HUMAN ACTIVITY RECOGNITION BASED ON ACCELEROMETER AND GYROSCOPE SENSORS

A THESIS

SUBMITTED TO THE GRADUATE SCHOOL

IN PARTIAL FULFILLMENT OF THE REQUIREMENT

FOR THE DEGREE

MASTER IN COMPUTER SCIENCE

BY

ISRAA MISHKHAL

DR. WU SHAOEN – ADVISOR

BALL STATE UNIVERSITY

MUNCIE, INDIANA

JULY 2017

DEDICATION

I dedicate this work:

To **ALLAH Almighty** who opens my mind before my eyes to guide me, enlighten my heart, and show me clearly the right way so I follow it, and show me clearly the wrong way so I avoid it.

To the person who is the source of inspiration, wisdom, knowledge, understanding, mercy, peace and the person who encourages people to be patient, our **Prophet Mohammad.**

To the person who is always praying for me and encouraging me, my **Mother.**

To my beloved husband, **Ammar Alqayyar** who has encouraged me all the way and whose encouragement has ensured that I give it everything it takes to finish that which I have started.

To the person who is always supporting and encouraging me, **Ziad Alqayyar.**

To my **brothers**, **sisters, and my husband's family** who are the second family for me. Also, to my children, **Rand** and **Raneem** who have been affected in every way possible by this quest. **Thank you**.

"My love for you all can never be quantified. God bless you all"

ACKNOWLEDGMENT

I want to pay a special thanks to my thesis advisor Dr. Shaoen Wu for his guidance, answers, and patience every time that I needed him during my work. I want to acknowledge my appreciation to him for all his effort and support in helping me complete my thesis. I wish to pay a special thanks to my committee members (Professor Jay Bagga and Dr. Xin Sun) who were more than generous with their expertise and precious time. I want to thank the chairperson of the Computer Science Department, Dr. Paul Buis, and the Director of Graduate Program, Dr. Samuel Hsieh, for their help throughout my master’s program and completing my graduation requirements. Also, I want to thank my classmate Xu Junhong for his help. Finally, I want to say thanks to all my family members and friends for all their help, support and love.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER ONE: INTRODUCTION

Human Activity Recognition (HAR) has recently become important in activity monitoring for public health care. In general, an activity recognition is used in different technologies to help people keep track of their daily activity movements. Some technologies are used to monitor the movement of users and to encourage them to move. The HAR system can keep a continuous observation on basic human activities of daily living, allowing various services such as life care from physical damage, nursing, rehabilitation, and health assistance to make a more intelligent environment for people.

In 2015, more than 25% of people used a smartphone and held their phone everywhere they went [1]. The popularity of using smartphones has increased dramatically because smartphones are suitable for users to access various online services. Every day, people use their smartphones to make calls; check email; log on social network applications, such as Facebook; and store personal information, data and files onto the clouds. So, the smartphone has become indispensable for most people. In addition, smartphones offer several practical uses in the medical and public health field. Numerous applications and themes are present in smartphones that are applicable to any area of medicine. When possible, relevance to public health is emphasized because the majority of healthcare and medical apps are marketed to health behavior changes, such as weight loss, physical fitness, smoking cessation etc. As smartphone applications become more developed, especially those that are evidence-based, there will be marked growth in relevant peer-review literature [2]. The main goal of using the smartphone in this research is to collect data from a user to classify them by using Convolutional Neural Networks (CNNs) deep learning algorithm [3]. There are many types of body activities. Ambulatory types of activities (walking, sitting, standing, walking upstairs, walking downstairs) are our focus of classifications [4], and we recognized these activities according to those categories.

For this thesis, we used open data sources by using an Android smartphone that has two kinds of sensors to recognize and predict human's daily activity by implementing the CNN algorithm. These sensors (Accelerometer and Gyroscope) are embedded in the smart phone to calculate the body activity for accurate results. These sensors gather signals endlessly and assign more emphasis on real-time signal achievement [3]. The result we obtained after completing the experiment shows that the proposed system can be 98.35% accurate in identifying human activities successfully.

The rest of this thesis is organized as follows. Chapter 2 shows the related work and the currently available solutions to the presented Human Activity Recognition problems. Chapter 3 presents the hardware we used to collect the data and build the system, as well as the algorithm that is used in the system. Chapter 4 describes the system design, while Chapter 5 shows the performance evaluation and comparison with other studies that used the same dataset. Finally, the conclusion and the future work is discussed in Chapter 6.

CHAPTER TWO: RELATED WORK

In this chapter, we are going to show the previous works that tried to solve the Human Activity Recognition (HAR) problem that is related with the healthcare field. It will be split into three sections: the first section is human activity recognition; the second section is sensors that we used to recognize human daily activities, and the last section is methodology used to solve this problem.

2.1 HUMAN ACTIVITY RECOGNITION (HAR)

There are several papers that study Human Activity Recognition (HAR) from different views and using different algorithms. In [12], the researchers study human activity that depends on periodic System Designations from a single instance using CNN and the height of feet above the ground as the feature. The changing in feet above or below cause classification errors in the experiment results. Andres and Cesar in [13] discuss the importance of monitoring and assessing the physical movements using smartphones for people who should balance their movements because of obesity or metabolism syndrome.

An increasing interest in these types of systems causes them to be one of the keys of several applications such as visual surveillance [5], video retrieval [6] and human–computer interaction [7], among others. Because activity recognition is a classification problem, there are several technologies for recognition of human daily activity, such as a smartphone, computer vision, etc. A smartphone [11], [14], [15] could shift in a person’s pocket, which is not ideal for tracking hand based activities. The smartphone-based Human Activity Recognition (HAR) is especially limited for women and elderly people since they typically do not tend to keep a phone in their pockets for a long time. Thus, a smart band that is worn in a consistent position and ideally situated for tracking hand-based activities has recently become popular for monitoring human’s daily activities. It can be utilized to decrease the death rate and predict early detection of a heart attack [16]. Recognition of human activities can be considered as the last step of a set of previous tasks, such as image capture, segmentation, tracking, identification, and classification. So, the HAR is an important area of healthcare research. There are many applications, including surveillance systems, patient monitoring systems, and a variety of systems that involve interactions between persons and electronic devices such as human-computer interfaces. Most of these applications require an automated recognition of high-level activities, composed of multiple simple (or atomic) actions of people.

Human body activity is the coordinated movement of different body parts and the connected joints. Researchers believe that knowledge of limb and joint angles is useful to detect the termination and commencement of different actions. Many studies have used information from the movement of body parts such as the trunk, arms and legs to analyze human motion in for healthcare purposes.

2.2 SENSORS

Recognizing activities for Human Activity Recognition systems have been challenging. Therefore, different types of sensors have been used to sense human's activities during daily lives. These sensors include state change sensors [7] attached to appliances and RFID tags and readers used with household items [8] to collect object usage data as an indirect way to infer user activity. In [9], the authors study HAR with a variety of sensors, including RFID sensors, switch sensors, and motion sensors, and offer an evaluation in real-world conditions. In addition, HAR becomes useful for elderly monitoring applications by using depth video sensors' technologies. Depth video sensors, which produce depth or distance information, have greatly improved HAR [10].

Body Sensor Networks (BSNs) and heterogeneous sensors have been used for automatic and intelligent human daily activities to monitor elderly people [11], [14]. They aim to capture the state of a user and its environment by utilizing information from heterogeneous sensors, which allows for continuous monitoring of numerous physiological signals by attaching these sensors to the subject's body. This can be immensely useful in activity recognition for identity verification, health, ageing, sport and exercise monitoring applications. In [17] a high-accuracy HAR system based on single tri-axial accelerometer ADXL330 sensor for use in a naturalistic environment was developed. It is manufactured by Analog Devices, which is capable of sensing accelerations with tolerances. The output signal of this sensor is sampled at 100 Hz. Additionally, the data generated by it was transmitted to a personal computer using a Bluetooth device. Data from this sensor has the following attributes: time, acceleration along x-axis, acceleration along y-axis and acceleration along z-axis. In [19], a key challenge in inferring human activities from multiple sensors is the fusion of low-level streams of raw sensor data into higher-level assessments of activity. These sensors are Binaural microphones, a USB camera, and a keyboard and mouse to recognize the overall office situation, such as the presence of a phone conversation, a face to face conversation, an ongoing presentation, a distant conversation, nobody in the office, or a user is present and engaged in some other activity. [20] studied the state of the art in HAR based on wearable sensors. Because these sensors have high computational power, small size, and low cost, users can incorporate them into their daily lives.

2.3 HAR ALGORITHMS

Human Activity Recognition (HAR) problem approach is broken into two stages. The first is Feature Generation, which is the process of taking raw unstructured data and defining it as a potential problem. The second stage is Feature Extraction, which tests transformations of the original features. It develops or enhances the movements into pure groupings [14]. Feature extractions are the different algorithms used for grouping the data. Recently, deep learning algorithms are generally used for recognition and classification, such as image recognition and face recognition. Oscar Lara and Miguel Lobrador in [20] present the Waikato Environment for Knowledge Analysis (WEKA), which is certainly the best-known tool in the machine learning research community. It contains implementations of several learning algorithms, and allows them to be easily evaluated for a dataset using cross validation and random split, among others. Thus, it helps to solve HAR. There are many papers that use different feature extractions other than what we used in this paper. Zhenyu and Lianwen [17] represent a high accuracy HAR system using discrete cosine transform (DCT), the Principal Component Analysis (PCA) and Support Vector Machine (SVM) for classification of human different activity. In [18], the researchers used Shift-invariant Sparse Coding algorithm for Learning Features for Activity Recognition. In [19], the authors train and test the performance of LHMMs and HMMs on recorded office activity data for (10 minutes per activity, 6 activities and 3 users). Additionally, the other papers use the WEKA Explorer mode to classify and categorize HAR by using some the classifier algorithms (such as Bayes, Functions, Lazy, Meta, Mi, Misc, Rules, and Trees) [21]. Finally, there are many different feature extractions, and the technology used for the problem will offer diverse results.

CHAPTER THREE: BACKGROUND

In this chapter, we will present more details about the parts that were used to create the proposed system and more general explanations about them. This chapter will be separated into two sections. The first section is about the hardware that was used to build this system, such as the sensors in some common smart devices that are used to collect data from people, and will explain the common use for each sensor. The second section explains the software that includes HAR as a real-life human-centric problem such as eldercare and healthcare, and the methodology that is used to determine human activities in daily lives.

3.1 HARDWARE

In this section, we explain the hardware devices that are used to build the proposed system, including the Body Sensors Network.

3.1.1 BODY SENSORS NETWORK (BSN)

The Body Sensors Network (BSN) connects the physical environment with electronic systems. It is also used to collect information from the physical environment. It has collections of sensors that are responsible for processing information by format conversion, logical computing, data storage, and transmission [22]. The proposed system uses two types of sensors (accelerometer and gyroscope sensors). These are a new type of inertial sensor with a small size, light weight, low cost, and low power consumption [23].

3.1.1.1 ACCELEROMETER SENSOR

The accelerometer sensor is a device that can measure acceleration (the rate of change in velocity), but in smartphones, it is able to detect changes in orientation and tell the screen to rotate. Basically, it helps the phone know up from down directions (see figure 1). The main use for this sensor within smartphones is to measures the linear acceleration of the device on the X- axis (lateral), Y-axis (longitudinal), and Z-axis (vertical) [24].

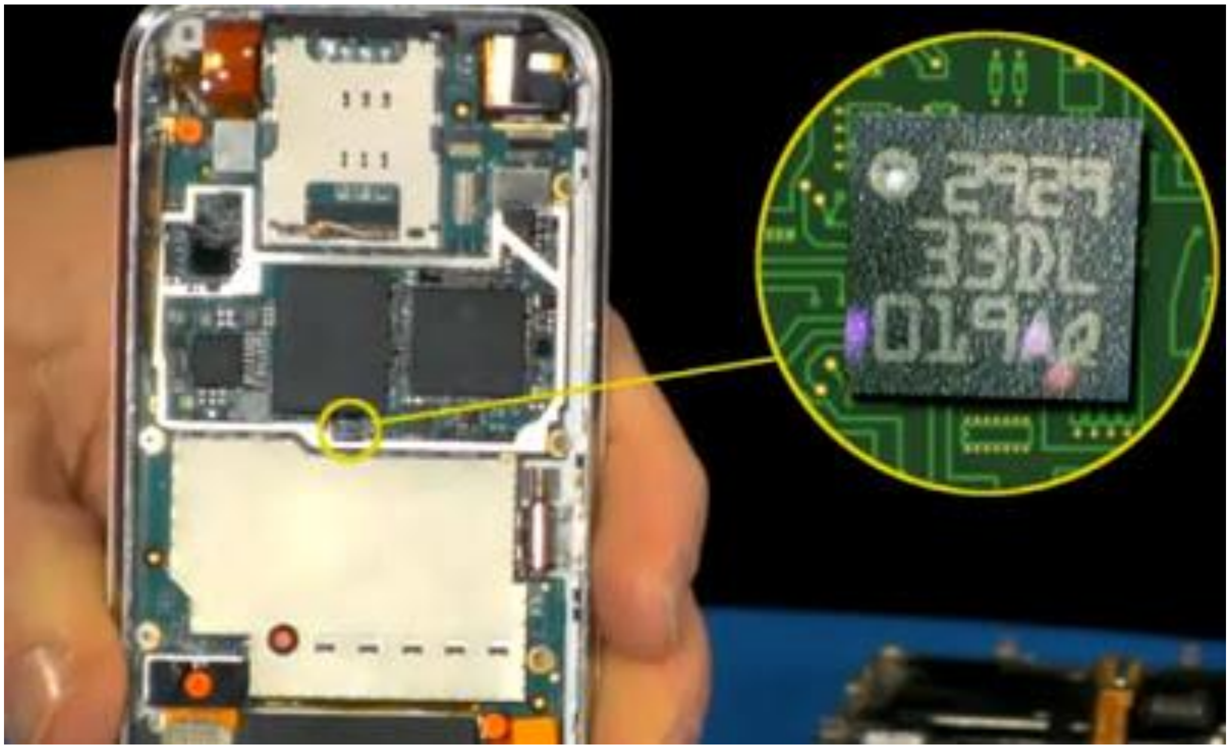


Figure 1 Accelerometer sensor in a smartphone (picture is from a website [41]).

It has been used heavily in smartphone sensors based activity recognition. For instance, if a user changes his/ her activity from walking to jogging, it will react on the signal shape of the acceleration reading: along the vertical axis there will be an abrupt change in the amplitude. Furthermore, the acceleration data could demonstrate the motion pattern within a given a period, which is helpful in the difficult HAR.

3.1.1.2 GYROSCOPE SENSOR

The gyroscope is a device that can provide orientation information as well, but with greater precision (see figure 2). The gyroscope also has been used in smartphones to measure the phone's rotation rate by detecting the roll, pitch, and yaw motions of them along the x, y, and z axis, respectively. It adds an additional dimension to the information supplied by the accelerometer by tracking rotation or twist. Also, it measures the angular rotational velocity and rate of change. The accelerometer sensor can give either a "noisy" information output that is responsive, or give a "clean" output that's sluggish. But when we combine the 3-axis accelerometer with a 3-axis gyroscope, we get an output that is both clean and responsive at the same time [25]. Table 1 shows descriptions of accelerometer and gyroscope sensors, as well as their functions that are supported by the Android platform [26].

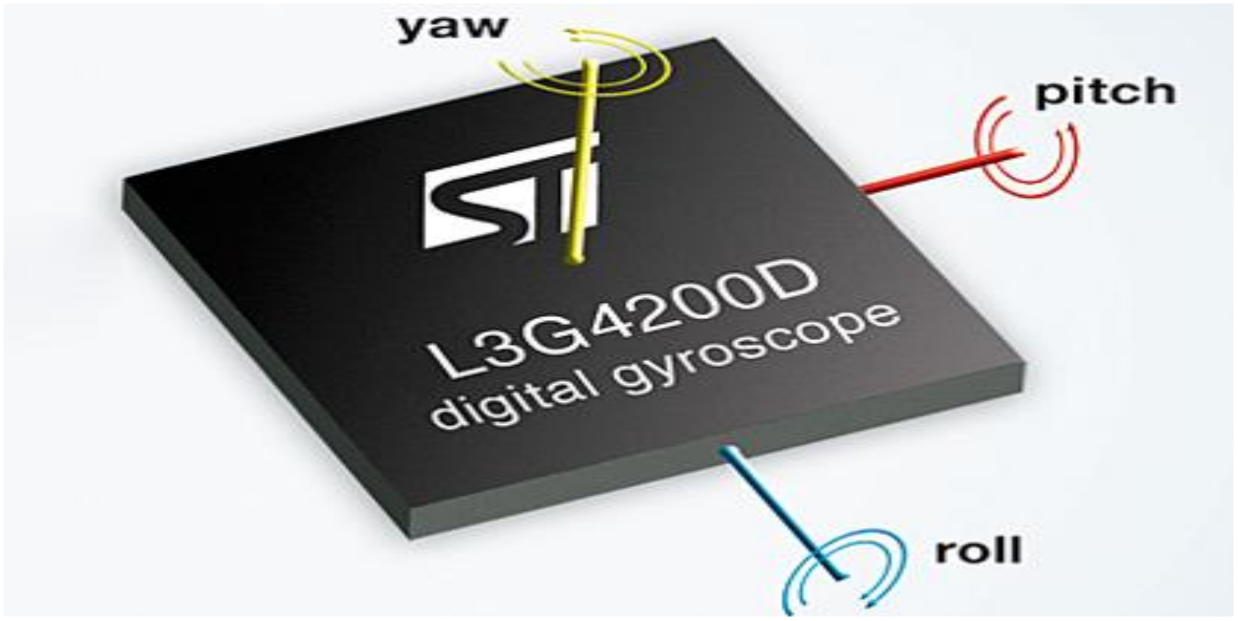


Figure 2 Gyroscope sensor in a smartphone (picture is from a website [42]).

Sensor	Description	Common use
Accelerometer	Measures the acceleration force in m/s^2 on all three physical axes (x, y and z)	Motion detection
Gyroscope	Measures a device’s rate of rotation in rad/s on all three physical axes (x, y and z)	Rotation detection

Table 1 Accelerometer and gyroscope sensors, descriptions, and functions

3.2 SOFTWARE

3.2.1 HUMAN ACTIVITY RECOGNITION (HAR)

Human Activity Recognition is an important technology in widespread computing because it can be applied to many real-life, human-centric problems, such as eldercare and healthcare. Activity recognition aims to recognize common human activities in real-life settings. Accurate activity recognition is challenging because human activity is difficult and highly diverse. Several probabilities based algorithms have been used to build activity. For example, the paper [11] explained several algorithms that are used to recognize human daily activity. HAR has been approached in two different ways, namely using external and wearable sensors. In the former, the devices are axed in predetermined points of interest, so the inference of activities entirely depends on the voluntary interaction of the users with the sensors. In the latter, these devices are attached to the user. An activity recognition requires two stages: training and testing. The training stage initially requires a time series dataset of measured attributes from individuals performing each activity. The time series are split into time windows to apply feature extraction, thereby altering relevant information in the raw signals. After that, learning methods are used to generate an AR model from the dataset of extracted features. In testing stage, data are collected during a time window.

3.2.2 CONVOLUTION NEURAL NETWORK (CNNet)

The Convolution Neural Network (CNNet) is a feed-forward type of machine-learning algorithm based on the human nervous system. The typical architecture of a CNNet (see Figure 3) is composed of one or more stages containing convolution layers and pooling layers. These stages are followed by one or more fully-connected layers prior to a top-level. Some activation layers include Softmax, Rectified linear units (RelU), TanH etc. (see figure 4 [27], [3]). Appendix A shows the Convolution Neural Networks (CNNs) that we built to predict Human Activity Recognition.

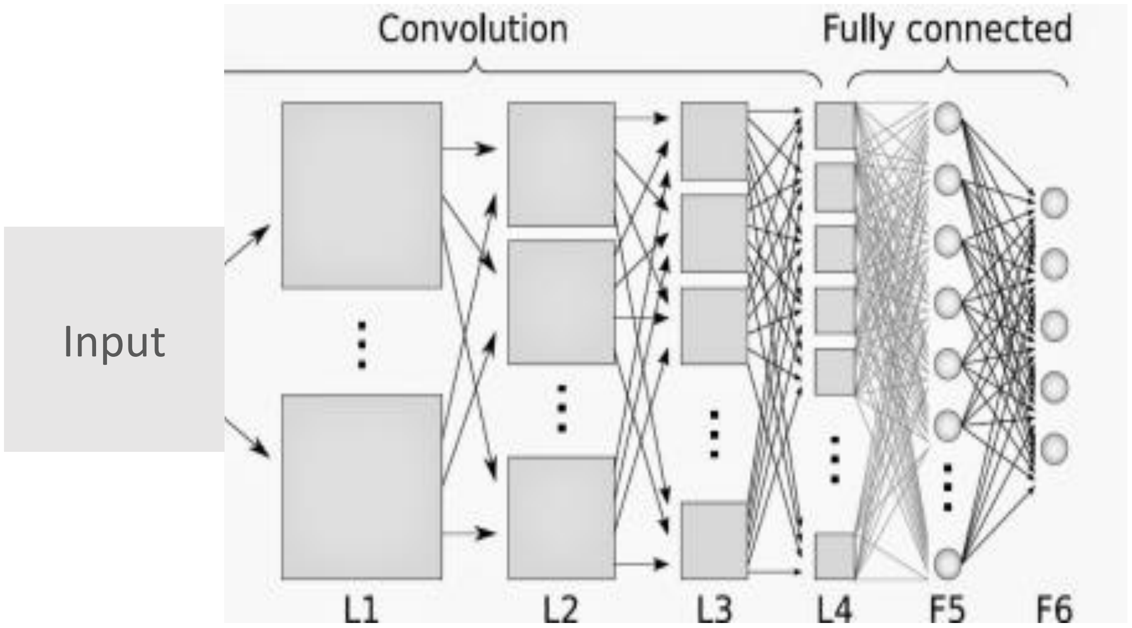


Figure 3 The Convolution Neural Network Layers

The CNNet has a great potential to determine the signals of HAR. Specifically, the processing units in the lower layers are used to characterize the nature of each basic movement in a HA.

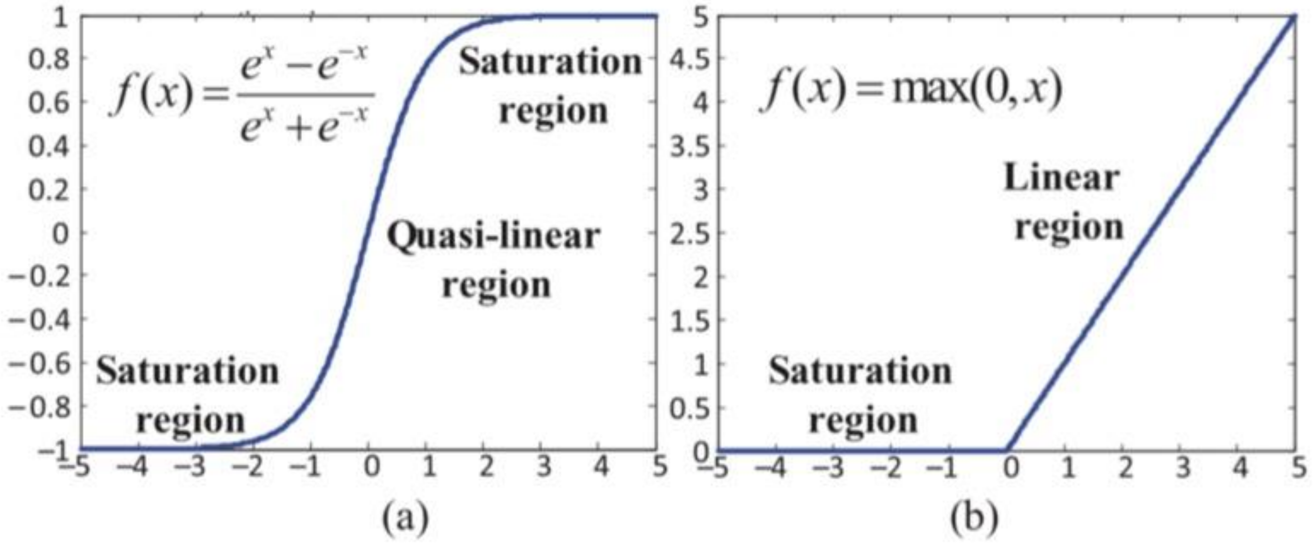


Figure 4 Two nonlinear activation functions. (a) TanH (b) ReLU [27]

In the higher layers, the processing units obtain the salient patterns of signals at high-level representation to distinguish the salience of a combination of several basic movements. The most important attribute of the CNNet is conducting different processing units, such as convolution, pooling, sigmoid/hyperbolic tangent squashing, rectifier and normalization alternatively [28]. The sliding window strategy is used with a length of w values and with a certain percentage of overlap to extract input data for the CNNet and to segment the time series into a collection of short pieces of signals [28]. We used the L-layer CNN-based model with three kinds of layers: 1) An input layer whose values are fixed by the input data; 2) hidden layers whose values are derived from previous layers; and 3) an output layer whose values are derived from the last hidden layer [3]. We have a convolution layer that convolves the input with a set of kernels (filters) to be learned. The max pooling layer with a sub-sampling factor can preserve scale invariants. The normalization layer normalizes the values of different feature maps in the previous layer. The convolution layers are convolved with several convolutional kernels to be learned in the training process. The output of the convolutional operators is put through the activation function to form the feature map for the next layer. The value $\mathrm{v}_{\mathrm{ij}}^{\mathrm{x,d}}$ is given by this equation as described in [28]:

$$\mathrm{v}_{\mathrm{ij}}^{\mathrm{x,d}} = \tanh\,(\mathrm{bij} + \textstyle\sum_{m}\sum_{p=0}^{pi-1} w_{ijm}^{p}\, v_{(i-1)m}^{x+p,d}) \tag{3.1}$$

$$\forall\ \mathrm{d} = 1, \ldots., \mathrm{D}$$

Where TanH represents the hyperbolic tangent function, bij represents the bias for this feature map, while m indexes over the set of feature map in the (i-1) the layer connected to the current feature map, and w_{ijm}^{p} represents the value at the position p of the convolutional kernel. Moving to the pooling layers, the resolution of the feature map is minimized to increase the invariance of features to distortions on the input. Feature maps in the previous layer are pooled by either max pooling function:

$$\mathrm{v}_{\mathrm{ij}}^{\mathrm{x,d}} = \max \leq p \leq \mathrm{Qiv}_{\mathrm{ij}}^{\mathrm{x,d}} \tag{3.2}$$

$$\forall\ d = 1, \ldots., D$$

Or a sum pooling function:

$$\mathrm{v}_{\mathrm{ij}}^{\mathrm{x,d}} = 1\backslash\mathrm{Qi}\textstyle\sum 1 \leq q \leq Qi\ (\mathrm{v}_{\mathrm{ij}}^{\mathrm{x,d}}) \tag{3.3}$$

$$\forall\ d = 1, \ldots., D$$

The number of layers of the CNNet classifier is varied as well to determine if deep learning will contribute to the performance of the system.

CHAPTER FOUR: SYSTEM DESIGN

In this chapter, we describe a new solution for Human Activity Recognition (HAR) by using multiple sensors in the smartphone. First of all, we used open source dataset collection (UCI) that collected the data by using more than one sensor in the smartphone. After that, we trained the data using Convolution Neural Networks (CNNs) algorithm. Also, we used this methodology to predict and evaluate the six activities that humans do every day. Figure 5 below explains the design of our system.

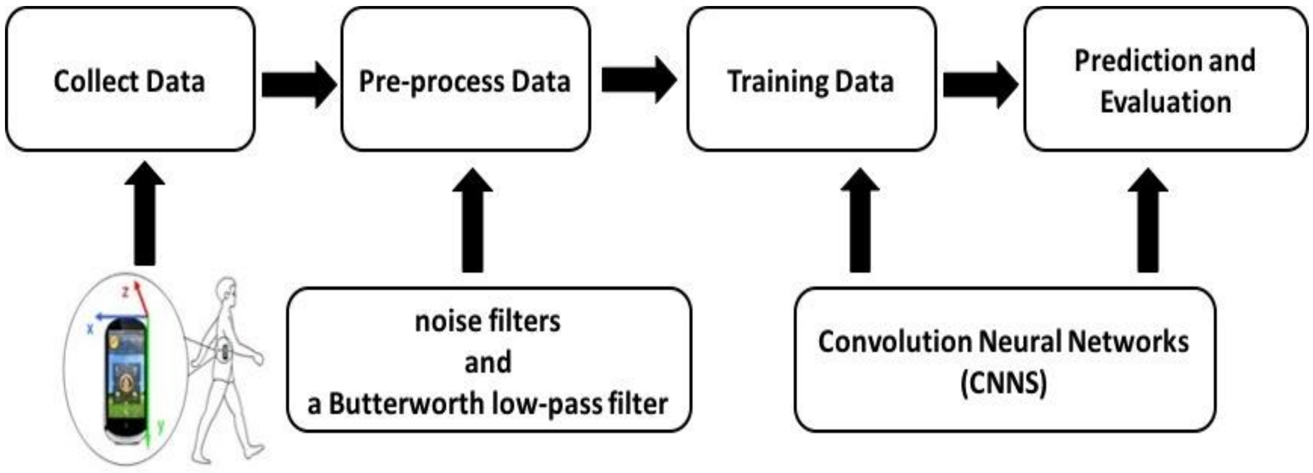

Collect Data
Pre-process Data
Training Data
Prediction and Evaluation
noise filters and a Butterworth low-pass filter
Convolution Neural Networks (CNNS)
x
y
z

Figure 5 Block diagram for our system design

4.1 DATASET AND PREPROCESSING

The data that we used for monitoring people was collected based on HAR by using accelerometer and gyroscope sensors from the UCI Machine Learning Repository [29] (see figure 6). The experiments were carried out with a group of 30 volunteers between the ages of 19 and 48 years. Each person performed six activities: standing, sitting, lying, walking, walking downstairs and walking upstairs. As we explained in (3.1.1), accelerometer and gyroscope sensors captured 3-axial linear acceleration and 3-axial angular velocity at the sampling rate of 50Hz. The obtained dataset was randomly partitioned into two sets, where 70% of the volunteers was selected for generating the training data and 30% the test data. The data were divided in two parts and they can be used separately. These are:

- Inertial sensor data: Raw tri-axle signals from the accelerometer and gyroscope sensor of all the trials with volunteer, and the labels of all the performed activities.
- Records of activity windows: Each one contains a (561-feature) vector with time and frequency domain variables, a connected activity label and an identifier of the subject who carried out the experiment.

The signals from accelerometer and gyroscope sensors were pre-processed by applying noise filters and then sampled in fixed-width sliding windows of (2.56 sec.) with a 50% overlap (128 readings/window). The signal from an acceleration sensor has gravitational and body motion components. It also was separated using a Butterworth low-pass filter into body acceleration and gravity. The gravitational force is assumed to have only low frequency components, therefore a filter with (0.3 Hz) cutoff frequency was used. From each window, a vector of (561) features was obtained by calculating variables from the time and frequency domain.

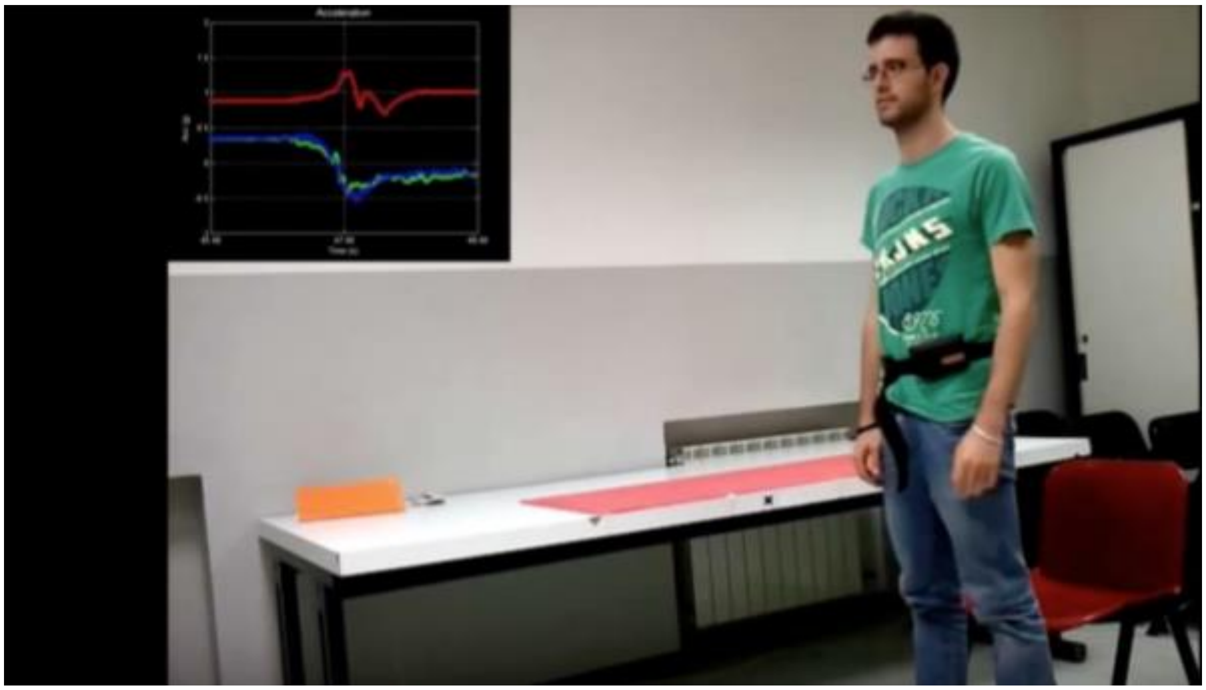


Figure 6 Human Activity Recognition (HAR) using smartphone sensors (picture is from a website [43]).

4.2 METHODOLOGY:

4.2.1 DEEP LEARNING ALGORITHMS:

Since 2006, deep learning has appeared as a new field of machine learning research. It allows computational models that are composed of multiple processing layers to learn representations of data with multiple levels of abstraction. These methods have dramatically improved the state of the art in speech recognition, visual object recognition, object detection and many other domains [30]. Deep learning discovers intricate structures in large data sets by using the backpropagation algorithm to indicate how a machine should change its internal parameters that are used to compute the representation in each layer from the representation in the previous layer. Deep convolutional nets have brought about breakthroughs in processing images, video, speech and audio. In our work, we used one of the common machine learning algorithms, CNNs.

We used it with data that was collected from two sensors, accelerometer and gyroscope, which are found in smart phones.

4.2.1.1 CONVOLUTION NEURAL NETWORKS FOR TRAINING DATASET TO RECOGNIZE HAR

We used the CNNs algorithm in our solution. It has three main layers and some activation layers. The architecture of our network is summarized in Appendix A. It contains fifteen main layers and eleven activation layers, seven 1D convolutional, four max-pooling and four fully-connected layers. We used 7,208 training samples to train our networks. As discussed in chapter 3, each layer of the CNN consisted of a convolutional layer, followed by a spatial pooling layer. The outputs of the second spatial pooling layer fed into a fully connected classification layer that performed either classification in the case of the human activity recognition. The CNNs described in this thesis are implemented in the Keras library in Python language (see Appendix D), a lightweight library to build and train CNNs. The model training and classification are run on a computer with two graphics processing unit (GPU) processors. The first one has a GPU with Intel Core i7, memory with 1733MHz clock speed and 64 GB RAM, and the second one has a GPU with 2560 cores, 1607 MHz clock speed and 8 GB RAM. We used the computer with two GPU processors because we have a lot of data that is about 68 GB. Also, the number of epoch are 300 to train the CNNet methodology. We used a computer with a CPU processor to implement our solution with all this information. So, the implementation took about nine to ten hours comparing with the computer has two GPU processors that took just a few minutes.

It is important to note that due to the relatively large networks we were using, we performed the fine-tuning over the GPU in a distributed setup. The use of a GPU and distributed computing is gradually becoming commonplace in large-scale machine learning [31, 32] and is often essential to constructing and training larger, more powerful, representational models. We trained 70% of the data, and the 30% remaining were used at the same time for the prediction, making our method very efficient.

The first layer, the convolutional layer, is convolved with a learnable filter and put through the activation function to form the output feature map. Each output feature map may merge convolutions with multiple input maps. Each convolution layer is followed by a max-pooling layer [33] (see figure 7). CNNs algorithm requires that in order to compute the sensitivities for a unit at the convolution layer, we first need to sum over the next layer's sensitivities corresponding to units that are connected to the node of interest in the convolution layer, and multiply each of those connections by the associated weights defined at max-pooling layer. We then multiply this quantity by the derivative of the activation function evaluated at the current layer's pre-activation inputs. In the case of a convolutional layer followed by a max-pooling layer, one pixel in the next layer's associated sensitivity map corresponds to a block of pixels in the convolutional layer's output map. Thus, each unit in a map at the convolution layer connects to only one unit in the corresponding map at the max-pooling layer. To compute the sensitivities at the convolution layer efficiently, we can up-sample the max-pooling layer's sensitivity map to make it the same size as the convolutional layer's map and then just multiply the sub-sampled sensitivity map from the max-pooling layer with the activation derivative map at the convolution layer element-wise.

The second layer, a max-pooling layer, serves to reduce the overall size of a signal. In many cases, it is done simply for the size reduction [33]. However, in the domain of the 1D convolution layer's outputs, the max-pooling layer can also be thought of as increasing the position invariance of the outputs. The max-pooling involves splitting up the matrix of the convolution layer's outputs into small non-overlapping grids (the larger the grid, the greater the signal reduction), and taking the maximum value in each grid as the value in the reduced matrix. Finally, by applying such a max-pooling layer in between convolutional layers, we can increase spatial abstractness as we increase feature abstractness.

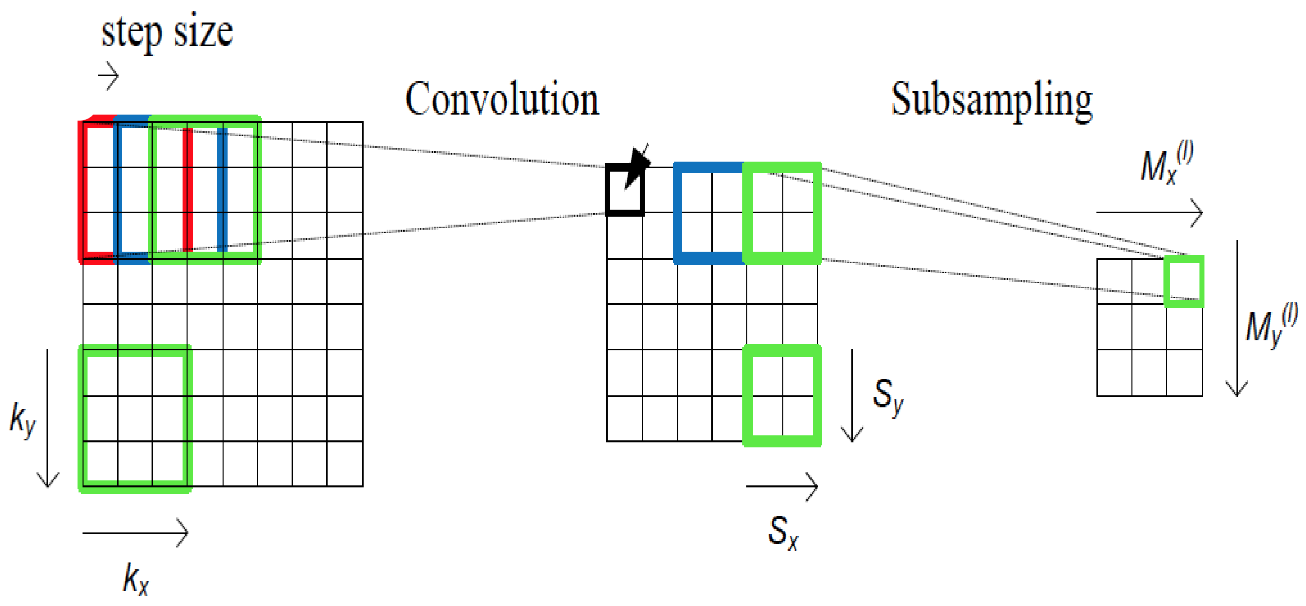


Figure 7 convolution and max-pooling layers in CNNs algorithm [34]

The third layer is a fully connected with a linear function where each output dimension depends on all the input dimensions. The layer takes all neurons in the previous layer (be it fully connected, pooling, or convolutional) and connects them to every single neuron that it has [35]. Finally, we used activation layers between our main layers to translate the input signals to output signals for training our data [7]. The equations for these layers are [36]:

$$\text{Tanh}(X) = 1/\ 1+e^{-X} \qquad (4.1)$$

$$\text{Softmax}(X)_I = e^{Xi} / \textstyle\sum_{k=1}^{k} e^{Xk} \quad \text{for i= 1……k} \qquad (4.2)$$

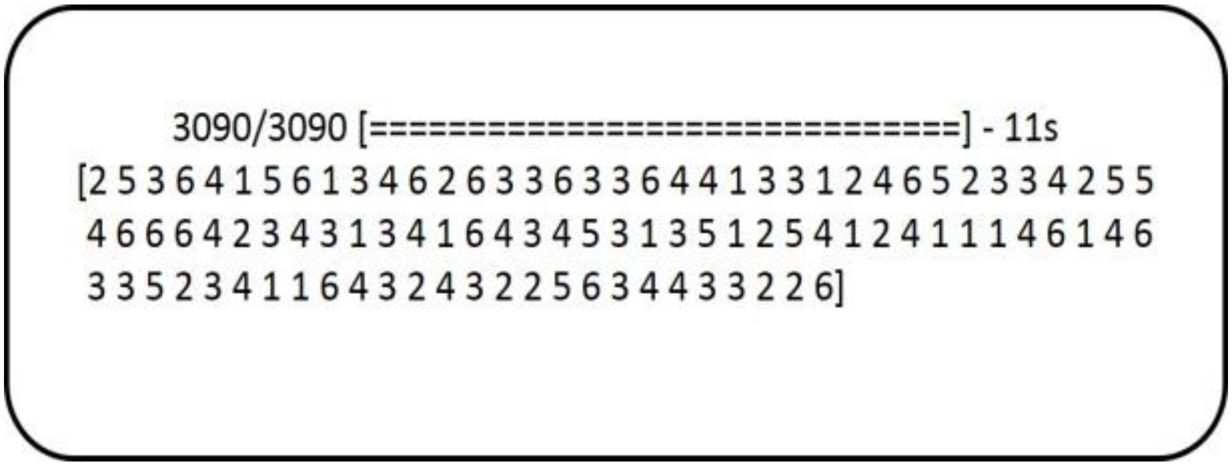
$$\text{ReLU (X)} = \begin{cases} o \; for \; X < 0 \\ X \; for \; X \geq 0 \end{cases} \tag{4.3}$$

Where X is an input from a previous layer.

4.4 THE PREDICTION AND THE EVALUATION OF OUR SYSTEM:

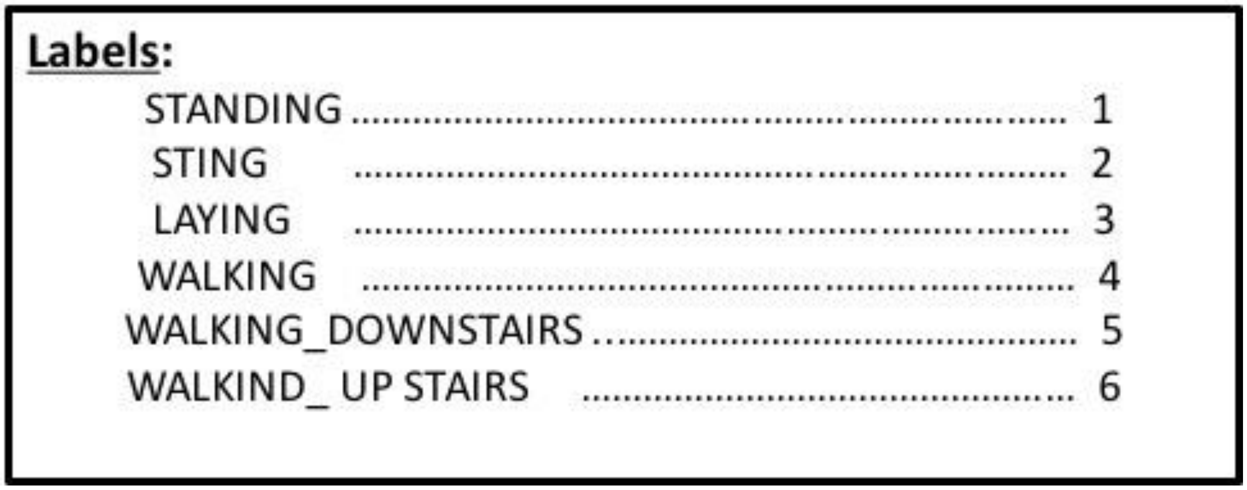
The final stages in the proposed software system are the prediction or the recognition and the evaluation stages. The previous stages in the software system are considered a learning process. At the prediction stage, a new reading is obtained by the sensors, where it is used to predict human daily activities. We use the convolution neural networks (CNNs) algorithm to train the data set. This is also used to predict human daily activities. As mentioned before in 4.2, we used Python language to implement our code. We trained our algorithm on 70% of the dataset that we have, and we made our prediction on the remaining 30%. After training the CNNs algorithm on 7,208 of the training samples. It has set the number of epochs to 300, which might be undesirably slow if you do not have a GPU processor. You might wish to decrease the epoch count and/or numbers of kernels if you are going to be training the network on a CPU. We used the CNNet algorithm to predict the 3,090 samples of testing dataset in the same number of epochs. The results of the prediction stage are shown in figure 8 where the human activities are labeled as numbers. Each activity gives a number from one to six (see figure 9).

In the evaluation stage, the performance of the CNN-based system for human activity recognition uses data that is collected from two kinds of sensors found in the smartphone. As shown in chapter 5, the four experiments were performed to conclude whether the results that we achieved from our solution are favorable when we compared them with the other proposed solutions. These experiments demonstrate that the Convolution Neural Networks methodology has qualitatively better performance than other methodologies that tried to solve HAR problem, reaching 98.58% winning rate (as shown in table 2).



```
      3090/3090 [==============================] - 11s
[2 5 3 6 4 1 5 6 1 3 4 6 2 6 3 3 6 3 3 6 4 4 1 3 3 1 2 4 6 5 2 3 3 4 2 5 5
 4 6 6 6 4 2 3 4 3 1 3 4 1 6 4 3 4 5 3 1 3 5 1 2 5 4 1 2 4 1 1 1 4 6 1 4 6
 3 3 5 2 3 4 1 1 6 4 3 2 4 3 2 2 5 6 3 4 4 3 3 2 2 6]
```

Figure 8 The results after using CNNs algorithm to predict Human Activity Recognition (HAR)



Labels:

STANDING	1
STING	2
LAYING	3
WALKING	4
WALKING_DOWNSTAIRS	5
WALKIND_ UP STAIRS	6

Figure 9 Labels for human activities

Figure 10 shows the frame classification accuracy rate on the core test set for the HAR. The X-axis represents the number of epochs that CNNet trains the training data, which were 300 epochs; while the Y-axis represents the accuracy rate that we achieved, as well as the baseline error. The accuracy that we achieved from our system design is higher. In addition, the error rate decreased during the training stage (see figure 11).

Testing data set	**Accuracy**	**Baseline Error**
The 3,090 samples of testing dataset	98.58%	1.42%

Table 2 Accuracy and error rate of the prediction

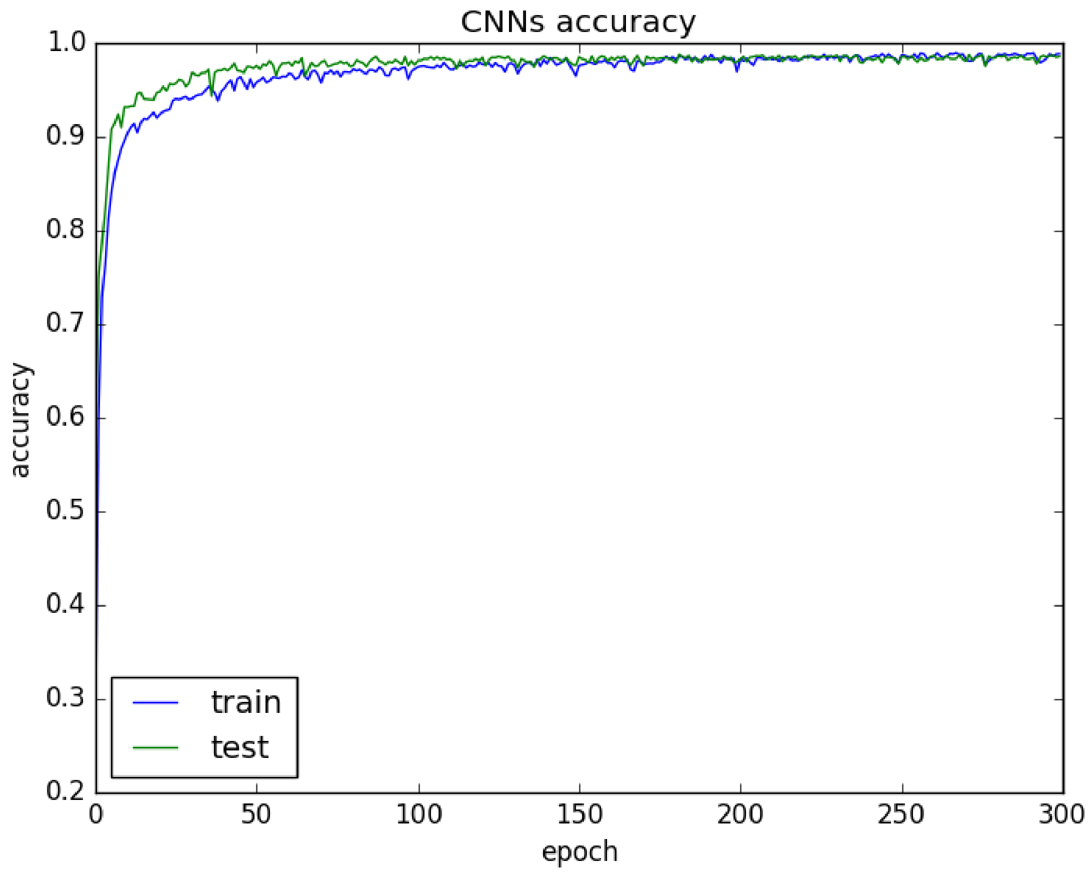


Figure 10 CNNs accuracy diagram

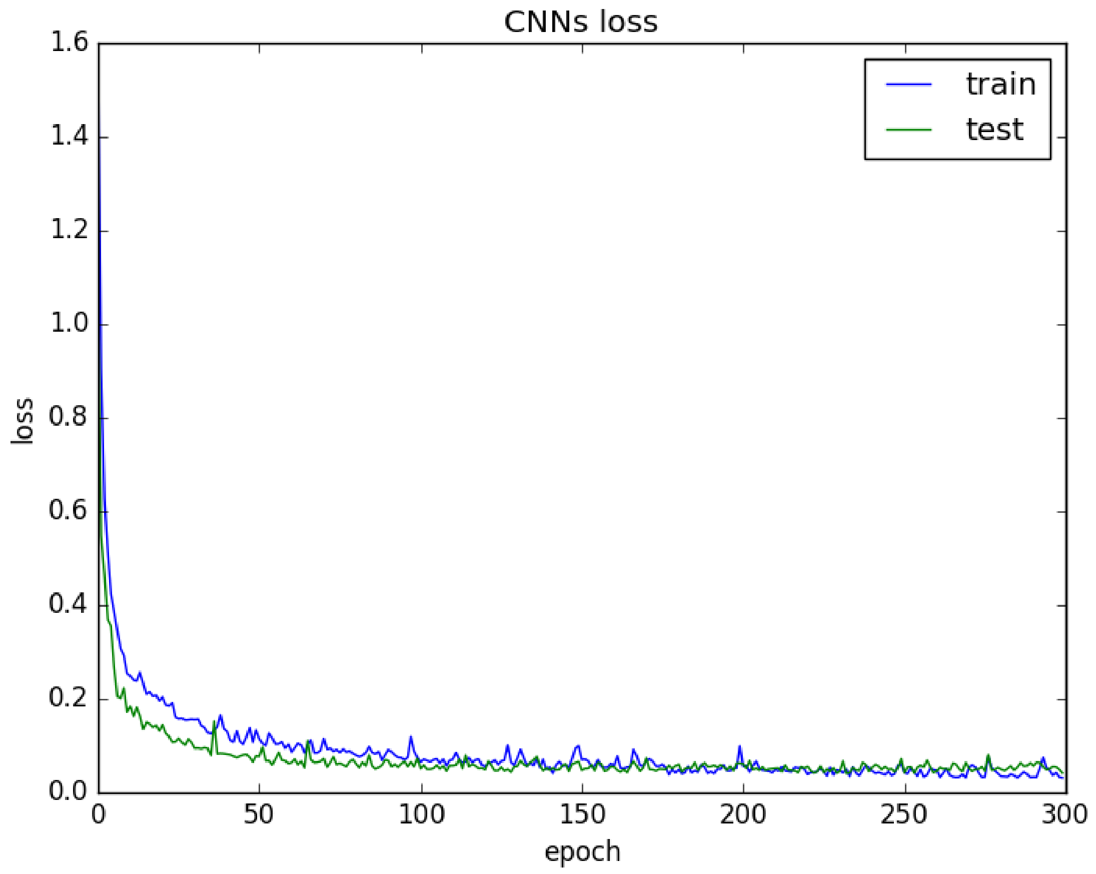


Figure 11 CNNs baseline error diagram

CHAPTER FIVE: THE PERFORMANCE EVALUATION

In this chapter, we are going to show the performance evaluation and compare the proposed solution in this thesis with other research that used the same dataset but different algorithms for solving the HAR problem related with the healthcare field. It will be split into two sections: the first section presents the details about the results that we achieved and analyzed with other experiments that we did in our network. The second section will compare the result that we got from implementing the convolution neural networks algorithm with other research that implemented the same dataset.

5.1 RESULTS AND ANALYSIS

To evaluate the performance of the CNN-based system for HAR and classification using two sensors that are found in a Samsung Galaxy S2 smartphone, four experiments were performed. The first and second experiments iterate on the filter size and number of filters, respectively. The third experiment investigates the possibility of changing the number of layers in the network for the CNNs algorithm, respectively, to see if that will enhance the network performance or not. The last experiment is for increasing the number of epochs for training the data that we used. For each scenario, the accuracy rate was recorded. The training and testing phase primarily operates for HAR, with 10,298 samples for all six human daily activities. 70% of the dataset is used in the training stage, while the remaining 30% is used in the testing stage. Each sample was obtained from accumulator and gyroscope sensors in the smartphone.

5.1.1 SIZE AND NUMBER OF FILTERS

For the first test, the filter size is increased from 3x3 to 11x11 to determine if the trend in the accuracy and baseline error of the system, and we used the same number of filters (seven) in all our experiments. The average performance for this test is equal to 98.63%.

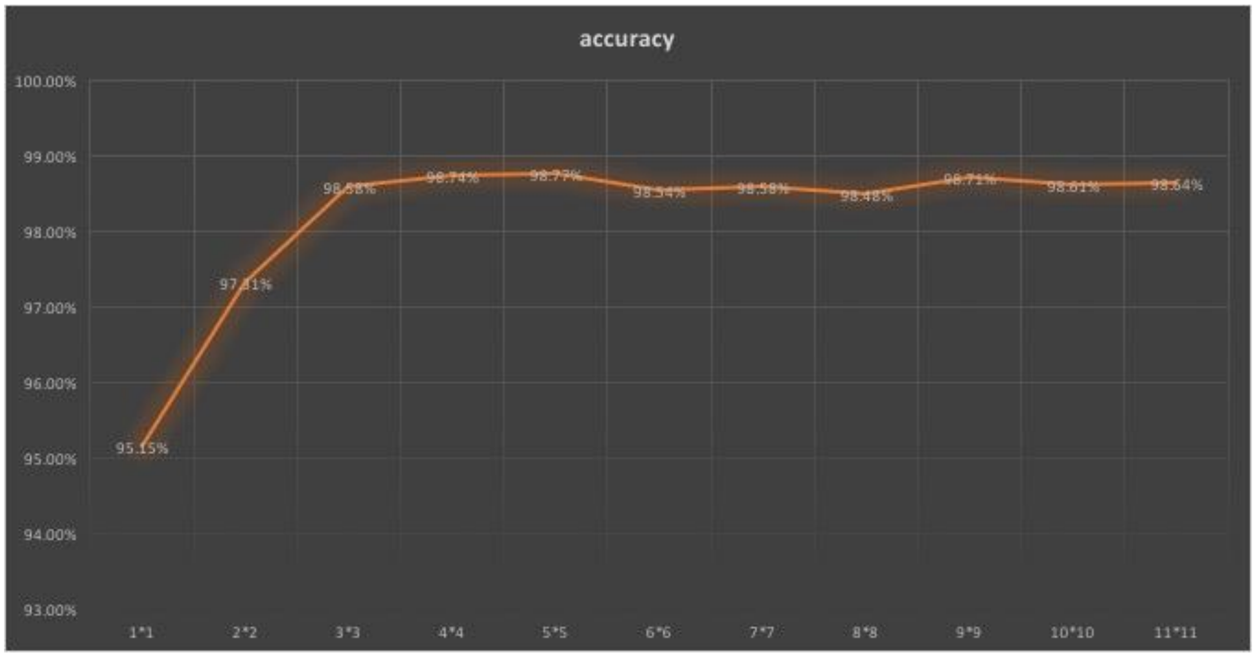


Figure 12 The rate of accuracy by increasing and decreasing filter size

In addition, we tried decreasing the filter size from 3x3 to 1x1 to determine the trend in the accuracy and baseline error of the system. As we seen in figure 12, the increase and decrease of the filter size in CNNs algorithm have an influence on the accuracy. Where the X-axis represents the filter's size that we chose during our experiments, the Y-axis represents the accuracy rate that each filter size achieved. We discovered that the accuracy will increase as the filter size increases due to the increasing robustness of pattern translation contributed by a larger weight overlap.

For the second test, we increased and decreased the number of filters to determine if adding more feature detectors will help in handling the input pattern in our solution. This experiment achieved an accuracy rate of 98.35% when we increased the number of filters and 98.77% when we decreased them. However, further increasing the number of filters show no significant change or trend in the performance of the system. This implies that feature information from any data is finite and adding more detectors will not provide additional information to the classifier.

5.1.2 CHANGING THE NUMBER OF LAYERS IN THE CNNS ALGORITHM

The third experiment investigates the possibility of changing the number of layers in the network for the CNNs algorithm (see Appendices B and C). This test is to determine whether the performance will improve as the network becomes deeper. However, during our experiment, we found that the increment layers in our solution achieved a lower accuracy rate of 96.67% compared to the decrement layers at about 99.00% (see figure 13). Where the X-axis represents the accuracy rate and baseline error that we achieved during our experiments, the Y-axis represents increment layers' test and decrement layers' test that we did. Thus, increased and decreased network size for CNNs methodology are an exceptionally effective use of computational resources. If the added capacity is used inefficiently (for example, if most weights end up being close to zero), then much of the computation is wasted. As the computational budget is always finite, an efficient distribution of computing resources is preferred to an indiscriminate increase of layer, even when the main objective is to increase the quality of performance.

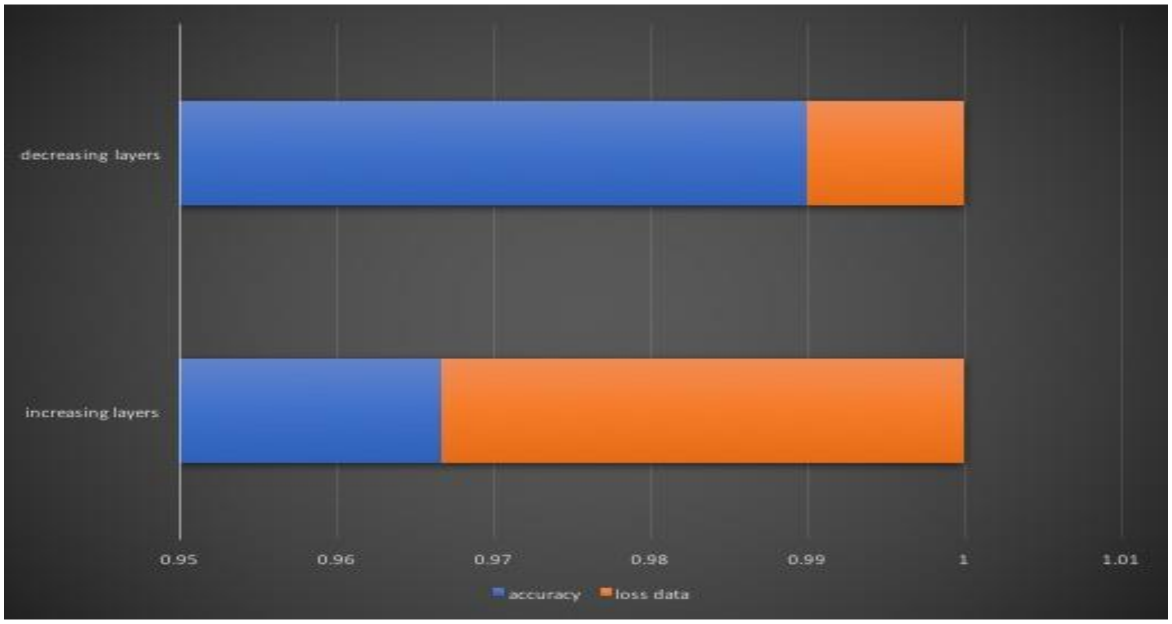


Figure 13 The accuracy rate performance and baseline error from the increased and decreased network size

5.1.3 THE NUMBER OF EPOCHS IN THE CNN ALGORITHM

For the last test, the number of epochs are incremented from 300 to 1,000 to determine the trend in the accuracy of the system (see figure 14). The X-axis represents the number of epochs that we increased from 300 to 1,000, while the Y-axis represents the accuracy rate that each epoch achieved during our experiment. The number of epochs represents the number of times the training CNNet algorithm will iterate over the entire training set before terminating. In this experiment, each increment of number of epochs requires more time than the previous one, and the average performance for this test is equal to 98.70%. We discovered from our experiment that we do not have to worry about setting the right number of training epochs because increasing further the number of epochs shows no significant change or trend in the performance of the system. We just need to use a GPU rather than a CPU to enhance the implementation rate for our solution.

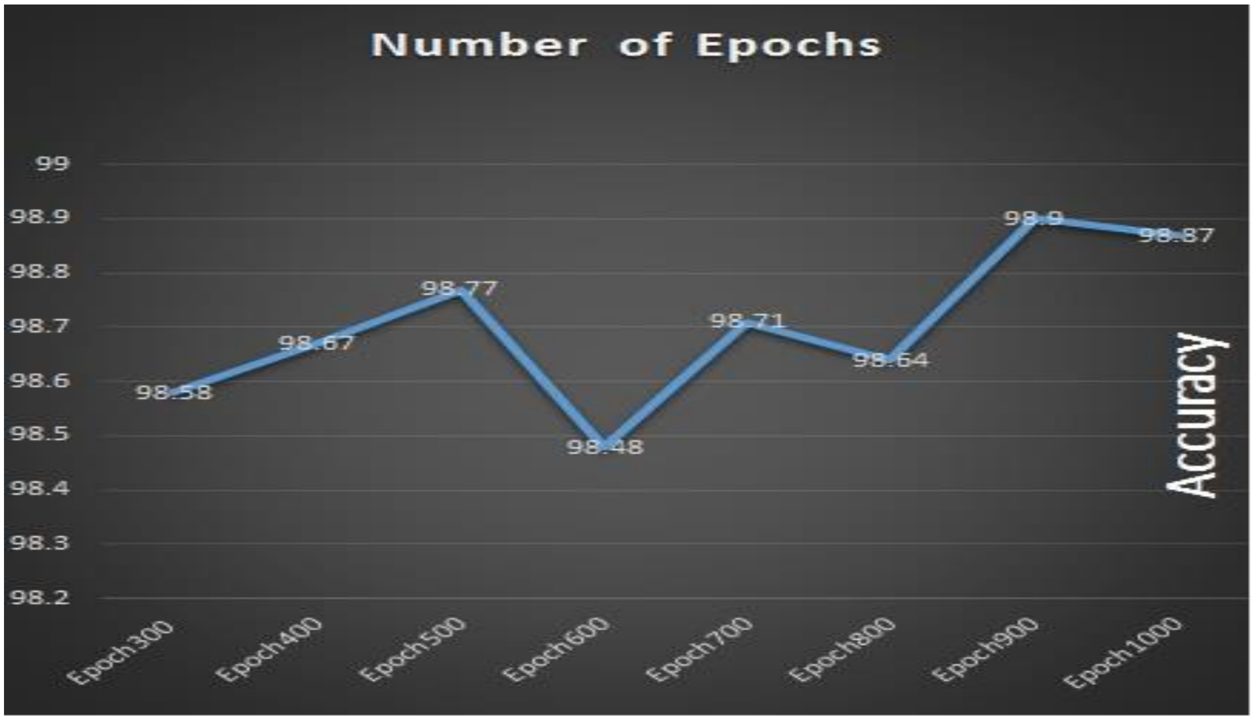
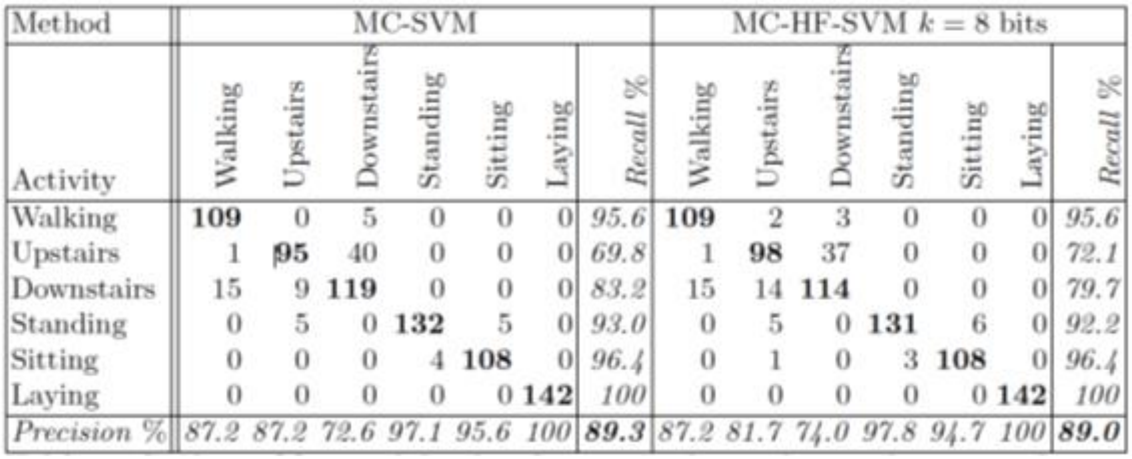


Figure 14 The accuracy rate performance from the fifth experiment

5.2 COMPARING THE RESULT THAT THE CNNet ALGORITHM ACHIEVED WITH OTHER METHODOLOGIES THAT IMPLEMENTED ON THE SAME DATASET.

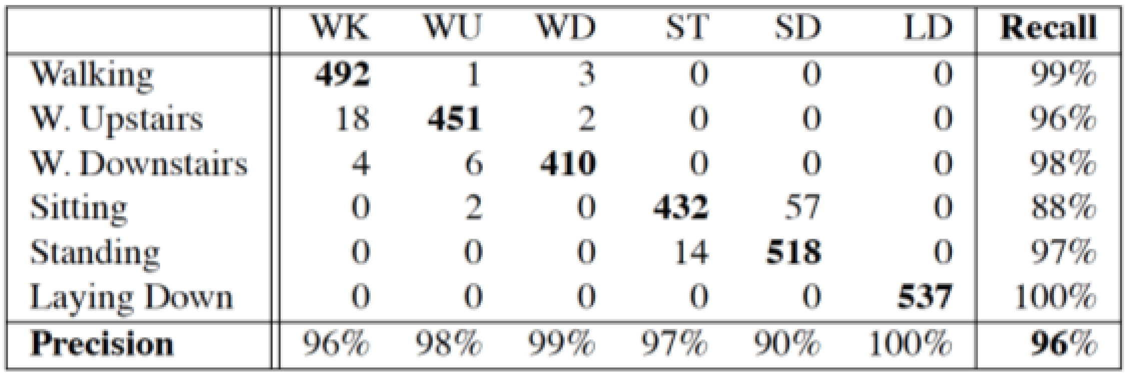
In this section, we compare our approach with other three papers that implemented the dataset in other methodologies. As shown in figure 15, the CNNet methodology that we proposed in this paper yields the highest accuracy for HAR than all the previous experiments. These experiments were using the same data that we used. It is possible to classify several postures and activities in real time. In the first paper [38], the authors used Multiclass Hardware-Friendly Support Vector Machine methodologies. They are an appealing approach for use in the exploitation of Ambient Intelligence (AmI) in daily activity monitoring for elderly people on smartphones. As shown in table 3, this proposal system achieved accuracy results of both the MC-SVM and the MC-HF-SVM with k = 8 bits 89% for the test data.



Method	MC-SVM							MC-HF-SVM $k = 8$ bits						
Activity	Walking	Upstairs	Downstairs	Standing	Sitting	Laying	*Recall %*	Walking	Upstairs	Downstairs	Standing	Sitting	Laying	*Recall %*
Walking	**109**	0	5	0	0	0	*95.6*	**109**	2	3	0	0	0	*95.6*
Upstairs	1	**95**	40	0	0	0	*69.8*	1	**98**	37	0	0	0	*72.1*
Downstairs	15	9	**119**	0	0	0	*83.2*	15	14	**114**	0	0	0	*79.7*
Standing	0	5	0	**132**	5	0	*93.0*	0	5	0	**131**	6	0	*92.2*
Sitting	0	0	0	4	**108**	0	*96.4*	0	1	0	3	**108**	0	*96.4*
Laying	0	0	0	0	0	**142**	*100*	0	0	0	0	0	**142**	*100*
Precision %	*87.2*	*87.2*	*72.6*	*97.1*	*95.6*	*100*	***89.3***	*87.2*	*81.7*	*74.0*	*97.8*	*94.7*	*100*	***89.0***

Table 3 The accuracy rate from paper [38]

In the second paper [39], the same authors also used this dataset, but they increased the numbers of the test sample from 789 to 2,947. They achieved their result by using the multiclass SVM (MC-SVM) for the 6 Activities of Daily Living (ADL) (presented in table 4). They show an overall accuracy of 96% for the test data.



	WK	WU	WD	ST	SD	LD	**Recall**
Walking	**492**	1	3	0	0	0	99%
W. Upstairs	18	**451**	2	0	0	0	96%
W. Downstairs	4	6	**410**	0	0	0	98%
Sitting	0	2	0	**432**	57	0	88%
Standing	0	0	0	14	**518**	0	97%
Laying Down	0	0	0	0	0	**537**	100%
Precision	96%	98%	99%	97%	90%	100%	**96%**

Table 4 The accuracy rate from paper [39]

The last study collected the research from 8 groups that dealt with HAR [40]. The authors divided the research into two groups. The first group includes the studies on HAR application areas such as smart homes, driving safety, and motion disorder recognition for the elderly.

Approach Implemented	**Accuracy**
OVO Multiclass linear SVM with majority voting.	96.40%
Kernel variant of learning vector quantization with metric adaptation.	96.23%
Confidence-based boosting algorithm Conf-AdaBoost.M1.	94.33%

Table 5 The accuracy rate from paper [40]

In a smart home, many researchers studied HAR that has importance due to the need of providing safer and more responsive environments to people. By using decision tree analysis, experiments obtained a variable performance ranging from 73.3% to 85.8%. Regarding driving safety and the detection of alcohol levels in car drivers, the results showed classification accuracy of 89.0% for the single-user approach and 78.0% for the multi-user case. In this study, the authors used the SVM regression model to predict alcohol levels, and they achieved their accuracy result by using either an ANN or SVM binary classifier methodology. The second group is related to HAR and collects the proposed methodologies for the classification. The HAR database was collected from the sensor recordings of 30 volunteers performing six daily activity while carrying an Android OS smartphone with an embedded accelerometer and gyroscope. Table 5 presented the three approaches and their performance of error accuracy. Finally, the accuracy that we achieved from our proposal solution is 98.58% (see chapter 4).

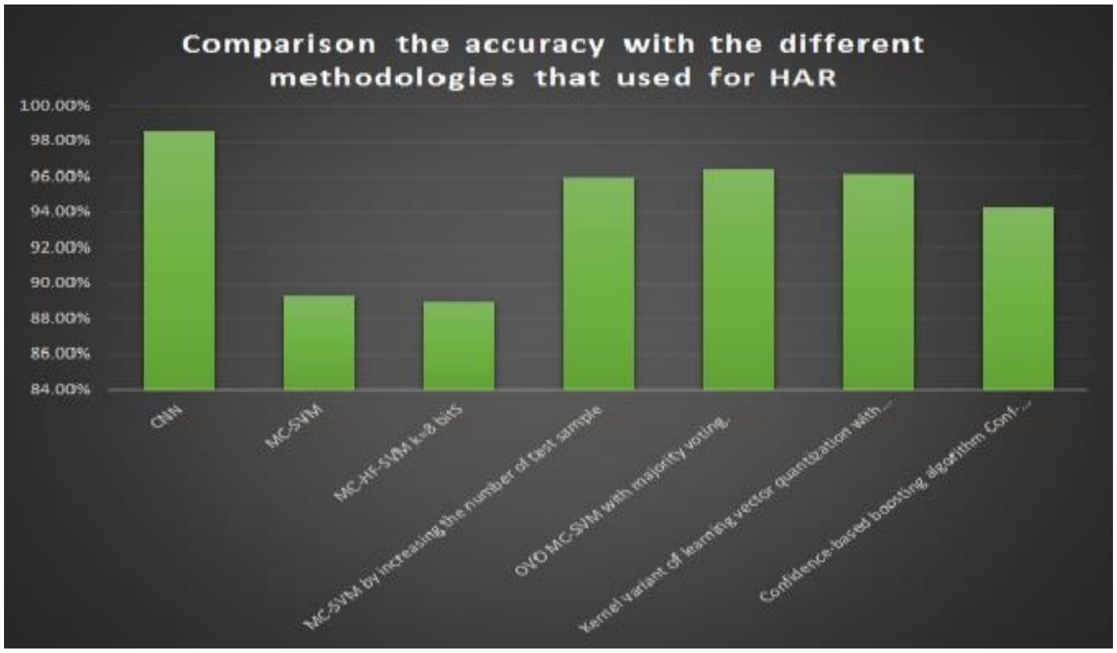


Figure 15 Comparison of the accuracy with the different methodologies that were used for HAR

CHAPTER SIX: CONCLUSION AND FUTURE WORK

The recently development technology is to try to make every device in our environment smart whenever possible. One of the goals of making smart systems is to recognize daily human activities by using a smart phone or a smart watch to encourage a safe, healthy life for people concerned about their health. In our thesis, we proposed a method to automate features from a smart phone dataset for the HAR task using a deep learning algorithm which is Convolution Neural Networks (CNNs). This proposed method builds a new deep learning machine architecture for the CNNs to investigate the multichannel time series data. It employs convolution and pooling operations to capture the salient patterns of the sensor signals at different time scales. The data that we used in our proposed software system was collected based on the HAR system by using two sensors found in a Samsung Galaxy S2 smartphone from the UCI Machine Learning Repository website. After that, we trained the data using CNNs model, which means using one algorithm for all of data that we have. We used Python language for coding this algorithm. The model training and classification are run on a computer with two GPU processors. The first one has a GPU with Intel Core i7, memory with 1733MHz clock speed and 64 GB RAM; the second one has GPU with 2560 cores, 1607 MHz clock speed and 8 GB RAM. Then, we used the same algorithm for the prediction and evaluation stages. The implementation of CNNs for this work is based on the knowledge that simulating the human nervous system to detect and identify objects will result in high accuracy over other machine learning techniques depending on all the experiments that we did on the dataset. A proposed future work is to complete this investigation with CNNs algorithm to increase our accuracy. In addition, we could use other sensors and combine them with our current sensors to improve the healthcare monitoring applications for better performance and better results. For instance, there are many applications, including surveillance systems, patient monitoring systems.

REFERENCES

[1] "Smartphone Users Worldwide 2014-2019", Statista, 2015. [Online]. Available: http://www. statista.com/statistics/330695/number-of-smartphone-users-worldwide/. [Accessed: 05- Apr-2016].

[2] D. Luxton, R. McCann, N. Bush, M. Mishkind, and G. Reger, mHealth for mental health: integrating smartphone technology in behavioral healthcare, professional Psychology: Research and Practice 2011, Vol. 42, No. 6, pp. 505–512.

[3] M. Zeng, L. Nguyen, B. Yu, O. Mengshoel, J. Zhu, P. Wu, and J. Zhang, Convolutional neural networks for human activity recognition using mobile sensors, in Mobile Computing, Applications and Services (MobiCASE), 2014 6th International Conference on, pp.197–205. IEEE, 2014.

[4] O. Lara and M. Labrador, A survey on human activity recognition using wearable sensors, IEEE Communications Surveys & Tutorials, 15(3):1192–1209, 2013.

[5] N. Haering, P. Venetianer, A. Lipton, the evolution of video surveillance: an overview, Machine Vision and Applications, 19 (2008) PP. 279–290.

[6] P. Geetha, V. Narayanan, A survey of content-based video retrieval, Computer Science 4 (6) (2008) 474–486.

[7] E. Tapia, S. Intille, and K. Larson, Activity recognition in the home setting using simple and ubiquitous sensors, in Proceedings of PERVASIVE, vol. 3001, 2004, pp. 158–175.

[8] M. Philipose, K. Fishkin, M. Perkowitz, D. Patterson, D. Fox, H. Kautz, and D. Hahnel, Inferring activities from interactions with objects, IEEE Pervasive Computing, vol. 3, no. 4, pp. 50–57, 2004.

[9] B. Logan, J. Healey, M. Philipose, E. Tapia, and S. Intille, A long- term evaluation of sensing modalities for activity recognition, in Proc. of Ubicomp, 2007.

[10] J. Ahmad, K. Shaharyar, K. Daijin, A Depth Video Sensor-Based Life-Logging Human Activity Recognition System for Elderly Care in Smart Indoor Environments, Sensors 2014, 14, PP. 11735-11759; doi:10.3390/s140711735

[11] G. Chetty and M. White, Body sensor networks for human activity recognition, in Signal Processing and Integrated Networks (SPIN), 2016 3rd International Conference on, pp. 660–665. IEEE, 2016.

[12] E. Ijjina and C. Mohan, One-shot periodic activity recognition using convolutional neural networks, in Machine Learning and Applications (ICMLA), 2014 13th International Conference on, pp. 388–391. IEEE, 2014.

[13] C. Torres-Huitzil and A. Alvarez-Landero, Accelerometer-based human activity recognition in smartphones for healthcare services, in Mobile Health, pp. 147–169, Springer, 2015.

[14] D. Anguita, A. Ghio, L. Oneto, X. Parra and J. Reyes-Ortiz, Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine, in International Workshop on Ambient Assisted Living, pp. 216–223. Springer, 2012.

[15] N. Lane and P. Georgiev, Can deep learning revolutionize mobile sensing?, in Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications, pp. 117–122. ACM, 2015.

[16] Y. Yadav and M. Gowda, Heart rate monitoring and heart attack detection using wearable device, international journal of technical research and applications e-ISSN: 2320-8163, volume 4, Issue 3 (May- June 2016), PP. 48-50.

[17] Z. He and L. Jin, Activity recognition from acceleration data based on discrete consine transform and svm, in Systems, Man and Cybernetics, 2009, SMC 2009, IEEE International Conference on, pp. 5041–5044. IEEE, 2009.

[18] C. Vollmer, H. Gross, and J. Eggert, Learning features for activity recognition with shift-invariant sparse coding, in International Conference on Artificial Neural Networks, pp. 367–374, Springer, 2013.

[19] N. Oliver and A. Garg, Layered representations for human activity recognition, proceedings of the Fourth IEEE International conference on multimodal interfaces (ICMI'02) 0-7695-1834-6/02 $17.00 © 2002 IEEE, pp. 1–6.

[20] Lara and M. Lobrador, A survey on human activity recognition using wearable sensors, IEEE communications surveys & tutorials, pp. 1192– 1209, VOL. 15, NO. 3, third quarter 2013.

[21] M. Ayu, S. Ismail, A. Abdul Matin and T. Mantoro, A comparison study of classifier algorithms for mobile-phone's accelerometer based activity recognition, international Symposium on Robotics and Intelligent Sensors 2012 (IRIS 2012), pp. 224 – 229.

[22] X. Lai, Q. Liu, X. Wei, W. Wang, G. Zhou, and G. Han, A survey of body sensor networks, Sensors, 13 (5):5406–5447, 2013.

[23] D. Yuan, X. Ma, Y. Liu, S. Yan, and C. Hao, Statistical modeling of additive noise and random drift for triaxial rate gyros and accelerometers, in Intelligent Control and Automation (WCICA), 2016 12th World Congress on, pages 308–313, IEEE, 2016.

[24] H. Cho, S. Kim, J. Baek and P. Fisher, Motion recognition with smart phone embedded 3-axis accelerometer sensor, 2012 IEEE International Conference on Systems, on pages 919– 924, Man, and Cybernetics October 14-17, 2012, COEX, Seoul, Korea.

[25] D. Johnson and M. Trivedi, driving style recognition using a smartphone as a sensor platform, 2011 14th International IEEE Conference on Intelligent Transportation Systems Washington, DC, USA. Pages 1609– 1615, October 5-7, 2011.

[26] W. Lee and R. Lee, Multi-sensor authentication to improve smartphone security, in: Conference on Information Systems Security and Privacy, 2015, pp. 1-11.

[27] G. Xu and H. Wu, Structural design of convolutional neural networks for steganalysis, IEEE SIGNAL PROCESSING LETTERS, on pagers 703– 712, VOL. 23, NO. 5, MAY 2016.

[28] J. Yang, M. Nguyen, P. San, X. Li Li, and S. Krishnaswamy, Deep convolutional neural networks on multichannel time series for human activity recognition, 2015.

[29] The UCI Machine Learning Repository, Smartphone-based recognition of human activities and postural transitions data set, 2015.

[30] L. Deng and D. Yu, deep learning: methods and applications, foundations and trends ® in signal processing, Vol. 7, Nos. 3–4 (2013) 197–387.

[31] D. Ciresan, U. Meier, J. Masci, L. Gambardella and J. Schmidhuber, high performance neural networks for visual object classification, technical report IDSIA- 01-11, Dalle Molle Institute for Artificial Intelligence, pp. 1-11, 2011.

[32] D. C. Ciresan, U. Meier, and J. Schmidhuber, Multi-column deep neural networks for image classification, technical Report IDSIA-04-12, Dalle Molle Institute for Artificial Intelligence, pp. 1-19, 2012.

[33] J. Bouvrie, Notes on convolutional neural networks. pp. 1-8 at 2006.

[34] F. Radzi, S. Liew, M. Khalil-Hani and R. Bakhteri, Convolutional neural networks with fused layers applied to face recognition, international Journal of Computational Intelligence and Applications, pp. 1-18, 2015.

[35] A. Vedaldi, K. Lenc and A. Gupta, MatConvNet convolutional neural networks for MATLAB.

[36] E. Jones, An introduction to neural networks: A white paper, The United States of America, pp. 1-39, 2004.

[37] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mane ́, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Vie ́gas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu and X. Zheng, TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems, google research, PP 1- 19, November 2015.

[38] D. Anguita, A. Ghio, L. Oneto, X. Parra and J. Reyes-Ortiz, Human Activity Recognition on Smartphones using a Multiclass Hardware-Friendly Support Vector Machine, 4th International Workshop of Ambient Assited Living, IWAAL 2012, Vitoria-Gasteiz, Spain, December 3-5, 2012, Proceedings. Lecture Notes in Computer Science 2012, pp 216-223.

[39] D. Anguita, A. Ghio, L. Oneto, X. Parra and J. Reyes-Ortiz, A Public Domain Dataset for Human Activity Recognition Using Smartphones, 21th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, ESANN 2013, Bruges, Belgium 24-26 April 2013.

[40] J. Reyes-Ortiz, A. Ghio, X. Parra-Llanas, D. Anguita, J. Cabestany, A. CatalÃ, Human Activity and Motion Disorder Recognition: Towards Smarter Interactive Cognitive Environments, 21th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, ESANN 2013, Bruges, Belgium 24-26 April 2013.

[41] J. Fingas, Engineer Guy shows how a phone accelerometer works, knows what's up and sideways (video).

[42] N. Srikar, what are different sensors are available inside a smartphone...?

[43] https://github.com/guillaume-chevalier/LSTM-Human-Activity-Recognition.

APPENDICES

APPENDIX A

Layer (type)	Output Shape	Param #	Connected to
convolution1d_1 (Convolution1D)	(None, 561, 32)	128	convolution1d_input_1[0][0]
activation_1 (Activation)	(None, 561, 32)	0	convolution1d_1[0][0]
dropout_1 (Dropout)	(None, 561, 32)	0	activation_1[0][0]
maxpooling1d_1 (MaxPooling1D)	(None, 280, 32)	0	dropout_1[0][0]
convolution1d_2 (Convolution1D)	(None, 280, 32)	3104	maxpooling1d_1[0][0]
convolution1d_3 (Convolution1D)	(None, 280, 32)	3104	convolution1d_2[0][0]
activation_2 (Activation)	(None, 280, 32)	0	convolution1d_3[0][0]
dropout_2 (Dropout)	(None, 280, 32)	0	activation_2[0][0]
maxpooling1d_2 (MaxPooling1D)	(None, 140, 32)	0	dropout_2[0][0]
convolution1d_4 (Convolution1D)	(None, 140, 128)	12416	maxpooling1d_2[0][0]
convolution1d_5 (Convolution1D)	(None, 140, 128)	49280	convolution1d_4[0][0]
dropout_3 (Dropout)	(None, 140, 128)	0	convolution1d_5[0][0]
maxpooling1d_3 (MaxPooling1D)	(None, 70, 128)	0	dropout_3[0][0]
convolution1d_6 (Convolution1D)	(None, 70, 64)	24640	maxpooling1d_3[0][0]
convolution1d_7 (Convolution1D)	(None, 70, 64)	12352	convolution1d_6[0][0]
dropout_4 (Dropout)	(None, 70, 64)	0	convolution1d_7[0][0]
maxpooling1d_4 (MaxPooling1D)	(None, 35, 64)	0	dropout_4[0][0]
flatten_1 (Flatten)	(None, 2240)	0	maxpooling1d_4[0][0]
dense_1 (Dense)	(None, 200)	448200	flatten_1[0][0]
dropout_5 (Dropout)	(None, 200)	0	dense_1[0][0]

```
dense_2 (Dense)                    (None, 50)     10050            dropout_5[0][0]
________________________________________________________________________________

dropout_6 (Dropout)                (None, 50)         0            dense_2[0][0]
________________________________________________________________________________
dense_3 (Dense)                    (None, 20)       1020           dropout_6[0][0]
________________________________________________________________________________
dropout_7 (Dropout)                (None, 20)        0             dense_3[0][0]
________________________________________________________________________________
dense_4 (Dense)                    (None, 7)        147            dropout_7[0][0]
________________________________________________________________________________
activation_3 (Activation)          (None, 7)         0             dense_4[0][0]
================================================================================
Total params: 1, 012,505
Trainable params: 1, 012,505
Non-trainable params: 0
```

APPENDIX B

```
Layer (type)                       Output Shape     Param #       Connected to
================================================================================
convolution1d_1 (Convolution1D)    (None, 561, 32)  128           convolution1d_input_1[0][0]

activation_1 (Activation)          (None, 561, 32)  0             convolution1d_1[0][0]

dropout_1 (Dropout)                (None, 561, 32)  0             activation_1[0][0]

maxpooling1d_1 (MaxPooling1D)      (None, 280, 32)  0             dropout_1[0][0]

convolution1d_2 (Convolution1D)    (None, 280, 32)  3104          maxpooling1d_1[0][0]

convolution1d_3 (Convolution1D)    (None, 280, 32)  3104          convolution1d_2[0][0]

activation_2 (Activation)          (None, 280, 32)  0             convolution1d_3[0][0]

dropout_2 (Dropout)                (None, 280, 32)  0             activation_2[0][0]

maxpooling1d_2 (MaxPooling1D)      (None, 140, 32)  0             dropout_2[0][0]

convolution1d_4 (Convolution1D)    (None, 140, 32)  3104          maxpooling1d_2[0][0]

convolution1d_5 (Convolution1D)    (None, 140, 32)  3104          convolution1d_4[0][0]

activation_3 (Activation)          (None, 140, 32)  0             convolution1d_5[0][0]

dropout_3 (Dropout)                (None, 140, 32)  0             activation_3[0][0]

maxpooling1d_3 (MaxPooling1D)      (None, 70, 32)   0             dropout_3[0][0]

convolution1d_6 (Convolution1D)    (None, 70, 128)  12416         maxpooling1d_3[0][0]

convolution1d_7 (Convolution1D)    (None, 70, 128)  49280         convolution1d_6[0][0]

dropout_4 (Dropout)                (None, 70, 128)  0             convolution1d_7[0][0]

maxpooling1d_4 (MaxPooling1D)      (None, 35, 128)  0             dropout_4[0][0]

convolution1d_8 (Convolution1D)    (None, 35, 128)  49280         maxpooling1d_4[0][0]

convolution1d_9 (Convolution1D)    (None, 35, 128)  49280         convolution1d_8[0][0]
```

```
dropout_5 (Dropout)                  (None, 35, 128)    0        convolution1d_9[0][0]
maxpooling1d_5 (MaxPooling1D)        (None, 17, 128)    0        dropout_5[0][0]
convolution1d_10 (Convolution1D)     (None, 17, 64)     24640    maxpooling1d_5[0][0]
convolution1d_11 (Convolution1D)     (None, 17, 64)     12352    convolution1d_10[0][0]
dropout_6 (Dropout)                  (None, 17, 64)     0        convolution1d_11[0][0]
maxpooling1d_6 (MaxPooling1D)        (None, 8, 64)      0        dropout_6[0][0]
convolution1d_12 (Convolution1D)     (None, 8, 64)      12352    maxpooling1d_6[0][0]
convolution1d_13 (Convolution1D)     (None, 8, 64)      12352    convolution1d_12[0][0]
dropout_7 (Dropout)                  (None, 8, 64)      0        convolution1d_13[0][0]
maxpooling1d_7 (MaxPooling1D)        (None, 4, 64)      0        dropout_7[0][0]
flatten_1 (Flatten)                  (None, 256)        0        maxpooling1d_7[0][0]
dense_1 (Dense)                      (None, 200)        51400    flatten_1[0][0]
dropout_8 (Dropout)                  (None, 200)        0        dense_1[0][0]
dense_2 (Dense)                      (None, 50)         10050    dropout_8[0][0]
dropout_9 (Dropout)                  (None, 50)         0        dense_2[0][0]
dense_3 (Dense)                      (None, 20)         1020     dropout_9[0][0]
dropout_10 (Dropout)                 (None, 20)         0        dense_3[0][0]
dense_4 (Dense)                      (None, 7)          147      dropout_10[0][0]
activation_4 (Activation)            (None, 7)          0        dense_4[0][0]
==========================================================================
Total params: 297,113
Trainable params: 297,113
Non-trainable params: 0
```

APPENDIX C

```
Layer (type)                              Output Shape      Param #          Connected to
=====================================================================================
convolution1d_1 (Convolution1D)      (None, 561, 32)       128     convolution1d_input_1[0][0]
_____________________________________________________________________________________
activation_1 (Activation)            (None, 561, 32)        0            convolution1d_1[0][0]
_____________________________________________________________________________________
dropout_1 (Dropout)                  (None, 561, 32)        0                activation_1[0][0]
_____________________________________________________________________________________
maxpooling1d_1 (MaxPooling1D)        (None, 280, 32)        0                   dropout_1[0][0]
_____________________________________________________________________________________
convolution1d_2 (Convolution1D)      (None, 280, 128)     12416            maxpooling1d_1[0][0]
_____________________________________________________________________________________
convolution1d_3 (Convolution1D)      (None, 280, 128)     49280           convolution1d_2[0][0]
_____________________________________________________________________________________
dropout_2 (Dropout)                  (None, 280, 128)       0             convolution1d_3[0][0]
_____________________________________________________________________________________
maxpooling1d_2 (MaxPooling1D)        (None, 140, 128)       0                   dropout_2[0][0]
_____________________________________________________________________________________
flatten_1 (Flatten)                  (None, 17920)          0            maxpooling1d_2[0][0]
_____________________________________________________________________________________
dense_1 (Dense)                      (None, 200)          3584200                flatten_1[0][0]
_____________________________________________________________________________________
dropout_3 (Dropout)                  (None, 200)            0                   dense_1[0][0]
_____________________________________________________________________________________
dense_2 (Dense)                      (None, 7)            1407                dropout_3[0][0]
_____________________________________________________________________________________
activation_2 (Activation)            (None, 7)              0                   dense_2[0][0]
=====================================================================================
Total params: 3,647,431
Trainable params: 3,647,431
Non-trainable params: 0
```

APPENDIX D

The developed software for the Convolution Neural Networks based on Human Activity Recognition can be found in the link below.

https://github.com/MishkhalIsraa/CNNet/blob/master/test/test.py