ACTIVITY RECOGNITION BASED RECOMMENDATION SYSTEM

Project report submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology

in

Computer Science and Engineering

by

Shivam Sharma (161242)

under the supervision of

Dr. Rajinder Sandhu

to

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology, Waknaghat, Solan-173234, Himachal Pradesh

Candidate's Declaration

I hereby declare that the work shown in this report entitled "Activity Recognition based Recommendation System" in partial fulfillment of the requirements for the degree of Bachelor of Technology in Computer Science and Engineering submitted in the Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Waknaghat is a valid and authentic record of our own work carried out over a duration from August 2019 to December 2019 under the supervision Dr. Rajinder Sandhu, Assistant Professor.

The matter written in the report has not been submitted for the award of any other degree or diploma.

Shivam Shama, 161242

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Dr. Rajinder Sandhu
Assistant Professor (Senior Grade)
Department of Computer Science & Engineering and Information Technology

Dated:

Acknowledgement

We would like to express our greatest regard to the people who have helped & supported us throughout our project. We are grateful to our mentor **Dr. Rajinder Sandhu** for his continuous support for the project, for initial advice & teachings in the early stages of conceptual inception & through ongoing advice & encouragement to this day.

A special thank of us to our group members who helped each other in completing the project & exchanged their interesting ideas, thoughts & made this project easy and accurate.

Table of Contents

List of Abbreviations

Abbreviation	Explanation
CSV	Comma-Separated Values
RBM	Restricted Boltzmann Machine
SAE	Stacked Auto Encoder
HAR	Human Activity Recognition
NA	Not Applicable
EDA	Exploratory Data Analysis
SVM	Support Vector Machine
DL	Deep Learning
kNN	k Nearest Neighbour
WRT	With Respect To
ML	Machine Learning
ANN	Artificial Neural Networks
AI	Artificial Intelligence

List of Figures

Abstract

In this project, our system will recognize activities in the home setting and then this system further learns the daily routine of the person and raising an alarm that recommends the person if an activity is missed by that person. This system uses a set of small and simple binary sensors, ML algorithms, DL algorithms and electronic experience sampling is introduced. The sensors are designed to be “tape on and for-get” devices that can be quickly and installed anywhere in home environments. The proposed sensing system presents an alternative to sensors that are sometimes perceived as intrusive, such as cameras and microphones. Unlike previous systems, this system was deployed in multiple residential environments with non-researcher occupants. Preliminary results show that it is possible to recognize activities of interest to medical professionals such as toileting, bathing, and grooming with detection accuracies ranging from 25% to 89% depending on the evaluation criteria used. Although these preliminary results were based on small datasets collected over a two-week period of time, techniques have been developed that could be applied in future studies and at special facilities to study human behavior such as the MIT Placelab. The system can be easily modified in existing home environments with no major modifications and can be used to enable IT and health researchers to study the behavior in the home. Activity recognition is increasingly applied not only in home-based proactive and preventive healthcare applications, but also in learning environments, security systems, and a variety of human-computer interfaces.

Chapter 1 Introduction

The study and research in (HAR)Human Activity Recognition is in big demand due to its emerging and important applications in Health care domain, Computer Vision domain, Household safety domain and Robot Learning domain. A very big amount of resources and money can be saved if sensors collects and monitor data of patients. System can itself automatically send those reports to medical personel or doctor in case of any offroad or unnatural behavior of patients.

So to achive this we have used low cost binary sensors to collect the data and identify human activities. Today, we can see a strong interest in the market of smart homes in future plans of large tech giants like Google, Samsung, Amazon, and many more. It is being expected by computer scientists that the market for smart homes will exceed $ 45 billion by 2021. But however, despite the exponential growth of market, smart homes still have a number of drawbacks and limitations, including they cannot analize the correct behavior of people. At this time there is not a single solution and approaches that can be used to completely solve this thing.

Main aim of HAR is to find the activities done by someone given some set of observations of him/her and their native environment. This task can be done by using the information which can be retrieved from different sources like resident's environment or body-worn sensors. Our primary aim is to classify some human activities in form of a dataset which will have 11no of labels.These activities which will become our labels are, Meal_Preparation (1606), Relax (2910), Eating (257), Work (171), Sleeping (401), Wash_Dishes (65), Bed_to_Toilet (157), Enter_Home (431), Leave_Home (431), Housekeeping (33), Resperate (6). The no. in ()parentheses tells the no. of times an activity appeared in our dataset.

Then after successfully recognizing the activities our system tends to learn the daily routine of the person with the help of dataset that contains the records of past 30 days.

1.1 Problem Statement

Recognizing the activities from home setting binary sensors dataset was a difficult work because of disordered dataset and ambiguous activities . Different ML approaches have been used in the past to recognize activities of human. So,we used a system which initially preprocesses our dataset and uses ML algos in optimized manner to produce perfect outcomes and learn the daily routine of user using some DL algorithms and when once learned the daily routine of the user, it can be used to raise alarms and recommend the missed activities from daily routine.

1.2 Objectives

i. Go clean the ambiguous dataset.

ii. Thoroughly study that dataset .

iii. To identifysuitable ML algos to analize that dataset.

iv. Applying ML algos to the cleaned datasets andgenerate the confusion matrix(cm).
v. Get error rates from cm.
vi. Calculates time taken in training the ML models.
vii. Compares time taken and error by different MLalgorithms.
viii. To monitor the recognised activity of the user and make an activity log by learning users day to day activity .
ix. Activity log will be learned by applying various deep learning models like RBMs and SAEs.

x. g. Raise an alarm/Recommend an activity to user if user goes off the schedule of his daily activities.

1.3 Methodology

Right now so many research project are focussing on collecting data in home setting. In this report dataset from CASAS Smart Home Project (Center for Advanced Studies in Adaptive Systems at Washington State University) is used. We are using the datasets naming ***Aruba***. Data is collected in the house of volunteered old woman for 1 month. Then thatData was impoted, cleared and normalised. This dataset is procesed through different supervised ML algos like RandomForest, SVM, ANN and KNN to classify data into 11 catagories. Accuracy of the model is calculated by making confusion matrix and by random simulation.

1.4 Organisation

Chapter2 describe some works which have association with our works. Method and SystemDesign is described in the Chapter3. Working and the results of various ML algos have been compare in Chapter4. Testing in performed in Chapter 5. Chapter 6 and Chapter7 has conclusions from the analysed work that we did.

Chapter 2 Literature Survey

2.1 Human Activity Recognition

It is find that here are minimum 4 way for a machgine to gather thedata about people's activity by sensors system:

(1) askfrom the man, as in experiience samplings

(2) remote observation the scenes using audios, visuals, em fields or other sensorsand interpretting the signals reading

(3) attaching sensor to the body-parts and interpretting the signals reading

(4) attaching sensor to things and devicess in the environments and interpretting the sensors reading.

Ask question directly is a good and effective way but 1 that is used provident. Frequents interruptions may annoy peoples. Althoughs progresses are being done on algos which monitors scenes and interpret the sensor signal, the accquired sensosr dataa is mostly highlly corrupted. The various sound founded in a house are however, considerabl, and couldbe particularlly hard to diferentiate sound generatted by different people.

Extra compllex sensor like camera in (CV)computer visions also being used for recognising activitys. Computer visions sensors for monitoring and action identification oftens work in the labs but fail in realality homes settingdue to litter, variable lightings, and highly varried activitie which takes places in real scenarios. Small of thistype of works have been broadly tested in fields because of difficulty in handling change in the scenes, like lights, different persons, cluter, etc. In the end due to sensor like microphone and camera are so normal and mostly common used as recording device, hey could also seen as intefering by many personss.

Attach sensor to the bodyparts isa encouraging and relativelly less costly method of acquiring data of some type of man movements.Also, they are effortlessly used in real environment.

The following methods are being used to classify different activities. A worable sensory jackets uses stretched sensor for detecting activities like walk and runn; an audiio processing worable computers for recognising socially interacttions in the workplaces, videeo procesing worable computers for finding patternss of activitys of everyday life, biometriics worable sensor to visualisingexercises in day-to-day activity, and a contextawre PDA and GPS to find activitys like “grocery shoping”, “go for working”,and “do laundry” and many more .

Various activitys, include physicalyl complexed motions and somemore interactions with an environmentt. Signal from very much varriying arms and legs movementin time of coooking and cleansing maynot allows the activity to be differentiated from otherones. Movementsare no dependent on things in our environments. Also, if sensor is embedd in oure environments, activity like co\king may be shown by different set of sensor activations (e.g the stove, cabinetry) with only minor day to day variations. Simple sensors can often provide powerful idea about activity. For example, a switch sensor in the bed can strongly suggest sleeping, and pressure mat sensors can be used for tracking the movement and position of people.

Last works wher sensor were kept on things in the environmentts include an adapitative controlling of house environment loike(ACHE) systemsmade for controlling the basical resident system like(airing, heat, light, ventilator and waterheating) for monitoring the liveestyles and drerams of the residents. Less complexed sensor in kitchens (tempt on stove, mat’s sensor, or cabinetts door;s sensor) were used for finding activities like food preparing in 1 special wire deployed house kitchenss.

Taking care and using the intellizent versaile house (MavHome) are systems which tends to increase resident's comfortness and decrease cost by prediction movability patttern and digital use .The main diference in the privious works and t the sis works are those system are not useed in these residents environments with real inhabitants. They were usually used in labs or houses of the research workers and their assistants. again, each of those systems would use very carefull installation and maintenance by research staff and students.

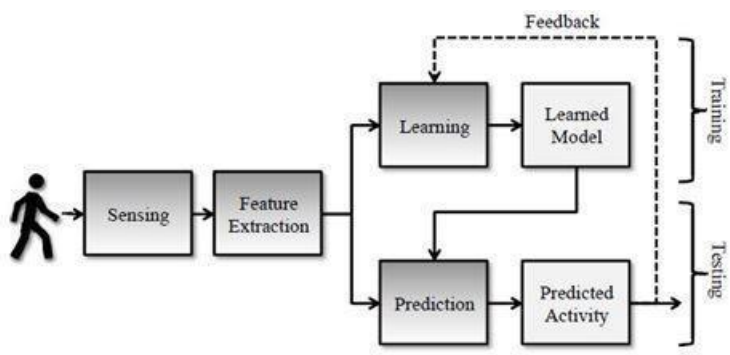

Figure 2.1: HAR development Process

2.2 Sensors used

2.2.1 Binary Sensors

Binary sensors collects data or info of devices;s state info rthat gives digitalized values like 0or1. They could be switc,h contact pin, etc. Theyhave only 2 state: **0/off/low/closed/false** and **1/on/high/open/true. As we k**now that there ais only 2 state allowed in House Assist to present the sensor in a more efficient ways in the leading end
acc. to the works or the functions.

We uses 3 types of sensor inthis research: that areL:motion sensors(their r IDs starts with "M"), door closing sensors (their r IDs starts with "D"), and tempt sensor (their r IDs startswith "T").

Motion Sensor

Motion on shows that sensor is detecting a motiuonal activity and Motion Off means there is no motion being detected by that sensor.

Figure 2.2: Motion sensor denotations

Door closure sensors

ON means door open and OFF means door closed.

Figure 2.3: Door closure sensor denotations

Temperature sensors

On means HOT and OFF means normal.

Figure 2.4: Temperature sensor denotations

In this project we collected, sensory activities were gathered with the time. An eg of the data you canbe seen in the following figure:

2010-11-04	05:40:51.303739	M004	ON	Bed_to_Toilet	begin
2010-11-04	05:40:52.342105	M005	OFF		
2010-11-04	05:40:57.176409	M007	OFF		
2010-11-04	05:40:57.941486	M004	OFF		
2010-11-04	05:43:24.021475	M004	ON		
2010-11-04	05:43:26.273181	M004	OFF		
2010-11-04	05:43:26.345503	M007	ON		
2010-11-04	05:43:26.793102	M004	ON		
2010-11-04	05:43:27.195347	M007	OFF		
2010-11-04	05:43:27.787437	M007	ON		
2010-11-04	05:43:29.711796	M005	ON		
2010-11-04	05:43:30.279021	M004	OFF	Bed_to_Toilet	end
2010-11-04	05:43:45.7324	M003	ON	Sleeping	begin
2010-11-04	05:43:52.044085	M003	OFF		
2010-11-04	05:43:53.185335	M002	ON		
2010-11-04	05:43:53.253809	M003	ON		
2010-11-04	05:43:59.493281	M002	OFF		
2010-11-04	05:44:04.048766	M003	OFF		
2010-11-04	05:44:06.14204	M003	ON		
2010-11-04	05:44:11.229146	M003	OFF		

Figure 2.5 : Dataset example

Chapter 3 System Development

3.1 Introduction

Here, we will elaborate every-step procedure deployedto designing the model;. The AI application we are solving comes in the class of classification problems. We are using Spyder(Anaconda3) App Studio version 3.3.6 on windows 10 Operating ZSystemand GoogleColaboratory to do our whoole experimentss.

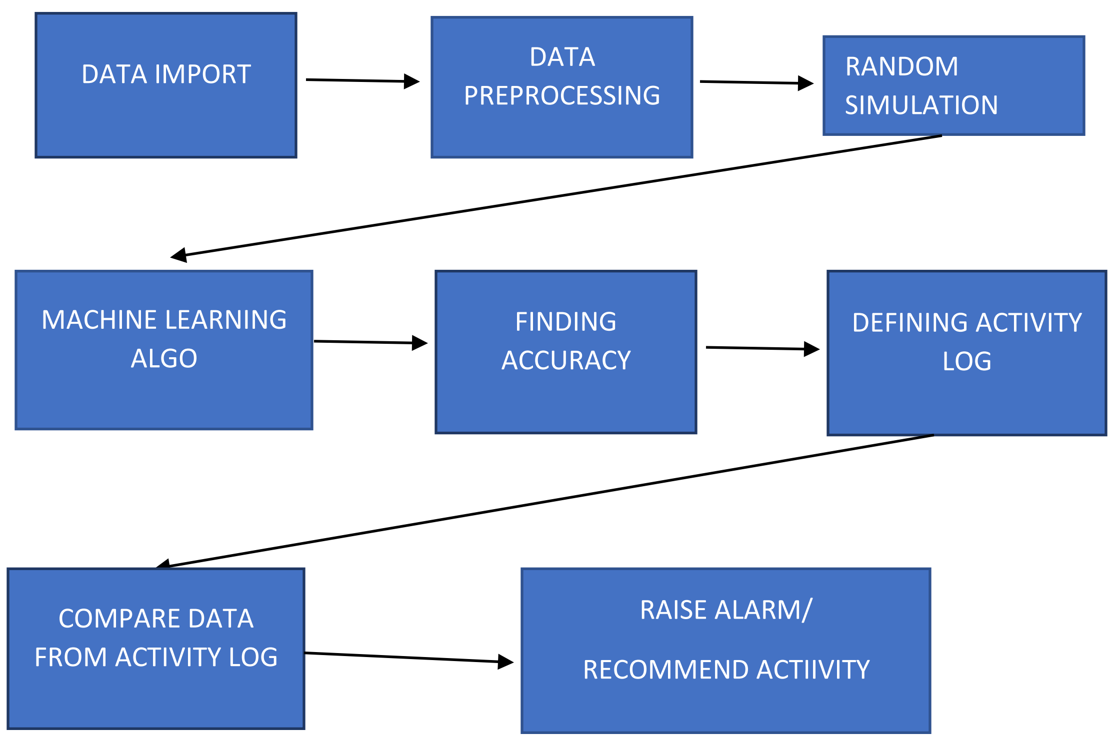

Figure 3.1: Flow diagram of Activity Recognition base recommendation System

3.2 Importing and Preprocessing Data

Pre-processing of data is very important part of datamining. The main aim of pe-pocessing is to filtring the datas and to take away the corrupted valued toget or selection offeatures. In the approach that we foloow rthgat is windows method the sensory signal are breaked into little time segmentations. They are used in extraction of features. After that, on every unique windowsegment, segmentalization and classification algos are used. We use three types of window technique that are (a) slidingwindow (b) definedevent (c) defined activity-. The mostly affective techniques is slidingwindow for real-time apps

The aim of pe-pocessing is to filtering datas, to exchange the corrupt nos and find selected featuresss.

3.3 RandomSimulation

RandomSimulisationn are done to find accuracy_yscore of predicting systems and preventing of overfit or underfit of dataset. Theis technssque includes random division of datas into train and test sets in the proportion of 7by3. This complete process is done 60 times for better acuracy of system according to statistics (CLT) CentralLimit theorem. Test data gives approximateof realtime dataset and gives a method for testing stableness of the system in reality examples.

3.4 Machine learning models

Various ML classification algos such as Randomized Forestts, SVMs, kNNs were tried and tested on our dataset and depending upon their performance one of the algorithm was selected for our model. More information on ML models is given in next chapters.

Chapter 4 Algorithms

4.1 ML algorithms

4.1.1 Random_Forest

The random_forest is a ensembled method which could be considered like nearest_neighbor prediction model. Ensemble learning appreoaches are like divide and conquerring ways for improving the performancess and accuracy. The idea of ensembledd method is that manya of “weak learners” couild comeand may work in groups to create a “strong learner.” Individually, every learner is a “weak learner,” but with alltogether they make“strong learner.”

The random_forest leads witha standard ML method which is named as“decision _tree.” In ensembeles language it belongs to learner that is weak. Here, an i/p is provided at the upper end that goesdown the tree, the datass get collected into increasingly little settress.

Whenever a new i/p is given to the systems, it is runs downs to all of the treess. The o/p may r be a averaged—or weight averaged—to everyend or leafnodes those were rech, or a voting majority in scenario of categoryfull datasets.

Random forest predictors lead to a dissimilarity between observations. A random forest can also be attractive because it handles missing values well. It also reacts well to outside observations.

The concept of “extremely randomized trees” or “Extra Trees” adds yet another layer of randomization. Each Extra Tree is trained using bagging and the random subspace method, but randomizes the top-down splitting in the tree, too. In this case, the system chooses a randomized valued splitss, rather than computing and assigning a local level best feature and split combos. The random value is selected from the empirical range of the feature.

Random_Forest were ensembledd learn techniques for classifiying, regression and many different works. It constructs somany decision_trees at time of training and o/ps the categorywhich is the avg result of all thedecision trees. Random_forest exempts over fitting the datasets. In our report and project we used Random Forest provided in Scikit-Learn library. There are very large no. of entities in this datasetwhich is why we used the first three weeks of data to train data and last some weeks to testing data. This dataset is very im-balanced so we did‘balanced’ valuess in ‘classweight’ parameter. The “balanced” data used the value of ys to auto adjusting weighsss inversed to frequency of each label of the i/p dataset.

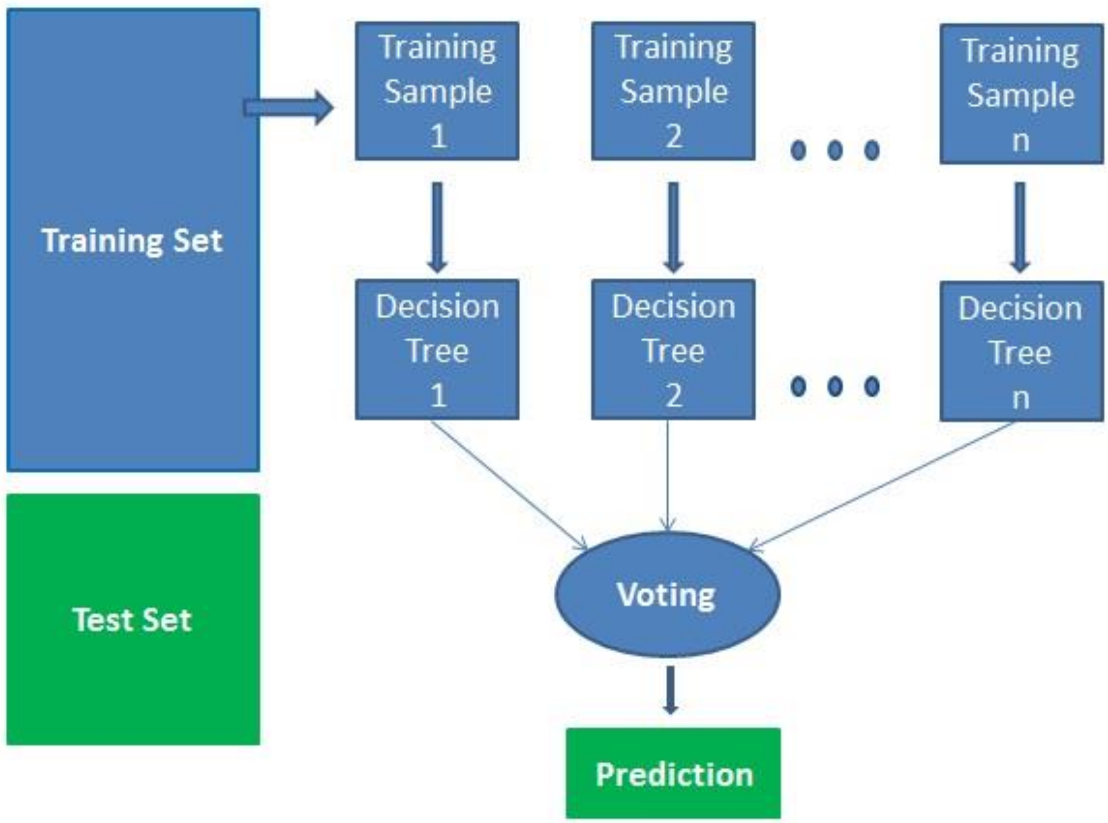

Figure 4.1: Flow diagram depicting the working of Random forest

4.2 Deep learning algorithms

4.2.1 Restricted -Boltzmann Machine

BMs (Boltzman_Machine) is a special type of Marakov Random Fields (MRF) that are log-linear which means the funct of energy is linear in the parameterssgay thatb are free. For making them bold and effective in depicting complicatedions distributes like., going from limit no of parametrized settingss to a nonparameterized 1 we thionks that someof the varies are not even observable and are known as hidden.If we have more of these variablllees (also known as hidd unitwe could maximize the model capacitacy of Boltzzzman Machiness (BM). Restricted_Boltzzmann Machine RBMs to these w/o visible-innvisible and hidden-unnhidden connecting lines. A graph depicting of RBM is:.

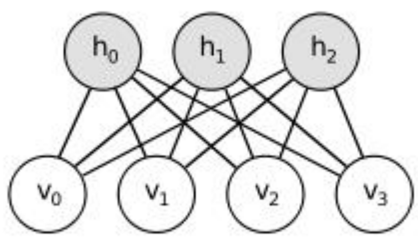

Figure 4.2:Structure of RBM

Funct which gives the energy $E(v,h)$ of an RBM is given as:

$$E(v,h) = -b'v - c'h - h'Wv$$

;here W shows the weighs lines which connectrs hidden to visible nodes and $b$and c are the Off serts of their model layers.

So the new formula of free energy is:

$$\mathcal{F}(v) = -b'v - \sum_i \log \sum_{h_i} e^{h_i(c_i + W_i v)}.$$

4.2.2 Stacked AutoEncoders

Auto-encoder are type of the learning that is unsupervised .Its architecture has 3 layersss: i/p layers, hid layers, and o/p layers as depectiсts in diagram . The training of An auto-encoder have two sections which are 1)encoding and 2)decoding. Encoding is done by encoders to map the i/p datavalues to hiden nodes, and decoding is reffers to again making i/p datavalues from the hiden presentation. The encoding is shown below:

$$h_n = f\left(W_1 x_n + b_1\right)$$

here above is an encoder's function, W_1 is the weigh matrices of the encodeing unit, and b_1 is a vector which contains biass.

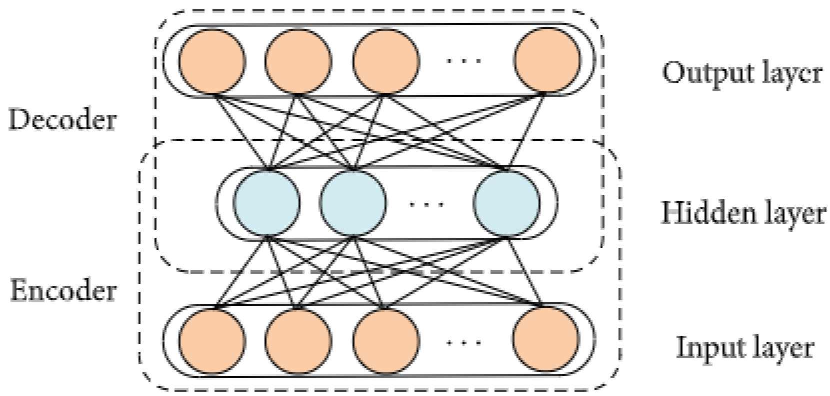

Figure 4.3: Structure of SAE

Chapter 5 Implementation

5.1 Dataset:

Our dataset is a .csv(comma separated values) file named ARUBAs.csv that has 1719557 continuesactivities written after record in 218 days of years2010-2011. This data has 6462 annotats of 11 several activitys.

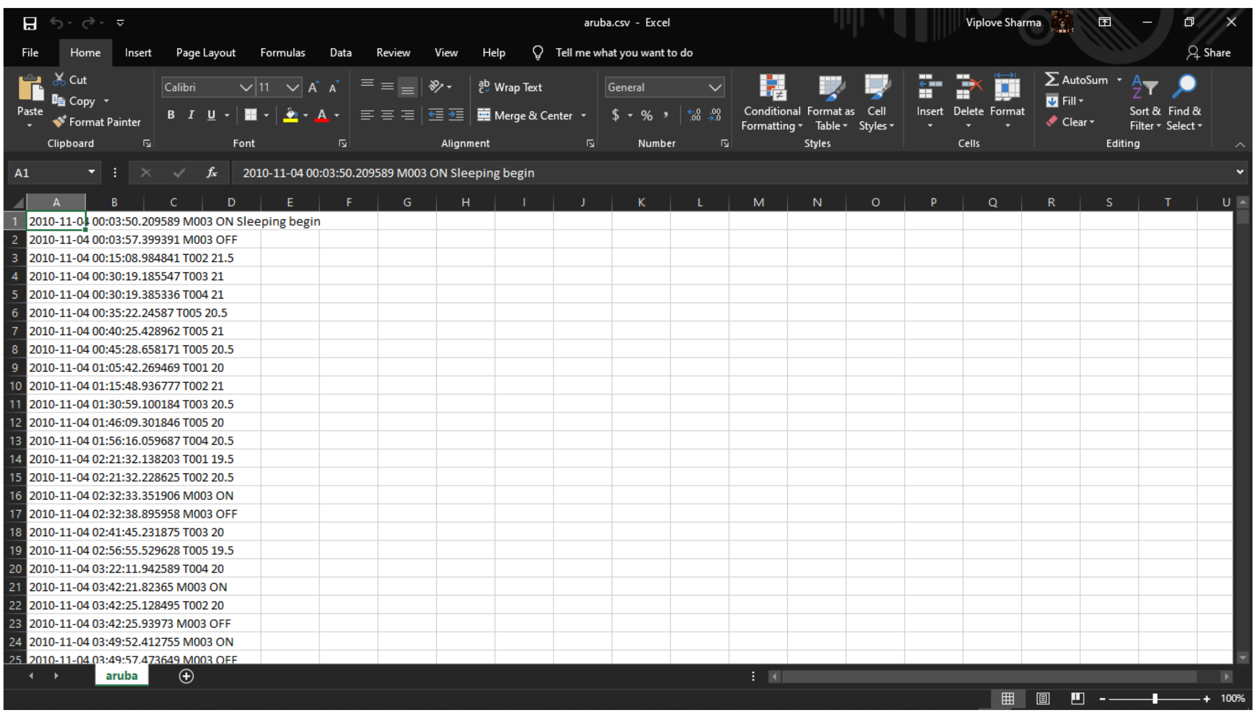

Figure 5.1: ARUBA.csv

CSVfile is a straight and forward excel sheet layout of the form of tables whioch contains infor, like, a spread sheet or databaseMS. Documents in this organization couldbe foreign made or traded of models which stores info in tabular form, like, MicrosoftExcel or Open CV OfficeCalculator. CSV stands for "coomma-seperated values".Rows indicatess detection of motion or not any emotion by motional sensorsss and detecting close or open of the doorss by those door's sensor.

5.2 Pre-processing the dataset

In reality, we would had to categorize every sensor's activate as single task, so thatwe couls state this project a multi-class classificationproblem. To make the data used to train we have to break thios unstructureddata with sliding window techniques. We have shown 3 different ways:

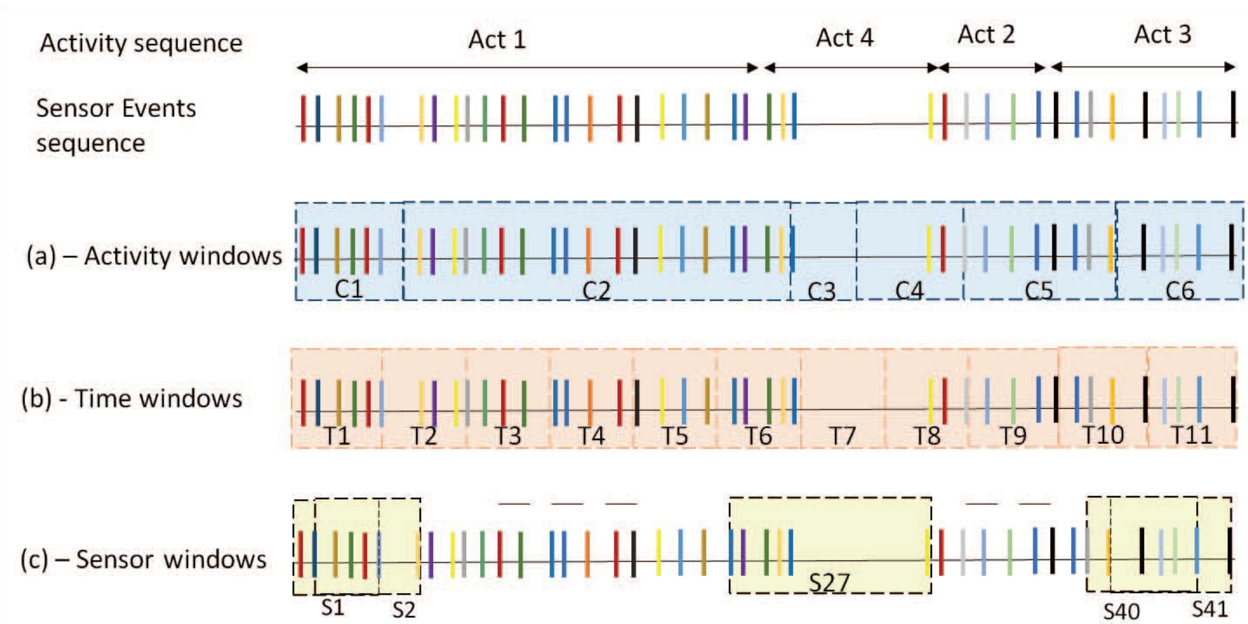

Figure 5.2: Types of segmentation

i.In a) type of segmentisation, we would break the data on different window in the caseof detecting of change in the activities, so that every windowsegment may have 1 activity each.

As most of the times, thgese are not properly differentiable so window-boundary were not accurate. Also it’s unsuitable for the recognization that is onluine due to time bondness constraints to take further results.

ii. Second way of segmening is time-based segmenting windows. We break the dataset to windows having fixed time duration. Small size of window will not have enough info for taking any decision. And for big window size which have sovarious activities, the 1 that is dominating would be depicted more as comparison with others, which can affect the result in a bad way.

iii. Third way is making windows which are sensor-based, using sliding window techniques we can break the data on differentwindows with each window having same no ofactivation of sensors. This technique also have disadvantages. Like 1 windowsegment may have sensors being activarted for different activities. So to deal with that we classifies only the last sensor’s activation, not any other.

For our project last method will be used

5.2.1 Feature Extraction

Firstly we started with creating dataset which can be used by ML or DL models. Using the approach that we discussed earlier that is sensors_based windows approach we extracted features like starting time, ending time,theit time difference, and the no of times each sensor is activated from each window, these are weighted by mutual info procedure. We, also added ids of sensors that activarted in last two spots in every window. So the no. feature becomes 5 + no of sensors in our whole data. For disadvantage of the method, we calculated the mutual info matrics,:

$$MI(i,j) = \frac{1}{|Q|}\sum_{k=1}^{|Q|} \partial(i,j)$$

Aim here is to minimize the inflluences of activatioon of sensors that occurs sometime and are not frequent. The formula that we used that is:$\partial(i, j)$ gives 1 if sensor i and j are used while an activity is being done. No Q here is the no of windows are in total. so we'll get a matrics of N X N size, where N is a no of sensor in dataset and each cell have values inrange of (0to1) which will get multiplied to the correspond values in the feature vector .That feature vector has activations of last 2 sensors.

5.3 Code

We divided the code into two .py files.

i. Data_processor.py – This .py file does the pre-processing work and preprocessesthe unorganised data.

ii. Activity_recognition.py – This .py file does the main work of applying the random forest classifier and predicting the accuracy.

```
import numpy as np
import pandas as pd
import itertools
import os.path
import matplotlib.pyplot as plt
from datetime import datetime
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix

class Processor:
    def __init__(self, sliding_window_length=6,
                 columns_names=('date', 'time', 'sensor_id', 'value', 'activity1',
                                'activity2', 'activity3', 'activity4',
                                'activity5'),
                 data_path=None, date='2011-01-21', date_sign='less',
                 sensor_encoder=None, activity_encoder=None,
                 mutual_matrix=None, sens_num=None, acts_num=None):
        self.columns_names = columns_names
        self.sliding_window_length = sliding_window_length
        self.data_path = data_path
        self.date = date
        self.date_sign = date_sign
        self.sensor_encoder = sensor_encoder
        self.activity_encoder = activity_encoder
        self.mutual_matrix = mutual_matrix
        self.sens_num = sens_num
        self.acts_num = acts_num
        self.data = None
        self.processed = None
        self.data = pd.read_csv(data_path, header=None, sep='\s+', names=self.columns_names)
```

Figure 5.3: Data_processor.py(i)

```python
    def markdown_acitivities(self):
        activity_pool = []

        for index, row in self.data.iterrows():
            if row.activity2 == 'begin':
                break
            else:
                self.data.drop(index, inplace=True)

        for index, row in self.data.iterrows():
            if row.activity2 == 'begin':
                activity_pool.append(row.activity1)
            elif row.activity2 == 'end':
                activity_pool.remove(row.activity1)
            elif activity_pool:
                if len(activity_pool) > 1:
                    self.data.drop(index, inplace=True)
                else:
                    self.data.at[index, 'activity1'] = activity_pool[-1]

    def calculate_mutual_info_matrix(self, mutual_matrix, sensnum):
        current_activity = self.data.iloc[0].activity1
        wind = []
        winds=[]
        winnum = 0

        for idx, row in self.data.iterrows():
            if current_activity == row.activity1:
                wind.append(row.sensor_id)
            else:
                sensors = np.unique(wind)
                for sensor in range(0, sensnum):
                    for sensor2 in range(0, sensnum):
                        if sensor in sensors and sensor2 in sensors:
                            mutual_matrix[sensor][sensor2] += 1
                            winds.append(wind)
                winnum += 1
                current_activity = row.activity1
                wind = [row.sensor_id]
```

Figure 5.4: Data_processor.py (ii)

```
        return mutual_matrix/winnum

    def process(self):
        self.data['datetime'] = pd.to_datetime(self.data['date'] + " " + self.data['time'])

        if self.date_sign == 'more':
            self.data = self.data[(self.data['datetime'] > self.date)]
        else:
            self.data = self.data[self.data['datetime'] < self.date]

        vls = ['begin', 'end']

        #There are datasets that have activity name in
        for index, row in self.data.iterrows():
            if not row.activity2 in vls and not row.activity2 is np.nan:
                if not row.activity3 in vls:
                    if not row.activity4 in vls:
                        self.data.loc[index, 'activity1'] = row.activity1 + "_" + row.activity2 + "_" + row.activity3 + "_" + row.activity4
                        self.data.loc[index, 'activity2'] = row.activity5
                    else:
                        self.data.loc[index, 'activity1'] = row.activity1 + "_" + row.activity2 + "_" + row.activity3
                        self.data.loc[index, 'activity2'] = row.activity4
                else:
                    self.data.loc[index, 'activity1'] = row.activity1 + "_" + row.activity2
                    self.data.loc[index, 'activity2'] = row.activity3

        #There are some typos in dataset in status column
        one_vals = ['ON', 'OPEN', 'ONc', 'ON5', 'ON55', 'ON5c', 'ONcc', 'ONc5c', 'ONc5', 'OPENc', 'O', 'ONM026',
                    'ONM009', 'ONM024']
        zero_vals = ['OFF', 'CLOSE', 'OFF5', 'OFcF', 'OFFc', 'OFFcc', 'OFF5cc', 'OFF5c', 'OFFc5', 'OcFF', 'OFFccc5',
                     'OF', 'CLOSED']

        self.data.loc[self.data['value'].isin(one_vals), 'value'] = 1
        self.data.loc[self.data['value'].isin(zero_vals), 'value'] = 0
        self.markdown_acitivities()
        #we need no data from temperature data
        self.data = self.data[~self.data.sensor_id.str.contains('c')]
        self.data = self.data[~self.data.sensor_id.str.startswith('T')]

```

Figure 5.5: Data_processor.py (iii)

```python
        self.data.loc[self.data['activity1'].isnull(), 'activity1'] = 'other'
        self.data.drop(columns=['date', 'time'], inplace=True)

        if self.sensor_encoder is None:
            self.sensor_encoder = LabelEncoder()
            self.sensor_encoder.fit(self.data.sensor_id)

        if self.activity_encoder is None:
            self.activity_encoder = LabelEncoder()
            self.activity_encoder.fit(self.data.activity1.astype(str))

        # We are not dealing with other activity here
        self.data = self.data[self.data.activity1 != 'other']

        self.data.sensor_id = self.sensor_encoder.transform(self.data.sensor_id)
        self.data.activity1 = self.activity_encoder.transform(self.data.activity1.astype(str))

        if self.sens_num is None:
            self.sens_num = len(self.data.sensor_id.unique())

        if self.acts_num is None:
            self.acts_num = len(self.data.activity1.unique())

        self.data = self.data.reset_index()

        if self.mutual_matrix is None:
            self.mutual_matrix = self.calculate_mutual_info_matrix(np.zeros((self.sens_num, self.sens_num)),
                                                                   self.sens_num)
```

Figure 5.6: Data_processor.py (iv)

```python
    def get_segments(self):
        windows_vectors, activities_vectors = [], [],

        for idx in range(0, self.data.shape[0], 1):
            current_sens = []
            if self.data.shape[0] <= idx + self.sliding_window_length:
                break

            freq = np.zeros(self.sens_num)
            windows = self.data.iloc[idx:idx + self.sliding_window_length]
            last_activity = windows.iloc[self.sliding_window_length - 1].activity1

            start_time = windows.iloc[0].datetime
            last_time = windows.iloc[self.sliding_window_length - 1].datetime
            last_time = datetime(last_time.year, last_time.month, last_time.day, last_time.hour, last_time.minute,
                                 last_time.second)
            start_time = datetime(start_time.year, start_time.month, start_time.day, start_time.hour, start_time.minute,
                                  start_time.second)
            last_time = last_time.hour / 24  + last_time.minute / 60. + last_time.second / 3600.
            start_time = start_time.hour / 24 + start_time.minute / 60. + start_time.second / 3600.
            last_sensor_id = windows.iloc[self.sliding_window_length - 1].sensor_id
            second_last_id = windows.iloc[self.sliding_window_length - 2].sensor_id

            for ind, act in windows.iterrows():
                freq[int(act.sensor_id)] += 1

            #weighting frequencies using mutual matrix table
            for ind, val in enumerate(freq):
                freq[ind] = freq[ind] * self.mutual_matrix[last_sensor_id][ind]

            timespan = last_time - start_time
            freq = np.append(freq, (last_time, timespan, second_last_id, last_sensor_id))

            windows_vectors.append(freq)
            activities_vectors.append(last_activity)

        return np.array(windows_vectors), np.array(activities_vectors)
```

Figure 5.7: Data_processor.py (v)

```python
def get_train_test_data(path, date1, date2):
    train_processor = Processor(date_sign='less', data_path=path, date=date1)
    train_processor.process()
    test_processor = Processor(date_sign='more', data_path=path, date=date2,
                               activity_encoder=train_processor.activity_encoder,
                               sensor_encoder=train_processor.sensor_encoder,
                               mutual_matrix=train_processor.mutual_matrix, sens_num=train_processor.sens_num,
                               acts_num=train_processor.acts_num)
    test_processor.process()
    train_x, train_y = train_processor.get_segments()
    test_x, test_y = test_processor.get_segments()
    return train_x, train_y, test_x, test_y, train_processor.activity_encoder.classes_

```

Figure 5.8: Data_processor.py (vi)

```python
def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):
    """
    This function prints and plots the confusion matrix.
    Normalization can be applied by setting `normalize=True`.
    """
    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
    else:
        print('Confusion matrix, without normalization')

    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    fmt = '.2f' if normalize else 'd'
    thresh = cm.max() / 2.
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, format(cm[i, j], fmt),
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")

    plt.ylabel('True label')
    plt.xlabel('Predicted label')
    plt.tight_layout()
```

Figure 5.9: Data_processor.py(vii)

```python

from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, balanced_accuracy_score
from data_processor import get_train_test_data, plot_confusion_matrix, confusion_matrix
import matplotlib.pyplot as plt
import numpy as np
import sys

DATA_PATH = 'aruba.csv'
DATA_1 = '2011-01-21'
DATA_2 = '2011-05-23'

np.set_printoptions(threshold=sys.maxsize)
np.set_printoptions(precision=2)

train_x, train_y, test_x, test_y, classes = get_train_test_data(DATA_PATH, DATA_1, DATA_2)

clf = RandomForestClassifier(n_jobs=-1, random_state=42, n_estimators=40,
                             max_depth=None, min_samples_split=2, verbose=False,
                             class_weight='balanced')

clf.fit(train_x, train_y)
predictions = clf.predict(test_x)

accuracy_score(predictions, test_y)

balanced_accuracy_score(predictions, test_y)

cnf_matrix = confusion_matrix(predictions, test_y)
plt.figure(figsize=(18, 16))
plot_confusion_matrix(cnf_matrix, classes=classes, normalize=True,
                      title='Normalized confusion matrix')
plt.show()

plt.figure(figsize=(18, 16))
plot_confusion_matrix(cnf_matrix, classes=classes, normalize=False,
                      title='Normalized confusion matrix')
plt.show()
```

Figure 5.10: Activity_recognition.py

5.4 Output

```
In [15]: accuracy_score(predictions, test_y)
Out[15]: 0.93493853273748

In [16]: balanced_accuracy_score(predictions, test_y)
Out[16]: 0.6177441787814696
```

Figure 5.11: Output

Chapter 6 Performance Analysis

6.1 Random Forest

The accuracy of our model on the test set which is measured after the completion of training of the model comes out to be pretty high, and taking account of imbalanced data in our dataset the use of this method of calculation of accuracymetric is not the best method. So, we used balanced_accuracy_score which gives the avg of recall value which we get for every class or label, the result than gets 0.617. Our cm:confusion matrix for this :

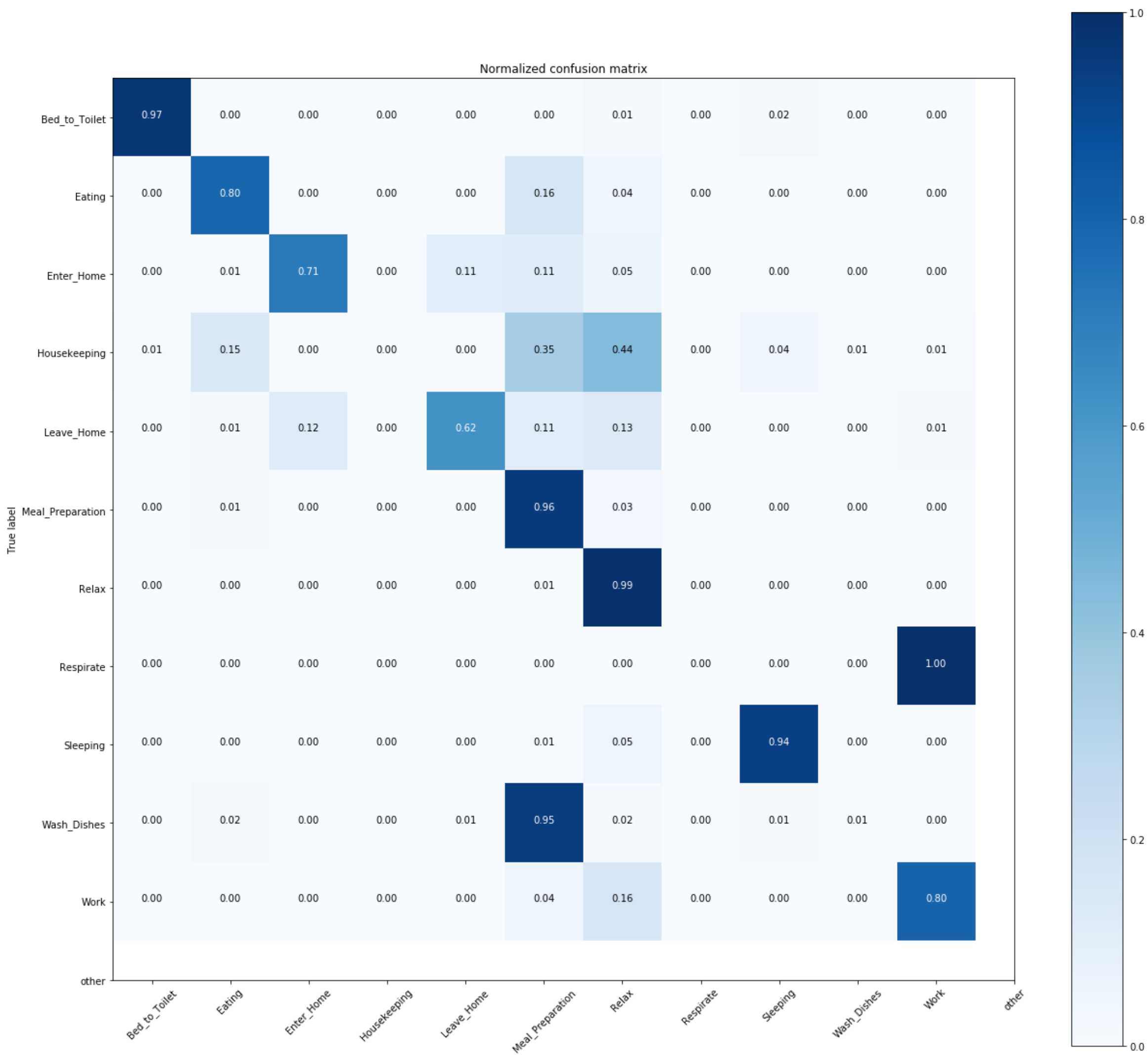

Figure 6.1: Confusion Matrix(i)

Acc to this matrics we have some activities for eg ‘Respirate’, which do not get any predictions all. Let’s see the notnormalized cm:confusion matrix.

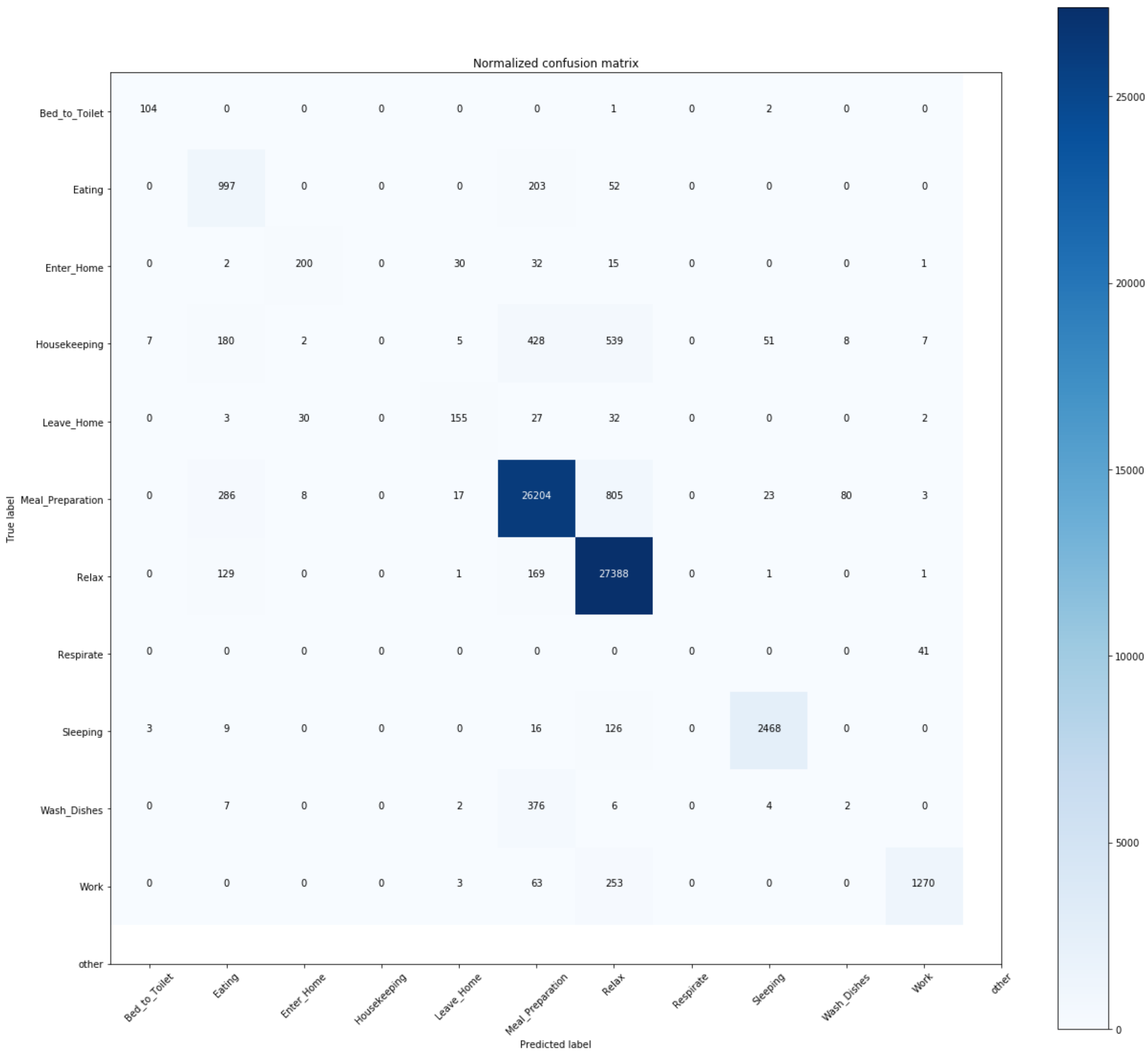

Figure 5.2: Confusion Matrix(ii)

According to thios cm some activities are very rare to be predicted or as happen.So to increase the accuracy of our model we can delete such activities.Activities like‘Housekeeping’ or ‘Wash_Dishes’ were also getpredicted wrong. This happens because oflimitations of binary sensors, as our windowing technique may did overlapped the activities making it difficult to differentiate.

Chapter 7 Conclusion

To conclude, we identified human activities from home setting binary sensors dataset by applying Random Forest Classifier and got accuracy up to 93%.

7.1 Future works

Recognising activities in a smart house is very difficult and this problem is still open to all. There are somany various methods to sove this problem. Like, there is a researchpaper which uses kNNs and modified versions to that. There is an technique which uses the Latent DirichletAllocation method. Another paper proposed a fmwork, having 2 steps: off-line model making and online activity recognization.

And to further expands the applications and implementations of thi project a Reccomendation system can be develoved or modeled to blow an alarm or recommend an activity or send signal to distant places in case of the activities goes off track to the usual routine or any activity which looks suspicious.This will complete the purpose of recognizing the activity.

References

[1] Krishnan, Narayanan & Cook, Diane. (2014). Activity Recognition on Streaming Sensor Data. Pervasive and mobile computing. 10. 138–154. 10.1016/j.pmcj.2012.07.003.

[2] Cook, Diane & Crandall, Aaron & Thomas, Brian & Krishnan, Narayanan. (2013). CASAS: A smart home in a box. Computer. 46. 10.1109/MC.2012.328.

[3] Yala, Nawal & Fergani, Belkacem & Fleury, Anthony. (2015). Feature extraction for human activity recognition on streaming data. 1–6. 10.1109/INISTA.2015.7276759.

[4] L. Bao. Physical activity recognition from acceleration data under semi-naturalistic conditions. M.Eng thesis, EECS, Massachusetts Institute of Tech-nology, 2003.

[5] T. Barger, M. Alwan, S. Kell, B. Turner, S. Wood, and A. Naidu. Objective remote assessment of activities of daily living: Analysis of meal preparation patterns. Poster presentation, Medical Automation Research Center, University of Virginia Health System, 2002.

[6] J. S. Beaudin. From personal experience to design: Externalizing the home-owner's needs and assesment process. MS thesis, MIT Media Laboratory, Mas-sachusetts Institute of Technology, 2003.

[7] M. Mozer. The neural network house: an environment that adapts to its in-habitants. In Proceedings of the AAAI Spring Symposium on Intelligent Envi-ronments, Technical Report SS-98-02, pages 110–114. AAAI Press, Menlo Park, CA, 1998.

[8] K. P. Murphy. Dynamic Bayesian Networks: Representation, Inference and Learning. PhD thesis, University of California, Berkeley, 2002.

major_shivam

ORIGINALITY REPORT

8%	5%	1%	5%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	medium.com Internet Source	3%
2	Submitted to Jaypee University of Information Technology Student Paper	3%
3	courses.media.mit.edu Internet Source	1%
4	Submitted to Nelson and Colne College Student Paper	<1%
5	Emmanuel Munguia Tapia. "Activity Recognition in the Home Using Simple and Ubiquitous Sensors", Lecture Notes in Computer Science, 2004 Publication	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: …………14/07/2020………………

Type of Document (Tick): PhD Thesis | M.Tech Dissertation/ Report | B.Tech Project Report | Paper

Name: ____Shivam Sharma __**Department:** ___CSE__ **Enrolment No** ____161242_________________

Contact No. ____8351018415_____E-mail. ______shivam.sharma1836@gmail.com________________

Name of the Supervisor: ___________Dr. Rajinder Sandhu_______________________________________

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): ________________________

_____________ACTIVITY RECOGNITION BASED RECOMMENDATION SYSTEM______________________________________

____

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

- Total No. of Pages =40
- Total No. of Preliminary pages =9
- Total No. of pages accommodate bibliography/references =2

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at …8%………..(%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

(Signature of Guide/Supervisor) **Signature of HOD**

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received on	Excluded	Similarity Index (%)	Abstract & Chapters Details	
	• All Preliminary Pages • Bibliography/ Images/Quotes • 14 Words String		Word Counts	
			Character Counts	
Report Generated on		**Submission ID**	Page counts	
			File Size	

Checked by
Name & Signature **Librarian**

………

Please send your complete Thesis/Report in (PDF) & DOC (Word File) through your Supervisor/Guide at plagcheck.juit@gmail.com