HUMAN ACTIVITY RECOGNITION USING SMARTPHONE SENSORS

A PROJECT REPORT

Submitted by

ASWIN S	**(923317104005)**
PRAVEENKUMAR I	**(923317104040)**
SATHISHKUMAR S	**(923317104047)**
VIGNESH D	**(923317104052)**

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

GOVERNMENT COLLEGE OF ENGINEERING

BODINAYAKKANUR – 625 582

ANNA UNIVERSITY :: CHENNAI 600 025

MARCH-2021

ANNA UNIVERSITY :: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report "**HUMAN ACTIVITY RECOGNITION USING SMARTPHONE SENSORS**" is the bonafide work of **"ASWIN S (923317104005), PRAVEEN I (923317104040), SATHISHKUMAR S (923317104047)** and **VIGNESH D (923317104052)"** who carried out the project work under my supervision during the Academic Year 2020-2021.

SIGNATURE

Dr.D.Mary Sugantharathnam, M.E., Ph.D.,
HEAD OF THE DEPARTMENT
Computer Science and Engineering
Government College of Engineering
Bodinayakannur – 625 582

SIGNATURE

Dr.V.Ramya
SUPERVISOR
Computer Science and Engineering
Government College of Engineering
Bodinayakannur – 625 582

Submitted for **MINI PROJECT (CS8611)** Viva Voce Examination held at Government College of Engineering, Bodinayakannur on ……

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We acknowledge with great pleasure, deep satisfaction and gratitude the contribution of many individuals in the successful completion of this project.

We express our special thanks of profound gratitude to **Dr.S.JAYANTHI Principal**, Government College of Engineering, Bodinayakkanur, for all the support and encouragement given to us throughout our project work.

We thank **Dr.D.MARY SUGANTHARATHNAM, Head Of the Department**, computer science and engineering, Government College of Engineering, Bodinayakkanur for her valuable suggestions throughout our project.

We express our gratitude to our guide **Dr.V.RAMYA, Associate Professor**, Computer Science and Engineering, Government College of Engineering, Bodinayakkanur, for her guidance and help in doing this project work successfully.

S.ASWIN

I.PRAVEEN

S.SATHISHKUMAR

D.VIGNESH

TABLE OF CONTENT

ABSTRACT

Activity recognition is one of the most important technology behind many applications such as medical research, human survey system and it is an active research topic in health care and smart homes. Smart phones are equipped with various built-in sensing platforms like accelerometer, gyroscope, GPS, compass sensor and barometer, we can design a system to capture the state of the user.

Activity recognition system takes the raw sensor reading from mobile sensors as inputs and estimates a human motion activity using data mining and machine learning techniques. In this paper, we analyze the performance of SVM and CNN classification algorithm.Usually first we use the KNN classification algorithm and next we utilize an improvement of Support Vector Machine(SVM) and Convolutional neural Network(CNN) classification algorithm.

We can predict the performance of these classifier from a series of observations on human activities like walking, running, lying down, sitting, going upstairs, going downstairs and standing in an online activity recognition system. In this paper, we are intended to analyze the performance of classifier with limited training data and limited accessible memory on the phones compared to off-line.

LIST OF FIGURES

LIST OF SYMBOLS

S.No	ACRONYM	ABBREVATION
1.	SVM	Support Vector Machine
2.	CNN	Convolutional Neural Network
3.	KNN	K-Nearest Neighbors
4.	RAM	Random Access Memory
5.	IDE	Integrated development environment
6.	OS	Operating System

CHAPTER 1

INTRODUCTION

1.1 About the project

Nowadays smartphones became more and more popular in human daily life. Most of the people used it for searching news, watching videos, playing games and accessing social network but there were many useful studies on smartphones. Activity recognition is one of the most important technologies behind many applications on smartphone such as health monitoring, fall detection, context-aware mobile applications, human survey system and home automation etc., Smartphone-based activity recognition system is an active area of research because they can lead to new types of mobile applications.

Understanding human activities creating a demand in health-care domain, especially in rehabilitation assistance, physiotherapist assistance, and elder care support services and cognitive impairment. Sensors will record and monitor the patient‘s activities and report automatically when any abnormality is detected, so, huge amount of resources can be saved. Other

applications like human survey system and location indicator are all getting benefits from this study.

Training process is always necessary when a new activity is added in to the system. The same algorithm parameters are needed to be trained and fine-tuned when the algorithm runs on different devices with various built-in sensors. However, labeling a training data (time-series data) is a time consuming procedure and it is not always possible to label all the training data by the users. As a result, we present an active learning technique to accelerate the training process. Given a SVM classifier, an active learning technique intuitively queries the unlabeled training samples and learns the parameters from the correct labels answered by the human. In this way, users will label only the samples that the algorithm demanded to do and the total amount of required training samples is reduced.

In this paper, we are also interested in analyzing the performance of classifiers with limited training data considering the limited memory available on the phones. In this system, we can collect the training data in a few minutes and it can be directly used for classification steps, which reduce the burden on the users. Being one of the first Android applications used for human activity recognition is another important motivation for this study.

In the literature, it has been reported that minimum distance classifier does not work well when used alone. SVM results are always better than KNN classifier in terms of accuracy. However, SVM requires high computational burden so, it is not an online classifier and due to limited resources on smart phone, it does not appear as a preferable method

1.2 Background of Study

From background study, we have shown that it is difficult for classifiers to classify similar activities like going upstairs and going downstairs, fast walking and slow running etc.As for the choice of classifier, one can use either temporal classifiers or non-temporal ones. In case of non-temporal classifiers like Decision Tree, Artificial Neural Network, k-Nearest Neighbor, K-Mean etc, one cannot provide the raw data as input directly to the classifier. First, one needs to extract some features from the raw data and then pass these features to the classifier. So there is a preprocessing on the raw data before sending it to classifier. In this case, the features that are mostly used are the following:

- Arithmetic Mean
- Standard Deviation

- Max, Min
- Median Absolute Deviation
- Frequency Signal Weighted Average

In case of temporal classifiers like AR and DTW, etc., there is no need for feature extraction. These algorithms take input of data as a time series. Moreover, one can easily plot the accelerometer and gyroscope data acquired from the smart phone into a time series. Hence, a temporal classifier is a better choice for classifying activities using the data acquired from smart phone sensors.

1.3 System Architecture

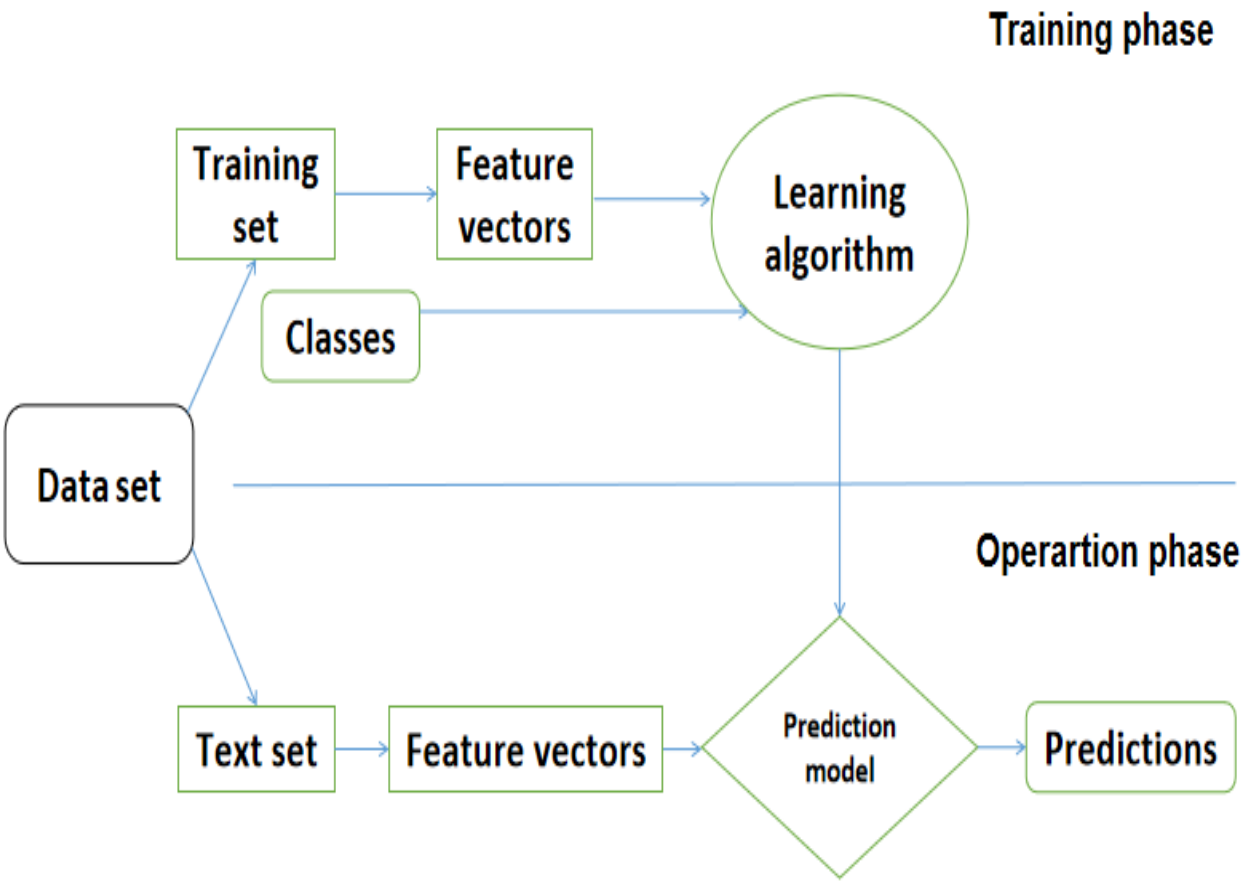


Fig 1.1 SYSTEM ARCHITECTURE

CHAPTER 2

LITERATURE REVIEW

1. S.Ali and M. Shah, "Human action recognition in videos using kinematic features and multiple instance learning", *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 32, no. 2, pp. 288-303, Feb. 2010.

We propose a set of kinematic features that are derived from the optical flow for human action recognition in videos. The set of kinematic features includes divergence, vorticity, symmetric and antisymmetric flow fields, second and third principal invariants of flow gradient and rate of strain tensor, and third principal invariant of rate of rotation tensor. Each kinematic feature, when computed from the optical flow of a sequence of images, gives rise to a spatiotemporal pattern. It is then assumed that the representative dynamics of the optical flow are captured by these spatiotemporal patterns in the form of dominant kinematic trends or kinematic modes. These kinematic modes are computed by performing principal component analysis (PCA) on the spatiotemporal volumes of the kinematic features. For classification, we propose the use of multiple instance learning (MIL) in which each action

video is represented by a bag of kinematic modes. Each video is then embedded into a kinematic-mode-based feature space and the coordinates of the video in that space are used for classification using the nearest neighbor algorithm. The qualitative and quantitative results are reported on the benchmark data sets.

2. A. Oikonomopoulos and M. Pantie, "Human Activity Recognition Using Hierarchically-Mined Feature Constellations", pp. 150-159, 2013.

In this paper we address the problem of human activity modelling and recognition by means of a hierarchical representation of mined dense spatiotemporal features. At each level of the hierarchy, the proposed method selects feature constellations that are increasingly discriminative and characteristic of a specific action category, by taking into account how frequently they occur in that action category versus the rest of the available action categories in the training dataset. Each feature constellation consists of n-tuples of features selected in the previous level of the hierarchy and lying within a small spatiotemporal neighborhood. We use spatiotemporal Local Steering Kernel (LSK) features as a basis for our representation, due to their ability and efficiency in capturing the

local structure and dynamics of the underlying activities. The proposed method is able to detect activities in unconstrained videos, by back-projecting the activated features at the locations at which they were activated. We test the proposed method on two publicly available datasets, namely the KTH and YouTube datasets of human bodily actions. The acquired results demonstrate the effectiveness of the proposed method in recognising a wide variety of activities.

3. M. Javan Roshtkhari and M. D. Levine, "Human activity recognition in videos using a single example", *Image Vis. Comput.*, vol. 31, no. 11, pp. 864-876, Nov. 2013.

This paper presents a novel approach for action recognition, localization and video matching based on a hierarchical code-book model of local spatio-temporal video volumes. Given a single example of an activity as a query video, the proposed method finds similar videos to the query in a target video dataset. The method is based on the bag of video words (BOV) representation and does not require prior knowledge about actions, background subtraction, motion estimation or tracking. The hierarchical algorithm codes a video as a compact set of spatio-temporal

volumes, while considering their spatio-temporal compositions in order to account for spatial and temporal contextual information. This hierarchy is achieved by first constructing a codebook of spatio-temporal video volumes. Then a large contextual volume containing many spatio-temporal volumes (ensemble of volumes) is considered. These ensembles are used to construct a probabilistic model of video volumes and their spatio-temporal compositions. The algorithm was applied to three available video datasets for action recognition with different complexities (KTH, Weizmann, and MSR II) and the results were superior to other approaches, especially in the case of a single training example and cross-dataset action recognition.

4. A. A. Chaaraoui, J. R. Padilla-López, P. Climent-Pérez and F. Flórez-Revuelta, "Evolutionary joint selection to improve human action recognition with RGB-D devices", *Expert Syst. Appl.*, vol. 41, no. 3, pp. 786-794, Feb. 2014.

Interest in RGB-D devices is increasing due to their low price and the wide range of possible applications that come along. These devices provide a marker-less body pose estimation by means of skeletal data consisting of 3D positions of body joints. These can be further used for pose, gesture or action

recognition. In this work, an evolutionary algorithm is used to determine the optimal subset of skeleton joints, taking into account the topological structure of the skeleton, in order to improve the final success rate. The proposed method has been validated using a state-of-the-art RGB action recognition approach, and applying it to the *MSR-Action3D* dataset. Results show that the proposed algorithm is able to significantly improve the initial recognition rate and to yield similar or better success rates than the state-of-the-art methods.

5. N. Noorit and N. Suvonvorn, "Human Activity Recognition from Basic Actions Using Finite State Machine", *Proceedings of the First International Conference on Advanced Data and Information Engineering (DaEng-2013)*, vol. 285, pp. 379-386, 2014

High-level human activity recognition is an important method for the automatic event detection and recognition application, such as, surveillance system and patient monitoring system. In this paper, we propose a human activity recognition method based on FSM model. . The action stream with related features (movement, referenced location) is recognized using the predefined FSM recognizer modeling based on rational activity. Our experimental result shows a good recognition accuracy (86.96 % in average).

CHAPTER 3

SYSTEM ANALYSIS

3.1 Objective

- The goal of human activity recognition is to examine activities from mobile phone sensors like accelerometer, gyroscope, etc. Motivated by this fact, human activity recognition systems aim to correctly classify input data into its underlying activity category.
- SVM(Support Vector Machine) algorithm is used for predicting the human activity recognition.

3.2 Existing System

Algorithms used are Binary Decision Tree, Decision Tree(20), Decision Tree(100), k-NN(k=1), k-NN(k=3) and AdaBoost with an accuracy rate of 53.1%, 91.7%, 94.4%, 96.1%, 95.5%, 95.4% respectively.

3.2.1 Disadvantage

- The User Interface(UI) is not user friendly and the User Experience(UX) is not great.
- The model is not faster than expected .

CHAPTER 4

SYSTEM REQUIREMENTS

4.1 Software Requirements

OS	Windows 8.0/10/Mac
Language	64-bit python
IDE	Jupiter

4.2 Hardware Requirements

Processor	Intel(R) Pentium(R) CPU AI018 2.10GHz
RAM	4 GB
Hard Disk	500 GB

CHAPTER 5

SYSTEM DESIGN

5.1 Flow Graph

A Flow Diagram is a graphical representation of the "flow" of data through an information system, modeling its process aspects. It is often used as a preliminary step to create an overview of the system without going into great detail. Flow graph for Human Activity Recognition is given in Fig. 5.1.

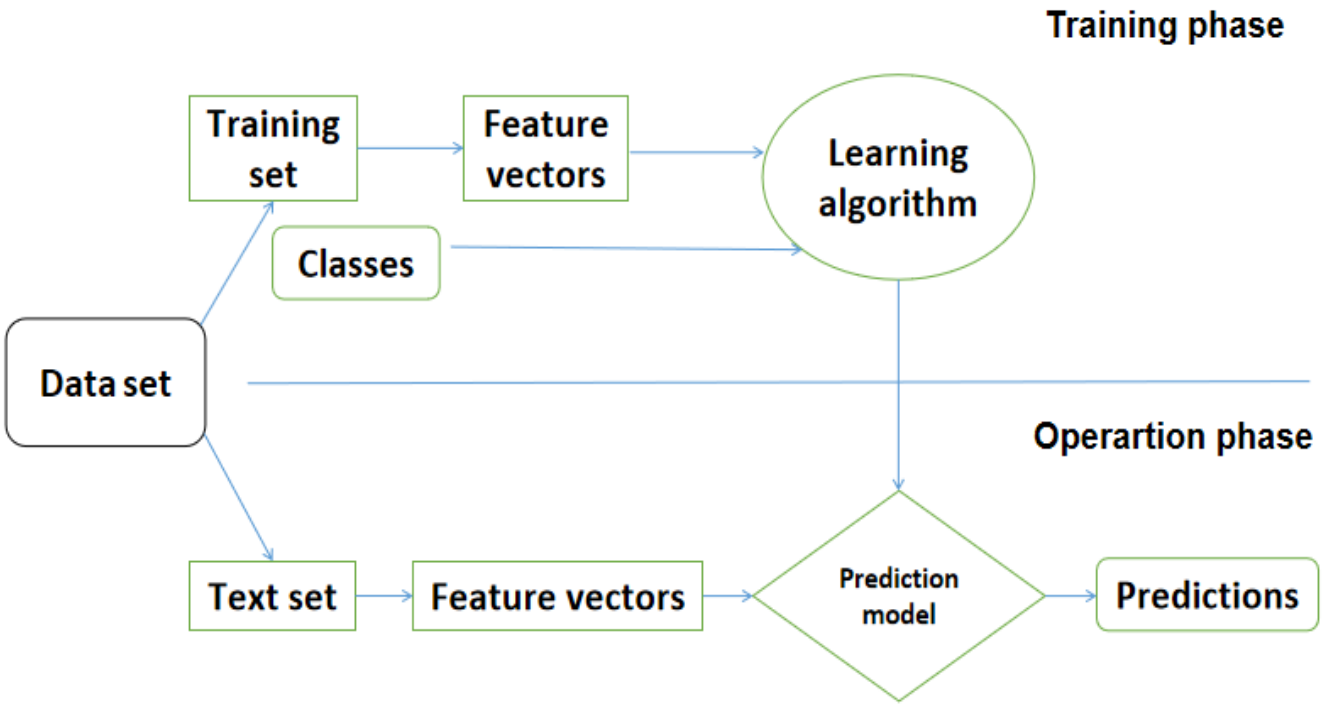
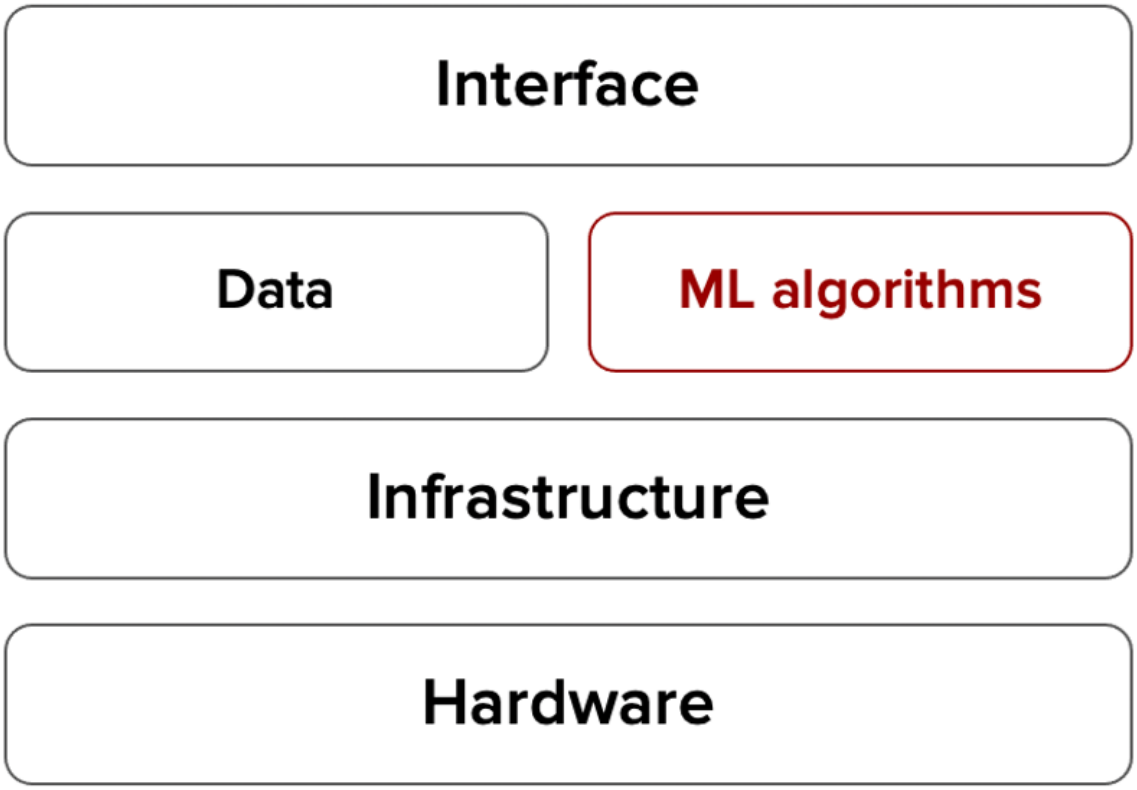


Fig. 5.1 FLOW GRAPH

5.2 Schematic Diagram for ML System



For, designing the ML system we need to consider all the above-Data, ML algorithms, related infrastructure, hardware, and Interface. Few of the high-level considerations for design,

Data

1. Feature expectations are captured in a schema – ranges of the feature values well captured to avoid any unexpected value, which might result in garbage response e.g. human age/height have expected value range, it can't be very large e.g. age value 150+, height – 10 feet, etc.

2. All features are beneficial – features added in the system should have some usefulness in terms of predictive power or some identifier because each feature added has an associated handling cost.

3. No feature's cost is too much– trade-off of cost vs benefits must be done for each feature added to remove features which has too little to add and too much to handle.

4. Features adhere to meta-level requirements – features used should match the project requirements e.g. there might be certain features that can't be used in the model e.g. sex, age, race, etc.

5. The data pipeline has appropriate privacy controls – e.g. personally identifiable information (PII) should be properly handled, because any leakage of this data may lead to legal consequences.

6. New features that can be added quickly-this will help in adding new features to improve system performance if any new external factor is impacting the system.

7. All input feature code is tested – all features e.g. one hot encoding/binning features or any other transformations code must be tested to avoid any intermediate values going off the expected range e.g. handling of unseen levels in one-hot encoded features.

Model

1. Model specs are reviewed and submitted– proper versioning of the model learning code is needed for faster re-training.

2. Offline and online metrics correlate– model metrics (log loss, mape, mse) should well correlated with the objective of application e.g. revenue/cost/time.

3. All hyperparameters have been tuned– hyperparameters, such as learning rates, number of layers, layer sizes, max depth, and regularization coefficients must be tuned for the use case. Because the choice of hyperparameter values can have a dramatic impact on prediction quality.

4. The impact of model staleness is known-how frequently re-train models based on changes in data distribution should be known to serve the most updated model in production.

5. A simpler model is not better - Bench-marking models with baseline i.e. simple linear model with high-level features is an effective strategy for functional testing and doing cost/benefit trade-off analysis against sophisticated models.

6. Model quality is sufficient on important data slices-model performance must be vetted against sufficiently representative data.

7. The model is tested for considerations of inclusion-model features should be well-vetted against importance in forecasting as in some applications certain features may bias results towards certain categories mostly for fairness purposes.

Infrastructure

1. Training is reproducible-training twice on the same data should produce two identical models. Generally, there might be some variations based on the

precision of the system/infra used. But, there should not be any major difference.

2. Model specs are unit tested-It is important to unit test model algorithm correctness and model API service through random inputs to detect any error in code/response.

3. The ML pipeline is Integration tested– complete ML pipeline – assembling of training data, feature generation, model training, model verification, and deployment to a serving system must be tested for the correct function of the ML system.

4. Model quality is validated before serving-After a model is trained but before it actually serves the real requests, an offline/online system needs to inspect it and verify that its quality is sufficient.

5. The model is debuggable-When someone finds a case where the model is behaving bizarrely, it should be well logged to make it easy to debug.

6. Serving models can be rolled back- considering the behavior of ML systems which performance very much depends on non-stationary quality/distribution of input data, there should be well designed fallback mechanism if something went wrong in ML response.

Monitoring

1. Dependency changes result in notification-any changes in the downstream inputs of the ML system should be immediately notified to quickly check for any ML performance deterioration.

2. Data invariants hold for inputs-input data quality and distribution should remain statistically constant, and if any significant data drift is observed, the model should be refreshed/re-trained accordingly.

3. Training and serving are not skewed– feature generation code for both training and inference should be the same.

4. Models are numerically stable-Invalid or implausible numeric values that can potentially crop up during model training without triggering explicit errors, and knowing that they have occurred can speed diagnosis of the problem.

5. Computing performance has not regressed-model has not experienced regressions in training speed, serving latency, throughput.

6. Prediction quality has not regressed– Check for model performance metrics as soon as in production, after serving prediction request, true levels

are available. If there is any significant model performance drift is observed, re-fresh/re-train the model.

Another important consideration that comes into ML system design is how to serve prediction, batch vs online based on business and resource availability requirements.

CHAPTER 6

PROPOSED SYSTEM

6.1 Proposed method

1. The Algorithm used is SVM(Support Vector machine) with the accuracy of 96.57%.
2. These sub-spaces are the ones that provide the largest margin separation from the classes of the training data with the intention of providing a model with low generalization error for its use with unseen data samples. SVMs are the basis for the classification of activities in this work.
3. When compared with other algorithms, SVM gives a better result.

6.1.2. Feature Extraction

After collecting the data, it had to go through a transformation process in order to extract features that provide all the necessary information to the algorithm used for ML. For every set of readings, we computed five types of features, each generating a number of inputs for the learning algorithm. A brief description of the features can be found below.

6.1.3 Average

There were nine inputs for this feature, which represented the average value of readings per axis, computed as follows (where N is the number of readings for each sensor per 10 s, for this and all the following equations):

$$1N\sum i=1Nxi$$

6.1.4 Average Absolute Difference

This feature (also with nine inputs) is the average absolute difference between the value of each of the readings and the mean value, for each axis, computed as:

$$1N\sum i=1N|xi-\mu|$$

6.1.5 Standard Deviation

The standard deviation was employed to quantify the variation of readings from the mean value, for each axis (resulting in nine inputs):

$$1N\sum i=1Nxi-\mu----------\sqrt{\ }$$

6.1.6. Average Resultant Acceleration

This feature, having the inputs, was computed as the average of the square roots of the sum of the squared value of each reading:

$$1N\sum i=1Nxi2+yi2+zi2−−−−−−−−−−−−√$$

(4)

6.1.7. Histogram

Finally, the histogram implies finding the marginal values for each axis (minimum -maximum), dividing that range into ten equal-sized intervals and determining what percentage of readings fall within each of the intervals (resulting in 90 inputs):

$$1N\sum i=1N[(xiinbj)→1,j=1…10]$$

6.2 Advantages

1. The UI(User interface) and the UX(User Experience) are good compared to the base paper.

2. The model is faster.

3. Health monitoring, fall detection, context-aware mobile applications, human survey system and home automation etc.,

4. Both users and capabilities(sensors) of smartphones increase and users usually carry their smartphone with them

CHAPTER 7

ALGORITHM

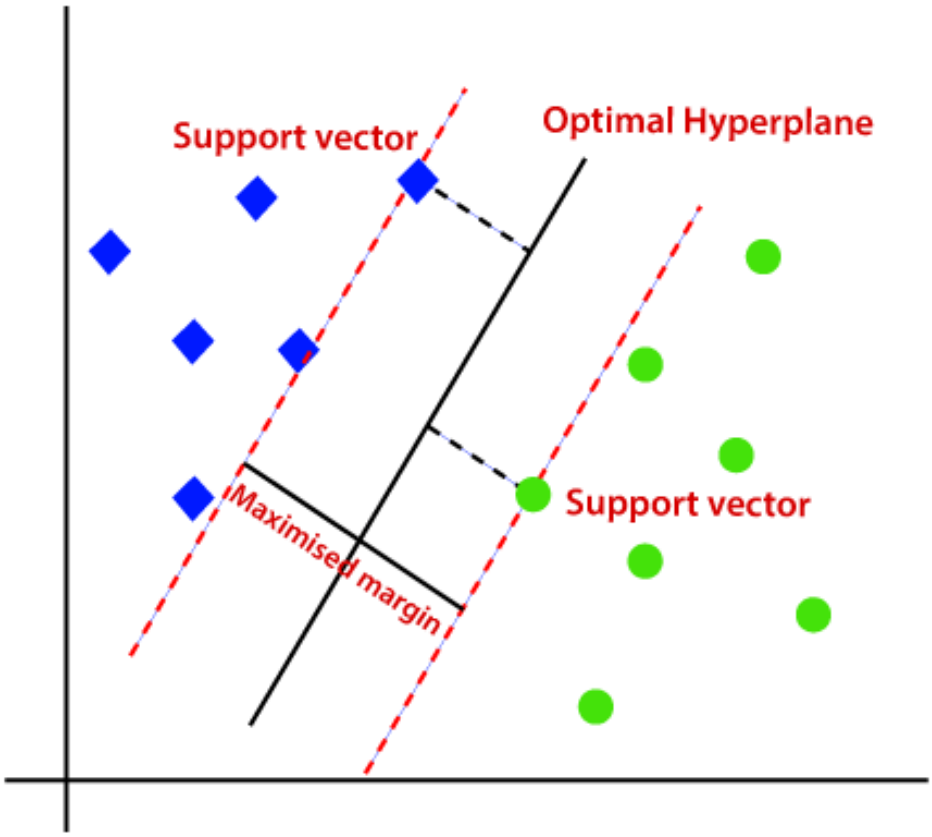
7.1 Support Vector Machine

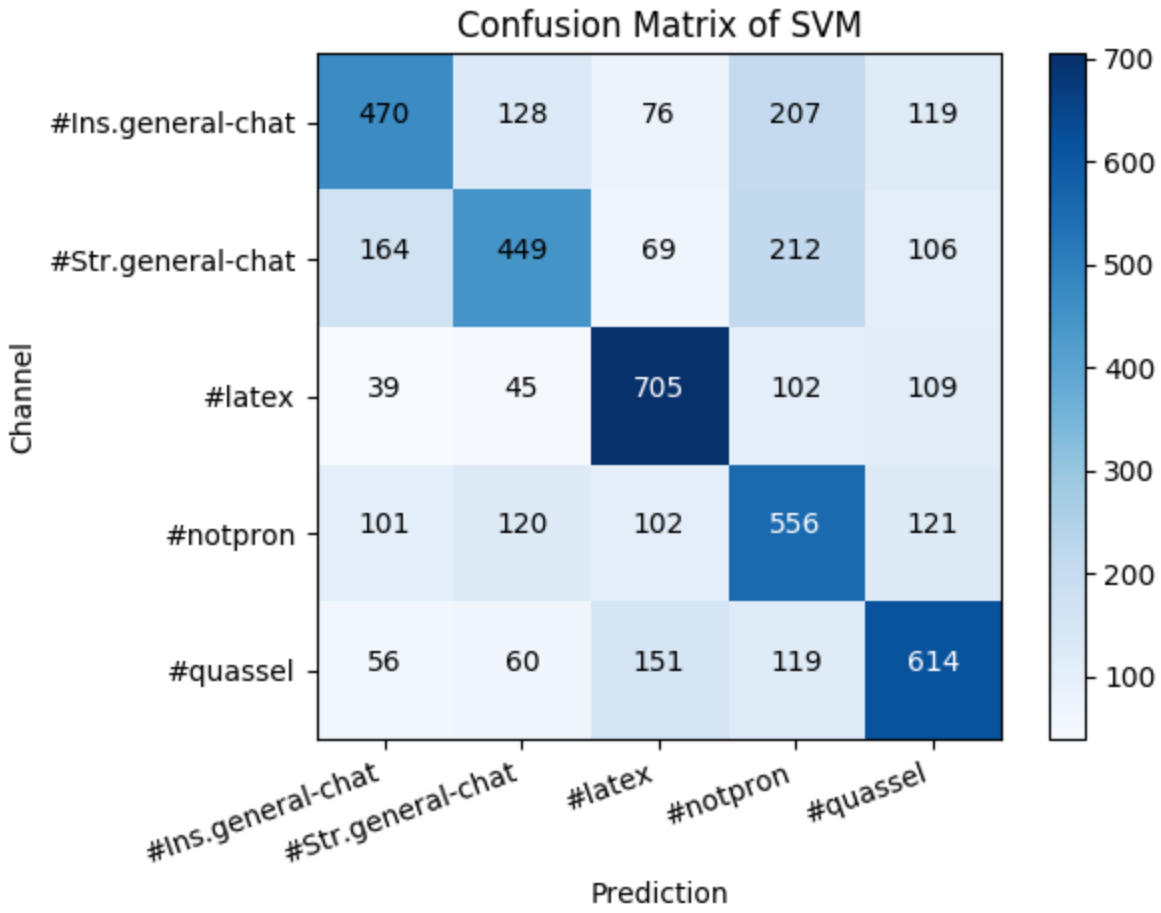
SVM (Support Vector Machine) A Support vector machine is one of the most commonly used supervised ML algorithms with the aim of solving linear and non-linear binary classification problems. Afterward, this algorithm has been adapted for its application in multi-class classification and regression analysis. The SVM for classification is a deterministic approach that aims to find the hyper planes that best separate the data into classes.

These sub-spaces are the ones that provide the largest margin separation from the classes of the training data with the intention of providing a model with low generalization error for its use with unseen data samples. SVMs are the basis for the classification of activities in this work. For this reason, we now introduce them, starting from the binary SVM model which is its simplest representation, to the extended case that allows

the classification of more than two classes: the multiclass SVM. This algorithm will be further revised throughout the development of this research to tackle specific requirements for our application in aspects such as kernel type, arithmetic used and algorithm output type.

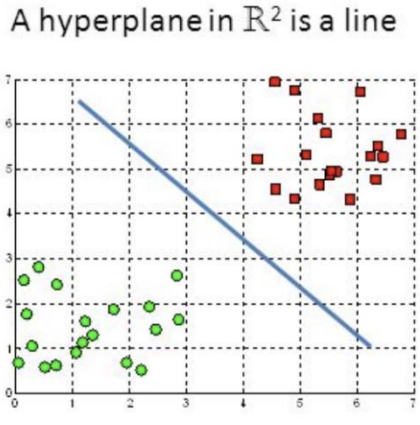
7.2 Diagram

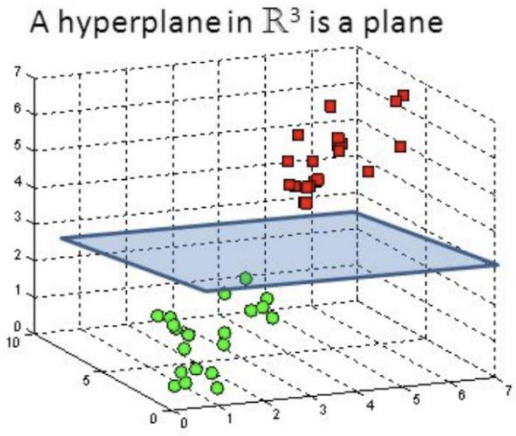



Confusion Matrix of SVM
#Ins.general-chat
#Str.general-chat
#latex
#notpron
#quassel
Channel
470 128 76 207 119
164 449 69 212 106
39 45 705 102 109
101 120 102 556 121
56 60 151 119 614
700
600
500
400
300
200
100
#Ins.general-chat
#Str.general-chat
#latex
#notpron
#quassel
Prediction


The image part with relationship ID rId13 was not found in the file.


The image part with relationship ID rId13 was not found in the file.


A hyperplane in $\mathbb{R}^2$ is a line


A hyperplane in $\mathbb{R}^3$ is a plane

7.3 CNN (Convolutional Neural Network)

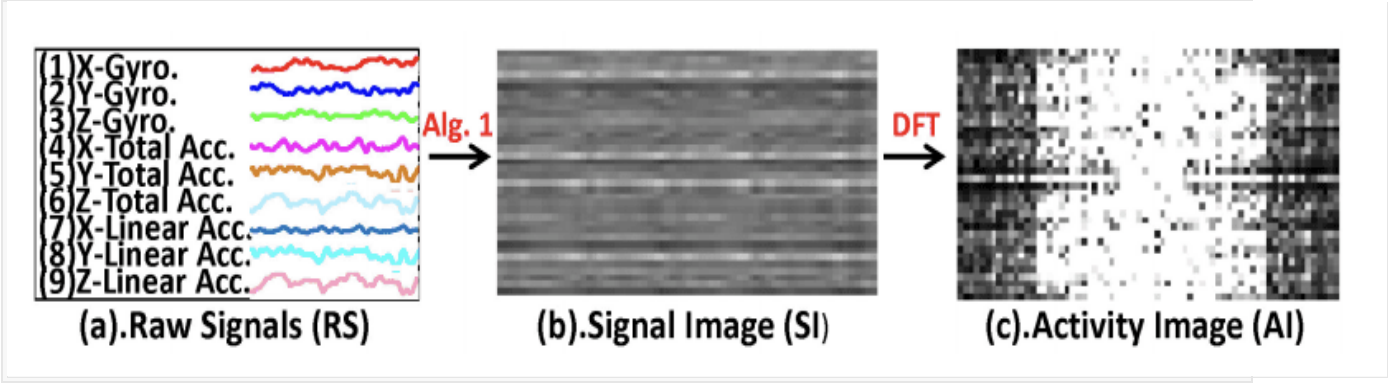
Develop 1D Convolutional Neural Network

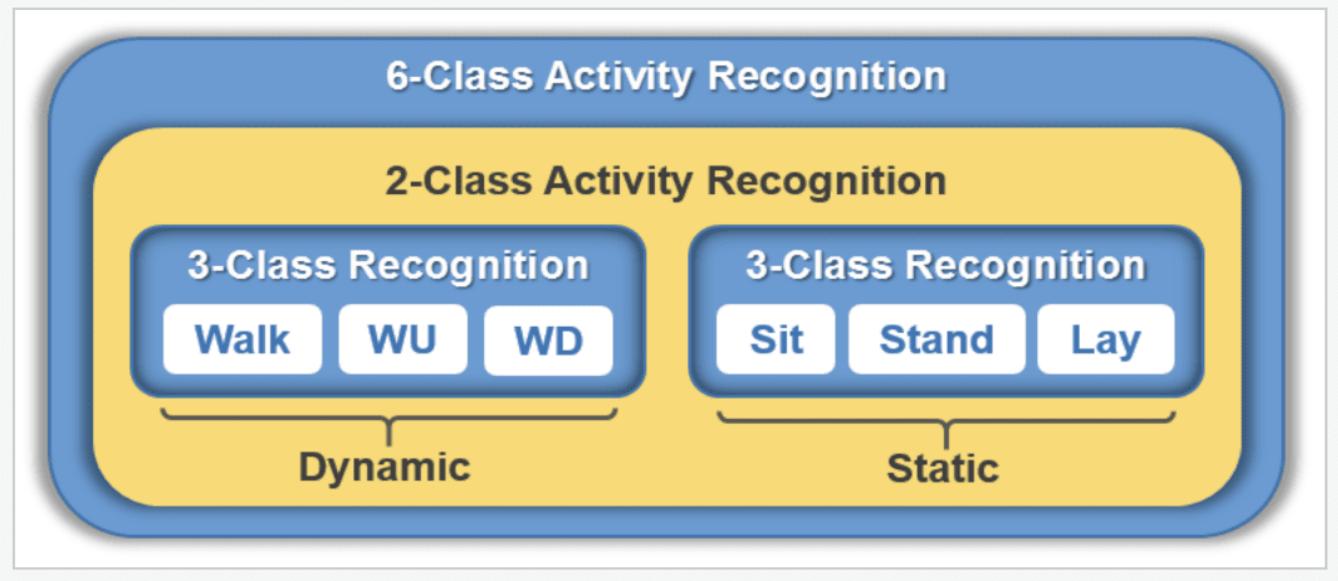
In this section, we will develop a one-dimensional convolutional neural network model (1D CNN) for the human activity recognition dataset.
Convolutional neural network models were developed for image classification problems, where the model learns an internal representation of a two-dimensional input, in a process referred to as feature learning.
This same process can be harnessed on one-dimensional sequences of data, such as in the case of acceleration and gyroscopic data for human activity recognition.

The model learns to extract features from sequences of observations and how to map the internal features to different activity types.
The benefit of using CNNs for sequence classification is that they can learn from the raw time series data directly, and in turn do not require domain expertise to manually engineer input features. The model can learn an internal representation of the time series data and ideally achieve comparable performance to models fit on a version of the dataset with engineered features.

This section is divided into 4 parts; they are:

- Load Data
- Fit and Evaluate Model
- Summarize Results
- Complete Example





Separation of Activities as Dynamic or Static
Taken from "Divide and Conquer-Based 1D CNN Human Activity Recognition Using Test Data Sharpening"

In it, they divide activities into those that involve movement, called "*dynamic*," and thosewhere the subject is stationary, called "*static*," then develop a CNN model to discriminate between these two main classes. Then, within each class, models are developed to discriminate between activities of that type, such as "*walking*" for dynamic and "*sitting*" for static.

CHAPTER 8

WORKFLOW

8.1 Machine Learning Workflow

Machine learning workflows define which phases are implemented during a machine learning project. The typical phases include data collection, data pre-processing, building datasets, model training and refinement, evaluation, and deployment to production.

8.2 Steps

1. Gathering Machine Learning Data
2. Data pre-processing
3. Building Datasets
    - 3.1 Training Set
    - 3.2 Validation Set
    - 3.3 Testing Set
4. Training and Refinement
5. Machine Learning Evaluation

8.2.1 Gathering Machine Learning Data

Gathering data is one of the most important stages of machine learning workflows. During data collection, you are defining the potential usefulness and accuracy of your project with the quality of the data you collect.

To collect data, you need to identify your sources and aggregate data from those sources into a single dataset. This could mean streaming data from Internet of Things sensors, downloading open source data sets, or constructing a data lake from assorted files, logs, or media.

8.2.2 Data pre-processing

Once your data is collected, you need to pre-process it. Pre-processing involves cleaning, verifying, and formatting data into a usable dataset. If you are collecting data from a single source, this may be a relatively straightforward process. However, if you are aggregating several sources you need to make sure that data formats match, that data is equally reliable, and remove any potential duplicates.

8.2.3 Building Dataset

This phase involves breaking processed data into three datasets—training, validating, and testing:

8.2.3.1 Training Set

Training set is used to initially train the algorithm and teach it how to process information. This set defines model classifications through parameters.

8.2.3.2 Validation Set

Validation set is used to estimate the accuracy of the model. This dataset is used to fine-tune model parameters.

8.2.3.3 Test Set

Test set is used to assess the accuracy and performance of the models. This set is meant to expose any issues or mis-trainings in the model.

8.2.4 Training and Refinement

Once you have datasets, you are ready to train your model. This involves feeding your training set to your algorithm so that it can learn appropriate parameters and features used in classification.

Once training is complete, you can then refine the model using your validation dataset. This may involve modifying or discarding variables and includes a process of tweaking model-specific settings (hyper parameters) until an acceptable accuracy level is reached.

8.2.5 Machine Learning Evaluation

Finally, after an acceptable set of hyper parameters is found and your model accuracy is optimized you can test your model. Testing uses your test dataset and is meant to verify that your models are using accurate features. Based on the feedback you receive you may return to training the model to improve accuracy, adjust output settings, or deploy the model as needed.

CHAPTER 9

MODULES

9.1 Sensor Activity Recognition Dataset Download

```
# Download dataset
#! wget https://www.utwente.nl/en/eemcs/ps/dataset-folder/sensors-activity-recognition-dataset-shoaib.rar -P ../data/
dataset=wget.download("https://www.utwente.nl/en/eemcs/ps/dataset-folder/sensors-activity-recognition-dataset-shoaib.rar")
# Extract dataset using unrar
!pip install unrar
#!unrar e ../data/sensors-activity-recognition-dataset-shoaib.rar ../data/
```

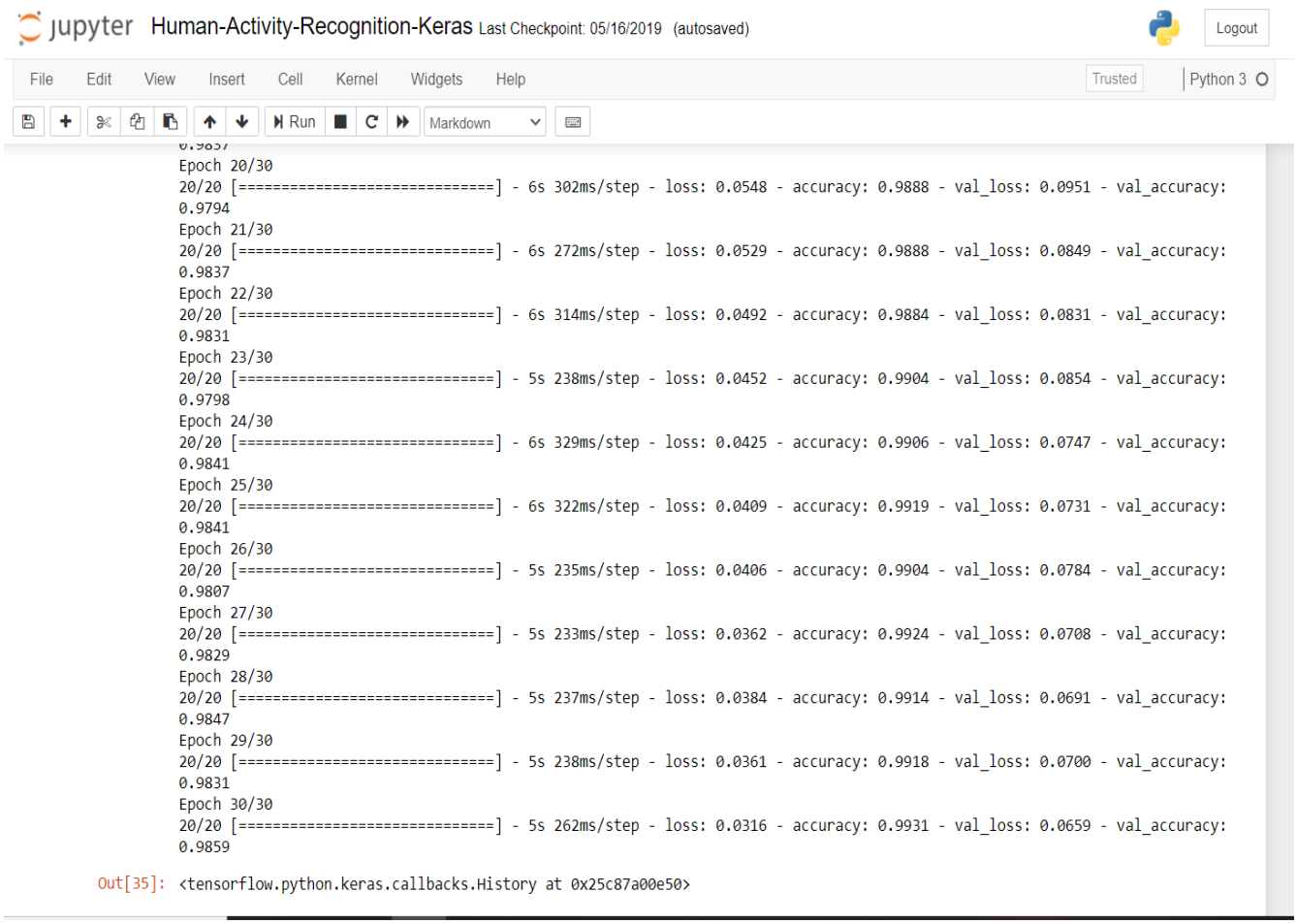
```
100% [..........................................................................] 100080442 / 100080442Requirement already satis
fied: unrar in c:\users\aswin\anaconda3\lib\site-packages (0.4)
```

9.2 Sensor Activity Recognition Dataset

The Human Activity Recognition database was built from the recordings of 30 study participants performing activities of daily living (ADL) while carrying a waist-mounted smartphone with embedded inertial sensors.

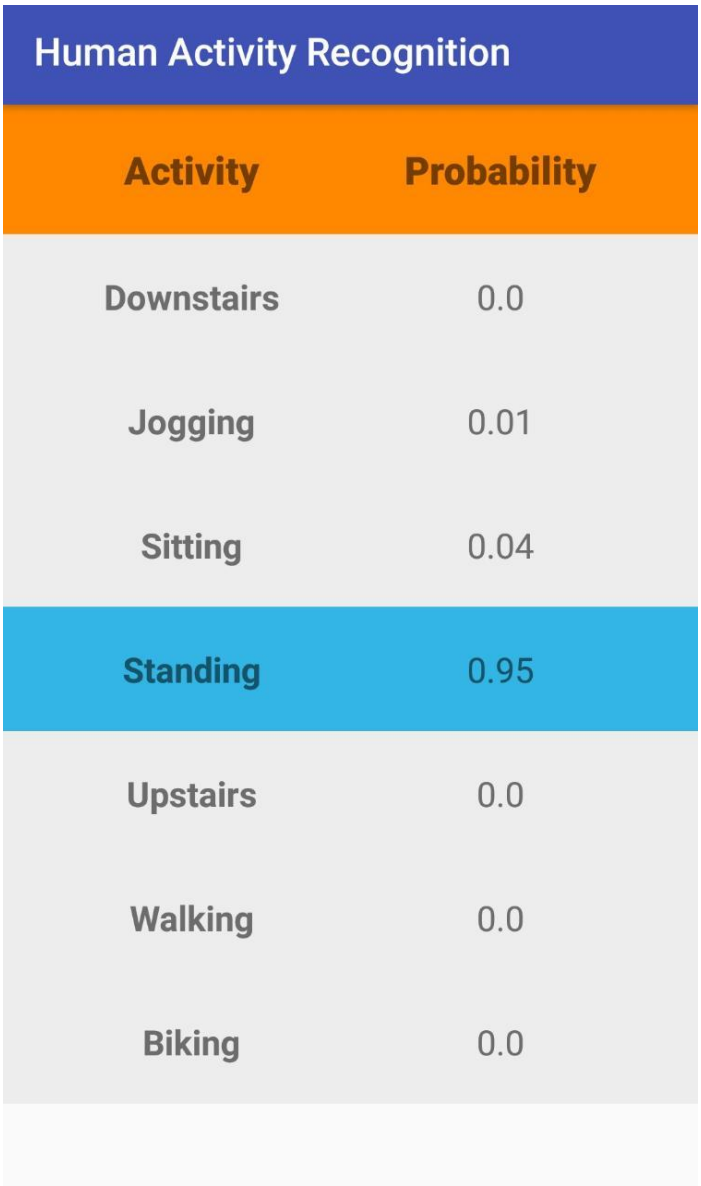
9.3 Accuracy

Accuracy can be computed by comparing actual test set values and predicted values. Well, you got a classification rate of 99.31%, considered as very good accuracy. For further evaluation, you can also check precision and recall of model.


jupyter Human-Activity-Recognition-Keras Last Checkpoint: 05/16/2019 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Markdown
0.9837
Epoch 20/30
20/20 [==============================] - 6s 302ms/step - loss: 0.0548 - accuracy: 0.9888 - val_loss: 0.0951 - val_accuracy:
0.9794
Epoch 21/30
20/20 [==============================] - 6s 272ms/step - loss: 0.0529 - accuracy: 0.9888 - val_loss: 0.0849 - val_accuracy:
0.9837
Epoch 22/30
20/20 [==============================] - 6s 314ms/step - loss: 0.0492 - accuracy: 0.9884 - val_loss: 0.0831 - val_accuracy:
0.9831
Epoch 23/30
20/20 [==============================] - 5s 238ms/step - loss: 0.0452 - accuracy: 0.9904 - val_loss: 0.0854 - val_accuracy:
0.9798
Epoch 24/30
20/20 [==============================] - 6s 329ms/step - loss: 0.0425 - accuracy: 0.9906 - val_loss: 0.0747 - val_accuracy:
0.9841
Epoch 25/30
20/20 [==============================] - 6s 322ms/step - loss: 0.0409 - accuracy: 0.9919 - val_loss: 0.0731 - val_accuracy:
0.9841
Epoch 26/30
20/20 [==============================] - 5s 235ms/step - loss: 0.0406 - accuracy: 0.9904 - val_loss: 0.0784 - val_accuracy:
0.9807
Epoch 27/30
20/20 [==============================] - 5s 233ms/step - loss: 0.0362 - accuracy: 0.9924 - val_loss: 0.0708 - val_accuracy:
0.9829
Epoch 28/30
20/20 [==============================] - 5s 237ms/step - loss: 0.0384 - accuracy: 0.9914 - val_loss: 0.0691 - val_accuracy:
0.9847
Epoch 29/30
20/20 [==============================] - 5s 238ms/step - loss: 0.0361 - accuracy: 0.9918 - val_loss: 0.0700 - val_accuracy:
0.9831
Epoch 30/30
20/20 [==============================] - 5s 262ms/step - loss: 0.0316 - accuracy: 0.9931 - val_loss: 0.0659 - val_accuracy:
0.9859
Out[35]: <tensorflow.python.keras.callbacks.History at 0x25c87a00e50>

9.4 User Interface Model

In this model,it shows the percent rate of the activity such as jogging, sitting, upstairs, walking , downstairs, biking and standing.



CHAPTER 10

IMPLEMENTATION AND RESULTS

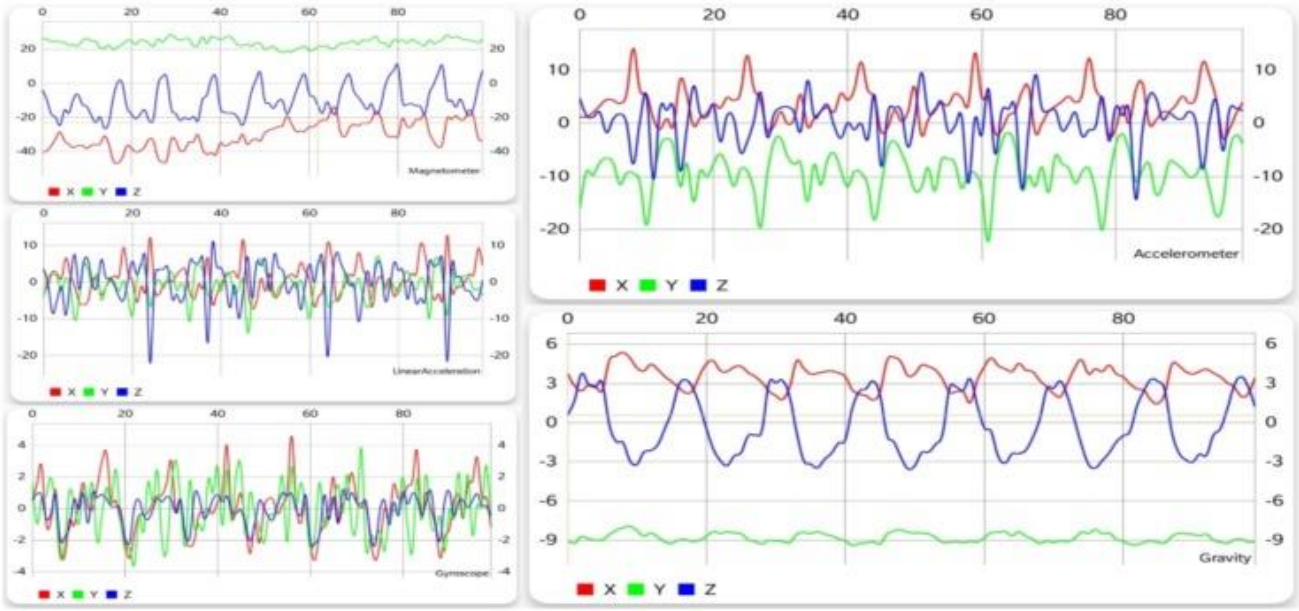
10.1. Data Interpretation

The two datasets used did not contain data from exactly the same sensors, but because the graphical shape of the data was the same in both of them, this section presents the testing data, which were the same regardless of the training data. The following charts were also offered in the application when users requested collecting new data. In the following sections, besides the interpretation of each chart independently, a comparison of the similarities and differences is offered where necessary.

For each action, the figures present (in this order) data from the accelerometer, gravity sensor, gyroscope, linear acceleration sensor, and magnetometer. Although both datasets contained data recorded while holding the device in different positions, the charts presented here were generated while holding the device in the right trouser pocket, oriented downward and facing the inside of the trouser.

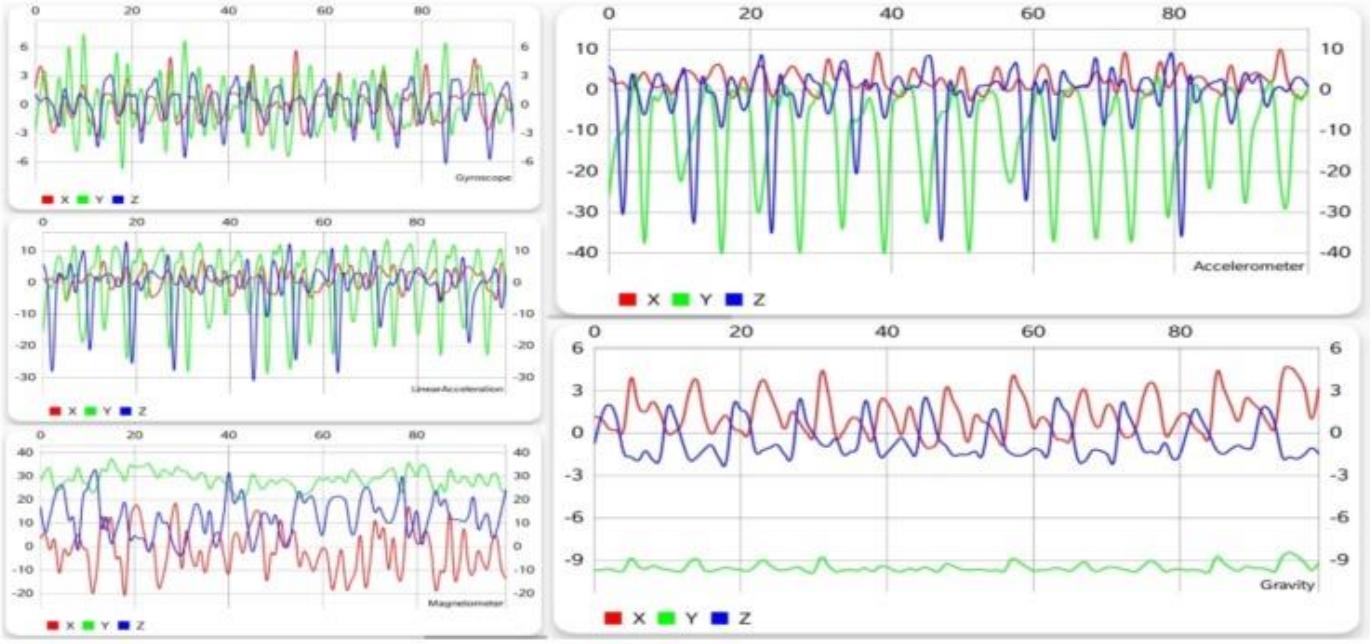
10.1.1. Walking

Walking generates a periodic pattern for each of the sensors, thus offering a large amount of information to the learning algorithm, so that the classifier can distinguish it from other activities easier. Of course, this is just one possible pattern, and depending on characteristics like age, gender, height, or weight, it can vary slightly. However, this affects only less significant metrics such as the interval of the repetitions or the range of values, but the shape and pattern of the movement is almost the same.



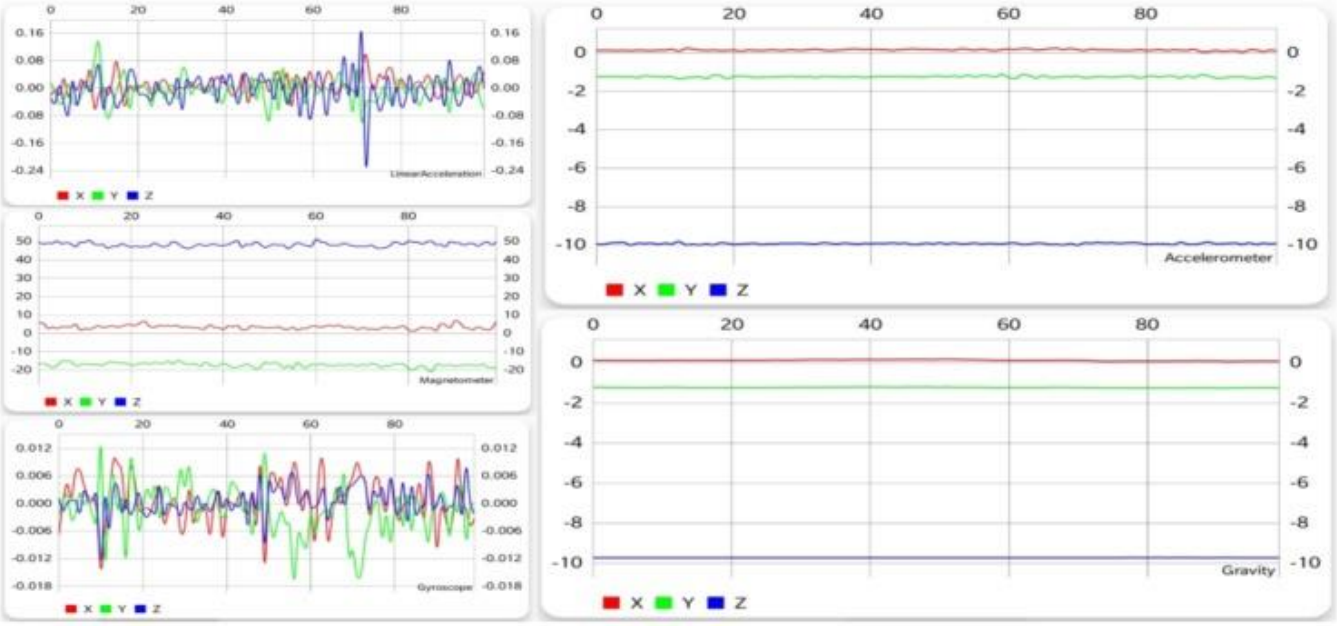
10.1.2. Running

Running implies a motion similar to walking, but executed faster. Periodic pattern can also be noticed for this activity, with a shorter time difference between periods for running than for walking. Although the patterns for the two activities have some common attributes, a clear difference can be seen from each sensor, especially the accelerometer and gravity sensor.



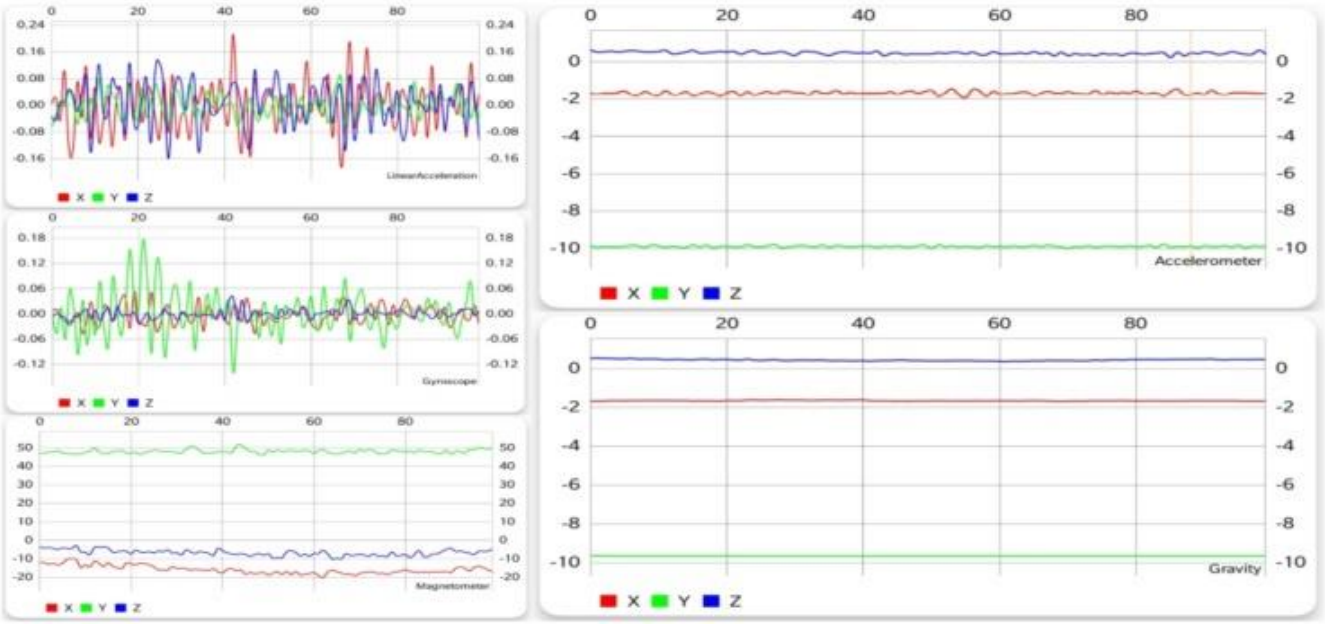
10.1.3. Sitting

Unlike the previous two activities, sitting is a motionless activity, if it can be called an activity at all, with the exception of the gyroscope and the linear acceleration sensor, all other sensors display little change in the values of each of the three axes, the accelerometer and gravitational data being almost unchanged. Even the first two mentioned sensors show much less fluctuation than walking and running, thus making sitting very easy to differentiate from them.



10.1.4. Standing

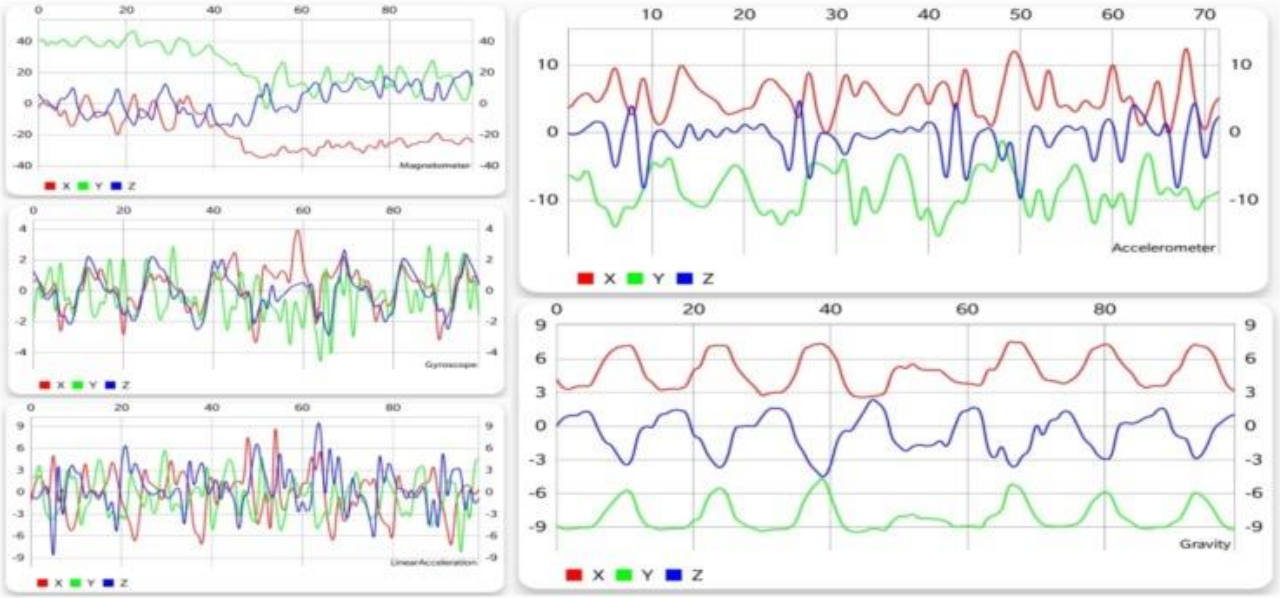
Standing is also an activity than involves staying still, Some differences can be seen in terms of gyroscope and linear acceleration, but the main indicator when trying to distinguish the two actions is the position of the axes on the graphs. When switching from one action to the other, an interchange between the Y and Z axes can be observed. Even though these two activities are somewhat similar, they are very easy to recognize.



10.1.5. Walking Upstairs

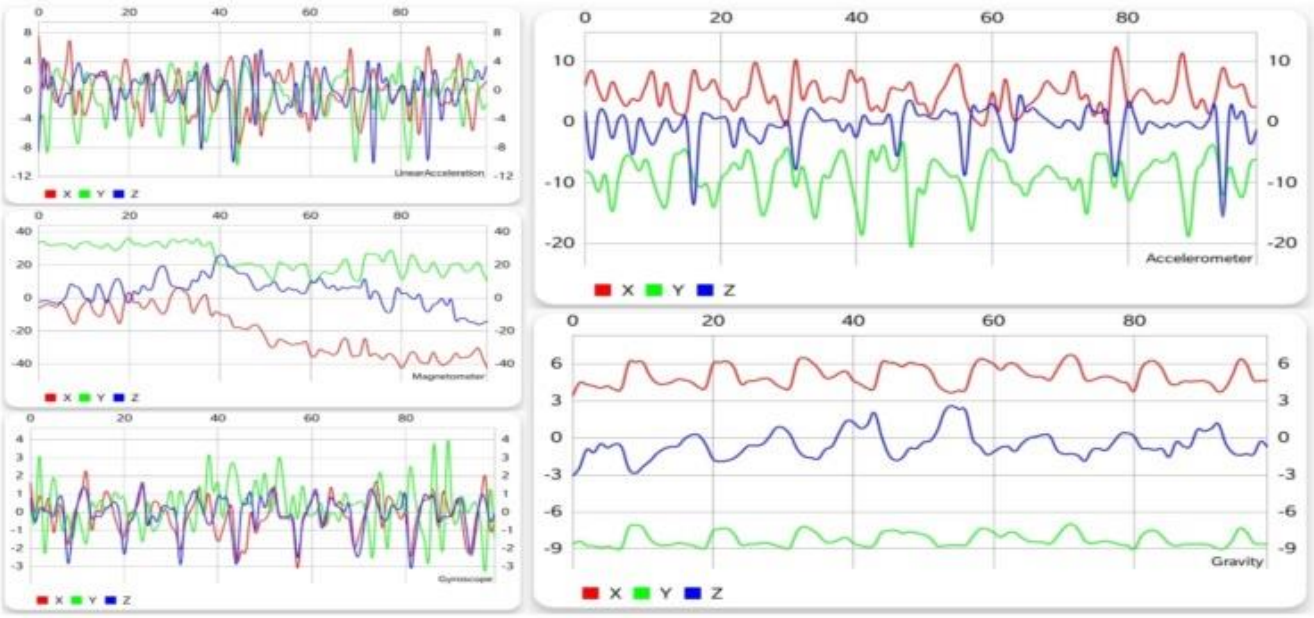
Walking upstairs is an activity that, unlike the previous ones, can be done in many ways. It is not that everyone walks or runs the same way, but with a few exceptions, the patterns they generate are more or less alike. As far as walking upstairs goes, people do it step by step, two steps at a time, some even three, slow, fast, or jumping, so recognizing it is not very straightforward.

Similarities can be observed between walking and climbing stairs, especially when comparing the results from the accelerometer and the gyroscope. This is partly due to the fact that there are not many stairs that take as long as 10 s to climb (since generally, one would need to spend a few seconds on the stairs, then walk on a flat surface, and so on). This resemblance generates confusion and some false positives, and negatives showed up when trying to distinguish and recognize these two activities.



10.1.6. Walking Downstairs

Just like going upstairs, walking downstairs is more complicated to detect when trying to differentiate it from other activities. Similarly to going upstairs, the patterns of going down the stairs tend to vary. Just like going upstairs, this activity is easily confused with walking. However, the biggest difficulty in recognizing both walking upstairs and downstairs is the fact that they are almost the same, which can easily be observed from , which means that the activity recognition struggled to find out which one was actually performed.


0 20 40 60 80
LinearAcceleration
X Y Z
Magnetometer
Gyroscope
Accelerometer
Gravity

CHAPTER 11

CONCLUSION

In this paper, we showed how human physical activity recognition can be achieved by using sensors available on a smartphone. During this work, a new data set was collected for the six activities that made up the subject of this paper; relevant features have been extracted from the gathered data; and results were obtained using a neural network. Evaluation of the results showed that most of the activities were recognized correctly, four of them averaging an accuracy of 93%. Even with the lower scores obtained for the other two activities, the accuracy did not drop under 86%. These numbers suggest that using sensors to recognize user activity is becoming more and more reliable. The proposed solution was also tested against an external standard data set, the results showing a slight decrease to 77%. However, the results may be affected because the external data set contained data from some different sensors. The paper was materialized through an Android application, easily usable by any type of user, including older adults.

CHAPTER 12

FUTURE ENHANCEMENT

After achieving satisfactory results for activity recognition, we wish to take this one step further and work on a way to make the proposed solution publicly available in a form that makes it easier to be used for further research. In order to do so, a couple of actions have been planned. Because machine learning algorithms, especially deep learning ones, use many system resources, running them on a mobile device (even with the capabilities of the latest smartphones) is not a viable option. Choosing large values for the depth of the network and the size of the training dataset can easily freeze a device. Bearing this in mind, it is clear that doing all the computational part right on the device is not the best solution. Therefore, in the near future, all this heavy processing is to be shifted to a remote server. This way, the performance of the application should be improved visibly, which would make real-time detection become more of a possibility than a probability. Furthermore, we would also like to explore various alternatives to the currently-implemented perceptron classifier, such as genetic algorithms.

Another goal is to improve activity recognition by adding additional activities like riding a bike, driving, and falling. This would not only help the project cover much more of the user activity, but also offer opportunities of developing new applications that use activity recognition. One example of such an application (that we have begun working on) involves combining activity recognition with user localization, with the purpose of tracking elderly people and receiving notifications whenever their behavior is not the one expected (i.e., they go to an area of the city that they do not know; they fall; they walk in strange patterns, etc.).

REFERENCES

[1] Academic Performance. International Journal of Computer Science and Information Security,7(1).292-295

[2] Science and Information Security,7(1).292-295F. Attal et al., "Physical Human Activity Recognition using Wearable Sensors," Sensors 15(12), pp.31314-31338, 2015.

[3] C. Ronao and S. Cho, "Human activity recognition with smartphone sensors using deep learning neural networks," Expert Systems With Applications 59(2016), pp. 235-244. Elsevier(2016).

[4] S. Kozina, H. Gjoreski, M. Gams and M. Lustrek, "Efficient activity recognition and fall detection using accelerometers" in Evaluating AAL Systems Through Competitive Benchmarking, Springer, 2013.

[5] A. Bayat, M. Pomplun and D. Tran, "A Study on Human Activity Recognition Using Accelerometer Data from Smartphones," Procedia Computer Science, vol. 34, pp. 450 – 457, 2014.

[6] D. Anguita, A. Ghio, L. Oneto, X. Parra and J. L. Reyes-Ortiz, "A Public Domain Dataset for Human Activity Recognition Using Smartphones." ESANN 2013 proceedings on European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, i6doc.com publ., Bruges, 2013.

[7] HMC Web Site: http://fourier.eng.hmc.edu/e84/lectures/ActiveFilters/node6.html Last accessed:2018.09.18

[8] M. Mohammed, M. B. Khan and E. B. M. Bashier, Machine Learning Algorithms and Applications. CRC Press, Florida, 2017.

[9] F. Giseke, "From Supervised to Unsupervised Support Vector Machines and Applications in Astronomy," The Carl von Ossietzky University of Oldenburg, Germany,2011 .

[10] Y. Freund, R. E. Schapire, “A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting,” in Journal Of Computer and System Sciences 97(55), pp. 119-139, Elsevier, 1997.

[11] C. A. Shipp and L. I. Kuncheva, “An Investigation into how ADABOOST Affects Classifier Diversity”, available at http://pages.bangor.ac.uk/~mas00a/papers/cslkIPMU02.pdf

[12] T. G. Dietterich, “Ensemble Methods in Machine Learning” Lecture Notes In Computer Science – Multiple Classifier Systems, vol.1857, pp. 1-15, 2000.

[13] L. Breiman, “Bagging Predictors.” in Machine Learning 24(2), pp. 123-140. Springer,1996.

[14] Mäntyjärvi J, Alahuhta P, Saarinen A (2004) Wearable sensing and disease monitoring in home environment. In: Workshop on ambient

intelligence technologies for wellBeing at home. Held in conjunction with 2nd European symp. on ambient intelligence. EUSAI, Eindhoven, November 2004.