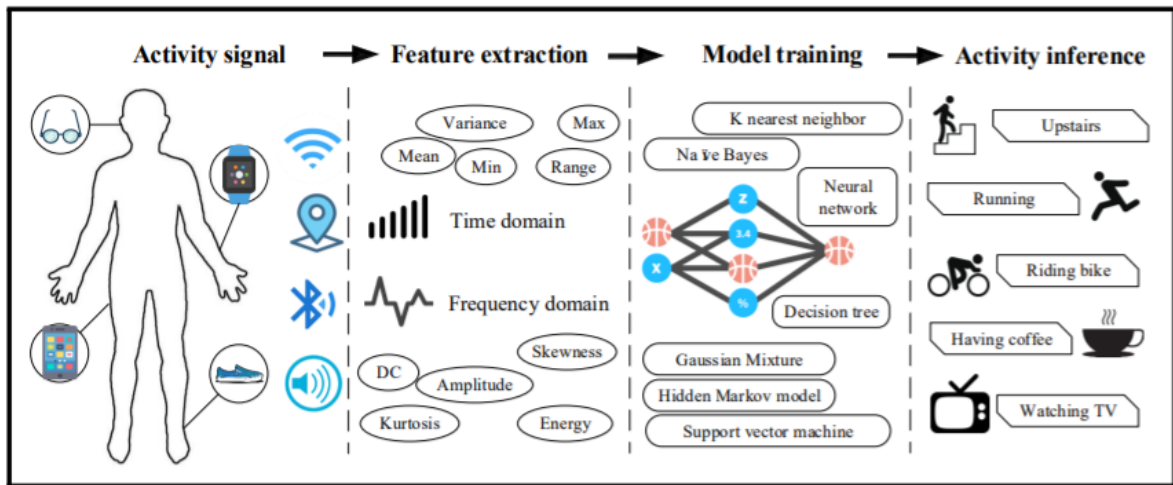


Figure 1 An example of sensor-based activity identification utilizing traditional pattern recognition techniques [13]

However, with the increased availability of data through off-the-shelf devices such as smartphones, smartwatches, and fitness trackers, as well as general increase in computing power, more work has focused on using deep learning methods.



Figure 2 move sensor [11]

In order for successful training to have a beneficial influence on cardiorespiratory fitness, appropriate technique must be used throughout the training session. It has been shown that incorrect technique is the most common cause of training injuries [8]. Furthermore, free weights workouts are responsible for the majority of weight training-related injuries in the United States (90.4 percent) [14]. Moreover, according to the same research, those who lift free weights are more prone to fractures and dislocations than people who lift weights using machines. Personal training by a professional trainer is the most often used method of preventing injuries and providing athletes with feedback on their technique in order to improve performance. Despite the fact that it is very successful, the presence of a trainer may not always be achievable owing to the expense and availability of such individuals. The number of athletes under personal supervision does not scale well with the quantity of athletes, especially among non-professionals. The use of ambient or on-body sensors for measuring workouts and delivering feedback on the quality of execution is a particularly promising method to exercise evaluation and feedback.

As a regular practice in sports science, trainers film their athletes with cameras and then utilize video digitizing systems to do on-line frame-by-frame annotation of the data collected from the athletes. A third option is to employ marker-based tracking systems, which automatically construct a digital skeleton of the athlete's body. There has been a significant amount of research on automated systems for distinguishing which activity was performed using on-body sensors in the field of activity recognition. Only a little

amount of research has been done to yet on the topic of assessing how (well) an activity was completed. This second kind of recognition is referred to as "qualitative activity recognition."

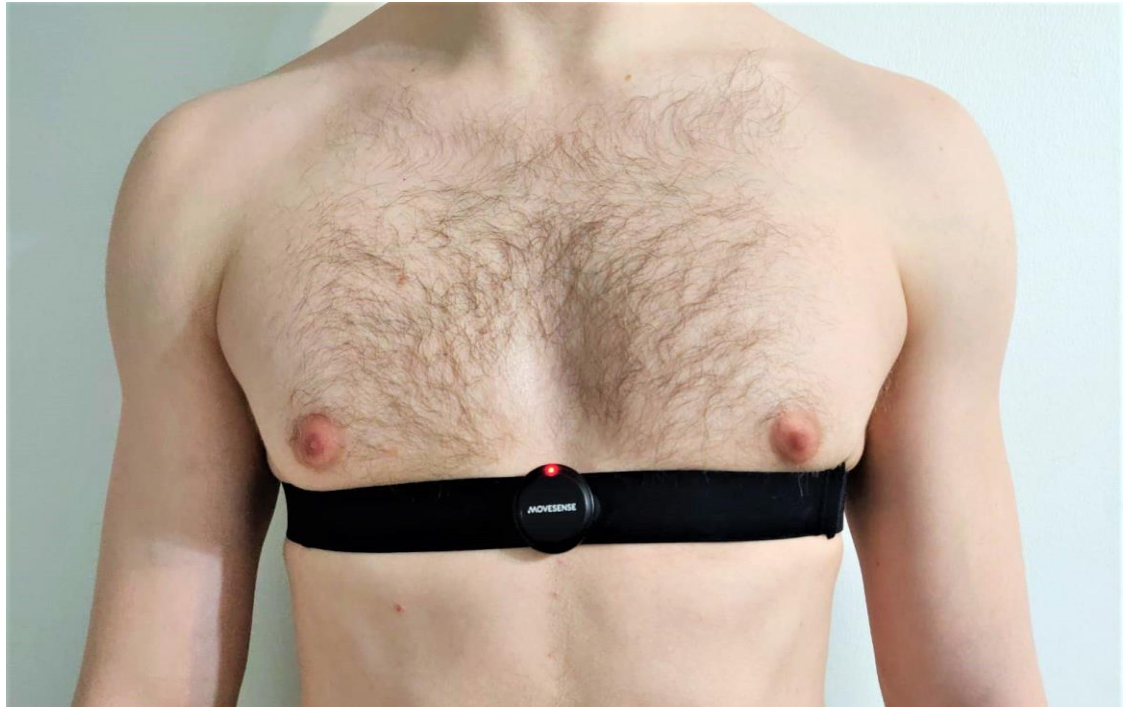


Figure 3 sensor worn on chest [11]

1.2 Problem statement

In this part, researchers used smartwatches, image processing, and sensors such as 3D accelerometers to identify different workout kinds such as pushups, sit-ups, squats, and jumping jacks throughout the implementation and early development phases. But how can we build a device without going overboard in terms of size, weight, connections, and other constraints. And, if we make the device as tiny and lightweight as feasible. How can we compensate for the reduced precision with a high-accuracy machine learning model that compensates for any sacrifice in the size of devices and sensors? As a result, the deep learning model that will aid in the design of these devices

is our primary emphasis here. Gym exercises pattern prediction will assist new trainees in keeping track of themselves.

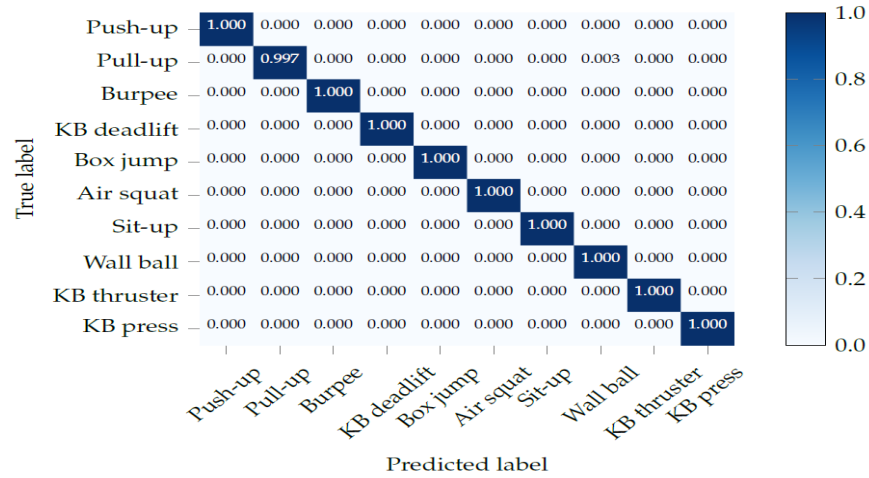


Figure 4 Confusion matrix of exercise recognition using all sensors from both smartwatches. [9]

1.3 Objective

Giving the problem statement, this project's main goal is to design a model that will understand the records of the trainees and classify it based on the features whether this record is Class A, B, C, D or E It is supposed to provide approaches for records understanding which is the key of the success of our system. As it will help a lot of trainees' old and new ones.

1.4 Motivation

Our key aim is that the outcome will assist new trainees in tracking themselves to have a better knowledge of whether they are performing the correct technique or not, and then following the directions for the correct one. Not understanding the appropriate motions for workouts is a possible concern since new trainees may not know exactly the right set of actions or may make mistakes, which may result in injuries, some of which may be fatal. As a result, this model should eventually assist trainees in understanding their workouts and how to perform them properly, resulting in benefits and preventing injuries.

1.5 Thesis layout

The first chapter provides a thesis outline as well as an introduction to the project. We want to provide a literature review and context for earlier efforts in the same topic of inquiry in the next chapter.

Chapter 2: Background and Literature Review

2.1 Background

- Sensor-based activity recognition

The goal of HAR is to better understand human behavior so that computer systems can provide proactive assistance to users depending on their requirements [13]. Consider the following scenario: a user is engaged in a variety of activities that fall within the purview of a predetermined activity set A .

$$A = \{A_i\}_{i=1}^m \quad (1)$$

where m is the number of different sorts of activities. There is a succession of sensor readings that collects the information about the activity.

$$s = \{d1, d2, \dots, dt, \dots, dn\} \quad (2)$$

in which dt signifies the sensor reading at the specified time t . Based on the sensor readings, we must construct a model F that predicts the activity sequence to be followed.

$$\hat{A} = \{\hat{A}_j\}_{j=1}^n = F_{(s)}, \quad \hat{A}_j \in A \quad (3)$$

In contrast, the genuine activity sequence (also known as the ground truth) is designated as

$$A^* = \{A_j^*\}_{j=1}^n, \quad A_j^* \in A \quad (4)$$

where n signifies the length of the sequence and $n \geq m$ [13].

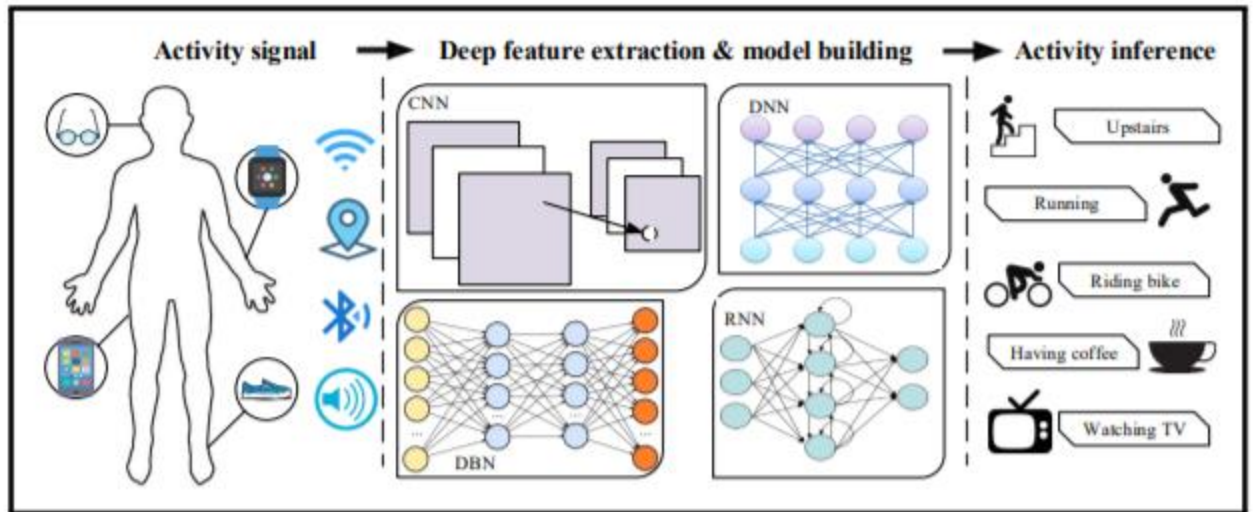


Figure 5 Using deep learning to recognize sensor-based activities. [13]

As briefly noted in the preceding chapter, there are a variety of methods for collecting data and identifying exercises to use. First and foremost, smartwatches are a fantastic piece of technology in which data is gathered by an application that may be running on two smartwatches, each worn on a separate body part and linked to the other using Bluetooth low-energy (BLE). A screenshot of the application's main menu is presented in Figure 6. This piece of technology is quite compatible with the objective of keeping sensors and tech components as compact as possible, although it is likely that this will not have an impact on accuracy. Additionally, when the program begins, the screen that is often used in such technology is presented, allowing you to choose from among the available routines. A time interval for the vibrations that signified the start of a repeat could also be programmed into the system. [9]



Figure 6 main menu screen [9]

Once the link between the two watches has been established, the wrist watch will transmit a command to the ankle watch, and the ankle watch will not need any user engagement on its own behalf. The ankle watch is used for the monitoring of everything to ensure its operating properly while doing the exercise, and the touch function is deactivated. Syncing the time on both watches was required in order for the sensor data to be recorded at the same time on both watches simultaneously. True Time, an NTP library, was used to tackle this problem for the sake of simplicity. It was determined that both watches were linked to a WIFI network and were able to acquire accurate times from NTP servers. It was shown throughout the data gathering process on the workout screen, which is seen in Figure 7. This page displayed the current workout as well as a button for initiating and terminating the recording of sensor data from the wearable device. After a 5-second countdown, the watch began recording in order to give the participant time to get into position. Following the completion of each activity, the subject was asked to indicate how many repetitions they had completed. The software was tested on two Huawei Watch 2 smartwatches that were

running Android Wear version 2.9. Each of the two timepieces was always worn in the same configuration on the right ankle and on the right wrist, and they were both securely fastened to prevent any rotation. [9]



Figure 7 Workout screen [9]

The second method is to train a system using a sample of segmented spectrograms that has been segmented. These 2D spectrograms are used to build the signal that is sent into the classifier modelling software.

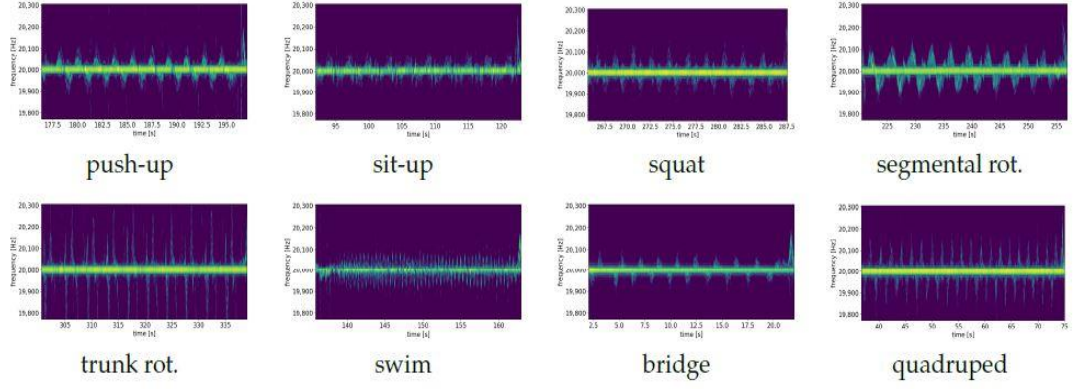


Figure 8 spectrogram of the sample activities [4]

We assessed our data on two end-to-end neural network topologies without incorporating significant domain expertise into the design process beforehand. The first architecture is a finetune model using VGG16 weights for the basic feature extraction layer, which is the first of two architectures. An LSTM bidirectional inference model is the second inference model to be discussed in this section. In the next part, we will go through the different model architectures as well as the hyperparameters that are utilized in each of the different designs. Using 5-fold cross-validation, the hyperparameters were refined to their optimal values. Using the Pytorch framework, the models were created and then trained on a GeForce RTX 2080 graphics card module. It was possible to directly get the weights of the finetune model VGG16 from the Pytorch model zoo. We further enhanced the generalization ability of the inference model by using a few-shot classification strategy on previously unknown test data to further increase its generalization capacity.[5]

In order to identify peaks, there are two factors to consider: the minimum distance between peaks (d_{min}), as well as the minimum height (h_{min}) of the peak. In the case of the latter, we establish an adjustable threshold α . The distance between this new candidate and our prior peak is determined after we have passed beyond the threshold of α . If the distance between the two points is more than d_{min} , the candidate is considered a peak.

$$\alpha = \text{mean}(\text{data}) + (\text{max}(\text{data}) - \text{mean}(\text{data})) * h_{min} \quad (1)$$

For various exercise kinds, we use different configurations of our peak detection system, which we describe below (Tab. 1). In most exercises, the number of peaks identified corresponds to the number of repetitions completed in that exercise, with the exception of jumping jacks, for which the number of repetitions equals half the number of peaks discovered. [9]

The Movesense sensor was chosen for its small size and low cost. Its diameter is 36.6mm and its thickness is 10.6 mm, making it ideal for small spaces. The controlling unit is a Nordic Semiconductor nRF52832, which consists of a 32-bit ARM Cortex-M4 processor with 64kB of on-chip RAM and 512kB of on-chip FLASH and is powered by a lithium-ion battery. The sensor communicates with the rest of the world via Low Energy Bluetooth 4.0. A number of parameters such as linear and angular acceleration, magnetic field intensity, temperature and heart rate may be measured by the Movesense system.[11]

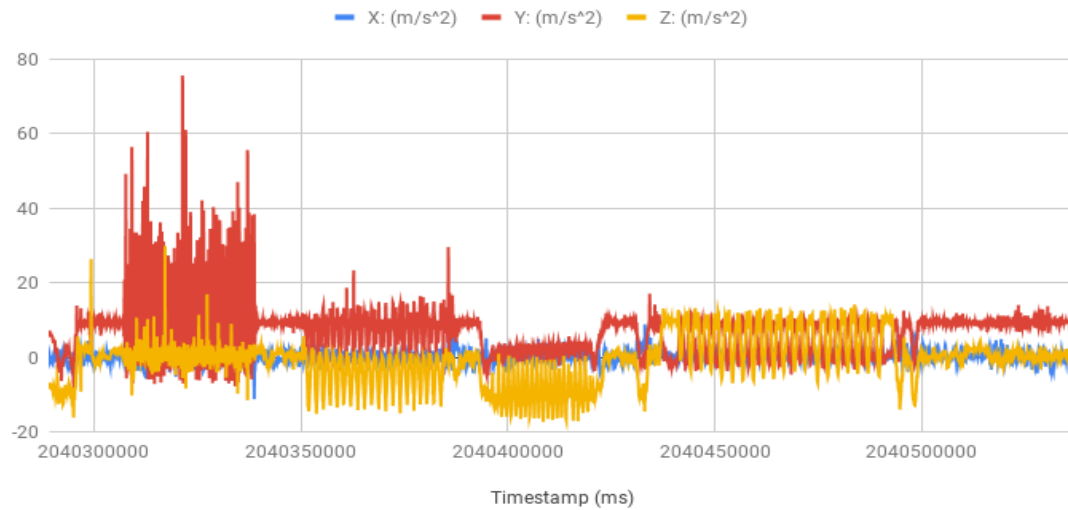


Figure 9 Acceleration [m/s²] for an exercise set recorded on the chest [11]

2.1.1 Machine Learning

Computer algorithms to automatically improve the experience and adapt over time, as more data from the environment and its interaction is specified. Machine learning (ML) is considered to be a component of artificial intelligence (AI), which is the study of computer algorithms to automatically improve the experience and adapt over time.

2.1.2 Neural Network and Deep Learning

In addition, Deep Learning techniques are regarded to be a component of artificial intelligence (AI). It has the same purpose as the human brain in that it analyses data and builds maps to aid in decision-making. You may also learn unsupervised in deep learning and utilize data that is neither organized or labelled, which is a great advantage.

2.2 Previous Work

In this section, we will provide the findings of a couple of researchers that attempted to address the topic's aim and provided some alternate approaches to doing it, which is the recognition of workout activities.

2.2.1 Research 1: Recognition and Repetition Counting for Complex Physical Exercises with Deep Learning (2019)

2.2.1.1 Strategy & Structure

Figure 10 depicts a high-level overview of our product development process. Raw sensor data is supplied into a neural network for exercise classification, which predicts the kind of exercise for each window based on the sensor data. We make use of overlapping input windows, majority voting, and subsequent smoothing to achieve our results. We employ distinct neural networks for each kind of exercise to count the number of repetitions performed. As a result, accurate exercise recognition is critical in order to choose the most appropriate neural network for counting. The repetition counting neural network generates a succession of 1's and 0's, with 1's designating the start of a repeat and 0's designating the end of a repetition. After that, we smooth the binary series and count the number of 1 sequence, which gives us the final number of repetitions in the binary series. Individual phases of the pipeline will be described in more depth in the sections that follow this one.

[9]

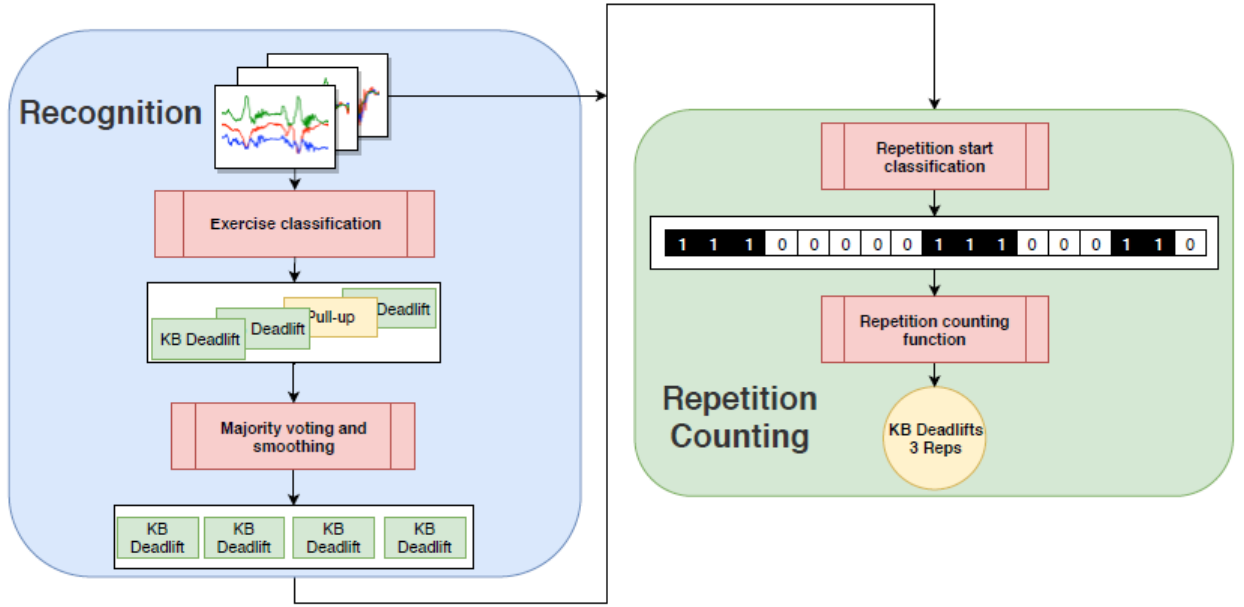


Figure 5 System pipeline [9]

2.2.1.2 Data

Currently, there are no publicly accessible data sets of sensor records for CrossFit that have been made available. As a result, we devised a variety of exercises and gathered data from willing participants. The information was gathered via the use of two smartwatches, one worn on the wrist and one worn on the ankle. The sensor data was then utilized to train and evaluate the various models developed as a result of the research. [2]

Table 1 lists the 10 most often performed actions in CrossFit, which we selected from the list. The following activities are shown in Figure 11 in order to familiarize the reader with some of the most CrossFit-specific movements: the kettlebell push (E9), the kettlebell thruster (E10), the wall ball (E8), and the burpee (E9) (E3). [2]

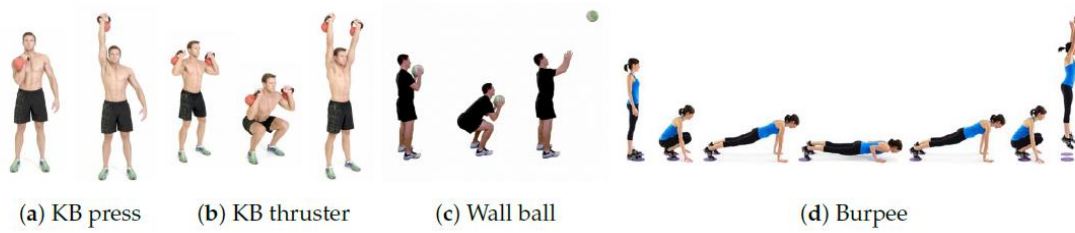


Figure 6 Execution of four typical CrossFit exercises [9]

Table 1 List of all exercises with respective code and equipment [9]

Ex. Code	Exercise	Equipment
E1	Push-up	Body weight
E2	Pull-up	Pull-up bar
E3	Burpee	Body weight
E4	Kettlebell deadlift	Kettlebell
E5	Box jump	Box
E6	Air Squat	Body weight
E7	Sit-up	Body weight
E8	Wall ball	Medicine ball
E9	Kettlebell press	Kettlebell
E10	Kettlebell thruster	Kettlebell

Using a program that ran on two smartwatches that were in contact with each other using Bluetooth low-energy, the sensor data was gathered and analyzed (BLE). Figure 6 depicts the primary menu of the application. This page was presented when the app was first launched, and it enabled users to choose one of the exercises, link the two smartwatches over Bluetooth, and sync the time on both devices. In addition, the time interval between the vibrations that marked the start of a repeat could be programmed into the system. [9]

An impressive number of 61 persons agreed to take part in the data gathering process. Because of technical issues with the smartwatches, the data from seven individuals had to be omitted from the analysis. For example, the ankle watch would occasionally cease collecting data or the watches would not remain synced.

50 persons took part in the time-constrained exercise out of the remaining 54 participants. There were five participants who participated in the unconstrained exercise, and four of those people also participated in the confined workout. The data for the null class was gathered from seven participants, four of whom also participated in the limited exercise session. The participants range in age from their early twenties to their early thirties. There were 43 males and 11 women among those who took part in the study. [9]

Table 2 Statistics for the collected data [9]

Exercise	Participants	Time [min]	Repetitions	Fraction of time
Push-up	50	25.05	718	11%
Pull-up	43	13063	300	6%
Burpee	47	30.97	551	13%
Kettlebell Deadlift	48	28.43	689	12%
Box jump	47	19.2	546	8%
Air squat	44	22.27	642	10%
Sit-up	42	26.5	555	12%
Wall ball	42	23.42	529	10%
Kettlebell Press	44	19.7	482	9%
Kettlebell thruster	39	20.88	449	9%
Total	-	230.05	5461	100%

2.2.1.3 Method Evaluation

Further adjustment may be accomplished by looking at Table A1 in Appendix A, which has a comprehensive list of the architectural parameters. Further, we tweaked the models for each workout type separately, in accordance with the data shown in Table 4. Table A2 in Appendix B has the final hyper-parameters of the repetition counting models for each of the exercises, and it contains the hyper-parameters for the final repetition counting models for each of these exercises. [9]

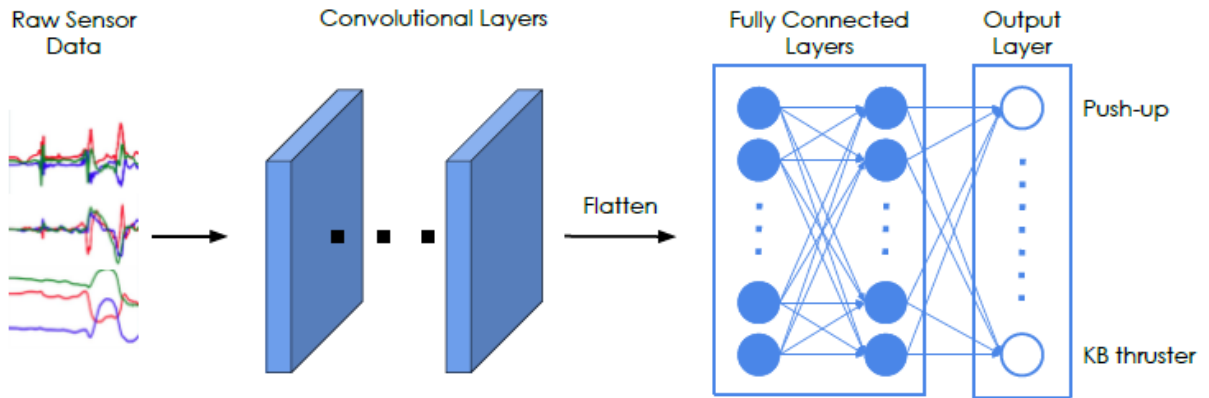


Figure 7 Neural network architecture for recognition and counting [9]

2.2.1.4 Results Evaluation

The performance of several sensor combinations is shown in further detail in Table 17. This also demonstrates the advantage of wearing a second smartwatch on the ankle. The optimum performance is obtained by combining the data from all of the sensors. [9]

Table 3 Comparison of 5-fold cross validation test accuracy for different combinations of sensors[9]

Inputs	5-cv Test acc
All	99.96%
Hand	95.90%
Foot	86.30%
Hand accelerometer	95.73%
Hand gyroscope	28.60%
Hand gyroscope and accelerometer	98.91%
Hand orientation	11.28%

2.2.2 Research 2: Workout Type Recognition and Repetition Counting with CNNs from 3D Acceleration Sensed on the Chest 16 May 2019

2.2.2.1 Strategy & Structure

Our goal is to automatically discover, identify, and compute representatives in four distinct kinds of training scenarios using machine learning. In addition to detecting four different sorts of exercises, they will take into consideration a fifth type known as no workout, which will include all of the other categories or the absence of any activity at all. They will be able to distinguish between exercises thanks to a single sensor named (3D accelerometer). Having just one sensor is justified by the fact that using more than one sensor would be inconvenient for users who want their exercises to be recognized and tallied.

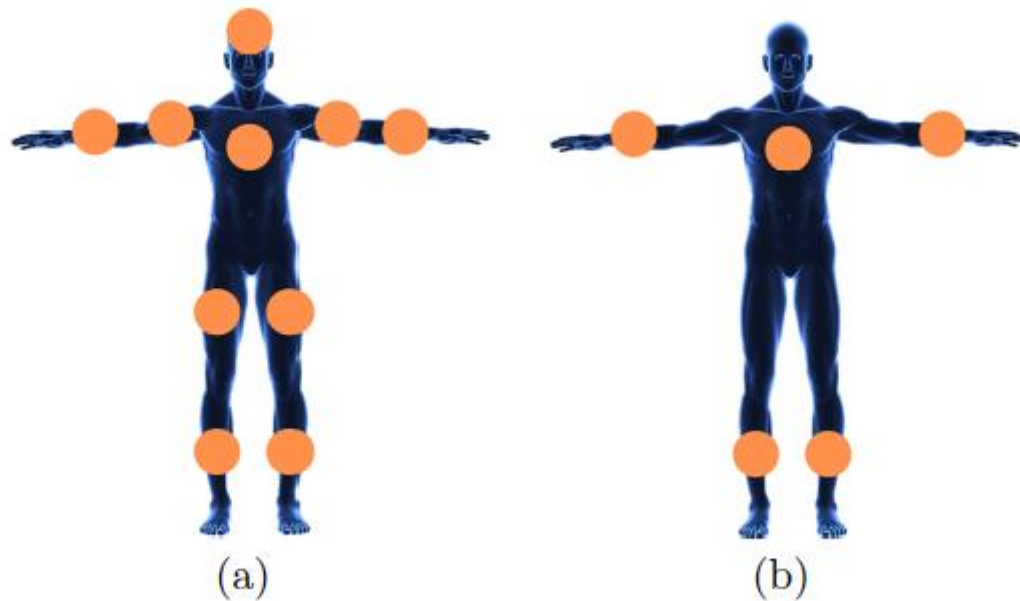


Figure 8 position of sensors [13]

In the literature, (a) the majority of solutions are based on data collecting, and (b) the environment is considered to be optimum.

2.2.2.2 Data

We collected the exercise data of ten distinct participants over the course of two to three workout sessions per subject, with each subject undertaking one of four different workout kinds, and with no-workout periods in between training sessions. As a consequence, a total of 44 workouts were recorded, with an average of 11 sessions per kind of exercise reported. Each exercise consists of 10, 20, or 40 repetitions of the program, depending on the kind of training and the individual. Figure 15 is a sample workout recording that includes jumping jacks, squats, pushups, and sit-ups in the following order: jumping jacks, squats, pushups, and sit-ups.

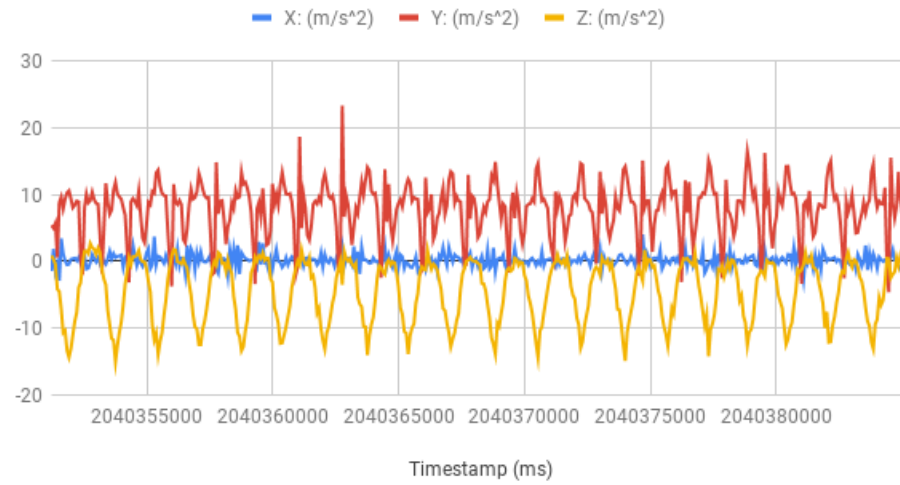


Figure 9 squats [11]

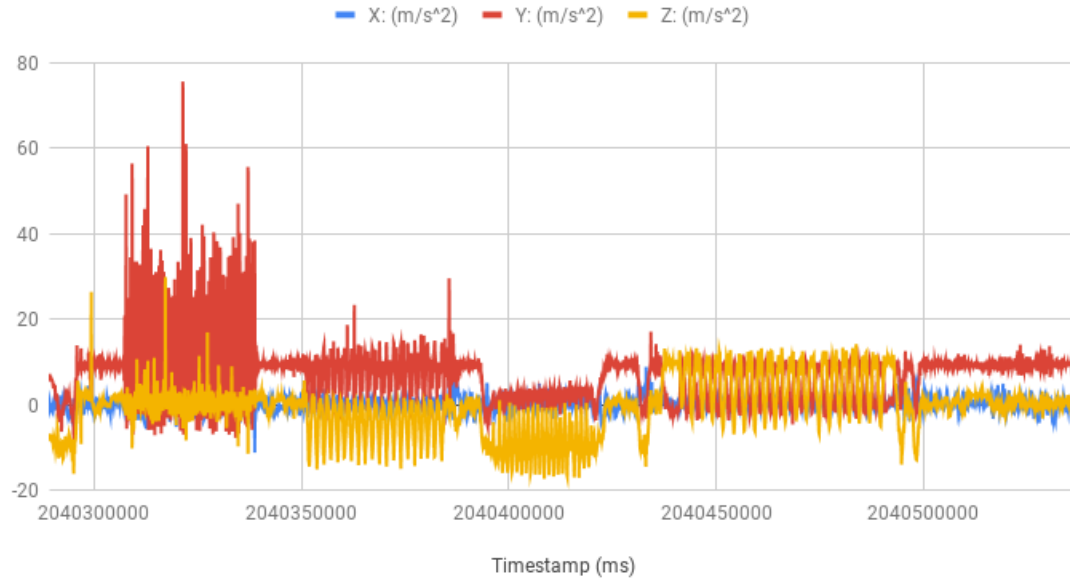


Figure 10 Acceleration for an exercise set recorded on the chest. [11]

Figure 15 illustrates the workout records, which include leaping, squatting, push-ups, and sit-ups, among other things. There are many rest-related activities in these samples (both uncontrolled and individual activities for each participant) such as sitting, standing, walking, drinking, and removing from one activity in the no-workout portion of the samples. Push-ups, for example, are a transitional exercise from one sort of training to another type of training.[11]

This resulted in a total of 55 exercise samples (of which 11 were non-exercise samples). Jumps, sit-ups, and squats are all shown in Figure 14 and Figure 15 as an example of a workout routine. As a result, the samples collected serve as the foundation for the identification and counting of different kinds of training and assessment activities.[11]

2.2.2.3 Method Evaluation

Each sample includes 156 functions and reflects the training data that was identified from the labelled samples. The model we trained to discriminate our five training kinds (four target training types and untrained types) took one second to train and was ready in one second. It is possible that you will have exercised, and that the model will be able to automatically identify between the new sample and the old sample, and if so, what form of exercise it is. Figure 16 depicts the convolutional neural network (CNN) with three hidden layers, which was chosen as the model type for this task. Because of this, the network contains an input layer (156 neurons), three fully connected hidden convolutional layers, and a SoftMax classification output layer following globalaveragepooling1D. (5neurons). The first convolution employs two filters and a core of size 15, the second employs one hundred filters and a core of size ten, and the third employs eight filters and a core of size two.

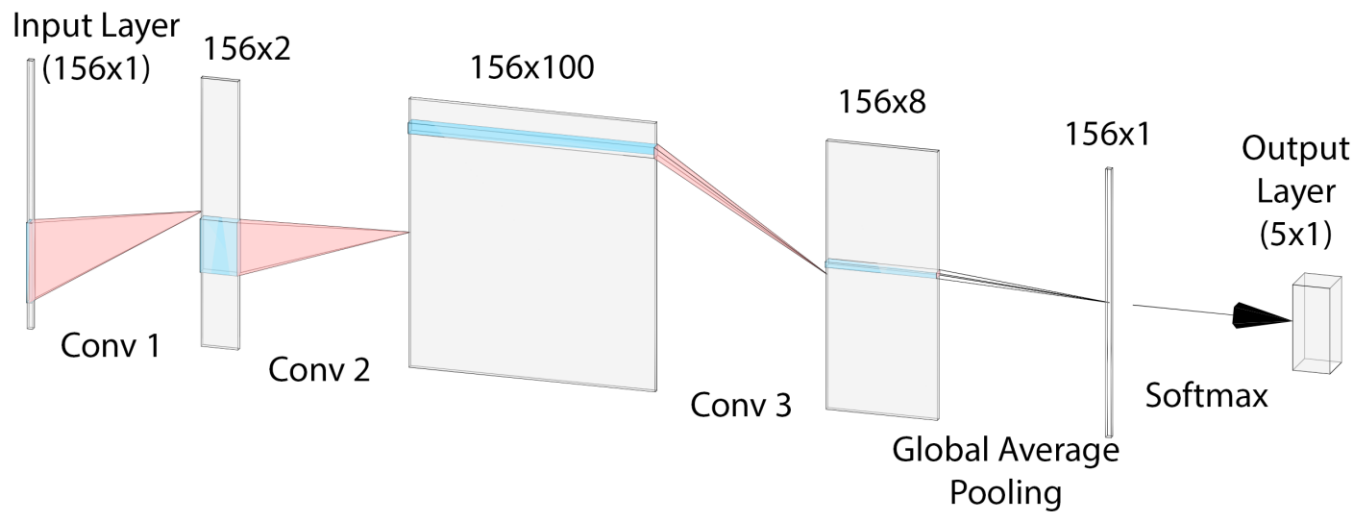


Figure 11 The CNN architecture utilized for recognizing workout types [13]

2.2.2.4 Results Evaluation

The recognition result of the training type, on the other hand, typically implies that the recognition result is satisfactory. Based on the results of Figure 17, the overall accuracy of the

Training is 90.6 percent, the overall accuracy of the test set is 89.9 percent, and the final loss of the verification set and the test data is 0.206 percent.

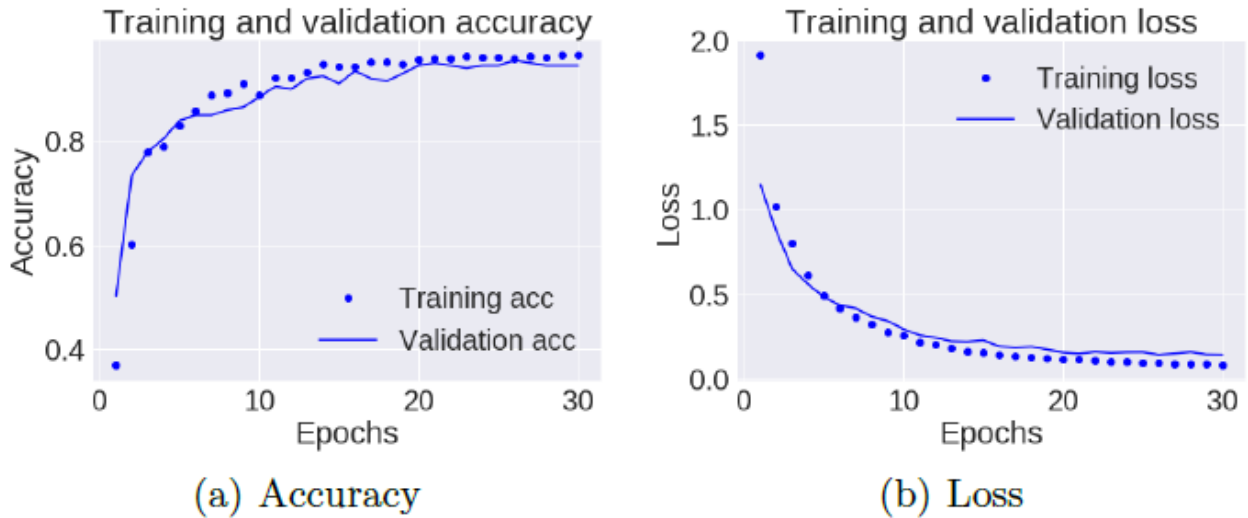


Figure 12 Accuracy (a) and loss (b) over training and validation data [11]

The CNN model is trained using training data and a batch size of 512, and it is used to predict the future. The check data is used to assess training progress, and training is stopped when the check set accuracy no longer increases, within a modest tolerance, for four consecutive epochs, as determined by the check data.

2.2.3 Research 3: Qualitative Activity Recognition of Weight Lifting Exercises

07 March 2013

2.2.3.1 Strategy & Structure

The purpose of this study is to examine the possibility of automatically analyzing the quality of weight lifting exercise execution and the influence of delivering real-time feedback to the athlete - also known as qualitative activity recognition - on the performance of the athlete. This paper focuses on three aspects of qualitative activity recognition that we believe are essential components of any qualitative activity recognition system: the problem of specifying correct execution, the automatic and robust detection of execution mistakes, and how to provide feedback on quality of execution to the end user. More particular, we investigate two ways for identifying faults in a computer-aided manner. In order to discover faults, the first step is to use machine learning and pattern recognition algorithms. Second, and as presented in this work, a model-based method is used to compare motion traces captured by ambient sensors to a formal definition of what defines proper execution. Specifically, this work makes three contributions: (1) a formalization of the term "quality" in the context of activity recognition; (3) the design and implementation of a novel framework for the development of qualitative activity recognition systems; and (7) the evaluation of a system developed with the framework in a user study on the example problem of assessing the quality of execution of weight lifting exercises.

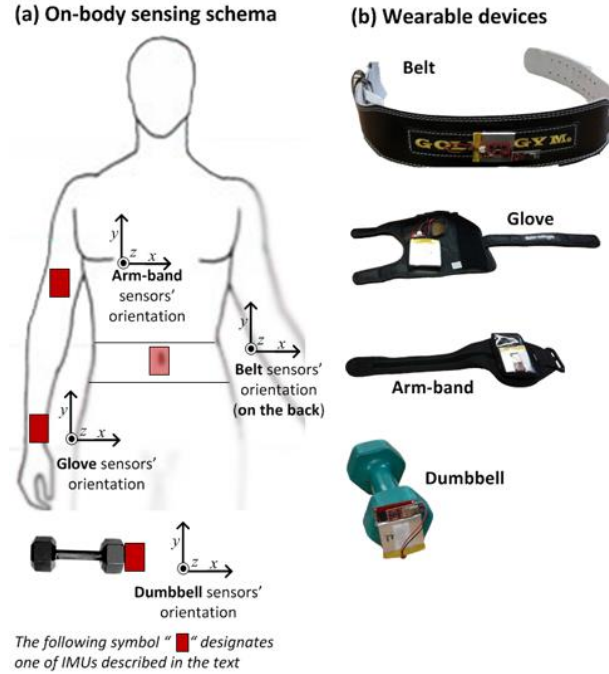


Figure 13 sensing setup [12]

2.2.3.2 Data

Approximate 1800 instances per dataset, derived from 39.200 readings on the IMUs), the number of subjects (6 young men), and the difficulty in distinguishing between different variations of the same exercise, which is a challenge in Qualitative Assessment Activities (Qualitative Assessment Activities). Because of the nature of this technique, it is necessary to collect a large amount of data from numerous individuals in order to arrive at an outcome that can be extended to a new user without the need for further training of the classifier [12].

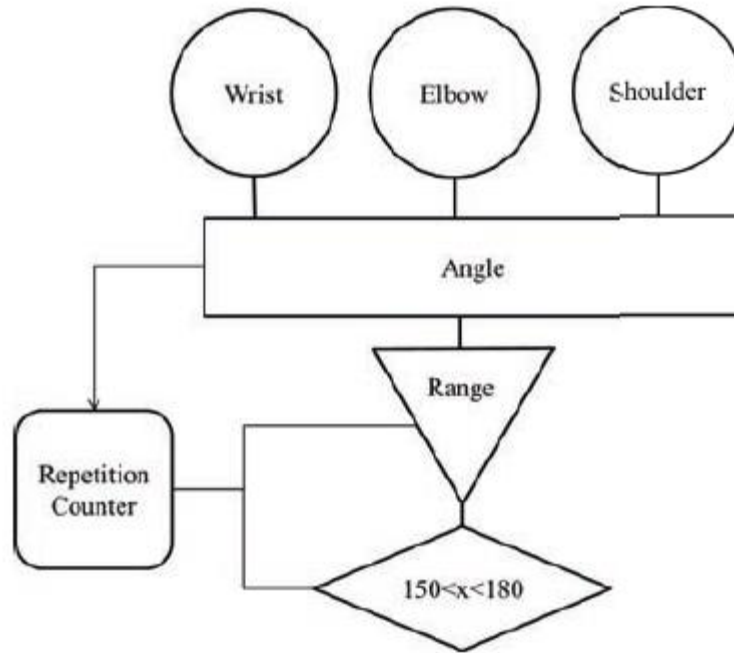


Figure 14 Repetition counted using joint position coordinates. At the end of each iteration, we verify the total range [12].

2.2.3.3 Method Evaluation

Due of the sensor data's inherent noise, we employed a Random Forest [28]. This approach selects a subset of characteristics at random, with the same distribution for each tree in the forest. To increase identification performance, we employed a bagged ensemble of classifiers [6]. We utilized 10 random woods, each with 10 trees. Each window size had 0.5s overlap, and the classifier was evaluated with 10-fold cross-validation (except the window with 0.5s). The optimal window size for this job was 2.5s, and the total recognition rate was 98.03 percent (see Table 1). The table displays the 10-fold cross-validation averaged false positive rate (FPR), precision, recall, and AUC for each of the five individuals (5 classes). The detailed accuracy by class was: (A) 97.6%, (B) 97.36%, (C) 98.2%, (D) 98.16%, (E) 99.1%. (98.2 percent weighted average).

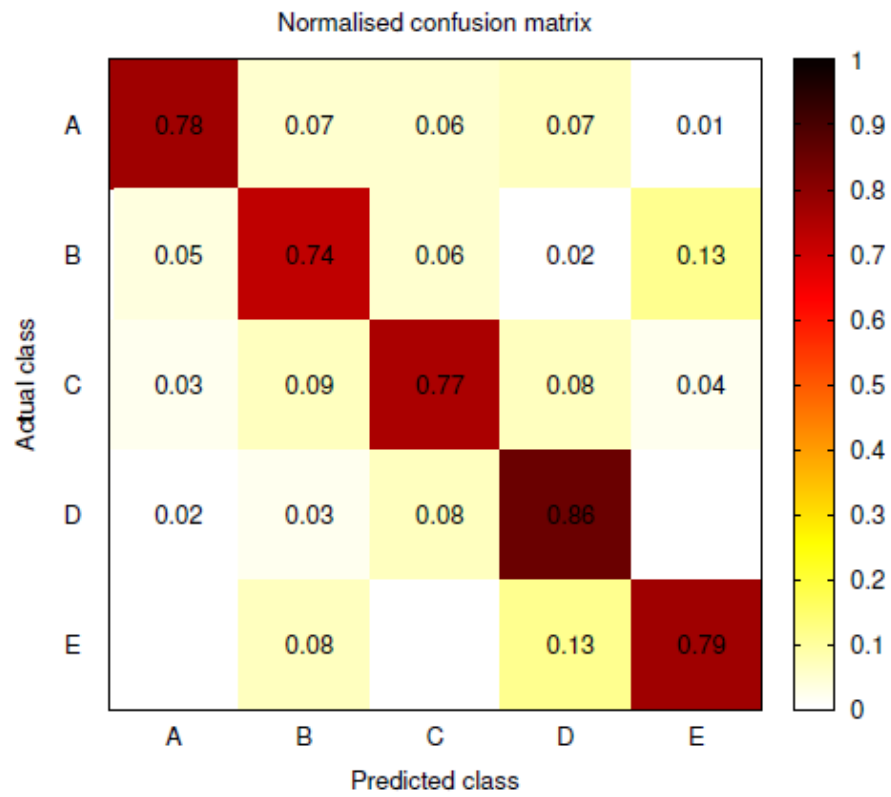


Figure 15 Averaged over all individuals and normalized across ground truth rows [12].

Chapter 3: Material and Methods

3.1 Materials

Here, we will detail the data, tools, programming languages, and computer environments that were utilized and were crucial in the development of this model, as well as the contexts in which the project was implemented.

3.1.1 Data

The dataset utilized was separated into four categories: training, assessing, testing, and validation, in order to get the best potential results. Deep-learning algorithms are being developed for analyzing different movements in videos and images, as well as understanding the appropriate set of movements in various types of exercises. They are being developed for evaluation in academic researches of deep-learning algorithms for analyzing different movements in videos and images and understanding the appropriate set of movements in various types of exercises. Images depicting hundreds, maybe thousands, of various categories and kinds of activities and exercises are included in the collection.

3.1.2 Tools

Python: an open-source programming language that helps developers carry out projects effectively.

Google colab: is a platform for applying machine learning and deep learning models.

3.1.3 Environment

CPU: 8th gen intel core i7 processor with 4 cores, 8 logical processors.

RAM: 8GB ram.

GPU: Local GPU Nvidia GeForce GTX 1660 TI or a cloud-based resources (GPU) like Google Colab.

3.2 Methods

- In this section we will mention & explain our solution methodology + used approach(algorithms)

3.2.1 System architecture Overview

There are three steps to the method that has been designed. It is responsible for reading the data, preparing it, and preprocessing it in preparation for the subsequent phases. The second step involves data reshaping in order to get a fixed form.

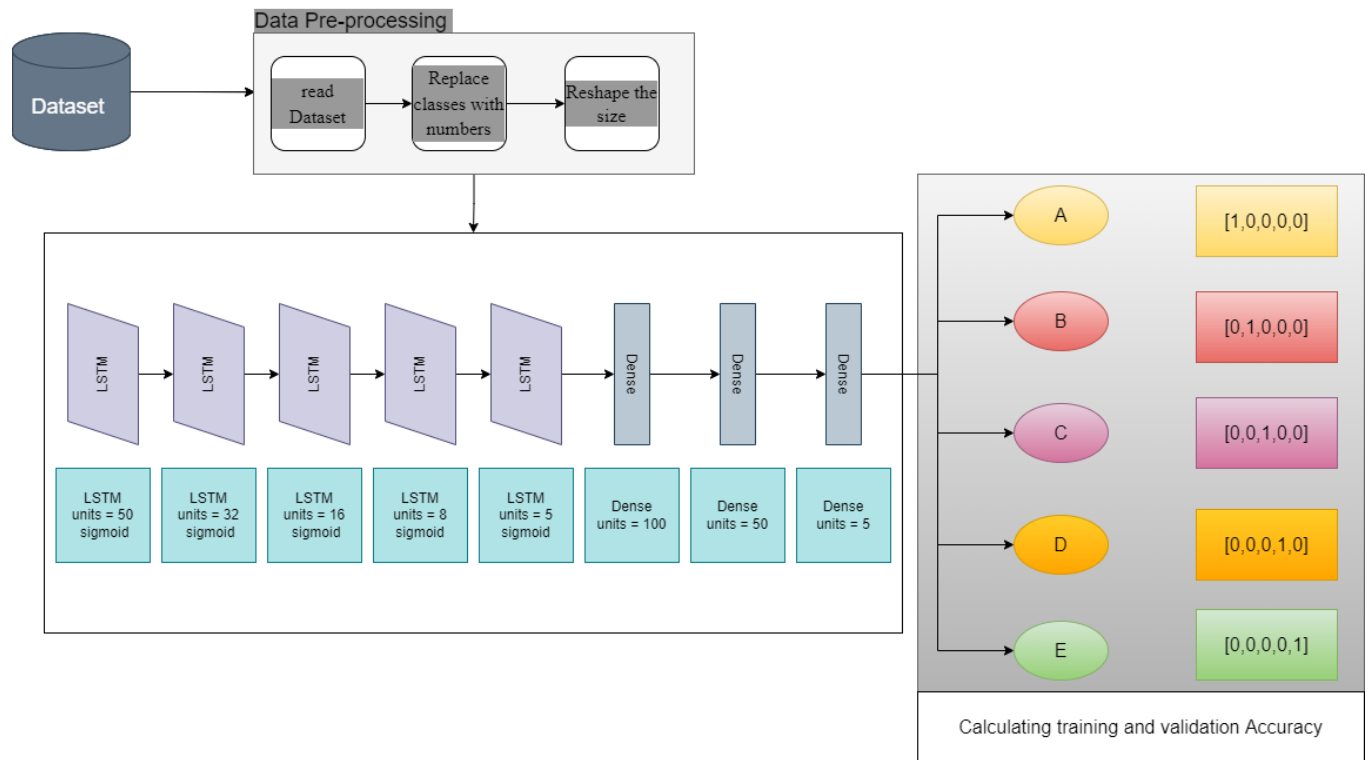


Figure 16 system architecture

The accuracy of the model prediction may be determined at the third step, once the model has produced the findings.

In the model first stage I read the dataset and do some preprocessing on it before the training my model starts with 5 LSTM blocks and 3 Dense Blocks each block contain different hyper parameters first layer contains 50 unit and a sigmoid as an activation function and then the output pass through the dense layers and the output is on of the following classes.

Chapter 4: System Implementation

4.1 System Development

- The system was built in many steps, the first of which was to comprehend the dataset, which included records for certain trainees and characteristics. The second phase was to implement the system.
- As an additional measure of accuracy, we used some machine learning algorithms, including (but not limited to) Random Forest, SVM, K-Nearest Neighbors' Classifier, Gaussian Nave Bayes, Adaboost Classifier, Extra Tree classifier, Bagging classification, Gradient Boosting Classifier, and Decision Tree classification, to test the algorithms.
- After that, I cleaned up the data by deleting any Null values and empty columns, and replaced the names of the trainees with numbers, and the classes of each record with an array representing each of the courses.
- After that, I'll start by inserting the records' features into an LSTM model using Keras to see if it can read and recognize the records' features. We'll use (19622,57) for the X and (19622,5) for the Y shapes, and we'll start by training on the dataset. To ensure that layers can read all of the features for each record, I developed a deep learning LSTM model to check the prediction accuracy.

- During the development process, we employed a variety of deep learning algorithms to run the model, including LSTM, Keras, and Colab.

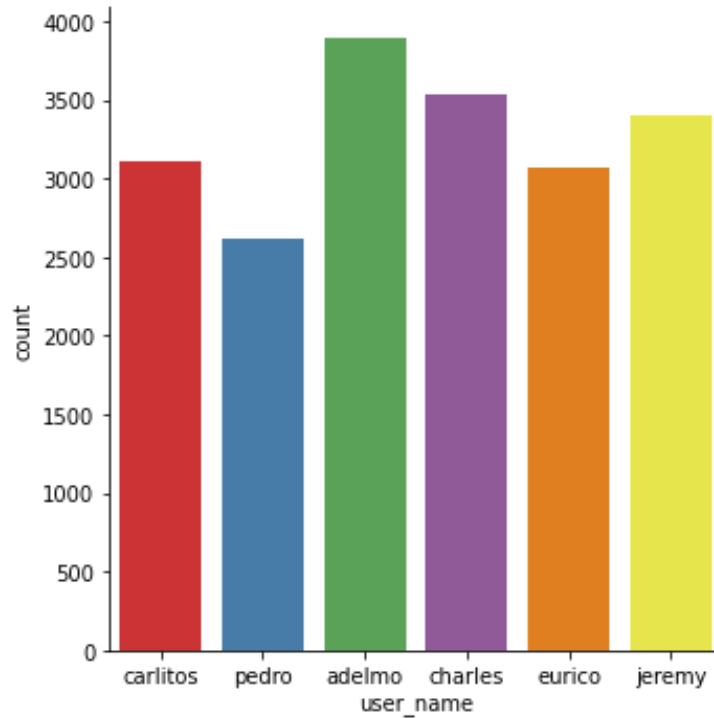


Figure 17 trainers records

Fig.17 shows the total records for each trainer doesn't matter what class he belongs to every trainer has records in all the classes this graph shows the sum of all of these records and display them in bar graph.

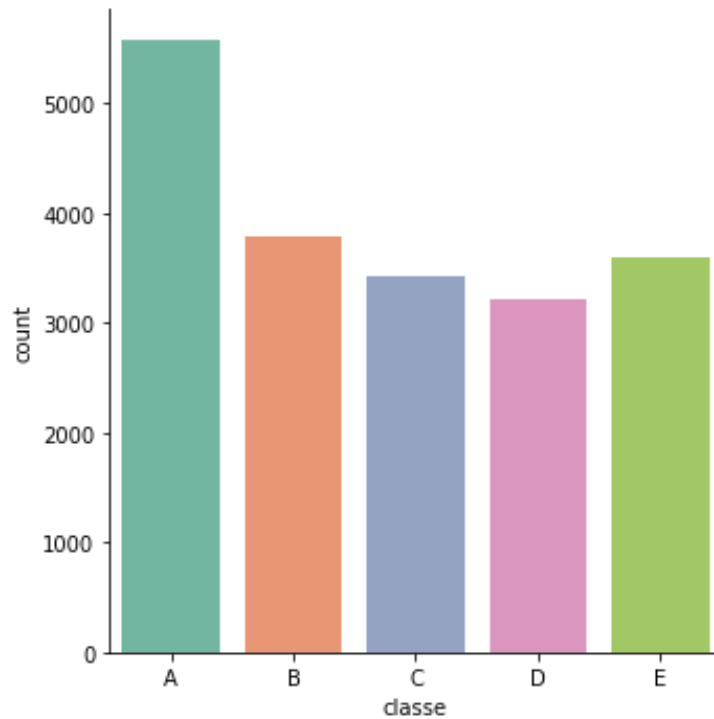


Figure 18 classes records

Fig.18 shows the total records for each Class doesn't matter what Trainer it belongs to every Class has records With all the trainers this graph shows the sum of all of these records and display them in bar graph.

4.2 System Structure

It is my responsibility to provide an overview of the final system we developed, including how data flows through the system and how each stage is responsible for what action to take. Additionally, in the subsections of this section, we will attempt to demonstrate the system by using a class diagram for the classes that we have used throughout the project, which will aid us in providing a clear overview of the system and demonstrating how data flows through the system.

4.2.1 System Overview

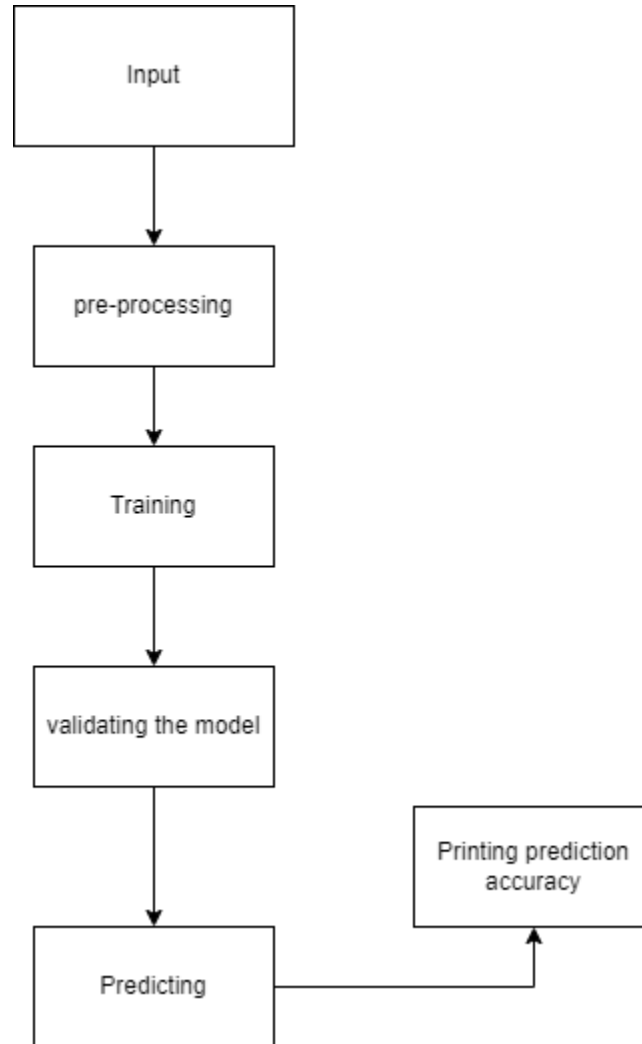


Figure 19 system structure

There are six steps to the method that has been designed. It is responsible for reading the data, preparing it, and preprocessing it in preparation for the subsequent phases. The second step involves data reshaping in order to get a fixed form. The accuracy of the model prediction may be determined at the third step, once the model has produced the findings.

4.2.2 Tensor Board

It is in this section that we will demonstrate the full structure of the LSTM by means of the tensor board, which will depict the LSTM layers, drop out layers, densest layers, and activation function.

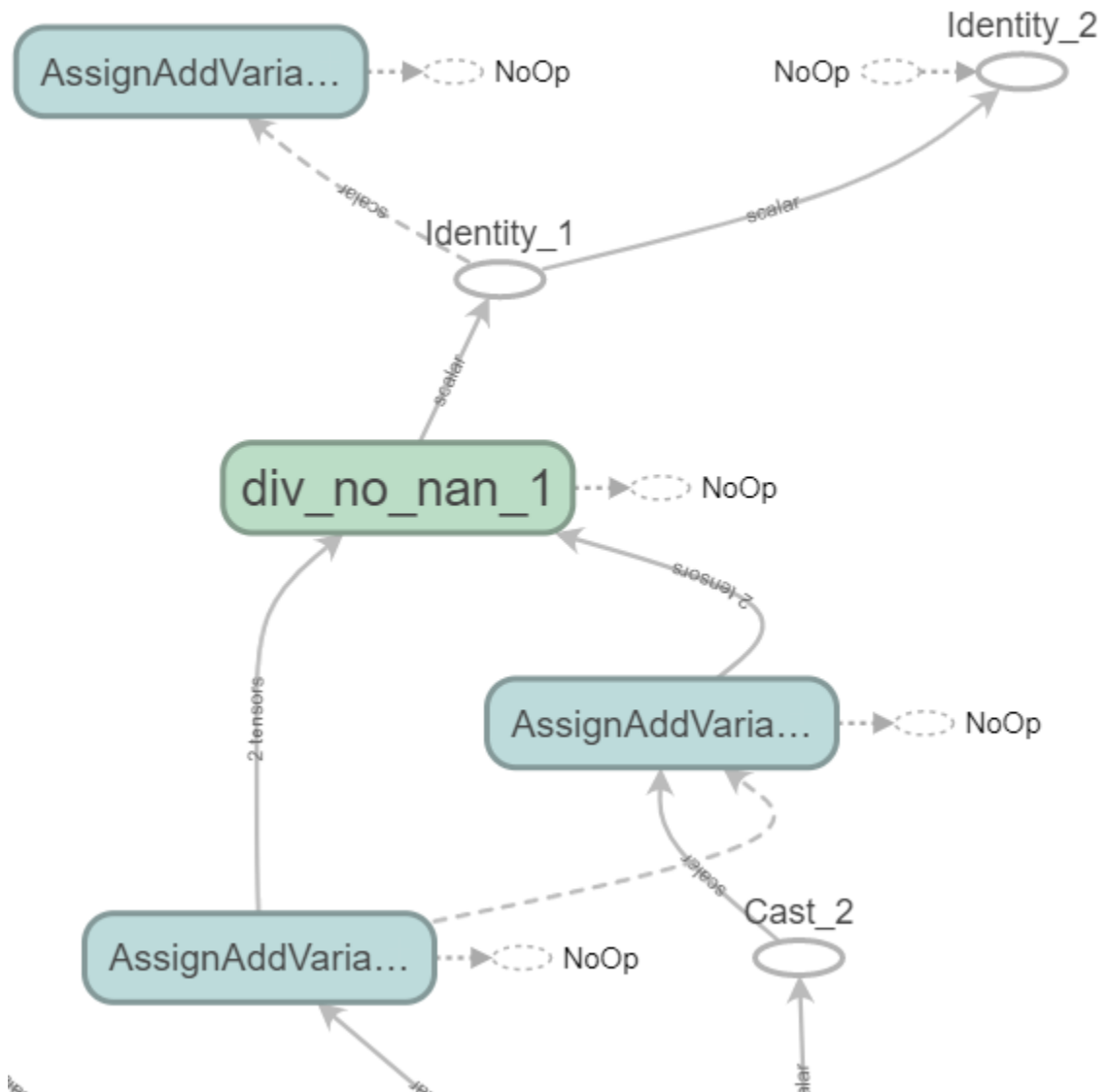


Figure 20 first section of tensor board

The second section of the tensor board depicts the model's flow in more detail.

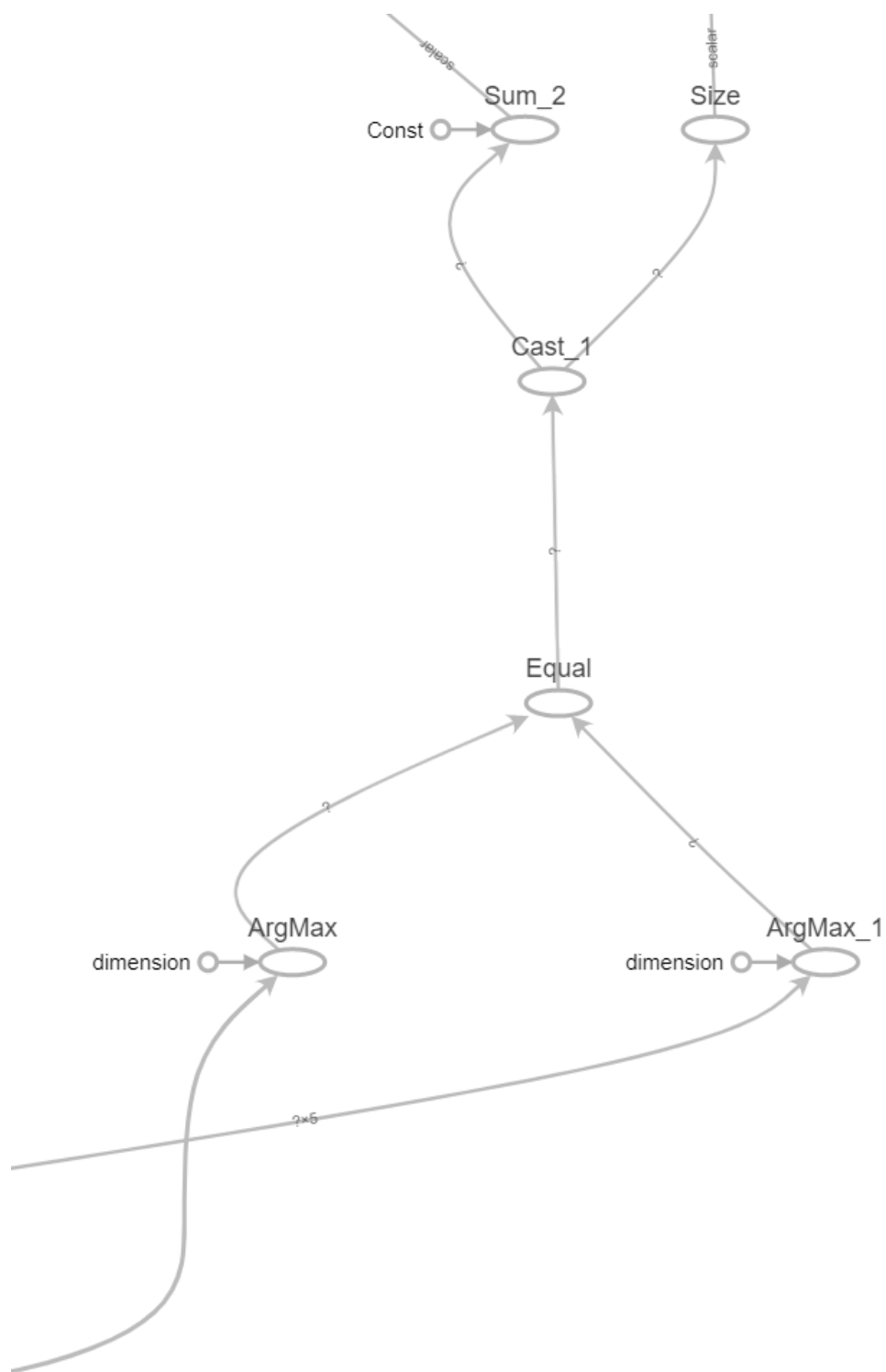


Figure 21 second section of tensor board

The third section of the tensor board depicts the flow of the model in more detail.

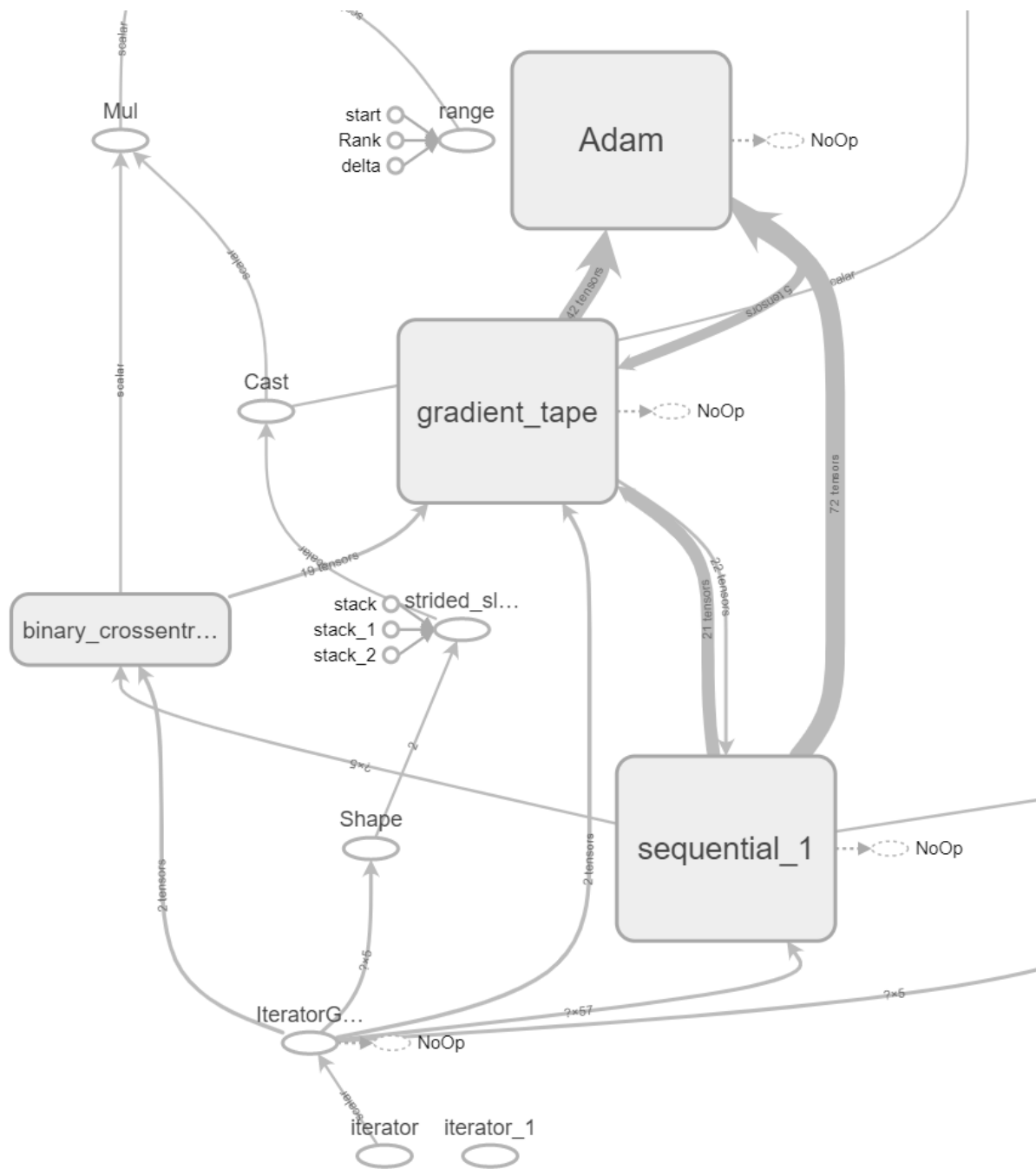


Figure 22 third section of tensor board

The fourth section of the tensor board depicts the model's flow in more detail.

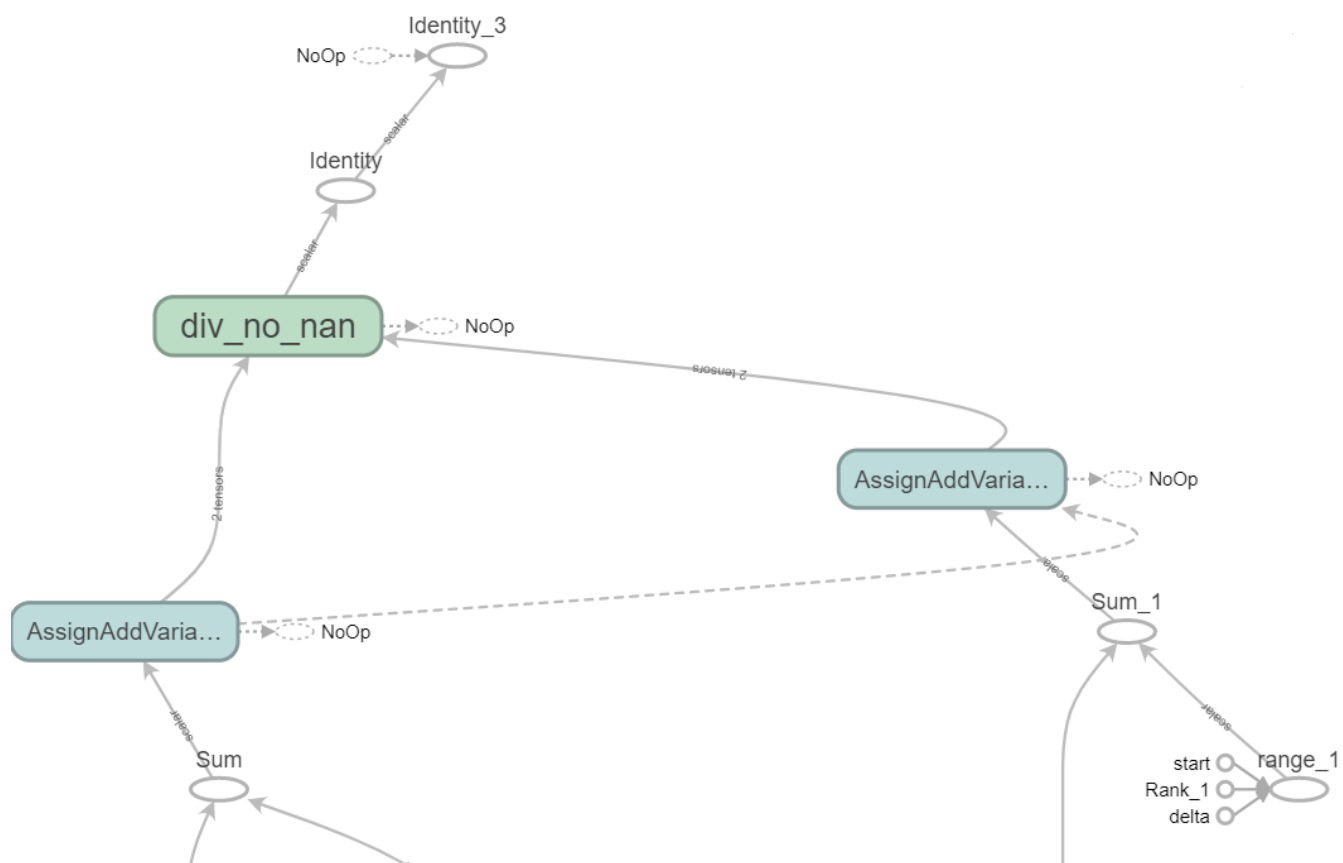


Figure 23 forth section of tensor board

The last section of the tensor board illustrates the model's flow in more detail.

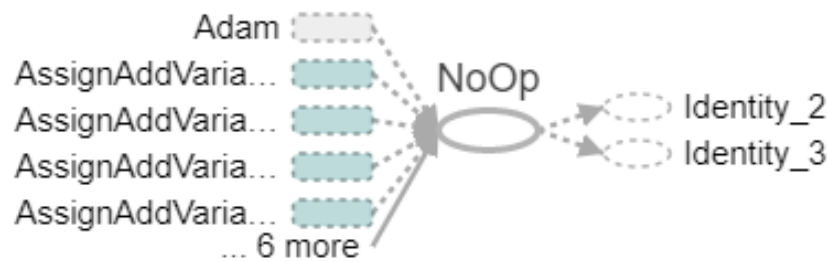


Figure 24 Last section of tensor board

4.3 System Running

In this section, we will provide the input-output relationships for each component of our final system.

4.3.1 Reading and Preprocessing the Dataset

The dataset excel file contains records of trainers who have been fitted with three separate sensors: a Gyroscope, an Accelerometer, and a Magnetometer. Each sensor has an X, Y, and Z axis as well as other features.

0	1	2	3	4	5	50	51	52	53	54	55	56	57	58	
0	1	1	1323084231	788290	0	11	0	-0.02	192	203	-215	-17	654	476	A
1	2	1	1323084231	808298	0	11	0	-0.02	192	203	-216	-18	661	473	A
2	3	1	1323084231	820366	0	11	-0.02	0	196	204	-213	-18	658	469	A
3	4	1	1323084232	120339	0	12	-0.02	0	189	206	-214	-16	658	469	A
4	5	1	1323084232	196328	0	12	0	-0.02	189	206	-214	-17	655	473	A

Figure 25 data before any processes

It is necessary to replace the Classes with numbers in order to confirm this data, and then I changed the Classes into a single hot coded as follows: Class A is [1.0.0.0.0], Class B is [0.1.0.0.0]; Class C is [0.0.1.0.0]; Class D is [0.0.0.1.0]; and Class E is [0.0.0.0.1]. The X size is (19622,57), and the Y shape is (19622,5). Finally, after doing a multilabel classification on the dataset, I divide it into X and Y categories.

	user_name	raw_timestamp_part_1	...	magnet_forearm_y	magnet_forearm_z
12519	5	1322673116	...	726	824
2858	0	1323084250	...	733	539
9142	1	1323095008	...	724	726
17329	3	1322837954	...	-486	-45
3558	5	1322673054	...	318	631
...
10955	5	1322673107	...	378	600
17289	3	1322837951	...	-565	-60
5192	5	1322673049	...	558	676
12172	3	1322837893	...	630	581
235	1	1323094971	...	763	730

[17659 rows x 57 columns]

Figure 26 Created X_train

```
[17659 rows x 57 columns]
12519    C
2858     A
9142     B
17329    E
3558     A
      ..
10955    C
17289    E
5192     A
12172    C
235      A
Name: classe, Length: 17659, dtype: object
```

Figure 27 Created y_train

```
Index(['Unnamed: 0', 'user_name', 'raw_timestamp_part_1',
      'raw_timestamp_part_2', 'cvt_d_timestamp', 'new_window', 'num_window',
      'roll_belt', 'pitch_belt', 'yaw_belt', 'total_accel_belt',
      'gyros_belt_x', 'gyros_belt_y', 'gyros_belt_z', 'accel_belt_x',
      'accel_belt_y', 'accel_belt_z', 'magnet_belt_x', 'magnet_belt_y',
      'magnet_belt_z', 'roll_arm', 'pitch_arm', 'yaw_arm', 'total_accel_arm',
      'gyros_arm_x', 'gyros_arm_y', 'gyros_arm_z', 'accel_arm_x',
      'accel_arm_y', 'accel_arm_z', 'magnet_arm_x', 'magnet_arm_y',
      'magnet_arm_z', 'roll_dumbbell', 'pitch_dumbbell', 'yaw_dumbbell',
      'total_accel_dumbbell', 'gyros_dumbbell_x', 'gyros_dumbbell_y',
      'gyros_dumbbell_z', 'accel_dumbbell_x', 'accel_dumbbell_y',
      'accel_dumbbell_z', 'magnet_dumbbell_x', 'magnet_dumbbell_y',
      'magnet_dumbbell_z', 'roll_forearm', 'pitch_forearm', 'yaw_forearm',
      'total_accel_forearm', 'gyros_forearm_x', 'gyros_forearm_y',
      'gyros_forearm_z', 'accel_forearm_x', 'accel_forearm_y',
      'accel_forearm_z', 'magnet_forearm_x', 'magnet_forearm_y',
      'magnet_forearm_z', 'classe'],
      dtype='object')
```

Figure 28 Dataset Features

4.3.2 Training and Validating Dataset

First and foremost, I needed to route the data via five layers of LSTM. Second. Secondly, it runs through 3 Dense Layers with Sigmoid Activation function before being finished. at the end, in the training I used shuffling to shuffle the data with splitting 30% testing to 70% training

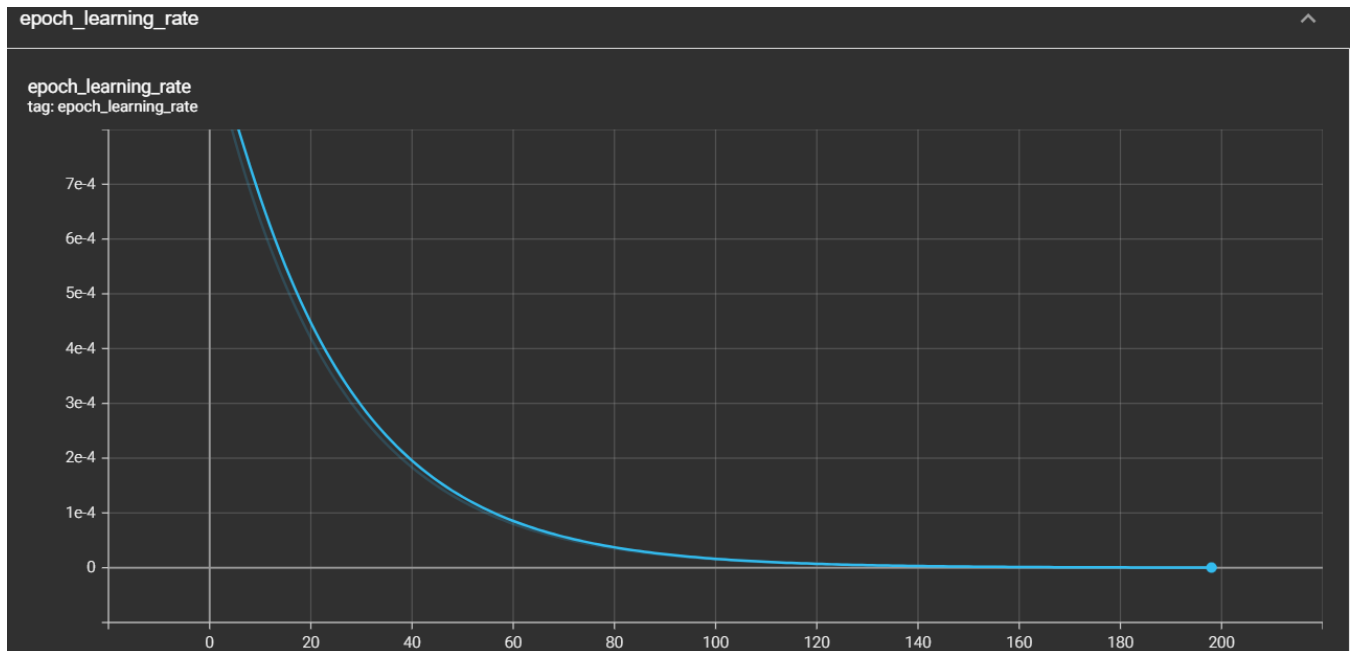


Figure 29 Epoch learning rate

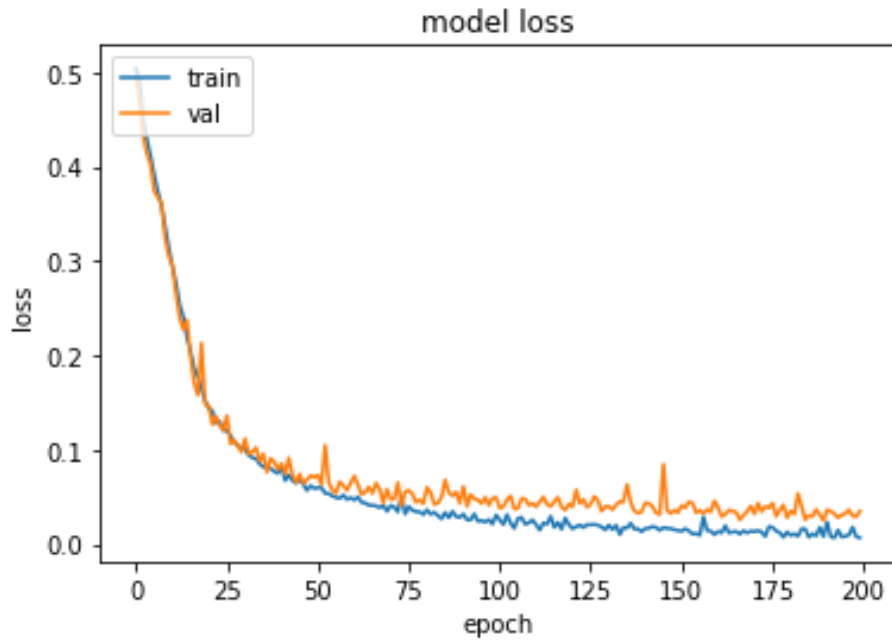


Figure 30 Epoch Loss

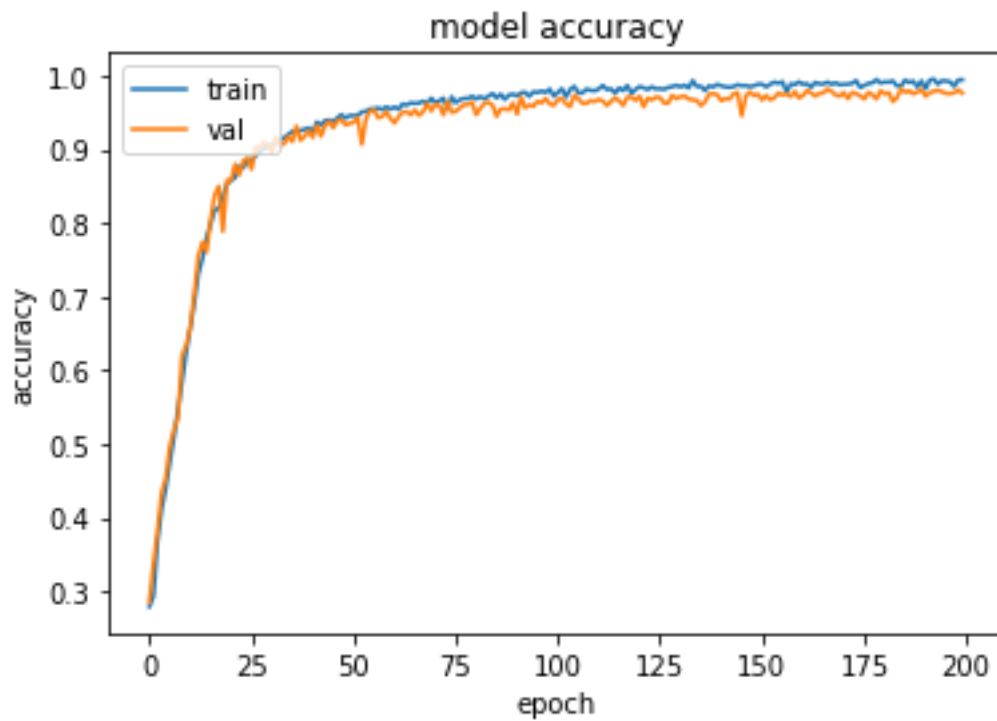


Figure 31 Epoch Accuracy

At the conclusion, we can observe how accurate the model's predictions were. After running 200 epochs, accuracy improved significantly.

Chapter 5: Results and Evaluation

5.1 Testing Methodology

Each record class will be predicted using the LSTM model. The details of how to predict each record class, whether it's A, B, C, D or E will be presented in depth in the following subsections. Class A relates to the specific execution of the exercise indicated in the exercise description. Throwing the elbows to the front is classified as Class B, rising the dumbbell only halfway is classified as Class C, lowering the dumbbell only halfway is classified as Class D, and throwing the hips to the front is classified as Class E [12].

First, the dataset was divided into two subsets: training and validation, and the dataset was fed into the model in accordance with the value of the batch size that was allocated to the dataset. Each of these approaches is assessed and tested independently based on its accuracy and loss in order to determine which preprocessing method is the most appropriate for the current classification challenge. The loss function was selected after many searches to see which one was the most appropriate for the dataset's five classes; binary crossentropy was the most appropriate after I tested Squared hinge, MeanSquarederror, and categorical crossentropy. The binary crossentropy algorithm has the lowest loss percentage of all the algorithms. When it comes to validation, I attempted to validate using a 40 % test size to 60% train and 30% test size to 70% train size split in order to get the most accuracy possible for the model. I also utilized sigmoid as an activation function rather than relu or any other activation function I tested.

5.2 Results

In this section, we will provide an overview about the results statistics with the aid of graphs and calculated Equations.

5.2.1 Best Results Cases

Perfect result happened after running 200 Epoch with 57 Batch Size Validation Accuracy was 95.79% and the prediction results was 95.78% with 30% test space.

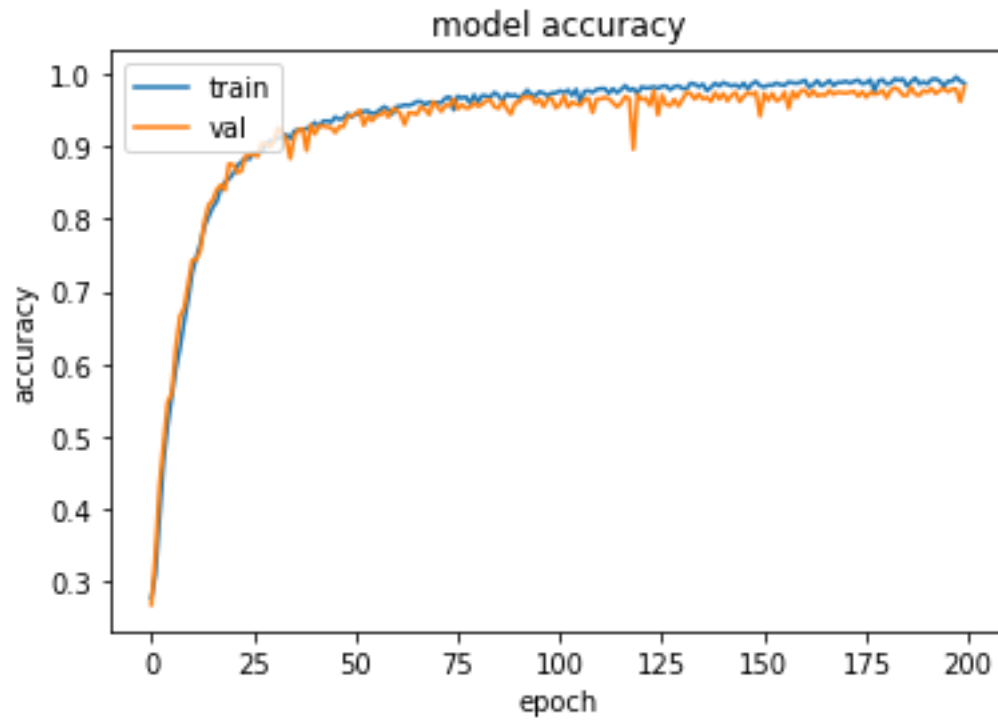


Figure 32 model accuracy best case

At the beginning of the graph, you can see that the model accuracy was too low, about 25%, and that it required 200 epochs to increase it to almost 99% for the training accuracy and roughly 99% for the validation accuracy.

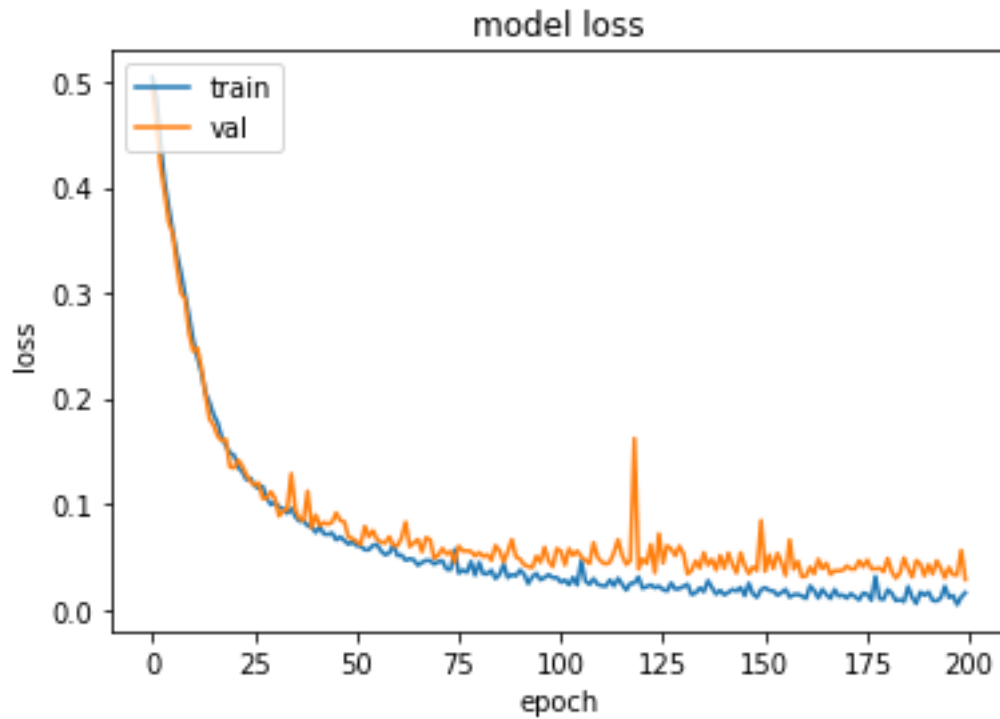


Figure 33 model loss best case

At the beginning of the graph, you can see that the model Loss was high, about 0.5, and that it required 200 epochs to decrease it to almost 0.04 for the training Loss and roughly 0.1 for the validation Loss.

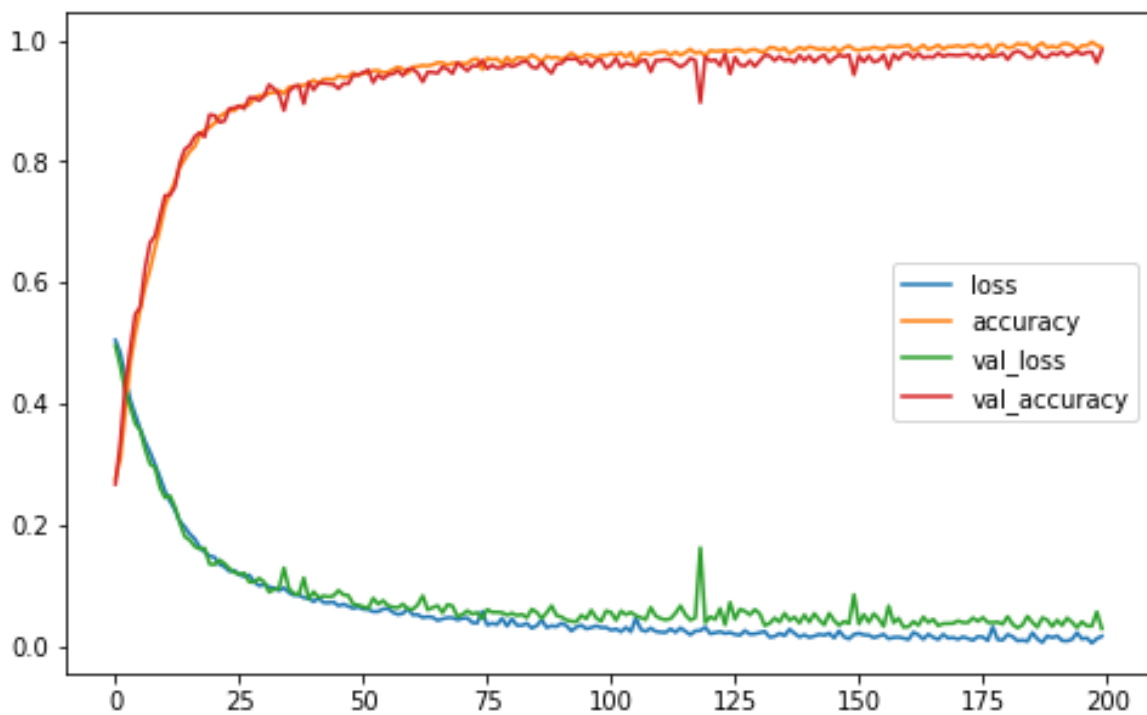


Figure 34 model summary best case

In Fig.34, you can see how the loss is diminishing and how the accuracy is growing as you go through the 200th epoch.

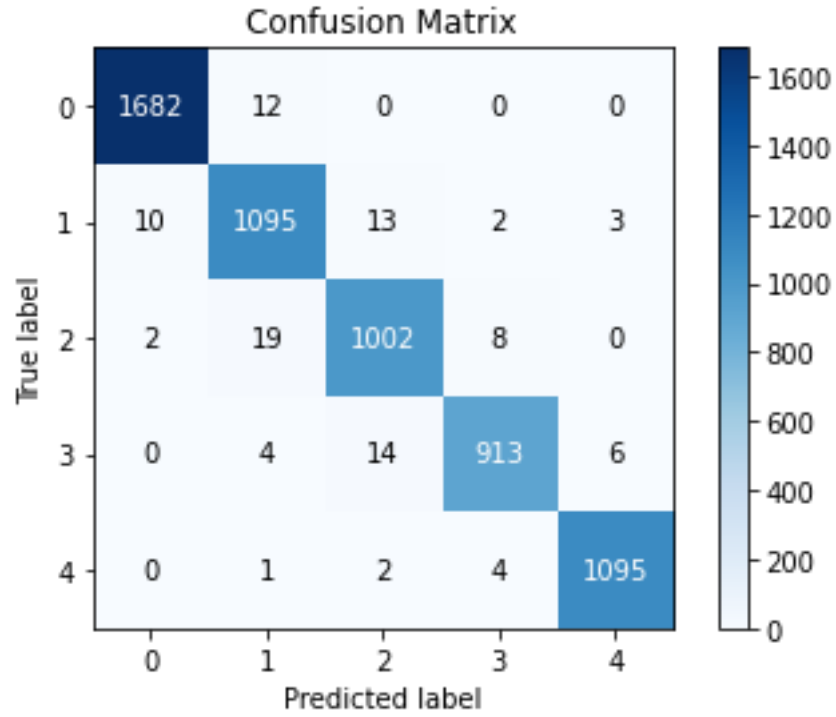


Figure 35 confusion matrix between true and predicted results

Fig.35 shows the confusion matrix for the correct and incorrect predictions

True positive Rate is a result in which the model accurately predicts the positive class.

$$TPR = \frac{TP}{P} = \frac{TP}{TP + FN} \quad (1)$$

A false positive Rate is an outcome in which the model forecasts the positive class inaccurately.

$$FPR = \frac{FP}{N} = \frac{FP}{FP + TN} \quad (2)$$

True negative Rate is a result in which the model accurately predicts the negative class.

$$TNR = \frac{TN}{N} = \frac{TN}{TN + FP} \quad (3)$$

False negative Rate is a result in which the model forecasts the negative class inaccurately.

$$\text{FNR} = \frac{TN}{N} = \frac{TN}{TN + FP} \quad (4)$$

Table 4 classification report for best case

	Precision	Recall	F1-score	support
0	0.99	0.99	0.97	1694
1	0.97	0.97	0.93	1123
2	0.97	0.97	0.94	1031
3	0.98	0.98	0.94	937
4	0.99	0.99	0.99	1102
Accuracy			0.98	5887
Macro avg	0.98	0.98	0.98	5887
Weighted avg	0.98	0.98	0.98	5887

Table 4 shows the precision, recall, f1-score and support for the acceptable case and the 0, 1, 2, 3, 4 represents the classes A, B, C, D and E.

Precision is defined as the ratio of true positives to the total of true and false positives.

Recall is defined as the ratio of true positives to the total of true positives and false negatives.

The F1 score is the weighted harmonic mean of accuracy and recall, calculated as follows: The closer the F1 score is to 1.0, the better the model's projected performance is expected to be.

Support is the number of actual occurrences of a class in a dataset is referred to as support. It does not differ across models; rather, it is used to diagnose the performance assessment process itself.

$$\text{Precision} = \frac{TP}{TP + FN} \quad (1)$$

$$\text{Recall} = \frac{TP}{TP + FP} \quad (2)$$

$$\text{F1} = \frac{2r.p}{r + p} = \frac{2TP}{2TP + FP + FN} \quad (3)$$

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (4)$$

The following figure, Fig. 36 illustrates what happens to the input as it passes through the model.

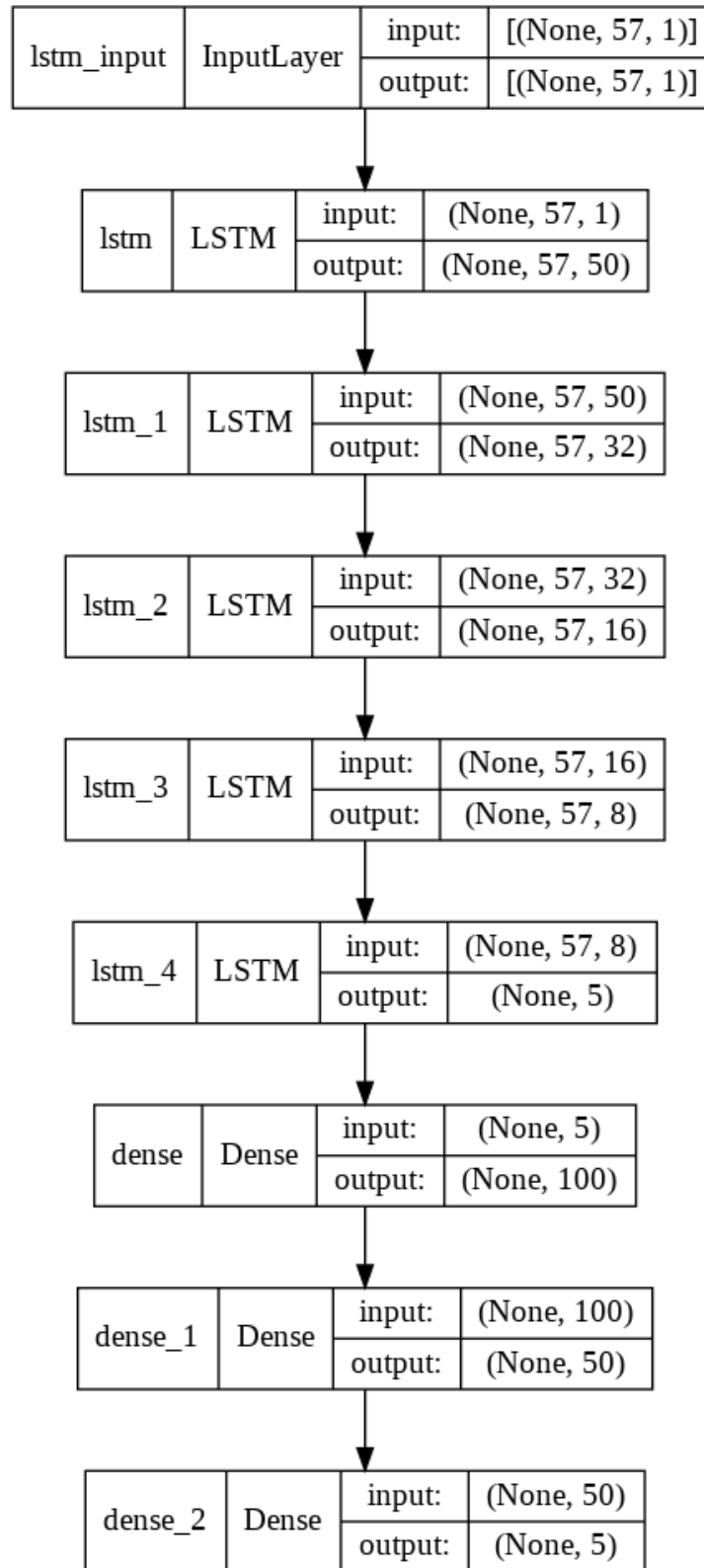


Figure 36 model flow best case

One of the best results happened after running 200 Epoch with 0 Batch Size Validation Accuracy was 93.56% and the prediction results was 93.56% with 30% test space.

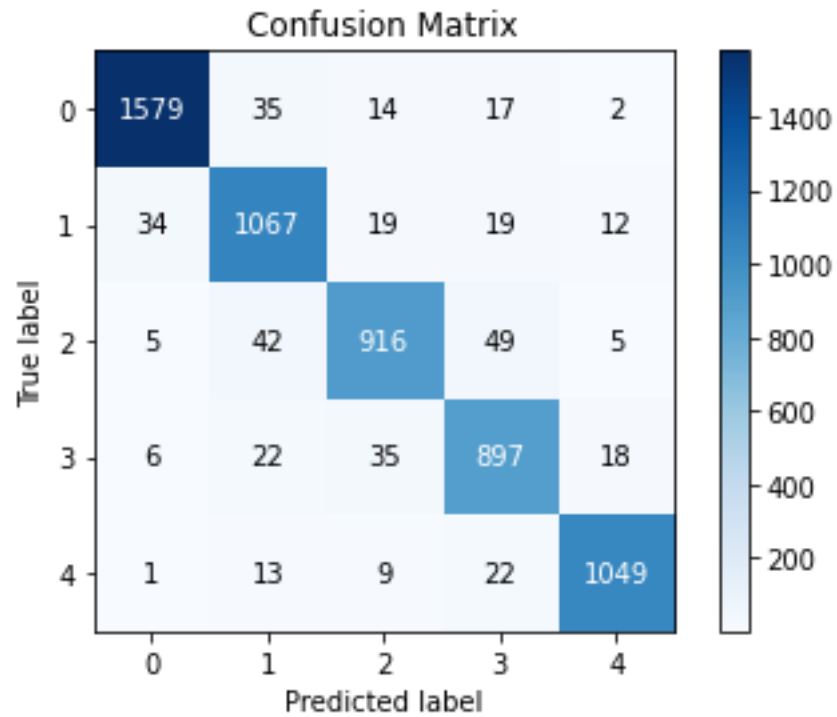


Figure 37 confusion matrix for the second-best result

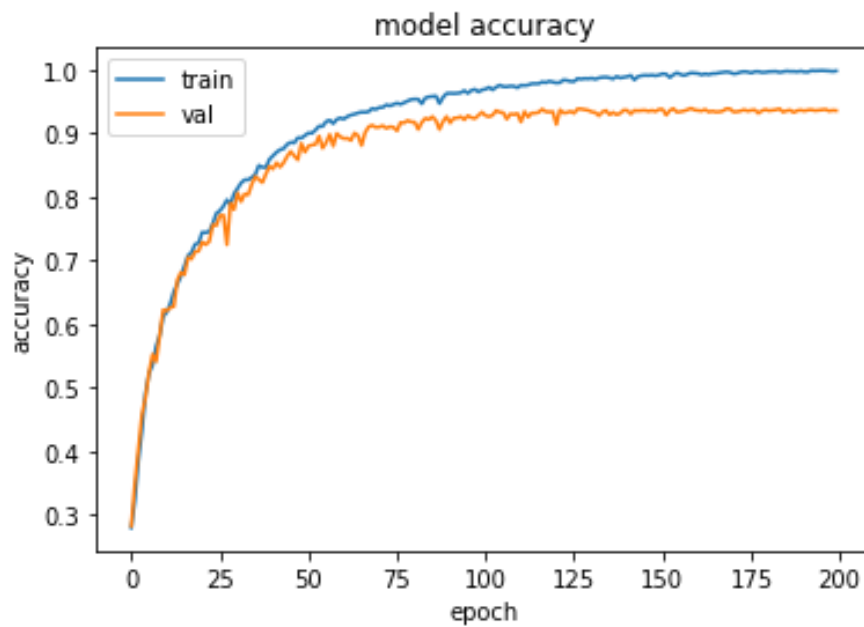


Figure 38 second-best result model accuracy

At the beginning of the graph, you can see that the model accuracy was too low, about 30%, and that it required 200 epochs to increase it to almost 99% for the training accuracy and roughly 95% for the validation accuracy.



Figure 39 second best-result model loss

At the beginning of the graph, you can see that the model Loss was high, about 0.5, and that it required 200 epochs to decrease it to almost 0.03 for the training Loss and roughly 0.25 for the validation Loss.

Table 5 Classification report for second best-result

	Precision	Recall	F1-score	support
0	0.97	0.96	0.97	1647
1	0.91	0.93	0.92	1151
2	0.92	0.90	0.91	1017
3	0.89	0.92	0.91	978
4	0.97	0.96	0.96	1094
Accuracy			0.94	5887
Macro avg	0.93	0.93	0.93	5887
Weighted avg	0.94	0.94	0.94	5887

5.2.3 acceptable Results Cases

Acceptable result happened after running 50 Epoch with 300 Batch Size Validation Accuracy was 89.67% and the prediction results was 89.67% with 30% test space

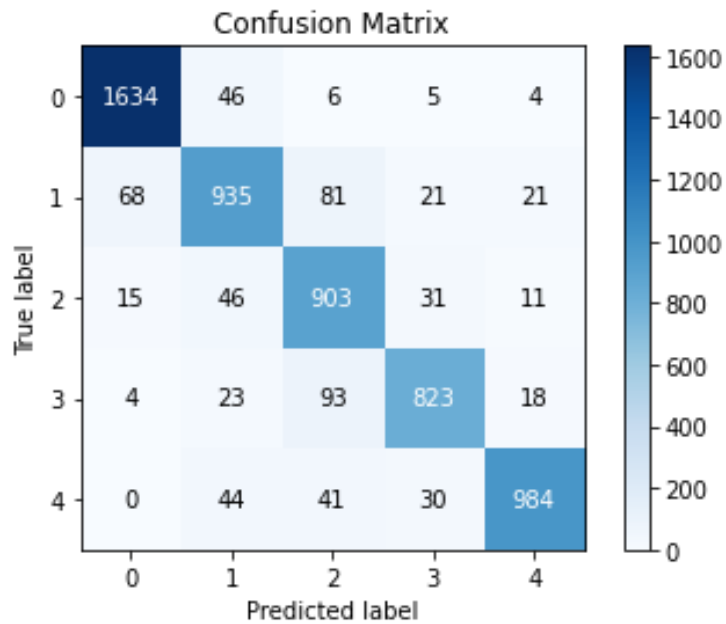


Figure 40 confusion matrix for the acceptable case

Fig.40 shows the confusion matrix for the correct and incorrect predictions

True positive Rate is a result in which the model accurately predicts the positive class.

A false positive Rate is an outcome in which the model forecasts the positive class inaccurately.

True negative Rate is a result in which the model accurately predicts the negative class.

False negative Rate is a result in which the model forecasts the negative class inaccurately.

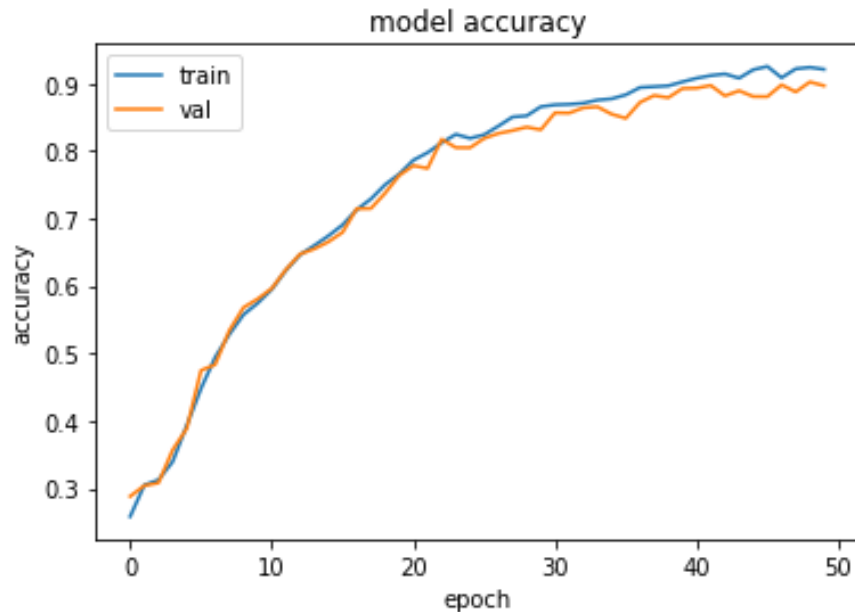


Figure 41 model accuracy for the acceptable case

At the beginning of the graph, you can see that the model accuracy was too low, about 25%, and that it required 50 epochs to increase it to almost 92% for the training accuracy and roughly 90% for the validation accuracy.

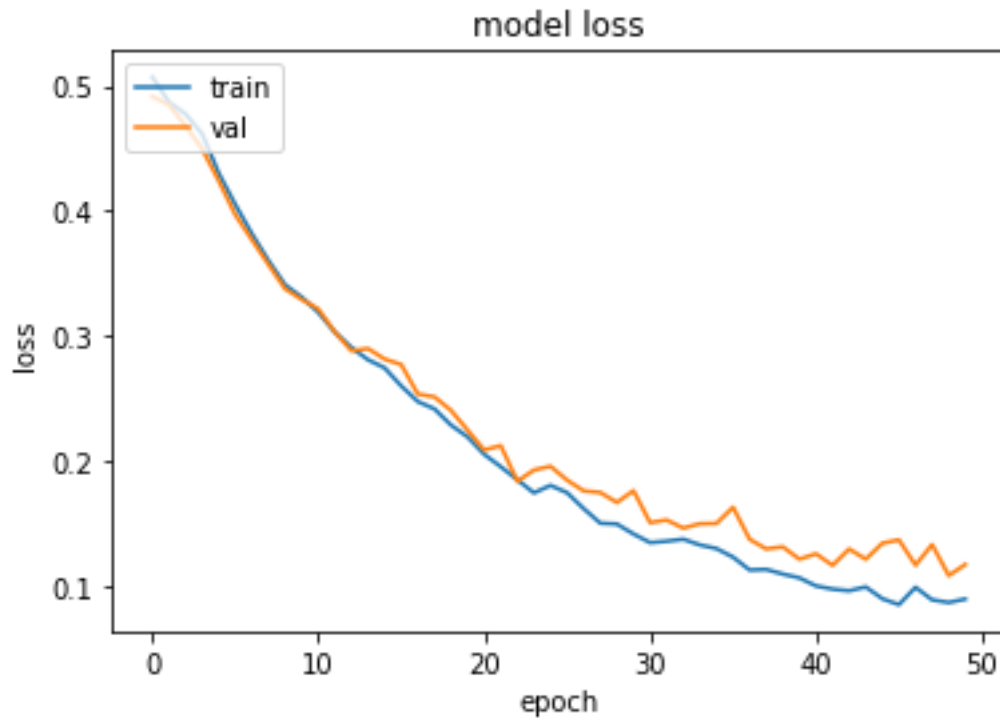


Figure 42 model loss for the acceptable case

At the beginning of the graph, you can see that the model Loss was high, about 0.5, and that it required 100 epochs to decrease it to almost 0.2 for the training Loss and roughly 0.25 for the validation Loss.

Table 6 Classification report for the acceptable case

	Precision	Recall	F1-score	support
0	0.95	0.96	0.96	1695
1	0.85	0.83	0.84	1126
2	0.80	0.90	0.85	1006
3	0.90	0.86	0.88	961
4	0.95	0.90	0.92	1099
Accuracy			0.90	5887
Macro avg	0.89	0.89	0.89	5887
Weighted avg	0.90	0.90	0.90	5887

Acceptable result happened after running 100 Epoch with 300 Batch Size Validation Accuracy was 88.36% and the prediction results was 88.36% with 30% test space

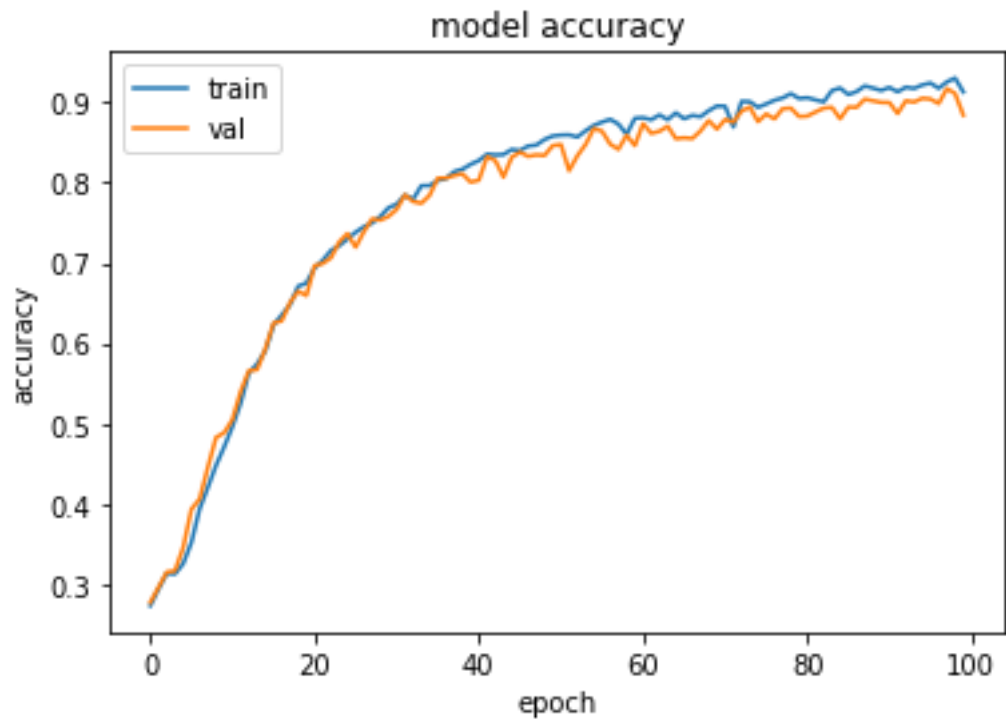


Figure 43 model accuracy acceptable case

At the beginning of the graph, you can see that the model accuracy was too low, about 25%, and that it required 100 epochs to increase it to almost 90% for the training accuracy and roughly 85% for the validation accuracy.

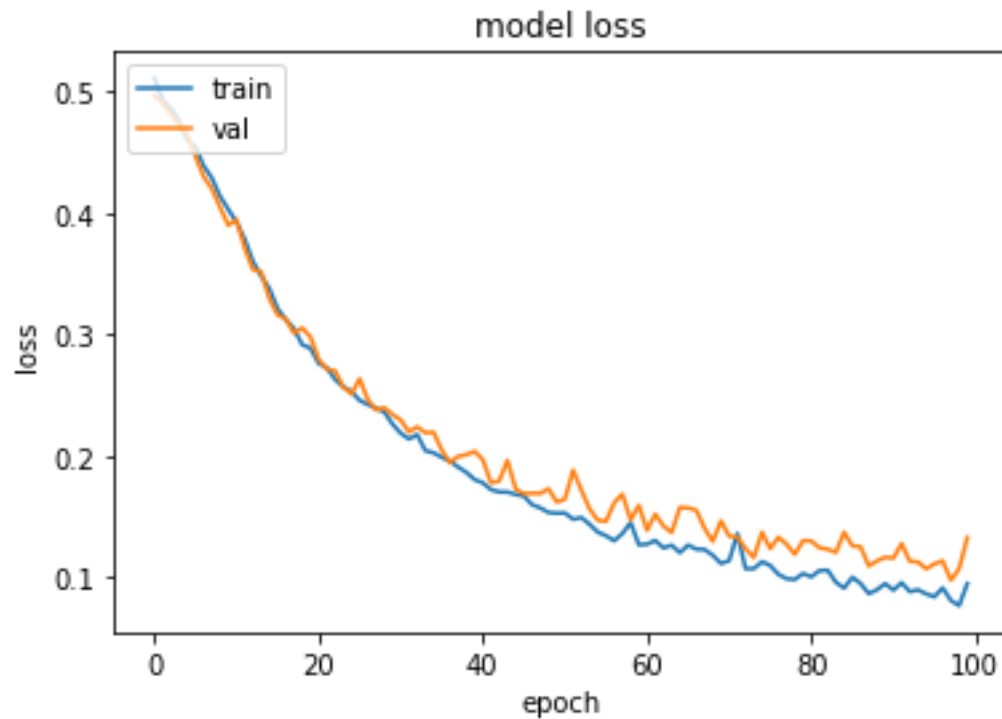


Figure 44 model loss acceptable case

At the beginning of the graph, you can see that the model Loss was high, about 0.5, and that it required 100 epochs to decrease it to almost 0.2 for the training Loss and roughly 0.25 for the validation Loss.

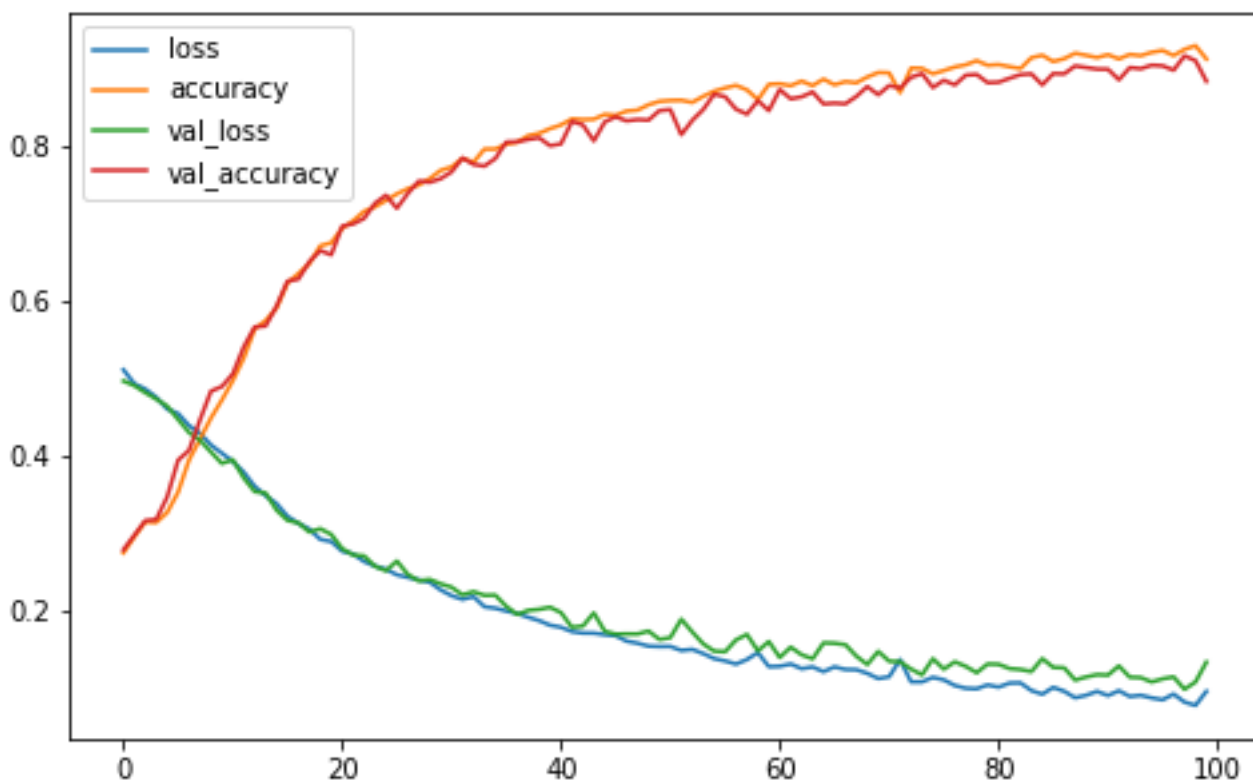


Figure 45 model summary acceptable case

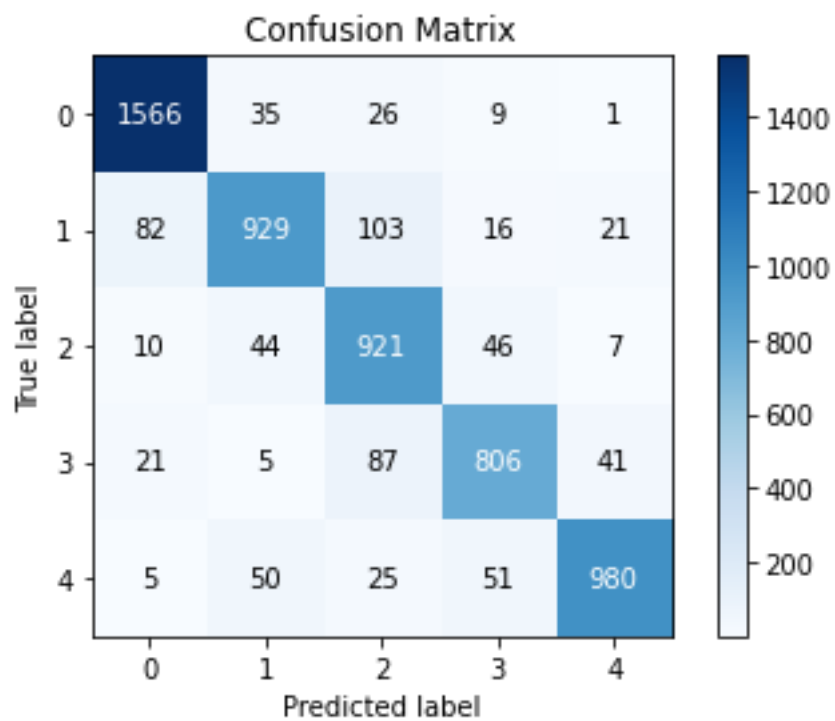


Figure 46 confusion matrix for the acceptable case

Fig.46 shows the confusion matrix for the correct and incorrect predictions

True positive Rate is a result in which the model accurately predicts the positive class.

A false positive Rate is an outcome in which the model forecasts the positive class inaccurately.

True negative Rate is a result in which the model accurately predicts the negative class.

False negative Rate is a result in which the model forecasts the negative class inaccurately.

Table 7 classification report for acceptable case

	Precision	Recall	F1-score	support
0	0.93	0.96	0.94	1637
1	0.87	0.81	0.84	1151
2	0.79	0.90	0.84	1028
3	0.87	0.84	0.85	960
4	0.93	0.88	0.91	1111
Accuracy			0.88	5887
Macro avg	0.88	0.88	0.88	5887
Weighted avg	0.89	0.88	0.88	5887

Table 7 shows the precision, recall, f1-score and support for the acceptable case and the 0, 1, 2, 3, 4 represents the classes A, B, C, D and E.

Precision is defined as the ratio of true positives to the total of true and false positives.

Recall is defined as the ratio of true positives to the total of true positives and false negatives.

The F1 score is the weighted harmonic mean of accuracy and recall, calculated as follows: The closer the F1 score is to 1.0, the better the model's projected performance is expected to be.

Support is the number of actual occurrences of a class in a dataset is referred to as support. It does not differ across models; rather, it is used to diagnose the performance assessment process itself.

The following figure, Fig. 47 illustrates what happens to the input as it passes through the model.

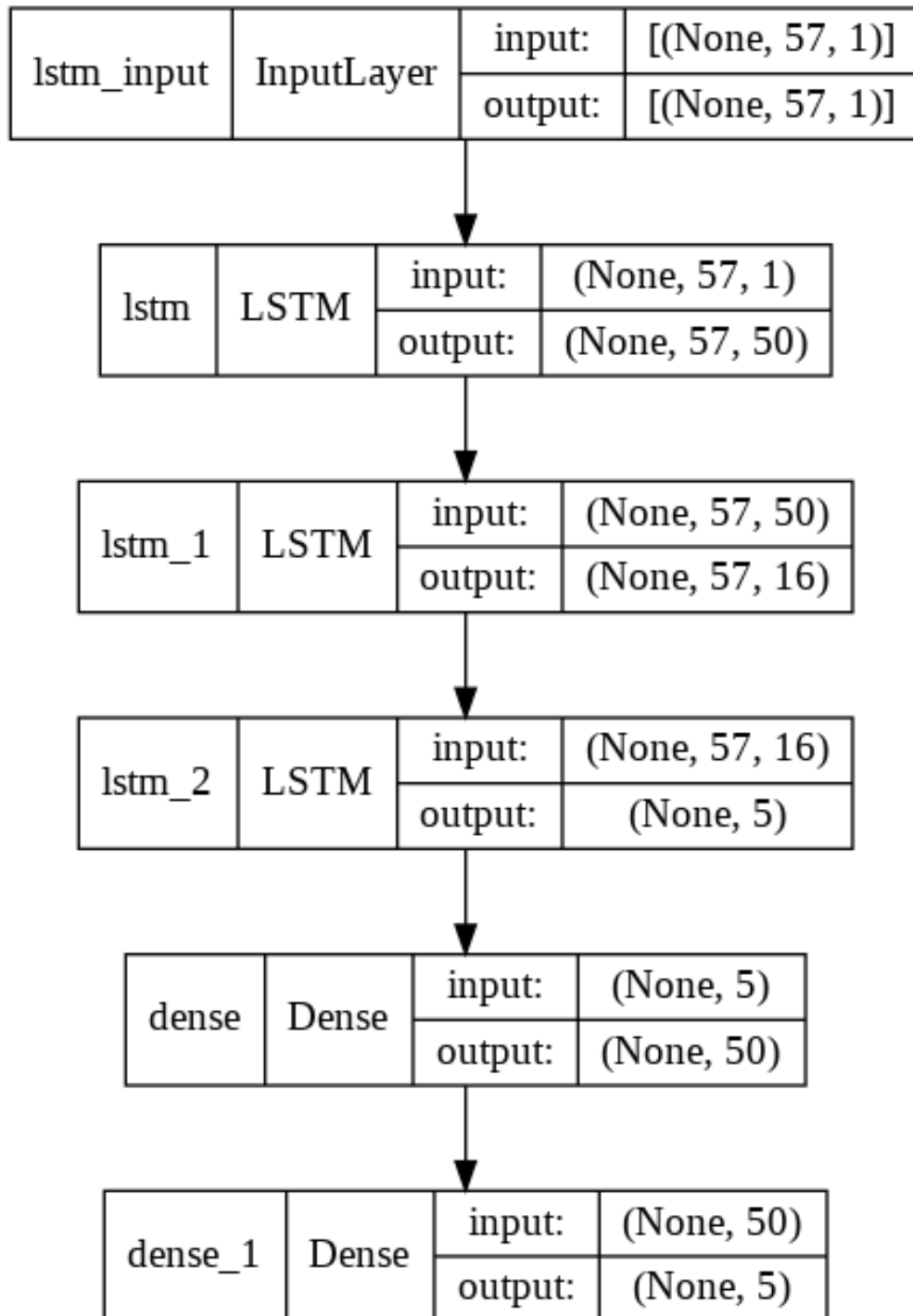


Figure 47 model flow acceptable case

5.2.3 Worst Results Cases

worst result happened after running 100 Epoch with 300 Batch Size Validation Accuracy was 72.85% and the prediction results was 72.85% with 40% test space

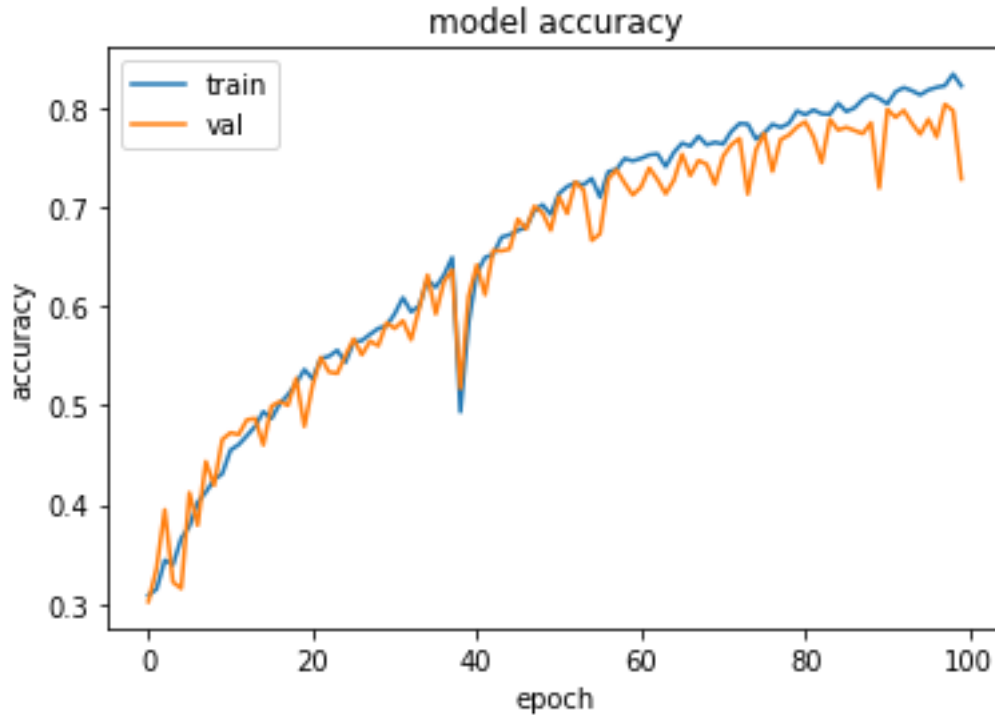


Figure 48 model accuracy worst case

At the beginning of the graph, you can see that the model accuracy was too low, about 30%, and that it required 100 epochs to increase it to almost 80% for the training accuracy and roughly 75% for the validation accuracy.

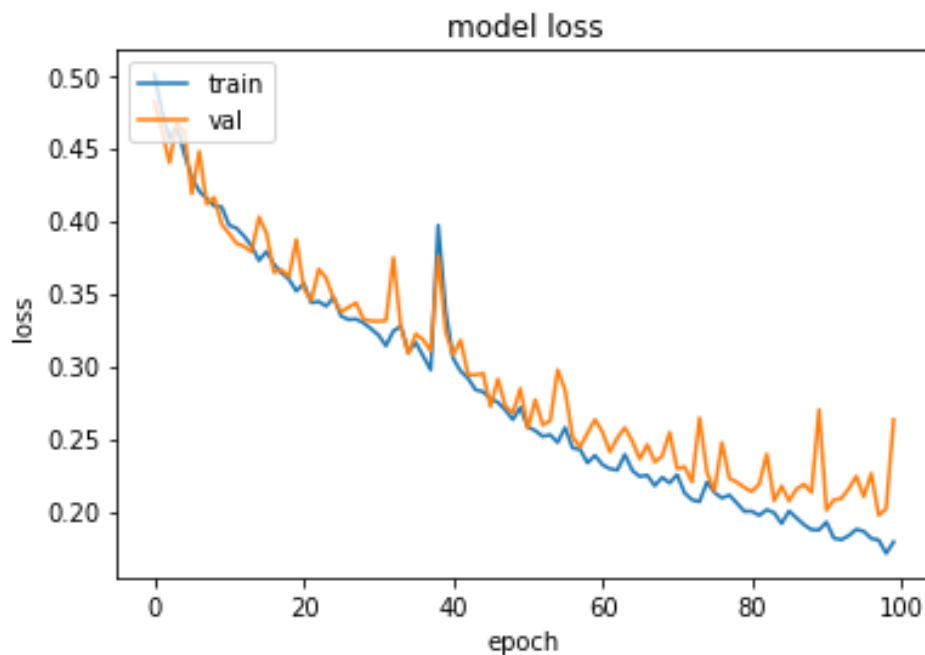


Figure 49 model loss worst case

At the beginning of the graph, you can see that the model Loss was high, about 0.5, and that it required 100 epochs to decrease it to almost 0.2 for the training Loss and roughly 0.25 for the validation Loss.

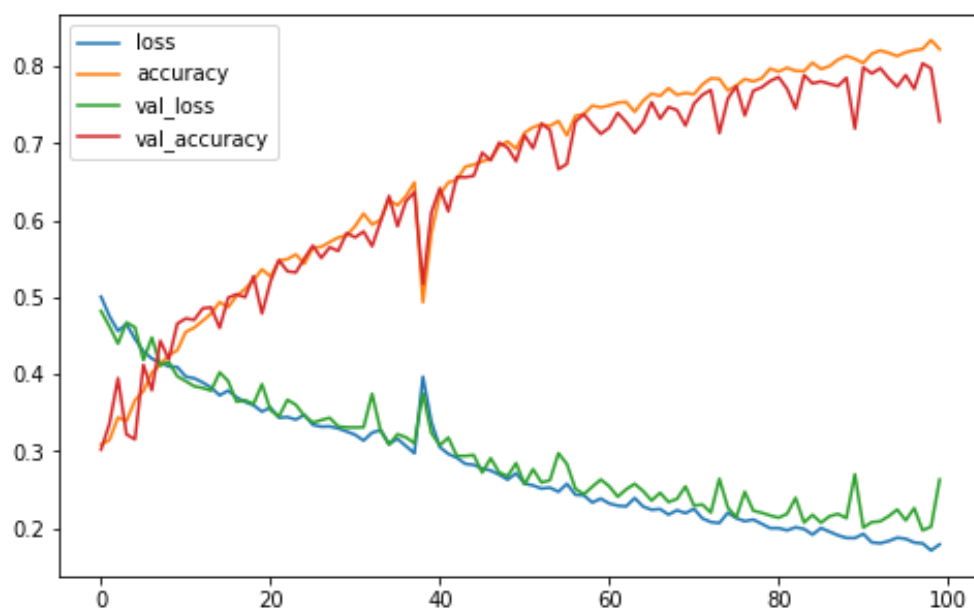


Figure 50 model summary worst case

In Fig.50, you can see how the loss is diminishing and how the accuracy is growing as you go through the 100th epoch.

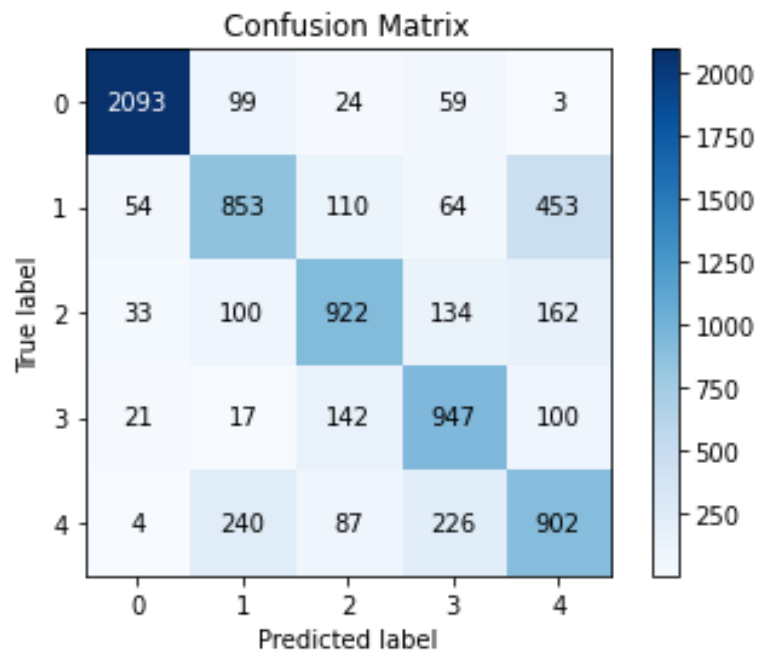


Figure 51 confusion matrix worst case

Fig.49 shows the confusion matrix for the correct and incorrect predictions

True positive: A result in which the model accurately predicts the positive class.

A false positive is an outcome in which the model forecasts the positive class inaccurately.

True negative: A result in which the model accurately predicts the negative class.

False negative: A result in which the model forecasts the negative class inaccurately.

Table 8 classification report for worst case

	Precision	Recall	F1-score	support
0	0.95	0.92	0.93	2278
1	0.65	0.56	0.60	1534
2	0.72	0.68	0.70	1531
3	0.66	0.77	0.71	1227
4	0.56	0.62	0.59	1459
Accuracy			0.73	7849
Macro avg	0.71	0.71	0.71	7849
Weighted avg	0.73	0.73	0.73	7849

Table 7 shows the precision, recall, f1-score and support for the acceptable case and the 0, 1, 2, 3, 4 represents the classes A, B, C, D and E.

Precision is defined as the ratio of true positives to the total of true and false positives.

Recall is defined as the ratio of true positives to the total of true positives and false negatives.

The F1 score is the weighted harmonic mean of accuracy and recall, calculated as follows: The closer the F1 score is to 1.0, the better the model's projected performance is expected to be.

Support is the number of actual occurrences of a class in a dataset is referred to as support. It does not differ across models; rather, it is used to diagnose the performance assessment process itself.

The following figure, Fig. 54 illustrates what happens to the input as it passes through the model.

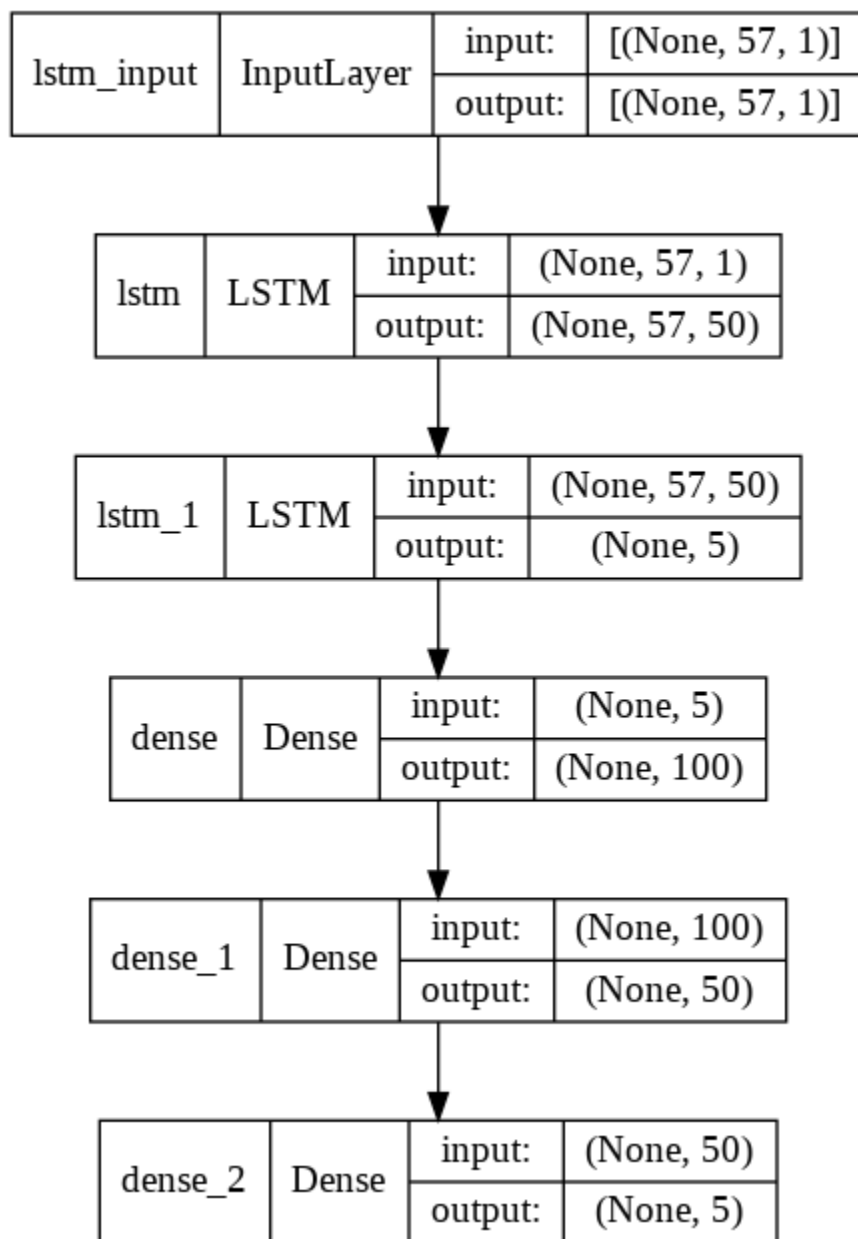
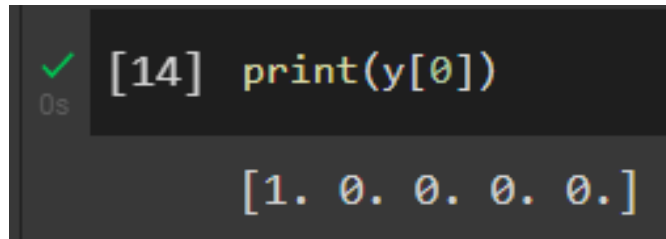


Figure 52 model flow worst case

5.2.6 Limitations

The data set was one of the challenges as the classes was presented as a character “A” not a number so I had to change its representation to numbers and then I made it categorical to be a one hot coded so I can represent it as [0,0,0,0,1] as in Fig. 55.



```
[14] print(y[0])  
[1. 0. 0. 0. 0.]
```

Figure 53 y_train and predicted y value

Also, I was using activation function “relu” which led the model to stuck and there was no performance increasing unlike the “sigmoid” which didn’t do as the “relu” and after using the “sigmoid” the performance increasing started to appear remarkably.

Also, I was using a loss function “categorical_crossentropy” which made the Loss higher than and after some searches I figured that “binary_crossentropy” is better for my model as the Loss went down so much after using it.

5.3 Evaluation

5.3.1 Accuracy Evaluation

Table 9: First Summary of the LSTM Models with their Accuracy

Models Number	No. of LSTM layers	No. of Dense layers	Test Size	No. of Epoch	Optimizer	Shuffle	Activation Function	Batch Size	Training Accuracy	Validation Accuracy
Model 1	3	1	30%	30	Adam	False	Sigmoid	100	78.01%	75.35%
Model 2	5	2	40%	80	Adam	False	Sigmoid	150	88.78%	86.86%
Model 3	3	3	30%	50	Adam	False	Sigmoid	300	94.85%	91.81%
Model 4	4	4	40%	100	Adam	True	Sigmoid	0	92.22%	91.52%
Model 5	3	3	40%	100	Adam	True	Sigmoid	300	80.85%	72.85%
Model 6	5	3	30%	200	Adam	True	Sigmoid	0	99.80%	93.56%
Model 7	5	3	30%	200	Adam	True	Sigmoid	57	98.81%	98.30%

In the above table, Table 8 it is shown that multiple implementations of the LSTM model were tested with different permutations of parameters such as the number of epochs, test size, shuffle, Number of dense layers, and batch size were tried until the optimum training and validation accuracy was achieved.

Training and validation loss are represented by the first graph, and it can be seen that the training and validation is shown by the second graph.

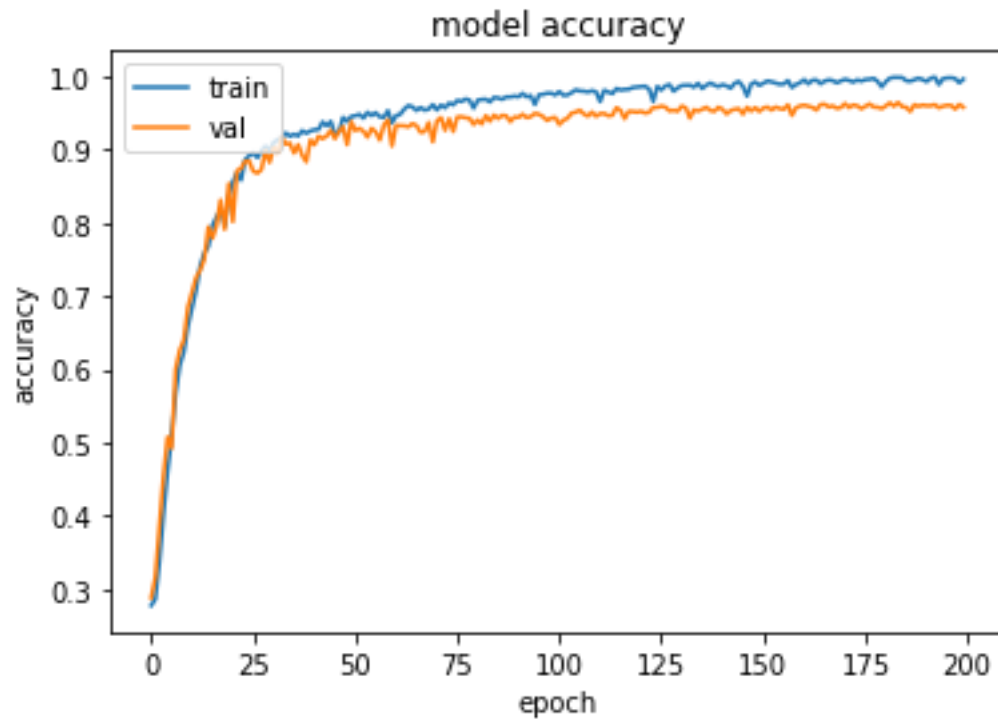


Figure 54 best case model accuracy

At the beginning of the graph, you can see that the model accuracy was too low, about 30%, and that it required 200 epochs to increase it to almost 99% for the training accuracy and roughly 95% for the validation accuracy.

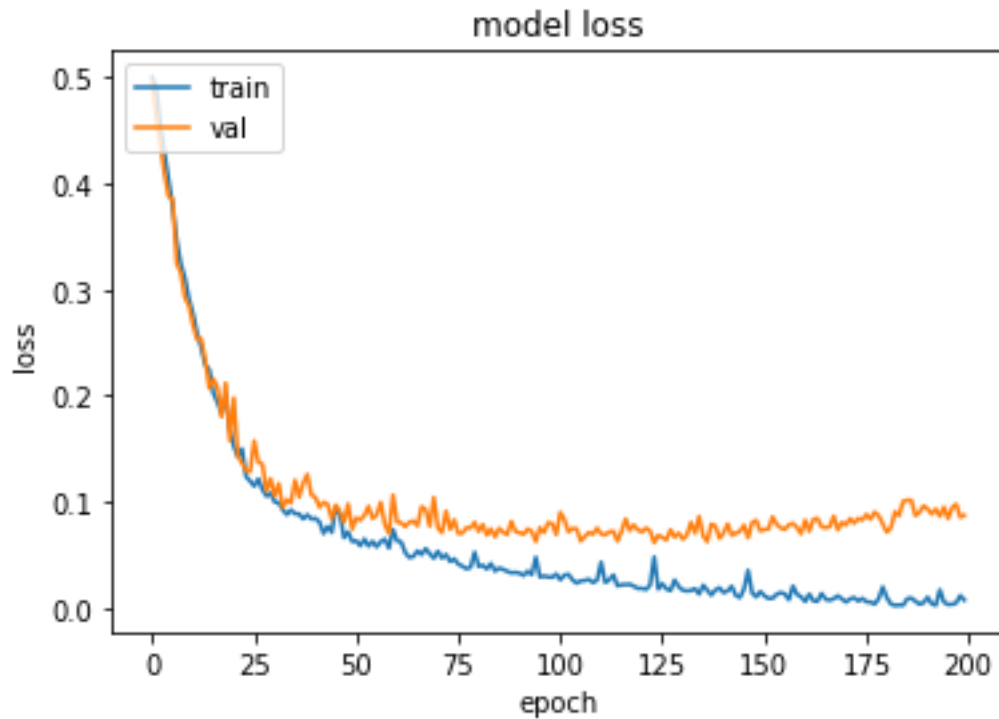


Figure 55 best case model loss

At the beginning of the graph, you can see that the model Loss was high, about 0.5, and that it required 200 epochs to decrease it to almost 0.03 for the training Loss and roughly 0.1 for the validation Loss.

5.3.2 Time Performance

In terms of time, utilizing Google Colab GPU resources was a huge benefit to using Spyder environments and Keras APIs on a local PC CPU, which was a considerable disadvantage. The GPU performance on Google Colab was 45s per epoch and 106ms each step.

Chapter 6: Conclusion and Future Work

6.1 Conclusion

Finally, in our thesis, we developed a system that can read trainee records and, based on the features system can predict which class the trainee belongs to. More importantly, following extensive reading and searching in the same area of our work, which introduced us to several approaches that allowed us to determine our system approach to achieve our goal, which was applying deep learning approach "LSTM" that is considered to be an appropriate approach for our Class Prediction and the extracted features to achieve the result of predicting the trainee record class, we were able to determine our system approach to achieve our goal of predicting the trainee record class.

6.2 Problem Issues

6.2.1 Data issues:

At the beginning when I was reading the data the system crashed because it couldn't read strings so I converted every string to a number like classes and trainees name also I had to search for a while to figure out what is the right way to write the input shape as I have 5 classes I have tried different ways until I figured it out.

6.2.2 Scientific issues:

As I wanted to get the best results out of the model in the beginning, I was using only 3 Blocks 2 LSTM and 1 Dense this model didn't give high accuracy so I needed to increase the number of layers so I can get an appropriate accuracy for the prediction.

6.3 Future Work

Determine and evaluate the appropriateness of several sites on the human body to wear the sensor so I can get better accuracy to make it easier for the chip which will be used to determine the trainee exercise and help the trainee more accurately.

References

1. A. Baca and P. Kornfeind. Rapid feedback systems for elite sports training. *IEEE Pervasive Computing*, 5(4):70{76, 2006.
2. Bulling, A., Blanke, U. and Schiele, B., 2014. A tutorial on human activity recognition using body-worn inertial sensors. *ACM Computing Surveys*, 46(3), pp.1-33.
3. D. Bannach, P. Lukowicz, and O. Amft. Rapid prototyping of activity recognition applications. *Pervasive Computing, IEEE*, 7(2):22 {31, 2008.
4. Fu, B., Kirchbuchner, F. and Kuijper, A., 2020. Performing Realistic Workout Activity Recognition on Consumer Smartphones. *Technologies*, 8(4), p.65.
5. Fu, B.; Kirchbuchner, F.; Kuijper, A. Unconstrained workout activity recognition on unmodified commercial off-the-shelf smartphones. In *Proceedings of the 13th ACM International Conference on Pervasive*.
6. G. O'Donovan, A. J. Blazevich, C. Boreham, A. R. Cooper, H. Crank, U. Ekelund, K. R. Fox, P. Gately, B. Giles-Corti, J. M. R. Gill, and et al. The abc of physical activity for health: a consensus statement from the british association of sport and exercise sciences. *Journal of Sports Sciences*, 28(6):573{591, 2010.
7. M. Bachlin, K. Forster, and G. Tröster. Swimmaster: A wearable assistant for swimmer. In *UbiComp '09: Proceedings of the 11th international conference on Ubiquitous computing*, pages 215{224, New York, NY, USA, Sept. 2009. ACM. 4th Augmented Human International Conference (AH'13) 122
8. M. Gallagher. Ten most common causes of training injury. *Muscle & Fitness*, June 1996.
9. Soro, A., Brunner, G., Tanner, S., & Wattenhofer, R. (2019). Recognition and repetition counting for complex physical exercises with deep learning. *Sensors*, 19(3), 714.
10. S. N. Blair. Physical inactivity: the biggest public health problem of the 21st century. *British Journal of Sports Medicine*, 43(1):1 2, 2009.

11. Skawinski, K., Roca, F. M., Findling, R. D., & Sigg, S. (2019, June). Workout type recognition and repetition counting with CNNs from 3D acceleration sensed on the chest. In International Work-Conference on Artificial Neural Networks (pp. 347-359). Springer, Cham
12. Velloso, E., Bulling, A., Gellersen, H., Ugulino, W., & Fuks, H. (2013). Qualitative activity recognition of weight lifting exercises. Proceedings of the 4th Augmented Human International Conference on - AH '13. <https://doi.org/10.1145/2459236.2459256>
13. Wang, J., Chen, Y., Hao, S., Peng, X. and Hu, L., 2019. Deep learning for sensor-based activity recognition: A survey. Pattern Recognition Letters, 119, pp.3-11.
14. Z. Y. Kerr, C. L. Collins, and R. D. Comstock. Epidemiology of weight training-related injuries presenting to united states emergency departments, 1990 to 2007. The American Journal of Sports Medicine, 38(4):765{771, 2010.