A
Preliminary Project Report
On

"Machine Learning On Real-time Data To Enhance Home Automation"

Submitted to

Savitribai Phule Pune University

FOR THE DEGREE
OF

Bachelor of Computer Engineering

BY

Mr. Ayyaj Attar	B120634202
Mr. Sameer Shaikh	B120634211
Mr. Siddik Pathan	B120634216

Under the guidance of
Prof. R. Shaikh

Department of Computer Engineering
Al-Ameen College of Engineering
Pune – 412 216

Year 2016-2017

Certificate

This is to certify that,

Mr. AYYAJ ATTAR	Exam Seat No: B120634202
Mr. SAMEER SHAIKH	Exam Seat No: B120634211
Mr. SIDDIK PATHAN	Exam Seat No: B120634216

have successfully completed their preliminary project report entitled "**MACHINE LEARNING ON REAL–TIME DATA TO ENHANCE HOME AUTOMATION**", under our supervision and guidance in partial fulfillment of the requirements for the degree of Bachelor of Engineering in Department of Computer Engineering of Savitribai Phule Pune University during the academic year 2016-2017.

Prof. R. Shaikh
Guide

Prof. G. S. Kothawale
HOD

Principal
AAEMF's COE & MS

Internal Examiner

External Examiner

Date:
Place:

Acknowledgments

It is a great pleasure for us to acknowledge the assistance and contribution of a number of individuals and entities who helped us in presenting this project report on **"Machine Learning On Real-time Data To Enhance Home Automation"**.

First and foremost we wish to record our gratitude and thanks to - our Project Guide **Prof. Rukaiya Shaikh**, the Head of Department **Prof. G.S. Kothawale**, and the **Principal, Mrs. S. Kapse** - for their guidance, instruction and enthusiasm leading to the successful completion of this report.

This acknowledgement would be incomplete without expressing our special thanks to the authors and organizations maintaining the various open-source tools that we have used in our work - especially to Linus Torvalds for his college project that turned into what we know today as the Linux Kernel.

Last but not the least, we would like to thank our parents, friends and our colleagues who helped us directly or indirectly in successfully completing this project report.

Mr. Ayyaj Attar
Mr. Sameer Shaikh
Mr. Siddik Pathan

Date:
Place:

Abstract

Traditional home automation systems are mostly hard-coded or require manual automation plan generation by users. This requires interaction between a control system (an interface) and the user, requiring quite some effort and time to be put in manual planning of the home environment. This project intends to explore applying ML on real-time usage data to generate personalized and time-variant home automation plans. These plans will save the user time and effort, leading to a smoother ML driven home automation experience. We collect streaming usage statistics from smart-home occupants and store it on a centralized server. Simultaneously, we also collect external data (which may consist of environmental factors like natural light intensity, wind speed, et cetera) which may influence occupants usage behavior. These datasets are combined, with data timestamps as a unique identifying field, into a super-set. Its then fed into a Machine Learning system to correlate user habits with time of the day and the external factors. The correlation hence established will be updated in real time. This correlation will be in the form of a prediction model that will be used to predict near future values of target devices for the occupant. Hence, by combining real-time usage data from a conventional home automation system and Machine Learning, we will be able to provide smoother and more comfortable environment to the users, as the burden of plan generation will be greatly reduced.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Detailed Problem Definition

Most present-day smart homes use simple reflex agents for automation. Simple reflex agents are non-flexible and can work with limited percepts and hard-coded actuation rules. These rules may not suit all people. This rigidity in usability of present consumer automation systems forms the core of our problem.

We intend to develop a software solution to this problem, that is centered around machine learning. This solution will be in the form of a learning agent that learns user habits by observation and anomaly detection.

1.2 Brief Description

This project aims to enhance the home automation experience by collecting usage data from the user and applying prediction algorithms on it to predict the next step the user may take. Furthermore, external data will also be collected an correlated with the usage data in order to determine what external conditions may influence the user's behavior. This involves data like weather data and traffic data.

Chapter 2

Analysis

2.1 Project Plan

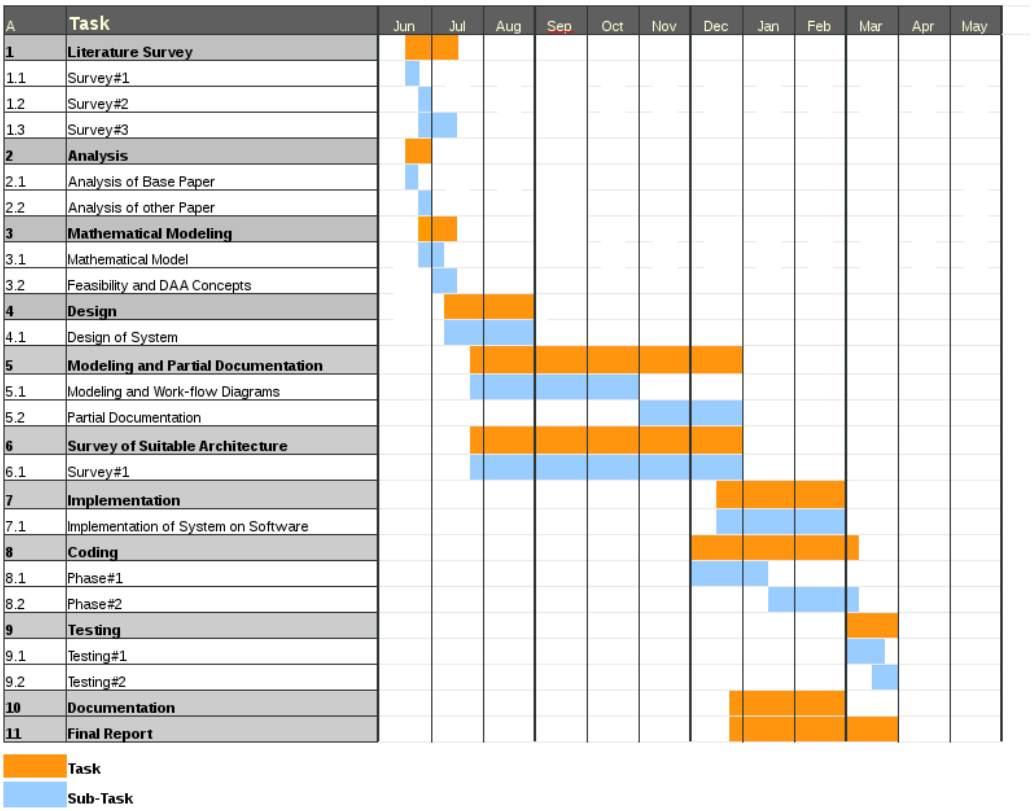

Figure 2.1: Project Plan

2.2 Requirement Analysis

¡Content¿

2.3 Team Structure

¡Content¿

Chapter 3

Design

3.1 Software Requirements

For the PC (LaaS server):

1. Any operating system with a Python interpreter available
2. Python 2.7
3. Flask micro-framework for implementing web service (package "flask")
4. SciKit Learn ML library for Python (package "sklearn")

For the Raspberry Pi (device controller):

1. Raspbian OS for the Pi
2. Python 2.7
3. Flask micro-framework for implementing controller's web interface (package "flask")

3.2 Hardware Requirements

1. A personal computer
2. A Raspberry Pi Single Board Computer (SBC)
3. Proper network infrastructure

3.3 Software Requirements Specification

3.3.1 Functionality

We intend the software to be divided over two tiers: the interface and the LaaS Server. The user is served up a web interface to the smart devices connected to the Raspberry Pi over a conventional TCP/IP network. It will be possible to view the states of individual devices and change them. Such usage of the system can happen even in the absence of the Machine Learning Service. That is to say, the first tier consists of an independent interfacing system that enables the user to control connected smart devices, and this may optionally utilize a usage-prediction-oriented machine learning service if it is available.

The second tier consists of a special case of SaaS (Software-as-a-Service), called LaaS (Learning-as-a-Service). This service is intended to provide proper WebAPI for the tier one application to communicate with. Data can be sent by the tier one application to the LaaS service using this API, which will then be processed continuously to generate predictions for individual devices. These predictions can also be requested by the tier one application so it can apply it in real-time to the smart environment it is providing an interface to. Non-availability of the services in this tier will not affect the user interfacing system implemented in the first tier, except for the unavailability of predictions.

Unavailability of the first tier, id est interface tier, is fatally deteriorating to the user experience. This tier must always be available and smoothly working, that is to say, it should be robust. The second tier provides a service which is not necessarily central to the functioning of the smart home by design. This means that unavailability of the service will not affect the core control system provided by the first tier.

3.3.2 External Interfaces

Interfacing between different components of this system is crucial to it's execution. We will use conventional communication mechanisms like RESTful WebAPIs for communication between both the client & first tier, and the first & second tier.

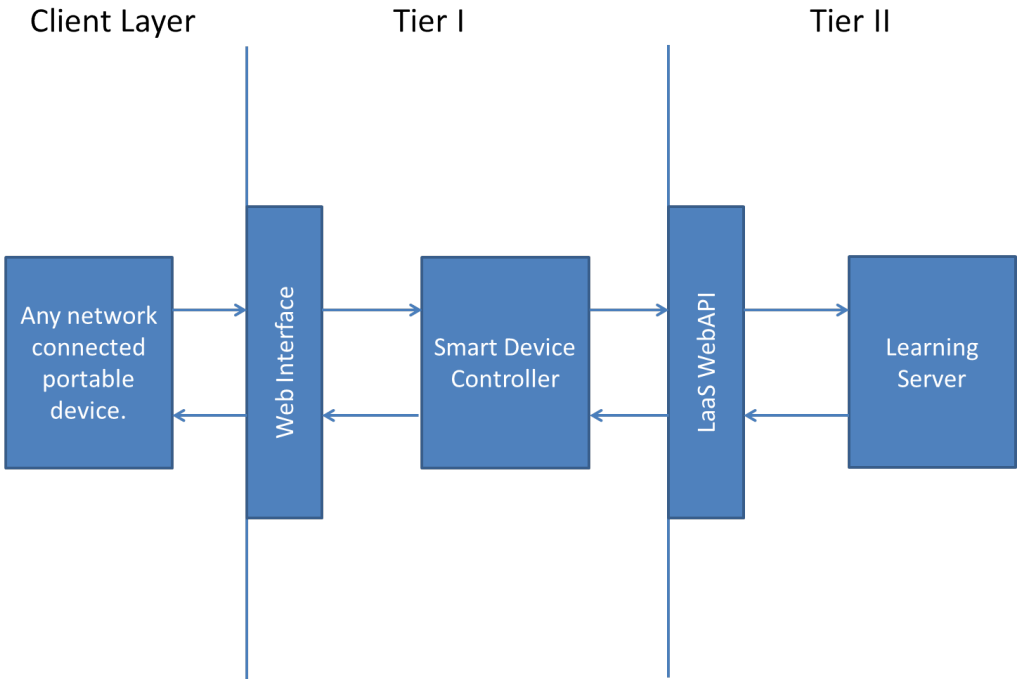

Figure 3.1: Overview of Tiered Components and their Interfacing

Client - Tier I Interface

The smart environment can be controlled via a web app. This app can be accessed by the user by connecting to the smart devices' controller (the Raspberry Pi) on the standard HTTP Port (port 80). The user can then view current states of the connected smart devices and change them according to their preference.

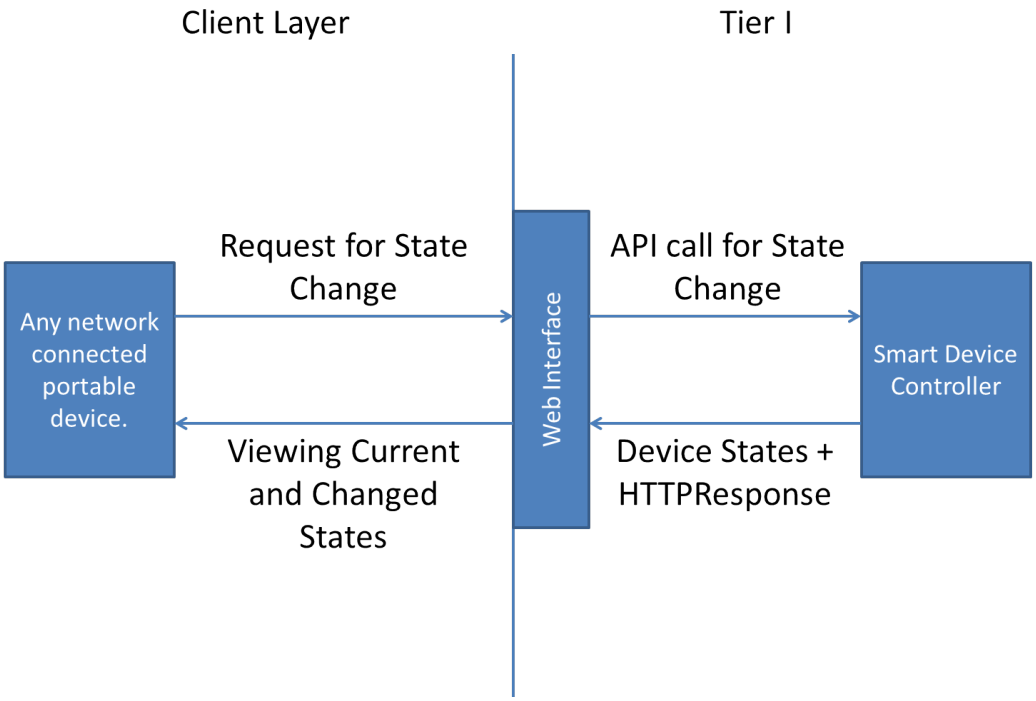

Figure 3.2: Interface between Client and Tier I

Besides collecting user generated actions, the controller may also collect external data (like current weather) in order to facilitate prediction of user actions based on regressive correlation with external influncers.

This interface is critical as it delivers the frontend for our system to the user. Network delay or inconsistencies within its implementation directly and drastically affect user experience.

Tier I - Tier II Interface

The LaaS server can be accessed by the Tier I application through a RESTful WebAPI. The API includes calls for collecting device state changes made by the user and for requesting the prediction based on recent usage data.

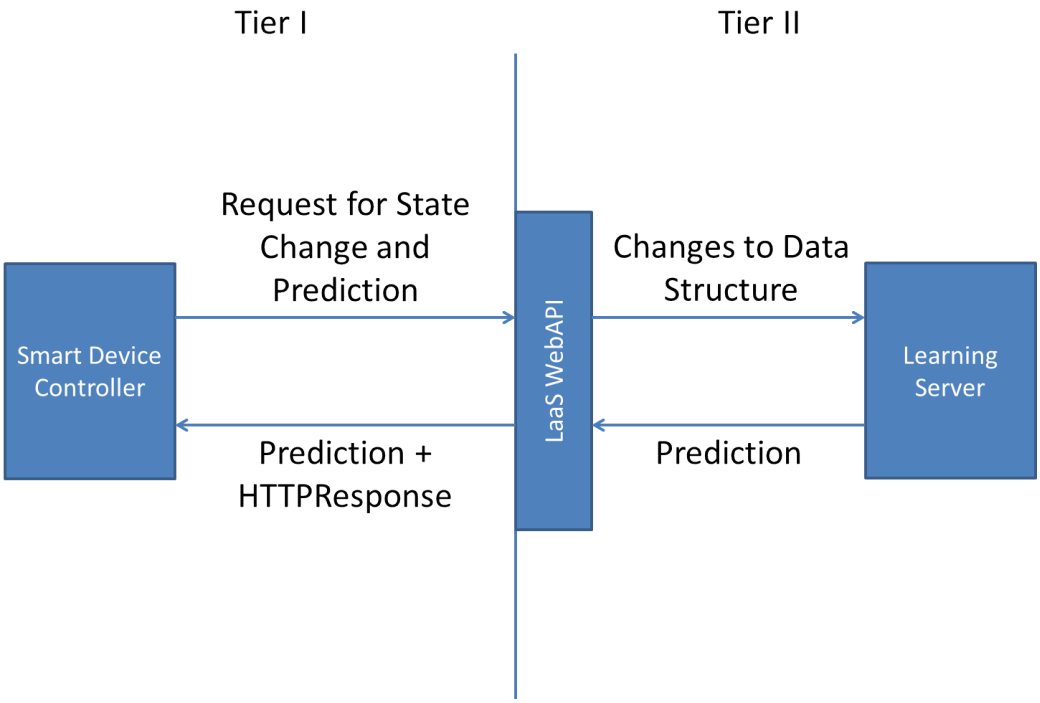

Figure 3.3: Interface between Tier I and Tier II

The only job of the LaaS server is to learn correlations between user actions and external influencing factors, and report the predictions thus generated back to the controller upon request. This server may be hosted locally within the controller's LAN, or publicly, as an on-cloud SaaS Service.

3.3.3 Required Performance

The performance of this system completely depends on the performance of individual machines and the speed their communication. This requires us to again describe the required performance with respect to individual tiers.

Tier I

This tier is performance critical and must always be available to the user. This is mainly because it serves the interface to the smart home environment, which is the most basic functional requirement of our system.

Besides, this tier should be able to work even with the unavailability of Tier II, which provides a supplementary, non-essential service. Also, the delay between the client and Tier I interface should be unnoticeable, as it can directly impact user experience as well as usability.

Tier II

This tier is not as critical to the performance of the system as Tier I. Although unavailability of the services provided by this system can deteriorate the user experience and usability, it will not render the system totally unusable, unlike Tier I.

While the system is running, it is expected to serve up predictions based on current actions quick enough that the user doesn't notice the delay. To achieve this, we need to make sure that the machine that hosts this server is fast enough to achieve this, and also that the network delay between the two tiers is the minimum possible.

3.3.4 Quality Attributes

This section describes the attributes that determine the quality of the software being written. The three attributes that we can identify within this system are Availability, Usability, and Functionality.

Availability

The system should provide at least a minimal device control interface at all times. Although, it should also provide proper and timely predictions for the user's actions most of the time. Then we can say that this system has an acceptable record of availability.

Usability

The system should be easy to handle for the user and provide a simple interface. Delay between an interface action (pressing a button) and response (changes in device states) should be minimal to the point of being unnoticeable. This requires the interfaces between client & Tier I, and Tier I & Tier II to be implemented with the main focus on speed. Such an implementation can be achieved using Responsive UI Design paradigms.

Functionality

Functionality of the system can be judged based on whether the right components and services are provided and work the way they're expected to. This involves observing how well the system responds to user actions and in what ways. Also, the responses generated should be in line with the developed tests. Each test case should be satisfied by the system at least under ideal condition, like availability of services and network connectivity.

Functionality of any system is subject to its usage, environment, et cetera, just as well as it is to its implementation. Nonetheless, we try to include as many possible test cases as are practically possible during the time of development.

3.3.5 Design Constraints

This system is entirely designed using Python (for the back-end on Tier I and II) and HTML/Javascript (for the front-end on Tier I). Tier I is implemented on a Raspberry Pi 2, model B+ - an ARM Cortex A7-based 32-bit quad-core Single Board Computer with 512 MB RAM and frequency of 900 MHz, and running the default Raspbian OS. Python is installed by default on these systems. With this, we need to install the python-flask package - a lightweight framework for implementing robust WSGI applications.

Tier II is targeted towards conventional PCs, and should execute smoothly on mid-level to performance-level PCs as well as workstations and servers. For our implementation, we've used a Linux distribution with Python 2, python-flask (Flask web framework library) and python-sklearn (SciKit Learn ML library) installed. Other operating systems that have a Python interpreter available for them can also be used.

The system can be initiated by running the following individual modules:

1. Raspberry Pi module, which implements the user interface to the smart devices and the connection and control logic for the connected devices. This module forms the Tier I of the system.

2. Server module, which can run on a conventional personal computer and implements the "brains" of the system. That is to say, it provides the system with usage-learning capabilities in the form of a conventional WebAPI. This module forms the Tier II of the system.

For communication, we use conventional TCP/IP network. This implementation is currently targeted towards LAN-based in-home usage and purposefully leaves out even common security features in order to manifest the core idea in a simple way.

Chapter 4

Modelling

4.1 Data Flow Diagram

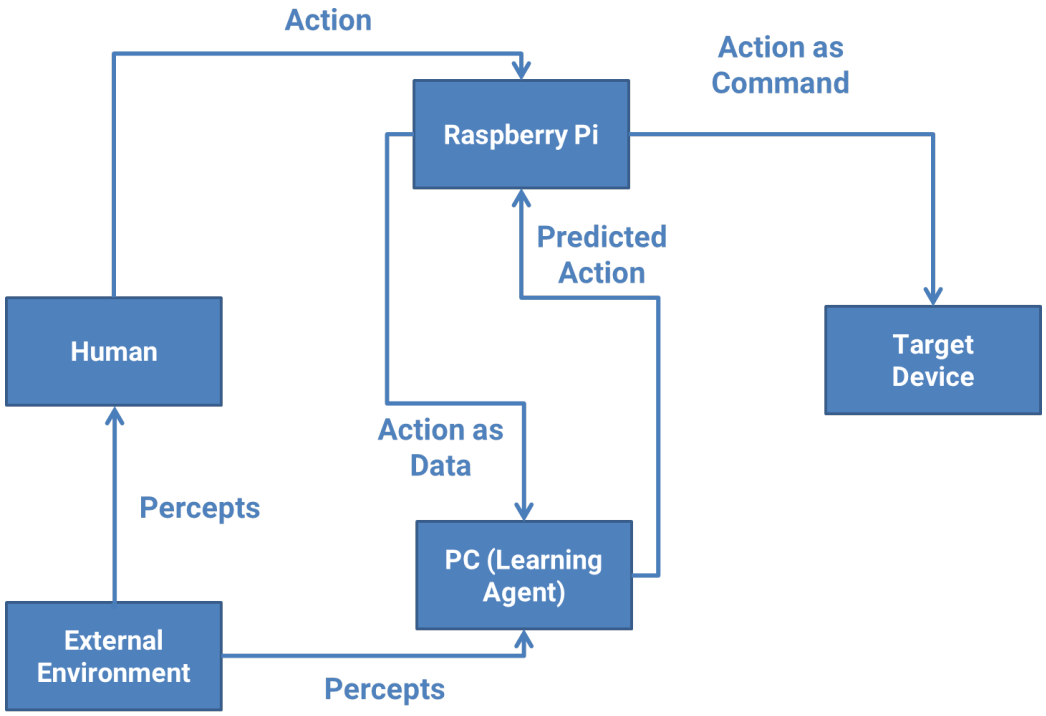

Figure 4.1: Data Flow Diagram

4.2 State Transition Diagram

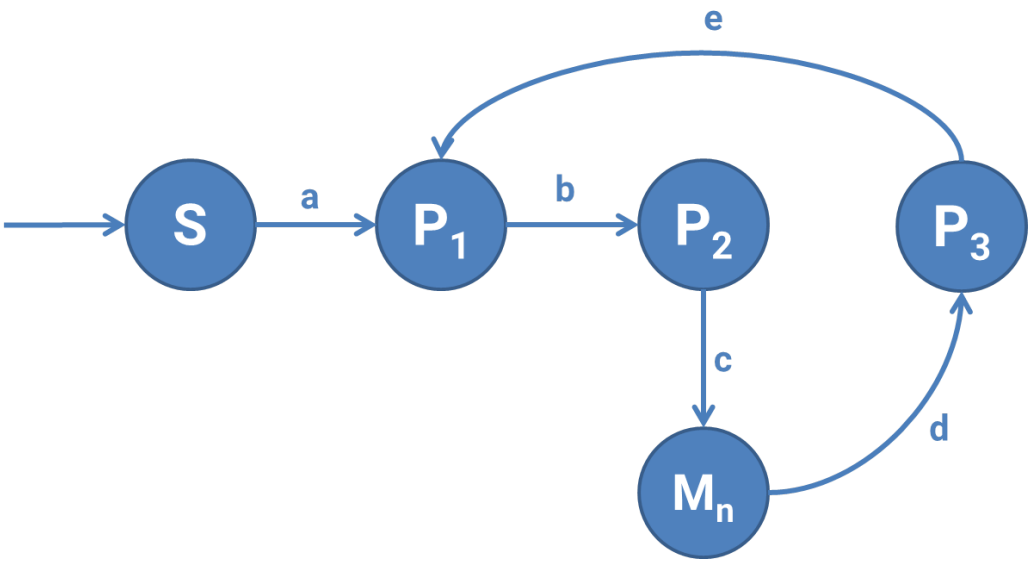

Figure 4.2: State Transition Diagram

Where,

S: Start state

P1: Phase 1- Observation and collection of usage data.

P2: Phase 2- Run machine learning algorithms on collected data to generate a prediction model.

Mn: Prediction Model- Current (nth) prediction model. n is the total number of discrete anomalies detected since start.

P3: Phase 3- Detect anomalies and modify data collected in Phase 1 to match the anomalies.

a: Create a mutable table for holding observed data for the learning agent to process.

b: Data collection threshold reached or data modification completed.

c: Prediction Model generated.

d: Anomaly detected. This implies change in user habit or detection of a new habit.

e: Make corrections in the table to avoid anomalies in the near future.

Our system does not have any state of absolute success. Success and failure are both temporary and the system is designed to learn from its mis-predictions. The canonical success in this project will be the usability of the machine learning system. The more data it is exposed to, the more successful the prediction model will be.

Chapter 5

System Design

5.1 System Architecture

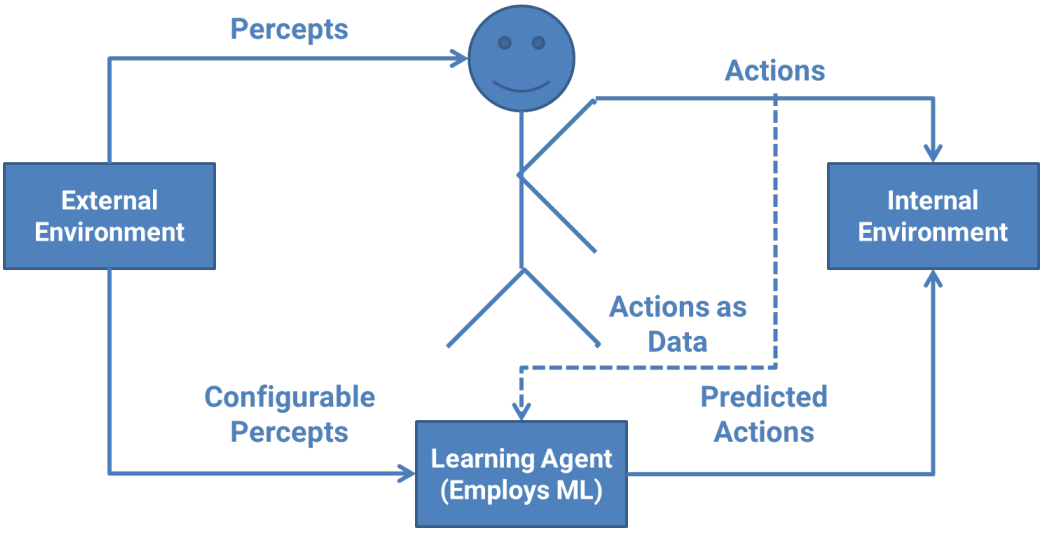

Figure 5.1: System Architecture

This defines the major elements within our system, their arrangement and interaction. It essentially represents an agent-environment model with the human and the Learning Agent as the two agents.

5.2 UML Diagrams

5.2.1 Use Case Diagram

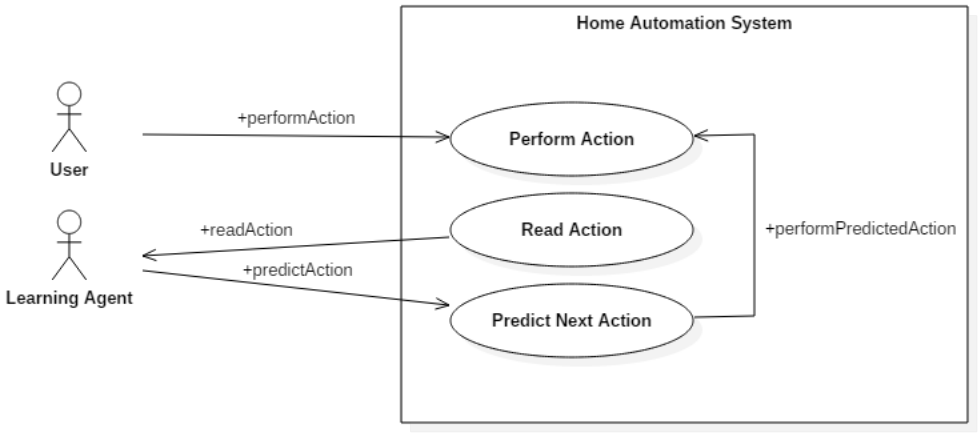

Figure 5.2: Use Case Diagram

This represents the various use cases that are possible in our system. The human user is the primary actor which may perform certain actions to initiate learning in a secondary learning agent that mimics the user's usage patterns.

5.2.2 Class Diagram

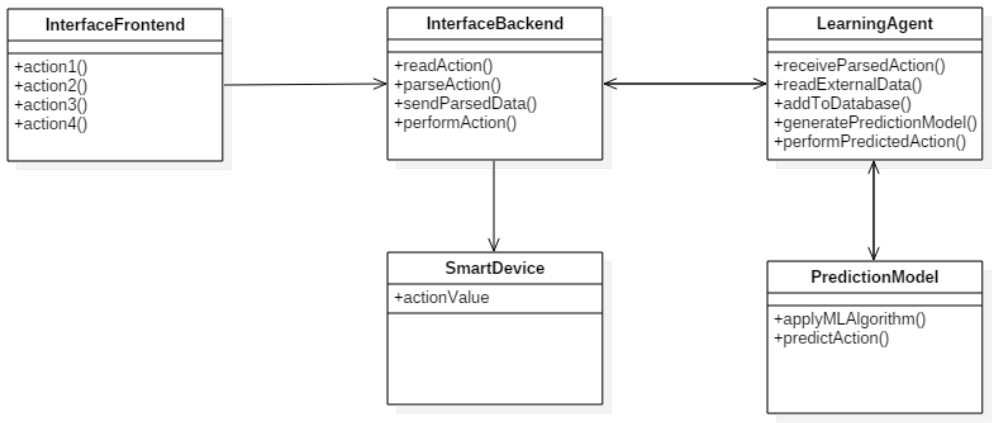

Figure 5.3: Class Diagram

This abstracts the major entities within our system as classes and shows the features they offer and the relationship between them.

5.2.3 Activity Diagram

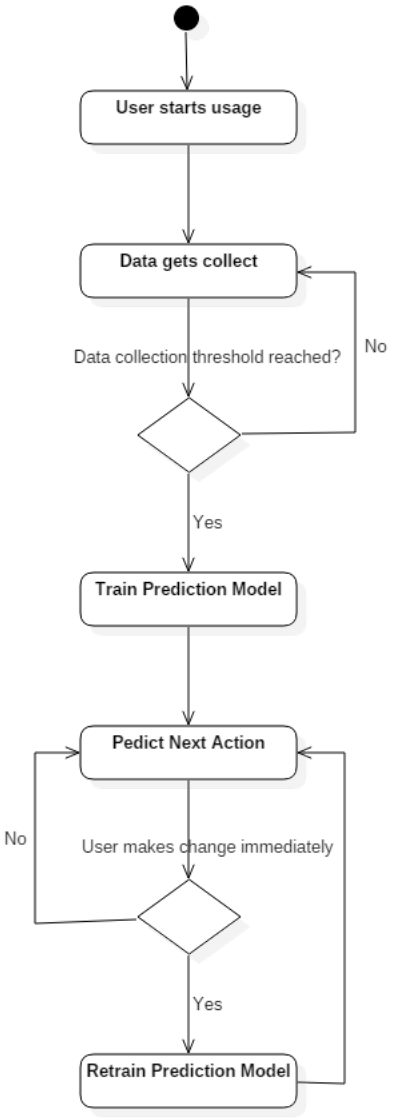

Figure 5.4: Activity Diagram

The activity diagram shows the flow of control through the process of our system. This system is designed to be non-terminating, so there is no halt in the flow. The system improves itself continually by realizing incorrect predictions.

5.2.4 Sequence Diagram

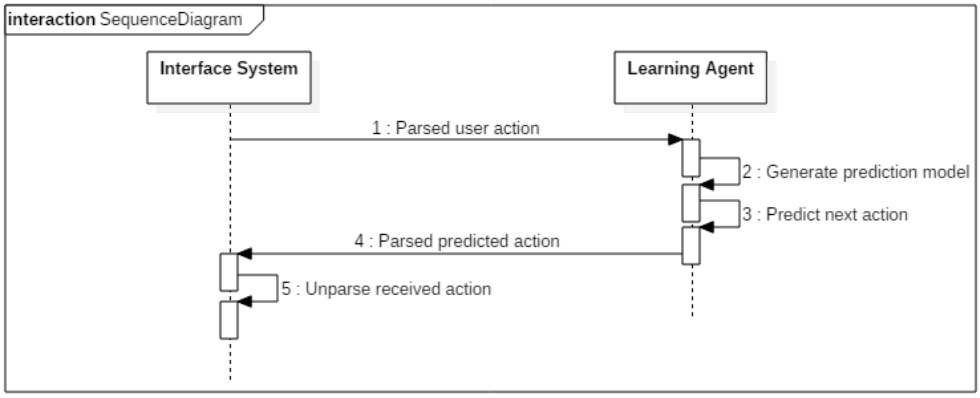

Figure 5.5: Sequence Diagram

It shows the flow of objective-critical data and signals between some major entities within the system.

Chapter 6

Coding

6.1 Algorithm

¡Stick LogReg history here.¿

6.2 References to Technology

6.2.1 Raspberry Pi

The Raspberry Pi is an ARM-based Single Board Computer. It was initially launched in the UK to teach programming to school children but being very cheap yet powerful, it became a favorite among computer hobbyists around the world. The latest revision available at the time of writing this report is revision 3. Raspberry Pi runs it's own flavor of Linux based on Debian called Raspbian OS.

This system is entirely designed using Python (for the back-end on Tier I and II) and HTML/Javascript (for the front-end on Tier I). Tier I is implemented on a Raspberry Pi 2, model B+ - an ARM Cortex A7-based 32-bit quad-core Single Board Computer with 512 MB RAM and frequency of 900 MHz, and running the default Raspbian OS. Python is installed by default on these systems. With this, we need to install the python-flask package - a lightweight framework for implementing robust WSGI applications.

6.2.2 Python

Python is an interpreted language with object oriented features.¡Add python history and specs here¿

6.2.3 Flask (Python Library)

¡Add flask history here¿

6.2.4 SciKit Learn (Python Library)

¡Add SciKit Learn history here.¿

6.2.5 HTML

¡Add description.¿

6.2.6 JavaScript

¡Add history and description¿

6.2.7 REST Web APIs

¡Add description.¿

6.3 Advantages

1. **Raspberry Pi** It's a cheap and yet complete, Linux-powered computer that can be used in lieu of many things like conventional microcontrollers, data sinks, and under-utilized workstations to name a few. It's size and cost also make it suitable to be used as a device interface in our project. Having a Python library to control its GPIO pins also makes it easier to program.

2. **Python** ¡Add advantage¿

3. **Flask** ¡Add advantage over other WSGI frameworks¿

4. **SciKit Learn** By using this library, we are spared the effort of rewriting machine learning algorithms that we intend to use. Being in Python, importing and using this library becomes as easy as writing conventional Python code.

5. **JavaScript** ¡Add advantage over other scripting languages¿

6. **REST Web APIs** ¡Add advantage over SOAP¿

6.4 Applications

1. **Machine Learning** Being a broad field of study, it finds application in a wide variety of fields. One such new field is enhancing classical automation systems for smart homes and non-critical industrial operations by working on behavioral data in real-time. Although we are more focused on home automation, this system can be modified to suit even industrial systems.

2. **Raspberry Pi** It is a complete computer system in itself and can be used in a variety of places. It's major applications include use as PC, smart device, hobby computer, teaching aid, IoT Hub etc.

3. **SciKit Learn** Applications of this library are just as vast as that of machine learning itself. It has been used by various projects dealing with everything from artificial intelligence to big data analytics.

Chapter 7

Result Sets

¡MORE INFO NEEDED¿

Chapter 8

Testing

8.1 Test Plan

¡Content¿

8.2 Test Cases

¡Content¿

8.3 Test Results

¡Content¿

Chapter 9

Conclusion

With the rise of Internet of Things and more powerful computers, we will be able to achieve Utopian homes using Smart Automation powered by Machine Learning Algorithms of higher complexity than Temporal Difference based Reinforcement Learning running on current data. More data will lead to better prediction of potential user action which will help us lead more comfortable lives. Physically challenged people can also benefit from such systems which will eventually make them more independent, as more and more data gets collected to predict such users' habits. The impact of this technology on human lives will be deep and possibly every human-machine interface in the future will have some form of machine learning powered intelligent assistance. Changes in user habits can also be used to predict a lot of other things about the user such as the user's physical and mental health. In fact, many current technologies aim to provide basic level medical assistance using machine learning systems.

References

[1] Panos Luridas, Christof Ebert,"Machine Learning", IEEE Computer Society, 2016

[2] Jeremie Saives, Clement Pianon, and Gregory Faraut, "Activity Discovery and Detection of Behavioral Deviations", in IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING, 2016

[3] Christopher Osiegbu, Seifemichael B. Amsalu, Fatemeh Afghah, Daniel Limbrick and Abdollah Homaifar, "Design and Implementation of an Autonomous Wireless Sensor-based Smart Home", IEEE, 2015

[4] Wesllen S. Lima, Eduardo Souto, Richard W. Pazzi, Ferry Pramudianto, "User Activity Recognition for Energy Saving in Smart Home Environment", IEEE Symposium on Computers and Communication, 2015

[5] Vahid Ghasemi, Ali Akbar Pouyan, "Activity Recognition in Smart Homes Using Absolute Temporal Information in Dynamic Graphical Models", IEEE Computer Society, 2015

[6] Beichen Chen, Zhong Fan, and Fengming Cao, "Activity Recognition Based on Streaming Sensor Data for Assisted Living in Smart Homes", IEEE Computer Society, 2015

[7] K.S.Gayathri, Susan Elias, S.Shivashankar, "Composite activity recognition in smart homes using Markov Logic Network", IEEE Computer Society, 2015

[8] Labiba Gillani Fahad & Muttukrishnan Rajarajan,"Anomalies detection in smart-home activities", in IEEE 14th International Conference on Machine Learning and Applications, 2015

[9] Z. Meng, and J. Lu, "A Rule-based Service Customization Strategy Context- aware Automation for Smart Home", in IEEE TRANSACTIONS ON MOBILE COMPUTING, 2014

[10] Chiming Chang, Paul-Armand Verhaegen, Joost R. Duflou, "A Comparison of Classifiers for Intelligent Machine Usage Prediction", IEEE Compuer Society, 2014

[11] Parisa Rashidi, Student Member, IEEE, and Diane J. Cook, Fellow, "Keeping the Resident in the Loop: Adapting the Smart Home to the User", in IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS, 2009

Plagiarism Check Report

Paper Published Details