FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Smart-Homes Activity Pattern Recognition: A Comparative Study

António Manuel Vieira Ramadas

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Hugo Sereno Ferreira, PhD

July 25, 2018

Smart-Homes Activity Pattern Recognition: A Comparative Study

António Manuel Vieira Ramadas

Mestrado Integrado em Engenharia Informática e Computação

July 25, 2018

Abstract

Nowadays, the Internet of Things industry is in expansion by being a market each time more profitable. Thus, smart devices already started to appear, and they distinguish themselves from normal devices because they are able to serve as collectors and as actuators.

Therefore, the research community has been releasing proposals that make use of these devices to empower a smart home. In fact, one of the goals has been the classification of user behaviour inside a house. Currently, there are two concerns being addressed: predict actions and/or activities. An action is a change of state of a device, and an activity is a set of actions. Once a system is able to successfully make these classifications, it can assist the inhabitant by automating the devices (e.g., turn off the lights and turn on the television once the user sits on the couch to watch a movie).

Actually, most of the proposals that address one or both of these concerns use a customised context or metrics that are scarcely shared by the rest of the community. Concretely, some proposals use a private simulated dataset, or use only a part of a public dataset, or even use different metrics to report the performance of the proposal. Additionally, some aim to predict actions and others predict activities. Hence, the task of naming a single best overall method reveals itself to be a burden.

Consequently, the current goal of this document is to develop a computational study on multiple methods regarding the classification of activities. Thus, the current work details about the learning environment (i.e., dataset or input) used, the metrics applied to enable the comparison, and the methods are described, discussed and compared. In fact, the data used is the one provided by the Massachusetts Institute of Technology, because it is labelled (each sensor used has its activity associated) and it is widely used by the research community despite its short length (just 2-weeks). Regarding the metrics, F1, Precision and Recall analyse the level of performance of the different methods.

Moreover, the methods used are Naive Bayes, Random Forest, Reinforcement Learning and Recurrent Neural Networks. All these make a different approach, but all yield the same output (i.e., the activity being predicted). Hence, with the comparison enabled, Recurrent Neural Networks, more specifically Gated Recurrent Unit, revealed to be the one with greatest generalisation power.

Finally, current work does not close the smart-homes research. This document concludes several opportunities to study different ways to impact the field, such as the creation of a larger dataset, or even a study to enable autonomous learning by the devices.

Resumo

Atualmente, a indústria da *Internet of Things* (Internet das Coisas) está em expansão por ser um mercado cada vez mais lucrativo. Assim, dispositivos inteligentes já começaram a aparecer, e distinguem-se dos dispositivos normais, pois são capazes de servir como coletores e como atuadores.

Portanto, a comunidade de investigadores tem divulgado propostas que fazem uso desses dispositivos para desenvolver uma casa inteligente. De facto, um dos objetivos tem sido a classificação do comportamento do habitante. Atualmente, há duas preocupações a ser abordadas: prever ações e/ou atividades. Uma ação é uma mudança do estado de um dispositivo e uma atividade é um conjunto de ações. Quando um sistema é capaz de fazer essas classificações com sucesso, ele pode ajudar o habitante automatizando os dispositivos (por exemplo, apagar as luzes e ligar a televisão assim que o habitante se sentar no sofá para assistir a um filme).

Na verdade, a maioria das propostas que abordam uma ou ambas das preocupações usam um contexto ou métricas personalizadas que são pouco compartilhadas pelo resto da comunidade. Concretamente, algumas propostas usam um conjunto de dados simulado privado, ou usam apenas uma parte de um conjunto de dados públicos, ou até mesmo usam métricas diferentes para relatar o desempenho da proposta. Além disso, alguns pretendem prever ações e outros prevêem atividades. Assim, a tarefa de nomear um único método, global e melhor, revela-se uma tarefa enfadonha.

Consequentemente, o objetivo atual deste documento é desenvolver um estudo computacional sobre múltiplos métodos em relação à classificação de atividades. Assim, o trabalho atual detalha sobre o ambiente de aprendizagem (ou seja, conjunto de dados ou entrada) utilizado, as métricas aplicadas para permitir a comparação e os métodos são descritos, discutidos e comparados. De facto, os dados utilizados são os fornecidos pelo Massachusetts Institute of Technology, porque ele é anotado (cada sensor usado tem uma atividade associada) e é amplamente utilizado pela comunidade de investigação, apesar do seu curto período de tempo (apenas 2 semanas). Em relação às métricas, *F1*, *Precision* e *Recall* permitem analisar o nível de desempenho dos diferentes métodos.

Os métodos utilizados neste trabalho são *Naive Bayes*, *Random Forest*, *Reinforcement Learning* (Aprendizagem por Reforço) e *Recurrent Neural Networks* (Redes Neuronais Recorrentes). Todos eles fazem uma abordagem diferente, mas todos produzem o mesmo resultado (ou seja, prever a atividade). Assim, com a comparação possibilitada, as *Recurrent Neural Networks*, mais especificamente a *Gated Recurrent Unit*, revelaram ser a que tem maior poder de generalização.

Finalmente, o trabalho atual não fecha o trabalho na investigação no campo das casas inteligentes. Este documento conclui várias oportunidades para estudar diferentes maneiras de colaborar na área, como a criação de um conjunto de dados maior ou até mesmo um estudo para permitir o aprendizado autónoma pelos dispositivos.

Acknowledgements

By closing this work I thank all, who in most varied forms, contributed to its realisation.

First and foremost, I thank my supervisor, Professor Hugo Sereno Ferreira, for the long-lasting support, orientation and friendship.

I also take this opportunity to thank my family for all the incentive and unconditional support they have shown.

Finally, I thank my lab colleagues for all the support and interest they have shown on my work.

António Ramadas

"There is nothing permanent except change."

Heraclitus

Contents

List of Figures

LIST OF FIGURES

List of Tables

Abbreviations

ANN	Artificial Neural Network
GRU	Gated Recurrent Unit
IoT	Internet of Things
LSTM	Long Short-Term Memory
MIT	Massachusetts Institute of Technology
RFID	Radio-Frequency IDentification
WSN	Wireless Sensor Network
WSU	Washington State University

Chapter 1

Introduction

This chapter presents an overview of the main key points of this document. First, section 1.1 introduces the context where the work currently being proposed fits in. After, section 1.2 presents the motivation and goals trying to be achieved. To finalise, section 1.3 shows the structure of the report and the scope of each chapter.

1.1 Context

By 2030, the United Nations predicts that 201.8 million people will be over 80 years old and around 2050 this number will double [Nat15].

Simultaneously, Internet of Things is predicted to value from 3.9 up to 11.1 American trillion dollars in 2025 where smart homes are expected to be a market in the range of 200 and 350 billion dollars [MCB+15]. Therefore, empowering houses with intelligence is a market expected to be very valuable. However, [RT17] states that nowadays smart homes implementations are still very rare despite the confirmed growing interest.

Thus, the same authors state the omnipresence of smart objects, and respectively in consumers' households, within the next few years as the need for services for smart homes, based on Internet of Things, will be inevitable. This confident statement is supported by the classifications mentioned before and indicates that, regardless of the attractive market, the solutions offered are scarce. In fact, [RT17] even state that "there is a lack of a unifying platform that would transform (...) separate individual applications into a single infrastructure, a platform that can be further used for advanced data mining and knowledge extraction".

Consequently, the research community has been releasing several contributions to the field. In fact, the problem itself it is not so much about the quantity, but more about the way it has been reported. Numerous proposals are discussed on metrics that hardens the comparison with other approaches, because they most often use a peculiar input or, less frequent, metrics that are not the most common.

Regarding the amount of research in the field, most of the research has the scope of either one of the two goals: predict activities or actions. An activity is a set of actions and an action is a change of state on a device. So, for instance, washing hands is an activity that uses the sink and the light of the bathroom.

Actually, the two types of classification mentioned do not appear to be evenly distributed in the research community due to the great number of publications in smart-homes that ought to accurately predict actions. However, there is at least one drawback of predicting actions instead of activities. Some methods tend to assume that the devices are only binary when in the real world they may not be (e.g., granularity on the control of the brightness of the lamps). Contrarily, activities are binary. In other words, watching a movie will always be just that (the activity is being done or not).

Another important topic to discuss is how to handle wrong classifications. This issue is problematic because it is undesirable to have a system that makes faulty classifications and does not listen to the feedback of the user. This feedback can be just a binary response (i.e., was it accurate or not?) or the user explicitly indicates his current activity. In fact, methods that explicitly take into account the user feedback are discussed in section 2.2.4.

1.2 Motivation and Goals

In this document, the main goal of this work is to develop a computational study on existent methods under the same environment (same input and same test metrics).

Actually, all methods chosen to fit in the context of smart-homes, because their goal is to assist the user inside a home - Ambient Assisted Living - by predicting the next activity of the user. An activity is a set of actions and an action is an input from a smart device, for instance, turning on the lights is an action. Therefore, the methods detailed in this document are able to predict if a user is, for example, going to watch a movie and recommend to the user - this is the classification - to turn off the lights and turn on the television. It will not act on the devices and this responsibility is handled to a different module capable of abstracting the learning module as a blackbox. This decision allows to decrease the scope of this module and enable a more parallelised research by the community.

Moreover, the system is only fed with information from the sensors around the house and just outputs classifications. Again, the responsibility of the communication with the user is moved to another module which is not on the scope of the current proposal for the reasons mentioned before.

In fact, once such a system is assembled, a smart-house can increase the comfort of the user and save energy consumption as it already happens with devices like Nest [Nes18].

To conclude, this work addresses one identified main challenge in smart homes. It aims to contribute to the improvement of state of the art by comparing some of the current methods found in the literature.

1.3 Report Structure

Besides this brief introduction, this document contains 5 more chapters. Chapter 2 reviews the current approach to empower a house - giving smartness to it - and it also highlights state of the art methods whose application is the classification of user actions/activities in smart-homes. Chapter 3 refers to the research statement going to further details about the goals, evaluation technique and contributions. Chapter 4 analysis one of the most used datasets of smart-homes. Chapter 5 details each method and compares them. Finally, chapter 6 concludes the work.

Chapter 2

Literature Review

This chapter describes the state of the art techniques related to the scope of this proposal. First, it is given a novel interpretation on the current Internet of Things topic - section 2.1 - enlightening its open challenges and goals. Afterwards, the following section - section 2.2 - uses the same critical approach giving an overview of the current methods that address the issues identified before, but falling into the range of this proposal.

2.1 Internet of Things

The Internet of Things term can be traced back as early as 1999 when it was coined in the context of managing a supply chain [Ash09]. Since then, this topic has deserved multiple and extensive studies from multiple players creating a large number of opportunities tackled by novel applications to "improve the quality of our lives" [LHSS10]. Actually, [Tan10] gives a possible premise for this result. They state that one of the biggest breakthroughs is "making the physical world and information world together". The collection is made by sensors that gather information about the environment opening opportunities for it to be monitored, analysed and acted on. The value added by *IoT*, apart from being recognised by the research community [WF15], is embraced by companies which make use of it to increase their prospects [DBNA15]. When dealing with a user, a good implementation supports him by carrying specific tasks in an intelligent way [MSDC12]. The awareness about its potential led *IoT* to be listed as one of the six Disruptive Civil Technologies [FU08]. *IoT* is acknowledged to be a "technological revolution" [WB14] empowering the industry 4.0. The integration of devices, sensors or actuators, equipped with multiple capabilities provides the foundational infrastructure for a smarter planet [WB14].

Nonetheless, despite all the focus being given on this topic, it is difficult to name a single consensual definition mainly due to a partial lack of standardisation [WB14]. In section 2.1.1, it is given more emphasis on this issue.

The presence of smart devices in the environment provides smartness to where it is located. It is important to understand the key enabling technologies - section 2.1.2 - and how to ensure the security and privacy of the data being collected - section 2.1.3. Currently, there are techniques aiming to extract the knowledge - section 2.1.4 - in order to increase the powerfulness of the applications - 2.1.5. However, there still are some gaps requiring attention - section 2.1.6 - which deserve study and attempt to solve - section 2.1.7.

2.1.1 Definition

As previously disclosed, there is no standard definition of what Internet of Things stands for. Different sources report similar definitions [GBMP13]. However, they do have common points and usually differ about the scope of *IoT*. Therefore, a definition referred to as "commonly accepted" is [XHL14]:

> a dynamic global network infrastructure with self-configuring capabilities based on standard and interoperable communication protocols where physical and virtual "Things" have identities, physical attributes, and virtual personalities and use intelligent interfaces, and are seamlessly integrated into the information network

Another attempt to explain and define what is *IoT* is given by [MSDC12] in which they show what the word has been used to refer to. They also offer several perspectives being the most relevant for this document the system-level:

> From a system-level perspective, the Internet-of-Things can be looked at as a highly dynamic and radically distributed networked system, composed of a very large number of smart objects producing and consuming information (...) able to sense physical phenomena and translate them into a stream of information data (...), as well through the presence of devices able to trigger actions having an impact on the physical realm

The "things" referred to are often called smart things given their intelligence to communicate, interact and be identified [MSDC12]. Another good insight is given at conceptual level [HFH15]:

> At a conceptual level, *IoT* refers to the interconnectivity among our everyday devices, along with device autonomy, sensing capability, and contextual awareness

To finalise, despite the lack of standard definition, it has been proposed standardisation classification schemes [WAD15].

2.1.2 Key enabling technologies

The wide and successful scope of Internet of Things is possible due to the major role of technologies that enabled the interoperability and connectivity of smart things - section 2.1.2.1 - as the architecture that enables to monitor and process the data extracted - section 2.1.2.2.

2.1.2.1 *RFID* and *WSN*

RFID stands for Radio-Frequency IDentification and it is an important aspect that allows to "monitor objects in real-time, without the need of being in line-of-sight (...). Therefore, they can be used in an incredibly wide range of application scenarios, spanning from logistics to e-health and security" [LXZ15]. The same authors add that *IoT* is, on its base, *RFID* systems full of *RFID* tags - ensuring interoperability - where there is a mapping of the object to the tag - enhancing identifiability. In the presence of a reader, the tags can be read and its data communicated allowing queries by a system.

In order to connect the tags, it is often used Wireless Sensor Networks that empower multiple types of monitoring [WB14]. The importance of such networks is considered critical to the development of *IoT* [RT17].

2.1.2.2 Cloud and Fog computing

The goal of Internet of Things is to empower *things* by connecting and make use of them. Hence, there are two possibilities to do such computations.

Cloud computing stands for "high reliability, scalability and autonomy to provide ubiquitous access, dynamic resource discovery and composability" [GBMP13]. Given this support, the requirement for "new methods and algorithms based on machine learning techniques, time series processing and advanced analytics" [RT17] are now possible to be employed. However, the stable growth of Internet of Things led to new challenges which cannot be properly addressed by Cloud computing - "stringent latency requirements, network bandwidth constraints, resource-constrained devices, cyber-physical systems, uninterrupted services with intermittent connectivity to the Cloud and new security challenges" [CZ16].

Due to Cloud computing being constrained to solve the new issues, Fog computing represents an enhancement by slightly moving the responsibility of what endpoint processes the data. The Cloud architecture is about the devices sending their raw data to another endpoint - usually more powerful servers -, but the Fog architecture allows the devices to pre-process the collected data and still sending to another endpoint. Because of this change, the volume of information can be greatly reduced, and more secure connections can be achieved [RT17, CZ16]. Lastly, the Fog concept was greatly possible to be realised due to the upgrade of the devices throughout the years because they are now powerful and efficient enough to conduct such computations.

2.1.3 Security and Privacy

Given an explanation of Internet of Things - section 2.1.1 -, one realizes the important role of having secure methods to ensure the privacy of the environment. If the devices may be accessed anywhere and are constantly active [HFH15], then malicious actors may attempt to disrupt the system [GBMP13]. Actually, there are multiple types of attack possible [HFH15] leading the National Intelligence Council of the United States of America to describe *IoT* "to the extent that everyday objects become information security risks" and "distribute those risks far more widely

than the Internet has to date" [FU08]. Consequently, multiple researches provide a coverage of the wide scope of attacks possible [RT17, MSDC12, HFH15, GMKS17] and how to shield against them [HFH15]. By tackling security, the data collected remains confidential, but its use may be positive - for instance, advertisement - or negative - for instance, blackmail [GBMP13]. Also, the extent to which the devices are mixed with the environment may lead to the collection of information without consent.

2.1.4 Data Mining

In Internet of Things, the data is characterised by its large volume, wide variety - from a broad range type of sensors - and fast velocity [RT17]. Consequently, its interpretation and reasoning about are challenging [MSDC12, GBMP13]. Also, the characteristics mentioned happen to be the same three dimensions of Big Data. In order to help in the decision making in the face of this issue, Artificial Intelligence represents a valid approach. It supports the creation of cognitive systems that "learn, adapt, and ultimately hypothesise and suggest answers" [WB14] that automates the decision making [GBMP13]. Actually, [RT17] gives an example of AI in the context of smart homes:

> All these tools need advanced algorithms that use much more parameters than those obtained by the home devices. They should perform the complex tasks of mining and knowledge extraction from the available data in the cloud in order to create consumer and household profiles, or in simpler words, the available smart home data should lead to the creation of personalised recommendations for all users

2.1.5 Applications

Internet of Things has a broad range of applications in many sectors, such as security and surveillance, health-care, environmental monitoring and workplace and home support [MSDC12, LXZ15, GBMP13]. Actually, there are more areas where *IoT* has a big impact like in the industry [WB14], but it won't be covered here, because it is out of scope to the goal established in this document.

One example of an application is the adoption of the cities by this paradigm which can be exploited to increase the transparency and promote more public-targeted civic actions by raising awareness and engagement. Consequently, it creates a win-win situation [MSDC12]. It also stimulates the creation of new services, creates an economic advantage and reduces operational costs.

Besides smart cities, there is also the smart home concept. Actually, the orchestration of smart devices by a hub can promote assistance to the life of the occupants. By targeting specific actions, the users can be helped in their daily activities - for instance, turn on the lights of the kitchen and start to make coffee - by "reasoning techniques for inferring activities of users and devising appropriate response strategies from the embedded devices" [MSDC12]. In fact, some research proposes a framework for smart homes. In [RT17], the authors lead to the conclusion that this concept is not a matter of if it is going to happen, but when it is going to happen. Their proposal

makes use of Wireless Sensor Networks and a hub - either Cloud or Fog computing - and mounts a system which can be controlled remotely.

To conclude, the example of applications is very extensive once one realises how many objects can already communicate and be operated autonomously. Therefore, it was only mentioned with particular focus two concepts, but the extension goes beyond scope.

2.1.6 Gaps

Internet of Things is a relatively new topic and gaps still exist. Besides the issue about standardisation - referred in section 2.1.1 - there are more open issues as enumerated by [LXZ15]. Some of them are mobility support, authentication, data integrity, privacy and digital forgetting. [WB14] also extends about this topic and [MMST16] even conducted a survey on the national *IoT* Finnish program and propose some features to close the existing gaps. In fact, the community is addressing the issues it has identified and described, and one example is Cloud and Fog computing - section 2.1.2.2.

2.1.7 Research

So far, Internet of Things provides multiple advantages, but implementations of interconnected objects are still rare. Also, there are challenges related to reliability, privacy and security - section 2.1.6. However, there is research in place aiming to mind these gaps. Actually, [WB14] identifies several trends: "Integrating Social Networking With *IoT* Solutions, Developing Green *IoT* Technologies, Developing Context-Aware *IoT* Middleware Solutions, Employing Artificial Intelligence Techniques to Create Intelligent Things or Smart Objects, Combining *IoT* and Cloud Computing". There is additional research with the goal to improve the quality of current implementations [RDD+17]. To conclude, [WAD15] went even further and analysed 127 documents and classified each one to perceive the current state of the art. They even added that "by bringing existing technologies together in a novel way, the *IoT* has the potential to reshape the world".

2.2 Artificial Intelligence

In the context of Internet of Things, one of its applications is smart-homes. They are filled with smart devices - sensors or actuators - and it is possible to create a system that operates them providing Assisted Ambient Living to the house occupants. Such system, often named a hub, is composed of algorithms and methods that reason about the data - section 2.1.4. Currently, "these implementations are still very rare" and "the need for *IoT*-based services for smart home will be inevitable" [RT17]. Basically, "there is a lack of a unifying platform that would transform (...) separate individual applications into a single infrastructure, a platform that can be further used for advanced data mining and knowledge extraction" [RT17].

Nonetheless, current techniques do work despite being yet not suitable for real-world use without constraints. Before naming the challenges current research faces, it is important to clarify

some of its nomenclature. In the context of a system that acts on a house, this is the module that provides smartness to the house, an action is an input from a device - for instance, to turn on the lights - and a set of actions is an activity - for instance, the activity of watching a movie involves turning off the lights and turn on the television. It is the relation between actions that enables the inference of activities. However, these relations can be of multiple kinds - section 2.2.2. The hub by being able to predict the user activities can assist him by automating some tasks. The level of automation depends on the goal. Should it be predicted actions individually (more complicated) or an activity (easier)? Should the system just make recommendations or act? Actually, this last question goes a bit beyond scope of Artificial Intelligence when implemented, because the communications with the devices are usually not a responsibility of this module. However, it is the probability of the prediction that allows modules to actuate with confidence.

After identifying the goal of the system, there are some challenges that for a realistic usage need to be addressed. A house can have multiple occupants, so how should the actions of different users be addressed? The layout of a house throughout the years is likely to suffer some changes, so how should the system handle new sensors/layouts not seen before? When given feedback from the user - for instance, the system wrongly predicted to turn on the lights and the user dismissed it -, does he refer to the action or to the activity - continuing on the previous example, is he referring just to the lights or he does not want to watch a movie? Lastly, the behaviour of the users is likely to differ throughout the years, so how can the system be aware of new activities?

Actually, the challenges identified are never faced altogether by current proposals, but there is progress towards it. To give a review on these implementations, the rest of the sections detail only the algorithms and methods used to automate home chores. Starting by how the patterns are modelled and matched - section 2.2.1 - and advancing to how it is defined the relation between actions - section 2.2.2. Having the relations between actions, it is possible to cluster them - section 2.2.3 and take into account the direct feedback of the user - section 2.2.4. In order for the system to adapt better from the training environment to the testing one, it is possible to transfer some knowledge between environments - section 2.2.5. If not successful enough, there are algorithm enhancements - section 2.2.6 - and hybrid approaches - section 2.2.7. Finally, it is important to be aware of the learning environments which aim to replicate the real world but produce datasets which are helpful to validate and test the techniques - section 2.2.8.

2.2.1 Patterns

As already said, actions may have some sort of relationship with each other and it is the connection of actions that comprise an activity. In order to recognise a pattern (an activity), there are two possible ways to do so. Either by pattern matching or by pattern recognition [MI13]. Figure 2.1 is a way to represent such connections. The match of a pattern is "the act of checking some sequence of tokens for the presence of the constituents of some pattern" [MI13]. This represents a partial match being possible to derive a probability of which is the next state. In opposition to pattern matching is pattern recognition which checks the exact sequence. This hard constraint makes pattern matching to be the most common implementation due to its flexible approach.

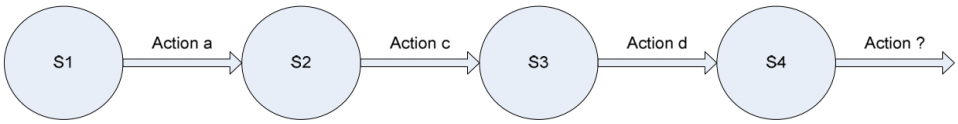

Figure 2.1: Chain of actions [MI13]

Actually, it is possible to name three algorithms that make use of a similar structure and their input can be directly from the dataset (no need to build relations between actions).

First, Active LeZi is an enhancement of the compression algorithm LZ78 [WRS+17] and it is adapted to smart environments. The representation inside the algorithm approximates a Markov model and the improvement from the previous algorithm is the inclusion of a sliding window which enhances the speed of the algorithm. The output is the next symbol with the highest prediction probability - pattern matching. [CHGY03] applied it to smart home environments and achieved very good results, but the algorithm shows some difficulties when the data is not clean and better represents real-world usage.

Therefore, it has been proposed an enhanced over Active LeZi which improves distinction of similar activities. [FR12] named it TALZ - Time-varyingLeZi - and it has improved recognition "based on the observation that human activity tends to occur on a periodic basis" [WRS+17] and also distinct activities that may be similar, but happen in different periods [WRS+17].

Finally, there is an algorithm which makes Sequence Prediction via Enhanced Episode Discovery - SPEED - and "the main purpose (...) is to learn from the periodic tendencies of the user of a smart home and attempt to make appropriate decisions based upon the data it has gathered" [WRS+17]. It discards unnecessary information and it is able to make predictions on sequences it has never seen before [WC07]. However, SPEED makes a trade-off between more accuracy by being more resource intensive. Additionally, it fails to predict actions that happen at the same time, because it does not take into account the timing of the events [WRS+17].

2.2.2 Association Algorithms

The data produced by a learning environment is used by the algorithms to learn and make predictions. The information usually comes with the timestamp in which the state of the device was changed which does not represent, in an explicit way, relations between actions. This is an important step given that it may be considered one of the first layers that influence the rest of the system. The algorithms discussed in this section may either be used to make a prediction about which next action will be or its output may be the input of another layer like Unsupervised Learning techniques - section 2.2.3.

One way to easily understand the type of relations existent is to start by Allen's Logic [AF94]. Figure 2.2 shows some of these rules and the conditions necessary to indicate the type of relation between actions. In fact, originally there were 13 rules [AF94]: before, after, meets, meet-by, overlaps, overlapped-by, starts, started-by, finishes, finished-by, during, contains, and equals. However,

when applying these rules there is a problem with ambiguity, because it is possible to have two events that can be correctly described by more than one temporal relation. [WC07] addressed this challenge by creating boundary conditions. Finally, the good performance led to multiple proposals to use this technique [BFSL+15, JC07, WC07].

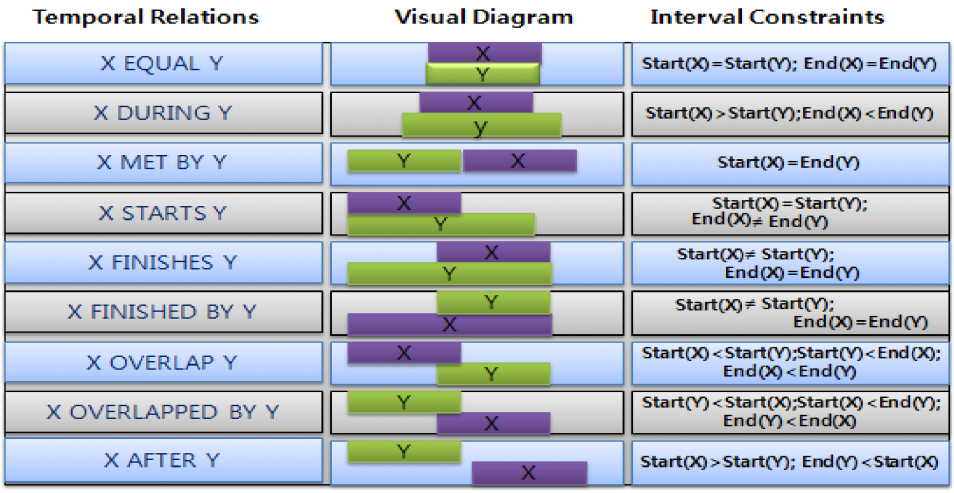

Figure 2.2: Example of some of the existing rules in Allen's Logic[BY15]

On a similar logic, the "Apriori Algorithm is a classic data mining algorithm for mining association rules" [WRS+17]. The first step is to process the input into subsets and identify the ones most common. Afterwards, some of the actions are pruned. The output is association rules in which one subset implies another [WRS+17]. However, this classical approach is very slow - requires multiple scans to the database -, so [ZQ15] proposed an enhancement in which they solve this problem and, also they noticed that adding an action to a non-frequent subset will not turn it frequent. The result was that their enhancement outperformed the classical approach.

Finally, the two algorithms referred to represent the biggest focus by the community on this topic. Nonetheless, there are more algorithms that achieve the same result using different approaches but have not been tested in the context of smart homes. They all find the most frequent items - similarly to the Apriori Algorithm. The Frequent Pattern Growth - FP-Growth - uses an extended prefix-tree structure to store compressed and crucial information about frequent patterns [BY15]. The Recursive Elimination Method - Relim - which finds the most frequent items by recursively eliminating the leading item [Bor05]. Lastly, Eclat is yet another algorithm for fast discovery of association rules [ZPOL97].

2.2.3 Unsupervised Learning

This module - Unsupervised Learning - clusters the input by grouping the actions into activities. Depending on the algorithm used, the input can either be from the Association Algorithms - section 2.2.2 - or directly from the database.

Starting with K-Means, this algorithm splits the data into K clusters and they are expected to be from similar size. A major drawback is its susceptibility to outliers. These characteristics are not very well suitable for smart homes environments. Hence, K-Patterns [SS13] was proposed following a similar approach. Efficiently handles very large environments [BY15]. In fact, the same authors state that it overcomes the problem of noisy data and it is able "to detect discontinuous and interleaved activity pattern of users". Despite the good performance, both these algorithms need the number of activities - K - to be present on the dataset to be settled a priori. So, introducing a number - K - smaller or higher than the correct number of activities lead to a partition of data which is not the best and may lead to incorrect clusterings. Additionally, in real scenarios, it is not known the number of activities performance in each house, so this constraint is of great importance [WRS+17].

On another hand, Flocking [Rey87] does not require an initial number of clusters to be set a priori [LBB+13]. In fact, this algorithm replicates the behaviour of several species in nature [Rey87]. The characteristics for clustering the actions into activities is alignment, separation and cohesion [WRS+17]. In order to further improve the performance of this algorithm, [LBB+13] added two more characteristics: similarity and dissimilarity. They noticed that the performance was superior to other clustering algorithms, but at the cost of more iterations. So, this algorithm is not suitable when one wants small learning periods.

Another clustering algorithm is Expectation-Maximization which has the particularity to either cluster given its initial number or to estimate it, but when estimating, the performance decreases [LBB+13].

Finally, some authors attempt to build a recommender system - still an unsupervised algorithm, but outside the scope of clustering because the data is not labelled - which continuously learns from the user habits updating when there is user feedback and it directly outputs recommendations to the user [Ras14]. In fact, their performance was good, but at the cost of an increased training time.

2.2.4 Reinforcement Learning

The proposals, on the context of smart homes, that fall into the category of Reinforcement Learning act autonomously without the need for other algorithms. Therefore, its input may be directly from the dataset and the output may go straight to the environment.

First, Planning Q-Learning Algorithm [HA15] is the application of the classical algorithm Q-Learning to generated plans by the Markov Decision Process. The authors state that their work "differs by its online and continuous learning as the system learns the change of user's behaviour by allowing him to make feedback actions directly on the devices manually" [HA15]. Their description of the algorithm is fairly simple, but one of the key points is their reward system because it is known before the execution of each state, because the reward is based on user satisfaction - the feedback. Lastly, the performance of this algorithm is not known given that the authors didn't release the performance of their experiments.

Also using Q-Learning is the framework Nash H-Learning [RRD06]. The scope of this technique is very wide because it tries to maximise the comfort, save energy and predict the location

of multiple users. Nonetheless, the performance is very high. The introduction of Nash Equilibrium aims to fix the challenge of Q-Learning in multi-agent systems by introducing the idea of a balance between all preferences from users of the system [WRS+17]. Lastly, this framework is mainly used to predict the location of the occupants [WRS+17].

Finally, Reaz et al. combined pattern matching - section 2.2.1 - with Q-Learning by proposing a formula that combined both [MI13]. The pattern matching component ensures that the sequence of events is preserved and used to empower a more robust formula than a direct implementation of a Reinforcement Learning algorithm. The other component, Q-Learning takes into account the user feedback and builds a matrix that preserves the knowledge acquired so far. The benchmark is reported on a simulated dataset and it is not detailed its complexity, but the accuracy hits 87% and passed in front of the algorithms it is compared with, which is a nice indicator of its quality.

2.2.5 Transfer Learning

The algorithms explained in sections 2.2.2, 2.2.3 and 2.2.4 require some initial data to start making predictions or acting on the environment. However, recent types of learning - Transfer Learning - address this challenge. This section does not extend deeply into this topic given that goal when applying this type of learning is not incorporated into the goals of this proposal.

The problem in smart homes is the high costs of building smart environments collecting data from scratch [CLH17]. However, recent proposals show that is possible to "transfer as much learned knowledge as possible from an existing environment to the new target environment" [CLH17]. Therefore, it is possible to train the system on a dataset which replicates a real-world scenario and then provide that knowledge to the deployed system, so it initially has some intelligence and it is able to make accurate predictions. Additionally, this type of learning relaxes the constrained found in the other types - sections 2.2.2, 2.2.3 and 2.2.4 -, because the learning environments - the one to train the system and the other where the system is deployed - are not required to be highly similar in distribution [CLH17]. This contrasts with the algorithms in sections 2.2.2, 2.2.3 and 2.2.4 because most of the researchers assume that the learning environment is the same for training and deployment.

The results of applying this type of learning lead to mixing results, because either the system almost paired performance with one without transfer learning [CLH17] or surpassed it [CCJ17].

2.2.6 Algorithm Enhancements

The algorithms in this section only have the goal to help other prediction algorithms - sections 2.2.2, 2.2.3, 2.2.4 and 2.2.5 - to increase their performance when paired with them [WRS+17]. Therefore, they do not work alone on a system.

Starting with Episode Discovery [ALS12], it "improves prediction algorithms by filtering the input stream before it is provided to the prediction algorithm as training data" [WRS+17]. The filter component works by splitting the input into maximal episodes. In one episode it is only present the actions that occur inside a specified time period. Afterwards, it is computed the episodes and

extracted to itemsets which are pruned, and the remaining episodes are potential candidates to be chosen by a greedy approach.

Next is Temporal Relations that can be simply explained by Allen's Temporal Logic - section 2.2.2 -, but the output instead of being to the user leads to the input of a learning algorithm.

Finally, there is the Movement Patterns algorithm that "can be regarded as an intermediate data level, based on which an unsupervised learning [- section 2.2.3 -] process about patterns of the occupant's higher level routines could be developed" [ZFYF17]. However, this algorithm makes use of Passive Infrared Sensors - PIR - to be able to provide the location awareness, but this type of sensors is not so common throughout smart home environments. Additionally, the authors, despite recognising the value of such additional knowledge, state that this algorithm requires more experimentation before being fully integrated.

2.2.7 Hybrid approaches

In sections 2.2.2, 2.2.3, 2.2.4 and 2.2.5 the techniques referred act alone and are that way explained. Nonetheless, some authors have been mixing different algorithms in order to study the impact on performance when doing it. However, such implementations are not common, yet. Additionally, they all have in common Artificial Neural Networks - *ANN*.

Actually, [BY15] connected an *ANN* to the output of K-Pattern clustering algorithm enhanced with Allen's Temporal Logic. Their performance achieved 88% accuracy on the Washington State University(*WSU*) CASAS dataset - section 2.2.8.1 -, but the authors reduced the size of the dataset to have only 5 activities instead of the 10 existents. This choice was probably made based on the fact of the training time that would be required to analyse such large volume of data.

Also using a hybrid approach is [CKO13] mixing Deep *ANN*- more than one hidden layer - either with Support Vector Machine - unsupervised clustering algorithm - or with another *ANN*. However, the maximum accuracy achieved was 51.8% on the Massachusetts Institute of Technology (*MIT*) dataset - section 2.2.8.1. The metric hardens the comparison with other algorithms given that the others used the new metric they proposed. Their metric - Rising Edge Accuracy - attempts to solve the problem that "an algorithm can report a high accuracy value while it makes poor predictions about when a sensor will be activated or deactivated" and the "second reason is that many sensors are not active for the majority of time, hence, the prediction about deactivation has a little value" [CKO13].

Finally, the few hybrid implementations previews that the performance can match or even surpassed the performance of standalone algorithms, but it is still required further experimentation. One flaw of using *ANN* is that they work better when the algorithm has a lot of features - variables - and it has an amount sufficient of time to learn or keep learning. However, this type of constraints does not completely suit smart home environments. Therefore, a novel implementation would be the mix of state of the art algorithms - sections 2.2.3 and 2.2.4.

2.2.8 Learning Environments

All the algorithms in section 2.2 require data to be able to learn or process information with the goal to help the user activities inside a smart house. In order to do it, there are learning environments which can either be real - section 2.2.8.1 - or simulated - section 2.2.8.2. [SNJ15] These environments replicate real-world scenarios and collect their data storing it into datasets which often are publicly available for the public to develop systems without the need to each time develop their own learning environment. Additionally, having common sources of data with common metrics eases the comparison among the proposals. Currently, there is no gold standard in smart home datasets, but the main source has been datasets from real environments - section 2.2.8.1.

2.2.8.1 Real

Collecting data from real environments - from the real world - has the drawback to having high costs due to the infrastructure necessary, but the data collected can easily capture natural user interactions with the environment. Due to this disadvantage, some laboratories have created this type of environments and released the datasets to the public. There are multiple datasets currently available, but most of the proposals take only into account three of them.

The *MIT* research group set up a website[1] where they detail the environment. The two datasets are from 2 subjects - one dataset per subject - where they live alone for 2 weeks in a house full of 77 or 84 sensors. They are spread across the house and on multiple utensils such as drawers, refrigerators and containers. Figure 2.3 illustrates some of the locations of the sensors. Also, the dataset is labelled with 16 activities such as preparing dinner, washing dishes or toileting.

Another popular learning environment is from the Washington State University's Centre for Advanced Studies in Adaptive Systems[2] [CCTK13]. They produced up to 26 datasets with a vast majority being annotated, but they do not have as many sensors as the *MIT*. The time interval for some of the datasets is 1 year which represents a big difference from 2 weeks from the previously discussed environment. Some environments had pets and others had multiple people inside a house. Lastly, the number of activities varies from each dataset, but often are less than *MIT*. So, when comparing this environment with the one from *MIT* it is evident a trade-off between the quality of the dataset and its time period.

Finally, LIARA[3] - *Laboratoire d'Intelligence Ambiante pour la Reconnaissance d'Activités* - is the least known of the three. The dataset produced compresses 14 activities performed by 20 different students with some errors introduced on purpose to provide noise to the data. This environment has the drawback of not capturing the continuous living, but sporadic activities.

[1] http://courses.media.mit.edu/2004fall/mas622j/04.projects/home/
[2] http://casas.wsu.edu
[3] http://liara.uqac.ca/index.htm

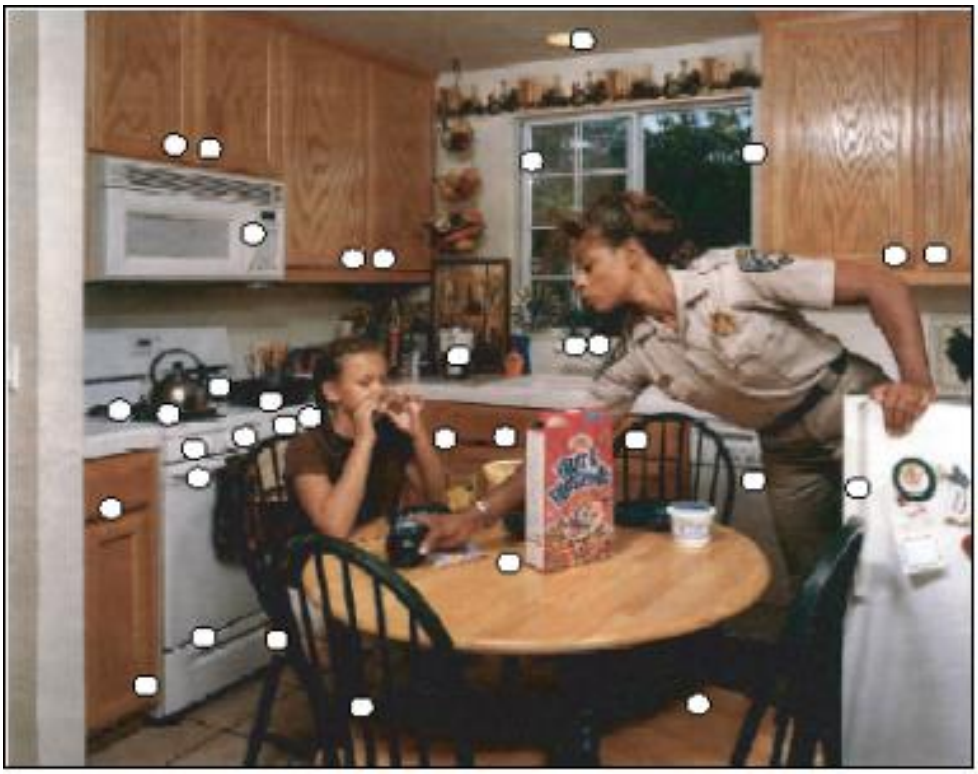

Figure 2.3: Example of the location of the sensors in the Massachusetts Institute of Technology learning environment [TIL04]

2.2.8.2 Simulated

Building real environments to study represents high costs and a big amount of time. In order to overcome these issues, some authors have created tools to simulate datasets being the major challenge the ability to accurately replicate human interactions with the environment.

Actually, there are two types of simulations [SNJ15]. The first is model-based, a generator based on an activity model - this is the key point to achieve the most possible real-like interactions - to generate vast datasets over extended periods of time. The second is an interactive approach much like a computer game in which a user plays an avatar in a simulated scenario. Currently, there are 12 simulated datasets [SNJ15], but none is often referred in the literature which limits the advantage of using them instead of real datasets which are more popular and do not have the problem of capturing the real world.

2.3 Conclusions

The goal of automating some actions inside a house is currently being tried to be achieved. In section 2.1, it is shown the progress of building a smart home and what is the status of Internet of Things. It is possible to conclude that current technologies and architectures already allow giving intelligence to smart homes by deploying smart devices. With the capture of this smart devices, it is possible to extract the data necessary for smart algorithms to help the user with his

actions/activities. The topic still has challenges to address as identified in section 2.1.7, but *IoT* can be considered in its mature state.

Regarding Artificial Intelligence in the context of smart homes, the level of automation achieved so far does not allow to deploy the solutions to the consumer market as identified in section 2.2. One of the biggest challenges currently is to compare results given that the proposals usually limit the size of the dataset, or do not release the results, or use different metrics - F1 score, accuracy, Rising Edge Accuracy, recall and precision - or, but less common, use different datasets. This customisation represents a barrier when pointing out the algorithm which achieved the best performance. Therefore, what helps the most when deciding which algorithm to use is its constraints. It is also worth noting that most of the investigation has been on Association Algorithms - section 2.2.2 - and Unsupervised Learning - section 2.2.3. Reinforcement learning - section 2.2.4 - has fewer proposals, but the most promising types of proposals seem to be Transfer Learning - section 2.2.5 - and hybrid approaches - section 2.2.7. In fact, hybrid approaches have the peculiarity to allow the system to address most of the challenges identified in section 2.2. Therefore, being, probably, more suitable for a real-world application. However, current proposals focus on *ANN* and do not attempt to mix different algorithms in types of learning where the investigation thrives more.

Chapter 3

Research Statement

This chapter, as the title points out, is the research statement of the work done. It starts with an introduction of the current challenges in the Smart Homes research - section 3.1 -, then it contextualises what the current goal is - section 3.2 -, how the methods are evaluated - section 3.3 -, and the outcome - section 3.4. Finally, the chapter ends with a short summary - section 3.5.

3.1 Introduction

classification in Smart-Homes is currently a field with an active community and research. The goal has been the same for several years, i.e. create a system capable of easing the living of a user inside a house. This objective has been unfolded into multiple challenges that when successfully tackled together create the system intended. First and foremost, the system should be autonomous or with enough autonomy to drive decisions for the user. On its most simplicity, the main focus is on classification and to achieve a good performance doing it.

Therefore, the focus of the research community is to tackle five main challenges:

1. Multiple people inside a house (section 2.2.8)

   - Is it possible to distinguish each person actions?

2. Dynamic environments (section 2.2.5)

   - How should the system handle new sensors/layouts not seen before?

3. Interpretation of the user feedback (section 2.2.4)

   - Is the feedback is for an action or a set of actions?

4. Detect and identify new activities (section 2.2.3)

   - How can the system be aware of new activities?

5. Improve state of the art results (section 2.2)

    - How to improve the performance of current state of the art methods?

A system that successfully addresses these challenges, besides having good classifications, should easily expand to real-world integrations. However, few proposals address all of them. In fact, the focus currently has been more narrowed to be on just part of the challenges in order to develop more well-performant solutions.

3.2 Main Goal

In section 3.1, it has been identified 5 main challenges that have been addressed but not yet successfully solved. The current challenge being addressed is to improve state of the art results.

In fact, the current state of the art, as stated in chapter 2, most frequently is proposed, and currently benchmarked, under different metrics and different input (i.e., the dataset). This certainly hardens the task of a researcher in order to understand what the best methods currently are, because there is no easy comparison. Also, most research implementations are not shared publicly and one to compare the proposals sees himself in front of multiple proposals that he has to implement just to compare with its own proposal. Given the great amount of work this represents, most proposal statements lack this important detail. To be fair, they do compare with the rest, but most only do on the part they operate (e.g., *The previous method did not take into account the period of the day*). Actually, just by reading the research published throughout the years is possible to see a progress in the quality of the work performed. Hence, a comparative study allows to get a better perception of the performance between multiple methods. Consequently, one possible outcome, depending on the acceptance by the rest of the community, is the standardisation of the investigation (same metrics and, possibly, dataset).

Additionally, few proposals have been published on the classification of activities, due to a bigger focus on actions, in smart-homes. One difficulty of predicting actions is that they can often be more stochastic than activities (e.g., I may not always wash my hands when I go to the bathroom). Additionally, some methods tend to assume that the devices are only binary when in the real world they may not be. Also, as mentioned in chapter 2, the industry is still expanding and new devices with new capabilities are being introduced that are more difficult to understand by a machine (e.g., turning on and off the lights is evolving from a simple switch to a more granular device capable of configuring the brightness of the lamps). Thus, developing systems with this assumption may be dangerous because they may be deprecated. On the other hand, activities do not have this level of granularity. Watching a movie will always be just that as it is binary (the activity is being done or not).

To summarise, the current goal is to perform a computational study on some of the current methods.

3.3 Evaluation Strategy

Starting with the input, the dataset used is from Massachusetts Institute of Technology, because it is labelled (each sensor used has its activity associated) and it is widely used by the research community despite its short length (just 2-weeks). As such, this choice eases the comparison with methods from other proposals - chaper 5.

Another important topic to discuss is how to measure the quality of the results. Hence, the choice of the evaluation strategy reveals itself to be somewhat complex.

Validation is an important step to ensure that the algorithms have a great power of generalisation and not just have memoized. This technique aims to split the dataset into 2. One part is used for training, thus it is named training data. The other part is used for testing and it is not used for training. This is an important step to ensure that the results benchmarked by the metrics are not biased. Therefore, this part is used for validation, hence the naming validation data.

Actually, the selection of both of these parts is of extreme importance for the research community because a naive use allows one to artificially leverage the quality of the results. For this study, it has been used a 10-fold approach. This means that each algorithm was tested 10 times and the results showed are the metrics averaged. Each time, the dataset was shuffled.

Moreover, the selection of training and validation sets used a 77%-33% ratio, respectively. Each time the algorithm was executed this process has been repeated (i.e., shuffling the days and split them).

Another important topic is the metrics used. In total, the results are reported under 3 formulas, but before addressing them table 3.1 presents a classification showing the meaning of several keywords used in the community. A correct understanding of these concepts is crucial to understand Precision, Recall and F1. It is important to mention that these metrics appear in a great number of papers and this increases the odd to compare the methods with other proposals.

Precision indicates the proportion of predicted values that are actually relevant, and its formula is:

$$Precision = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

Regarding Recall, it can be considered as the ability to find all the correct values. Its formula is:

$$Recall = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

Thirdly, F1 is a formula that creates a unique result relating recall and precision and is more accurate of the algorithm/model performance. Its formula is:

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Finally, the current problem is a classification task as the goal is to predict what activity is being performed. In total, there are 20 activities in *MIT* 1 and 22 in *MIT* 2. More information about the datasets is in chapter 4 and the adaptation of the metrics to the dataset is in section 5.1.

Table 3.1: Definition of True Positive, True Negative, False Positive and False Negative for classification tasks

		Actual	
		Positive	**Negative**
Predicted	**Positive**	True Positive	False Positive
	Negative	False Negative	True Negative

3.4 Main Contributions

The current work presented aims to achieve 2 outcomes while aiming to contribute with a computational study.

The first is the analysis made on the *MIT* dataset - section 2.2.8. The goal is to introduce the dataset to the reader and prepare it for the algorithms by discarding unnecessary information and highlighting the most pertinent one. In fact, this has successfully been done in chapter 4 with a statistical approach. The choice of the dataset from *MIT* is a consequence of being the one most used, representative of the real world and with enough quality to be able to be used by different methods.

The second is the implementation of multiple algorithms which have yielded positive results but are difficult to compare with each other because they have been published under different metrics and input. So, after understanding the proposal in which they are detailed, they were implemented, compared (chapter 5) and published online[1]. Moreover, the methods used are:

- Naive Bayes (section 5.2.1)

  - Assumes that the probability of an event is not independent of other events

- Random Forest (section 5.2.2)

  - Ensemble learning algorithm with multiple trees where the overall output takes into account the output yielded by each tree which is based on the input it has been provided

- Reinforcement Learning (section 5.2.3)

  - Q-Learning algorithm applied to smart homes

- Recurrent Neural Network (section 5.2.4)

  - Applicability of Long Short-Term Memory and Gated Recurrent Unit to smart-homes

Finally, current work aims to contribute by performing a comparative evaluation of multiple machine learning algorithms to detect user activities.

3.5 Conclusions

In this chapter, it has been introduced the current issues in the field (i.e., classification in smart homes) - section 3.1 -, and what issue this document addresses as also the problems associated

[1] https://github.com/antonio-ramadas/upis

with it - section 3.2. The metrics used to benchmark are also detailed - section 3.3 - and it is described the contributions to the field - section 3.4. Chapters 4 and 5 provide an in-depth insight and discussion of the topics introduced in this chapter.

Chapter 4

Dataset analysis

As mentioned before, multiple algorithms have been tested, but, to provide a common baseline for all of them, all use the same dataset from the Massachusetts Institute of Technology.

Diving into details, the data generated from both subjects share some similarities. Both last 15 days (2-week period) with about 70 sensors in each house. The data collection was not performed in parallel, but sequentially which means that one subject was observed and a week after was the other's turn. To avoid a possible wrong assumption, the houses where the study was performed have different architectures.

Each dataset has been processed to extract relevant information disperse across 3 different files into a single one. The information scrapped aims to preserve all the relevant features and provide a simple way to reason about. A glimpse of one of the datasets can be found in table 4.1. It is worth noting that actions (a row) can over-lapse. For instance, according to the table, sensor 68 is used within the lifespan of sensor 100. In fact, this sort of relations has been expressed before in section 2.2.2. More concretely, in figure 2.2. It is also worth clarifying that a sensor may be used for multiple activities. Lastly, all the time intervals are integers meaning that there is no decimal precision.

Table 4.1: First rows of *MIT* 1 dataset

ROW_NUMBER	SENSOR_ID	ACTIVITY	START	END
0	100	Bathing	2003-04-01 20:51:52	2003-04-01 21:05:20
1	68	Bathing	2003-04-01 20:51:58	2003-04-01 20:52:05
2	81	Bathing	2003-04-01 20:53:36	2003-04-01 20:53:43
3	101	Bathing	2003-04-01 20:53:49	2003-04-01 21:21:43
4	93	Bathing	2003-04-01 20:53:52	2003-04-01 20:58:42

So, these details provide no info or metrics about the data collected. Therefore, sections 4.1 and 4.2 show a more scientific approach.

4.1 *MIT* 1

Logically, actions may have different timespans, because using the lights of the bathroom will certainly last more than taking a shower. However, all algorithms used do not have this sort of information beforehand. So, it is useful to develop an insight into the 2-week period so we can provide better inputs and parameters to the algorithms we will later use (section 5.2).

If actions have different lengths, then it is useful to analyse its distribution. In fact, by looking at the dataset, it is possible to conclude that there are 545 different timespans. Also, by looking to figure 4.1 it is possible to perceive that shorter actions are the most common. Additionally, there are multiple actions whose duration only occurs once. This difference is shown when comparing both plots.

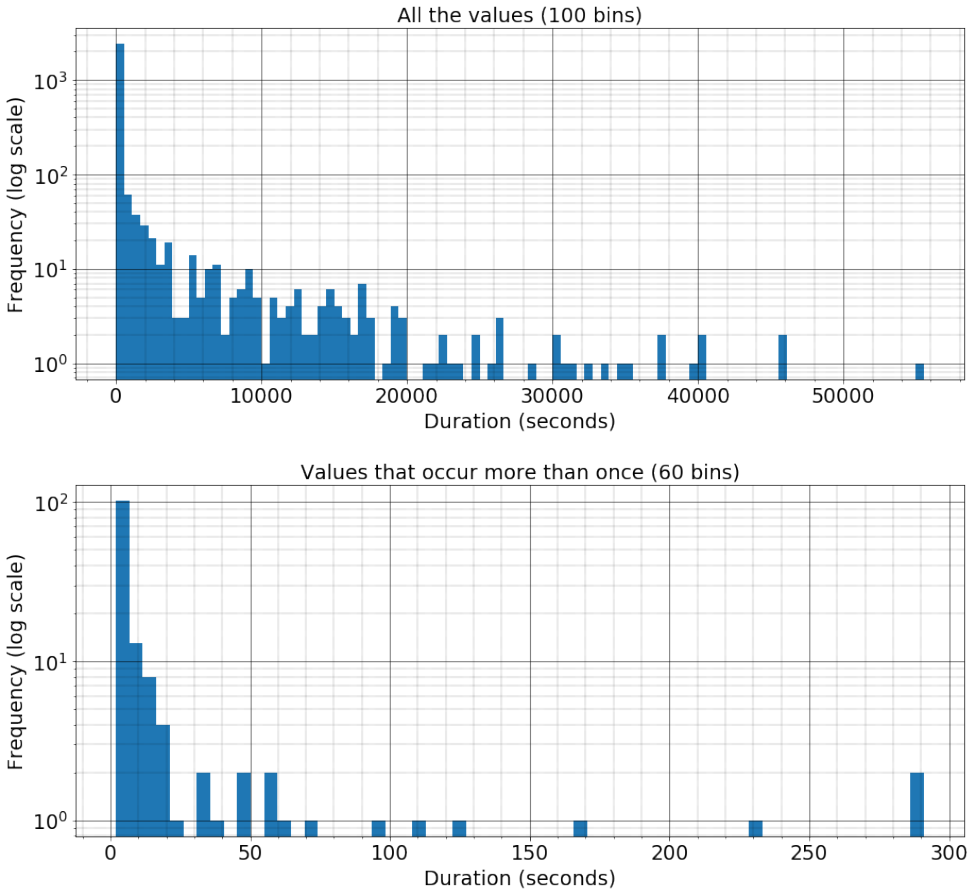

Figure 4.1: All timespan distribution of actions according to its frequency

Statistically, 75%, the 3rd percentile, of actions lasts less than 43 seconds. In this dataset, 43 seconds can already be considered long duration. Now, we can group and remove those actions from our analysis, so it is easier to analyse and label. The result is figure 4.2. It is possible to observe that there still is a high concentration of short values.

Again, 75% of the actions now last less than 12 seconds. So, actions that last the same or more than 12 seconds are now labelled with medium duration. Extracting these leads to figure 4.3.

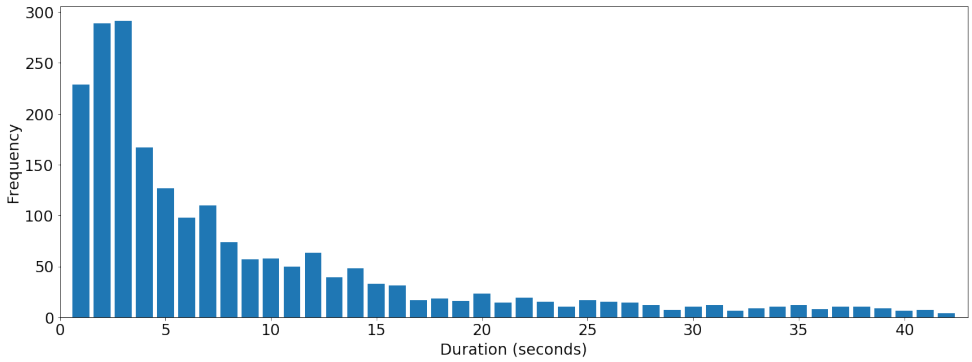

Figure 4.2: All, except long duration, timespan distribution of actions according to its frequency

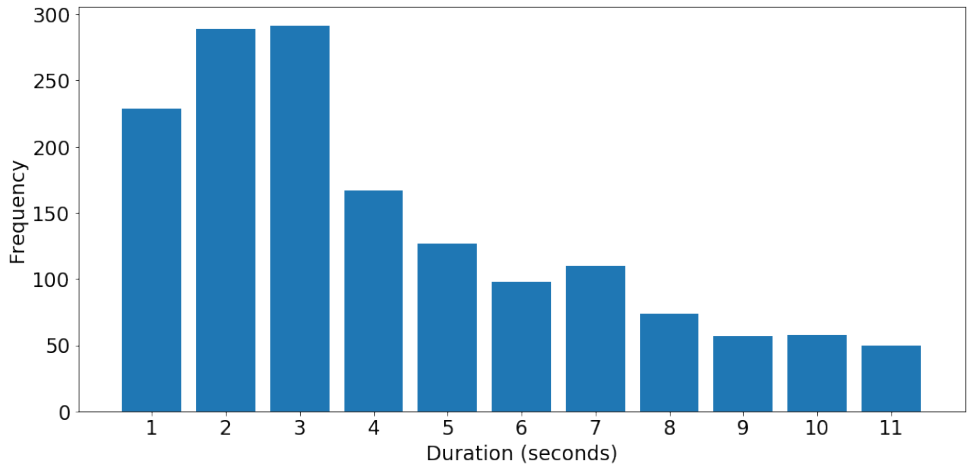

Figure 4.3: All, except medium and long durations, timespan distribution of actions according to its frequency

Now, the distribution is more uniform and given that the time interval is small, it is possible to make an analytical analysis. Table 4.2 represents this information. Actually, it is possible to deduce that fast actions are more common than when they take more time. This can either represent that someone opened the wrong drawer or closing the door with a fast movement. Also, the median now points to actions that last 3 seconds. We can now label these two groups as ultra-short and short duration.

So far, it has been analysed and labelled the actions according to their timespan. Given that there are no more durations to be analysed, it is possible to categorise the actions. The result is the one present in table 4.3.

To verify that nothing missed, in total, there are 2772 values which is the same number of actions of the original processed data.

Another important detail for the algorithms later used, section 5.2, is the separation by days is not the one traditionally made. This means that a day is not considered from midnight until midnight of the next day. The reasoning is that the usage of the devices shows that the user has different activity patterns until 5 a.m. So, a new day starts at 5 a.m. and algorithms may be better

Table 4.2: Frequency of short durations (sorted by their frequency)

Duration (seconds)	Frequency
3	291
2	289
1	229
4	167
5	127
7	110
6	98
8	74
10	58
9	57
11	50

Table 4.3: Frequency of actions according to their duration with the respective label

Time interval (seconds)	Label	Frequency
duration <= 3	Ultra-Short	809
3 <duration <12	Short	741
12 <= duration <43	Medium	524
duration >= 43	Long	698

tested if the cut-off of a new day is not made at midnight, but at 5 a.m. This decision can easily be understood in figure 4.4. At 5 a.m. there is no activity and it is the moment where the plot shows a more intensive utilisation of the house. Moreover, it is also possible to perceive from the plot that after 10 p.m. the number of actions drastically reduces, and it is a strong indicator of the inhabitant sleep patterns (from 11 p.m. until 5 a.m.).

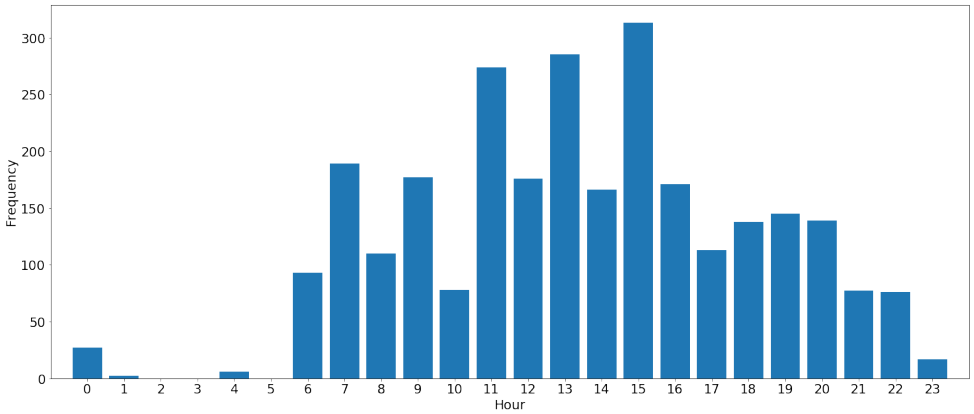

Figure 4.4: Frequency of actions distributed for each hour in a day

Despite this whole analysis, this is not enough, because it has only been analysed the timespans of the actions. Given that the dataset compresses multiple days, then it is also possible to observe its dispersion during the day and also throughout the week.

Starting by the day, it is commonly known that there are 4 periods: morning, afternoon, evening and night. People tend to stick to this schedule. If defined that morning is from 7 a.m. up to 12 a.m., and afternoon until 6 p.m., followed by evening until 12 p.m. and night until morning, then we obtain the results as displayed in table 4.4.

Table 4.4: Frequency of actions for each period of the day

Morning	**Afternoon**	**Evening**	**Night**
640	711	592	128

All periods show high activity except for the night which is when, presumably, the user is sleeping.

Regarding the activity during the week, table 4.5 shows the frequency of actions by the user for each day of the week. Please note that the results are the sum of the two weeks. So, if there are 2 Mondays in the dataset, then the number of actions is summed.

Table 4.5: Total number of actions for each day of the day of the week

Sunday	**Monday**	**Tuesday**	**Wednesday**	**Thursday**	**Friday**	**Saturday**
345	200	205	387	487	476	672

To finish, there is more to analyse like frequency of activities, but the current conclusions are enough to have an insight about the dataset and create a common baseline for the comparison of the algorithms. All the figures and logic applied is available online[1].

4.2 *MIT* 2

Similarly to *MIT* 1 - section 4.1 -, the same analysis has been performed and conclusions are similar.

Actually, for the timestamp of the actions, it has been used the same percentiles and the distribution is similar as shown in figure 4.5. In fact, it is possible to observe the same tendency for shorter values to be the most frequents. Therefore, labelling each action according to its length results in the same number of groups (ultra-short, short, medium and long), but at different intervals. This happens because this inhabitant produced data more disperse. The results of the grouping, using the same logic (i.e., the same percentiles), are displayed in table 4.6.

In total, there are 1962 values which is the same number as the number of rows of the original processed data.

Regarding the cut-off, a reasoning which has been explained in section 4.1, figure 4.6 allows to easily point out to the most probable time interval of sleep. In fact, from 9-10 p.m. until 4-5 a.m. the inhabitant shows a low activity (low frequency of actions) when compared to the rest of the day. Hence, the cut-off here applied is the same from section 4.1, 5 a.m.

[1] https://github.com/antonio-ramadas/upis/blob/master/notebooks/DiscretizeData-MIT1.ipynb

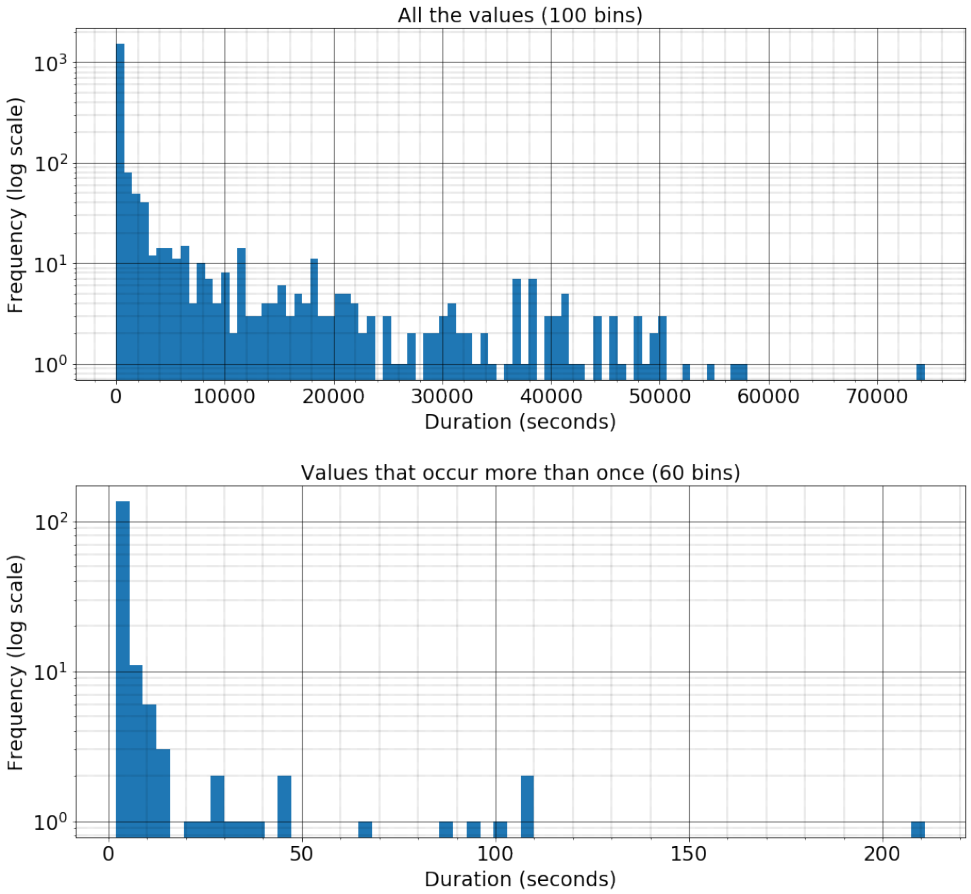

Figure 4.5: All timespan distribution of actions according to its frequency

Table 4.6: Frequency of actions according to their duration with the respective label

Time interval (seconds)	**Label**	**Frequency**
duration <= 5	Ultra-Short	620
5 <duration <19	Short	472
19 <= duration <233	Medium	379
duration >= 233	Long	491

As for the periods of the day, the same logic has again been applied and the split yielded the results displayed in table 4.7. Contrarily to expected, this inhabitant shows a great activity during the night period. A more detailed perspective is shown in figure 4.6 where for each hour of the day, it is presented the number of actions occurred.

Table 4.7: Frequency of actions for each period of the day

Morning	**Afternoon**	**Evening**	**Night**
588	269	211	389

Regarding the activity during the week, table 4.8 shows the frequency of actions by the user for each day of the week. The observation made for table 4.5 also applies to this one.

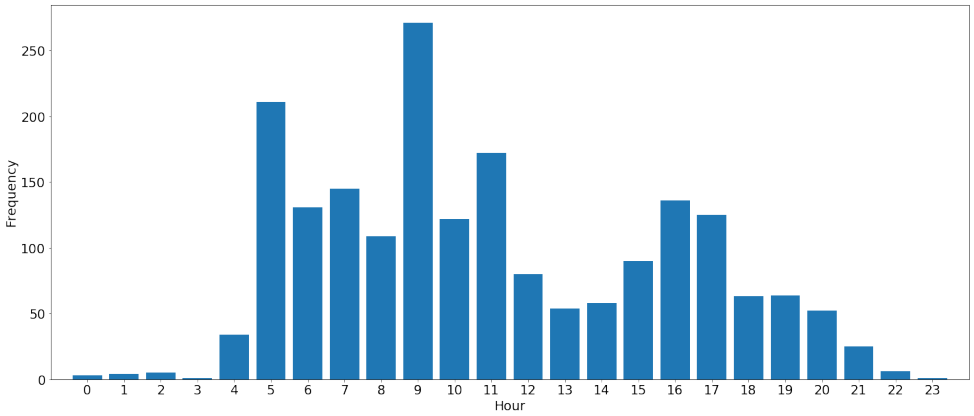

Figure 4.6: Frequency of actions distributed for each hour in a day

Table 4.8: Total number of actions for each day of the day of the week

Sunday	**Monday**	**Tuesday**	**Wednesday**	**Thursday**	**Friday**	**Saturday**
448	166	130	270	341	185	422

To conclude, the analyse of this dataset is shorter than *MIT* 1, because all the logic is the same. So, aiming to avoid repetition, only the raw processed data and results were shown. Lastly, this analysis is publicly available on GitHub[2].

4.3 Dataset conclusions

Both datasets share high similarities probably because the inhabitants had a similar schedule. However, to achieve this conclusion is necessary to make a more in-depth statistical analysis. It is not the case here, because it would require to moving time resources to it from the algorithms analysis. Also, for the goal established, comparison of multiple algorithms to predict user activities, this in-depth analysis is not necessary.

To finish, both sections 4.1 and 4.2 have the analysis publicly available on GitHub[3] as Jupiter Notebooks[4].

[2] https://github.com/antonio-ramadas/upis/blob/master/notebooks/DiscretizeData-MIT2.ipynb

[3] https://github.com/antonio-ramadas/upis/tree/master/notebooks

[4] https://jupyter.org

Chapter 5

Experimental Results and Discussion

This chapter first shows the experimental setup - section 5.1 - and then provides all the details and individual discussion to each method and an overall comparison - section 5.2. Finally, section 5.3 concludes this chapter.

5.1 Experimental Setup

This section can be seen as a sequence of section 3.3 because it introduces the strategy to evaluate the dataset. However, there the description is too generic. Hence, here lays one major detail of great importance that prevents the degradation of the quality of the dataset when employing the methods discussed in the other section.

In this case, the dataset was produced with 2 inhabitants being monitored, in separate houses, for two weeks. So, there are 10 weekdays (5 days for each week) and 4 weekend days (2 days for each week). In order to preserve the occurrences throughout the day, the order of the days given to the algorithms was random and the order of the actions inside each day was not rearranged. This ensures that the continuity of each day was not disturbed (e.g., moving the breakfast actions to the night period).

Actually, there is a high probability to choose more weekdays than weekend days. This breaks the week concept and induces the algorithm in error (if providing weekdays as training data and validate on weekends, then the algorithm will struggle to adapt and possibly be unaware of new actions/activities that are only performed during weekends). So, the approach used is to have an abstraction layer that treats weekdays and weekends and provides it together as the algorithm expects. Dealing with these two types of day separately means that the ratios 77%-33% are applied to each and then both splits are merged (the training data together and the two validation data also together). This logic is available online as a Python class[1]. For the algorithms, the split

[1] https://github.com/antonio-ramadas/upis/blob/master/sourcecode/DataProcessor.py

is abstracted and they are filled with input that is logically more accurate to reality than a naive division.

5.2 Algorithms and Results

As mentioned in chapter 2, classification of activities in smart homes is currently not a topic heavily researched by the community. This chapter shows how current methods behave when their focus is shifted from actions to activities. Additionally, as has been disclosed in chapter 3, one of the goals is to provide a baseline for future researchers. Hence, this section discusses the algorithm used and compares them all together.

In fact, Naive Bayes is forked into 2 different implementations - section 5.2.1 -, Random Forest is detailed with the features used - section 5.2.2 -, Q-Learning and Recurrent Neural Networks (*RNN*) are applied to smart homes - sections 5.2.3 and 5.2.4, respectively. Actually, the output of all these methods is the same, i.e. the activity (column ACTIVITY in table 4.1).

Lastly, the results of the metrics are truncated to 4 decimal places, because it is enough to untie possible similar results and it is also easier to visualise.

5.2.1 Naive Bayes

Naive Bayes is a supervised machine learning algorithm based on the Bayes' Theorem. This theorem describes that the probability of an event is not independent of other events meaning that it assumes a probability with prior events. The assumption made here is that an action is related to another action, for instance turning on the lights of the bathroom probably also means opening the water on the sink.

Regarding the implementation, in scikit-learn[2], there are three different implementations that vary on the formula to compute the probability (they all keep the assumption of not independent events): Gaussian Naive Bayes, Multinomial Naive Bayes and Bernoulli Naive Bayes. Given that the implementation is fairly simple, and the dataset is relatively small, a performance benchmark has been made on all with the default parameters.

Additionally, it conveys to denote that this algorithm has been implemented in two different ways: single, section 5.2.1.1, and multiple, section 5.2.1.2. Briefly, the variation between the two is the input. One only receives the previously used device on each step while iterating through the dataset. The other is provided, on each step, the complete state of the house for the activity, which means that it is given multiple devices.

Lastly, both types of implementation can be found on GitHub[3].

[2] http://scikit-learn.org/stable/modules/naive_bayes.html

[3] https://github.com/antonio-ramadas/upis/blob/master/sourcecode/Algorithms/NaiveBayes.py

5.2.1.1 Single

This variant of this algorithm is the simplest one because it is trained and evaluated with an input of just one device, an integer. In other words, the input is just the column SENSOR_ID from table 4.1 and each row is given as the correct output the element in the same row but in column ACTIVITY.

The result, as shown in table 5.1, indicates that this algorithm is slightly more proficient in *MIT* 1, but F1 cannot be considered good. In fact, the formula with the best results of the 3 is Gaussian with F1 reporting 0.0466 and 0.0296 for *MIT* 1, and 2, respectively. The performance on both datasets, *MIT* 1 and 2, is not desirable and there probably is a correlation with the fact that the input yields little information (just one device).

Actually, in chapter 4 has been disclosed that an activity has multiple devices and that one device may be shared by multiple activities. Hence, for each input (i.e., for each sensor) the output most likely is the activity that the sensor has been present more often. To address this issue, one possible solution is to provide the current entire state of the devices active (an array of sensors as contrary to just one sensor) given that it provides more information about the current activity (e.g., it is possible to make a better inference that we are cleaning an house if we are given the multiple lights that have been switched instead of just one single light). Additionally, this approach disregards the time that the action was performed. Consequently, it may downgrade the performance of the algorithm as we are subtracting temporal context.

Table 5.1: Performance of different metrics for an average on 10-fold

			Datasets	
			MIT 1	*MIT* 2
Naive Bayes	Gaussian	F1	**0.0466**	**0.0296**
		Precision	0.0442	0.0232
		Recall	0.0840	0.0625
	Multinomial	F1	0.0148	0.0087
		Precision	0.0086	0.0051
		Recall	0.0567	0.0398
	Bernoulli	F1	0.0165	0.0171
		Precision	0.0096	0.0103
		Recall	0.0573	0.0588

5.2.1.2 Multiple

By observation of the results in section 5.2.1.1, the implementation is the same except for the input which has been tweaked to accommodate more information to be filled to the algorithm. The idea is simple, for each activity, extract the state of every device in the house and provide that array of sensors to the algorithm. Hence the name of this variant. The logic for this change has been discussed in section 5.2.1.1.

Regarding performance, as shown in table 5.2, the formula yielding best results is Multinomial for both datasets. F1 is 0.3101 and 0.1867, for *MIT* 1 and 2, respectively. The algorithm also reveals issues in improving the performance of *MIT* 2 to *MIT* 1.

Moreover, this approach yielded worst results to the same algorithm as published by another researcher. However, it is worth mentioning that the goal was to predict actions and not activities [Ras14]. Concretely, the values reported are 0.4 and 0.51, for *MIT* 1, and 2, respectively. Here, the best is 0.3101, and 0.1867. Luckily, the implementation code was published[4] and after a dissection, it is evident that the approach is remarkably similar to the one of this section. Thus, the shift of the output (activities instead of actions) appears to be the culprit for this discrepancy.

Finally, one possible future improvement is to study the viability of adding temporal information to the algorithm. The implementation of this reasoning hardens with the fact that scikit-learn requires that all the inputs are of the same size (in this case it was used a LabelBinarizer[5] to encode all the actions into an array). Thus, in this case, it would be necessary to add an encoding function to make to group the array of sensors and the time. However, we would also be increasing the granularity of the input (i.e., there would be different encodings for the same 3 sensors at 9 a.m. and at 7 p.m.). Another possible tweak is to induce the activity by the time columns (START and END in table 4.1) and not by the actions. Actually, to do this it would be required the encoding of the analysis made in chapter 4 as the time given in the raw format may be too granular (precision to the second).

Table 5.2: Performance of different metrics for an average of 10-fold (in bold are the best F1 results)

			Datasets	
			MIT 1	*MIT* 2
Naive Bayes	Gaussian	F1	0.1570	0.1238
		Precision	0.1985	0.1362
		Recall	0.1756	0.1401
	Multinomial	F1	**0.3101**	**0.1867**
		Precision	0.3503	0.1869
		Recall	0.3167	0.2333
	Bernoulli	F1	0.2677	0.1745
		Precision	0.2871	0.1918
		Recall	0.2818	0.2056

5.2.1.3 Conclusions

Unsurprisingly, providing as input more information turns the algorithm more efficient. The multiple variant is consistently the best among the two. However, both are similar to the deterioration in

[4] https://github.com/krasch/smart-assistants

[5] http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelBinarizer.html

performance between *MIT* 1 and 2. One possible reason is that the assumption of the dependence between actions cannot be easily generalised to the smart-homes world.

5.2.2 Random Forest

Random Forest is an ensemble learning algorithm based on the construction of multiple decision trees. A tree infers rules from the training data which are then used to label the input. The algorithm of this section, instead of providing all the decision trees with the same input, gathers a sample from all the training data and provide it to a decision tree. Therefore, the trees, due to the fact that may be given different data, may output different results. If that is the case, then Random Forest applies a formula to choose one of the outputs. This formula may vary differently in implementation, for instance, it may be the most common output or the maximum average of the probability for each class.

The current implementation[6] relies, once again, on the implementation made by scikit-learn for Random Forest[7] with the default parameters. Moreover, the input has also been modified to provide more features, so the trees can make better inferences. The features created were based on the analysis made in chapter 4. So far, the input used contained the following columns:

- Sensor ID
- The time when the action started
- The time when the action ended

However, time, as has been used, is too granular for the intention. An activity is more constrained to the period of the day than to the time with precision to the second. Moreover, these two features contain valuable information which can be hardly decoded autonomously by the algorithm. For instance, it is possible to decode the day of the week and because one week is similar to other (e.g., wake-up, go to work, come from work, dinner, sleep), the algorithm can easily detect this periodicity, if present.

Consequently, after the extraction and computation, the features are now:

- Sensor ID

  Same as before

- Duration

  The difference, in seconds, from the time the action ended, and it started

- Duration categorised

  Duration labelled according to the analysis made in chapter 4

[6] https://github.com/antonio-ramadas/upis/blob/master/sourcecode/Algorithms/RandomForest.py

[7] http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

- Weekday

  Either Sunday, Monday, Tuesday, Wednesday, Thursday, Friday or Saturday

- Period of the day

  Either morning, afternoon, evening, or night

Now with the input already processed, the performance of this algorithm is shown in table 5.3. Actually, the F1 metric shows that this algorithm is currently not very well suitable for this type of task. Perhaps, by providing more features (do a more extensive analysis on feature engineering) such as the previous k actions or even adding more computed columns would give better results, but the current F1 value either shows that there is a lot of space for optimisations (carefully adding features to avoid overfitting the dataset) or that simply Random Forest is not a suitable algorithm. Out of the scope of this algorithm, but it would be useful to provide a more extensive dataset because there would be more training data which empowered a better decision process.

Table 5.3: Performance of different metrics for an average on 10-fold

		Datasets	
		MIT 1	*MIT* 2
Random Forest	F1	0.0886	0.0821
	Precision	0.0996	0.0908
	Recall	0.1015	0.1052

5.2.3 Reinforcement Learning

Q-Learning is a reinforcement learning algorithm that has been adapted to smart-homes by Mamun Bin Ibne Reaz et al. [MI13]. The main difference between the implementation proposed and the here implemented[8] is that the dataset is not simulated and the classification is activity and not actions (e.g., turn on the lights). **When implementing this algorithm, one struggles due to the lack of precision in the algorithm description.** Moreover, due to the lack of a public code implementation by the authors, the implementation here presented may not be the same as a consequence of the assumptions made.

Before approaching the pseudocode, it is worth mentioning that the single input is the column ACTIVITY in table 4.1 and the feedback for the reinforcement learning algorithm is either positive or negative according to the equality of the output and the next activity in the dataset (i.e., the one after the current).

The pseudocode of the algorithm as indicated by the authors is:

1. Let α be the threshold value $0 \leq \alpha \leq 1$, $f(s,a)$ is the action under consideration under the latest current state in the recent graph of event sequence.

[8] https://github.com/antonio-ramadas/upis/blob/master/sourcecode/Algorithms/QLearning.py

2. Calculate the longest pattern match starting from the current state in the recent graph of event sequence and store it in variable L.

3. Calculate the Q-value function for the current action under consideration based on the previous history of events using and store it in variable R.

4. $h(s,a)$ is the total number of occurrences of action a under the current state s and $\sum_i h(s,a_i)$ is the calculated total number of occurrences of other actions a_i at the same state s.

5. Calculate the probability for the current action a under consideration using the formula:
$$f(s,a) = (1-\alpha)L + \alpha\left\{\frac{h(s,a)}{\sum_i h(s,a_i)} + R\right\}$$

6. Repeat steps 2 to 5 for the subsequent actions for the same state s and select the action with the highest ranking.

7. After all the actions have been taken under consideration receive feedback from the home inhabitant. If the action was satisfying give the action a positive reward otherwise give the action a negative reward while taken the user chosen action under consideration. Calculate the Q-value for the selected action by the algorithm using Q-learning formula $Q^*(x,a) = (1-\alpha)Q^*(x,a) + \alpha(r + \gamma V^*(y))$.

Regarding step 7, α is the learning rate, γ is the value of future reinforcement and V^* is the future Q-learning value function (maximum reward possible on the next state).

From the pseudocode is possible to split the algorithm into 2 components: pattern matching and reinforcement learning. The first is performed with the assistance of 2 graphs: history graph (which stores the history of activities) and a recent event graph (which store current actions) - figures 5.1 and 5.2, respectively. The edges of both graphs have been adapted from action to activity (i.e., when an activity starts or ends, the state is changed). The states are a set of active activities (i.e., the ones currently being performed). **One assumption occurs here because it is not referred when the activities are added to the history graph nor when the recent event graph actions are added to the history.** If this step is not performed, then the algorithm will have the history unmodified throughout the lifetime of the algorithm and relied upon without considering new changes (new activities or new order of activities being performed). Also, the recent event graph will continuously grow and, eventually, became bigger than the history itself. So, in order to overcome this obstacle, at the end of each day, the event graph is cleared. **It is important to clarify that it is not merged with the history graph, because it is not clear that if an output is wrong, then which of the possible activities is correct**. The second component is the q-learning algorithm itself.

In fact, with these 2 components, this algorithm performs pattern matching with the two graphs (recent graph on the history graph). Afterwards, it is applied a metric on top of the value computed by the Q-Learning algorithm. It is this metric that matters because it is the one being maximised to choose the output. Actually, it is simple to understand because it is the sum of both of the current events and the history with a threshold variable to choose which operand side should have

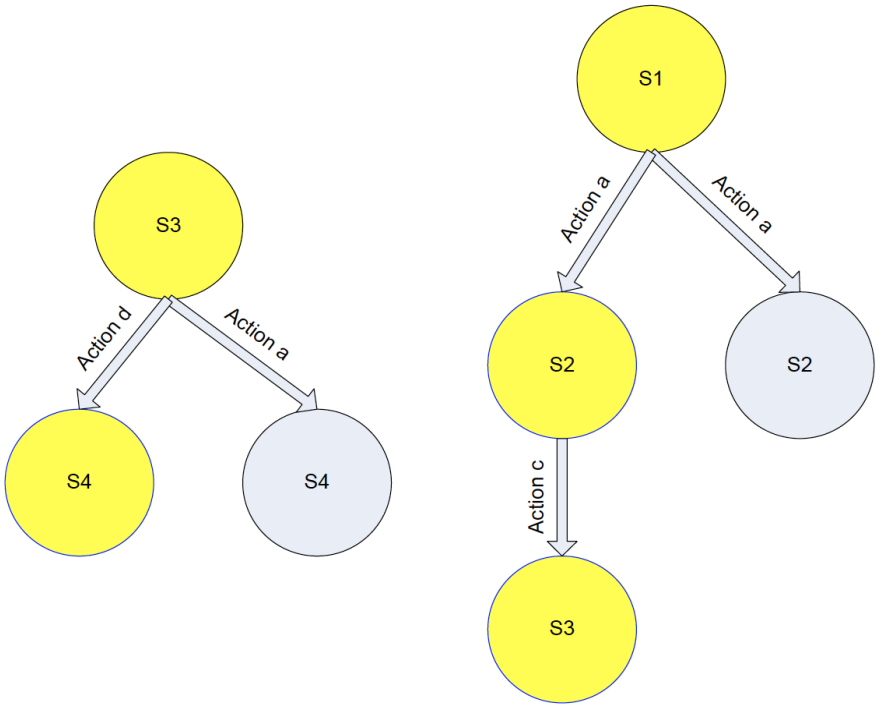

Figure 5.1: History graph [MI13]; the adaptation to current work has been the change from action to activities, so where it is read action, it should be activity

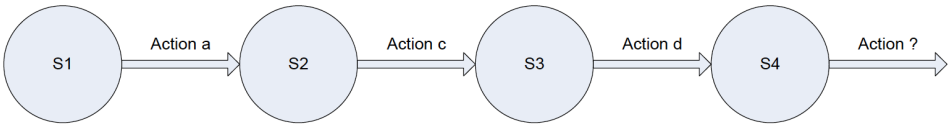

Figure 5.2: Current event graph [MI13]; the adaptation to current work has been the change from action to activities, so where it is read action, it should be activity

more weight. Here, it was used 0.5 as the threshold, meaning that both operands have the same weight as the decision, i.e. it is an average. However, the authors state that used a threshold of 6.0 which does not respect the first step of the pseudocode. Lastly, the feedback for the Q-Learning algorithm is the ground truth.

Regarding the performance of this algorithm, the results can be found in table 5.4. Before observing the metric F1, it is worth mentioning that *MIT* 1 has no overlapping activities. Hence the history graph will have height 1. On the other hand, *MIT* 2 has at most 2 activities overlapping, thus the height increases to 2. **Now, about F1, rounding it to 0.02 reveals a value too low.** The authors of the algorithm use a different metric, accuracy, but here it has been used F1 because it is more robust - section 3.3. **For comparison, the authors show that their implementation has an accuracy of 87% where here it was achieved, at best, about 12%**. Given that there is no information regarding its input, this result can be very easy to customise because it greatly depends on the complexity of the dataset.

Table 5.4: Performance of different metrics for an average on 10-fold

		Datasets	
		MIT 1	*MIT* 2
Q-Learning	F1	0.0190	0.0171
	Precision	0.0203	0.0174
	Recall	0.0418	0.0487
	Accuracy	0.0729	0.1169

Actually, there are some additional notes to be added to the paper being followed. The dataset is not described. It is said that it is synthetic, but it is not given extra information. What is it distribution? What is the number of sensors? How many times are they used? How much time does the dataset last?

Anyway, the algorithm approach definitely shows an adaptation to smart homes with the creation of additional data structures, when compared to the original Q-Learning algorithm, to accommodate the knowledge of the data already given. For future improvement, the algorithm awareness to extra features would, most likely, increase its performance (e.g. explicitly provide the day of the week). Also, the current approach does not take into account the sensors used, but only the activities. So, accommodating those in the algorithm can leverage its performance. Currently, this algorithm has room for improvement which could be accomplished through tuning of the logic of the algorithm.

5.2.4 Recurrent Neural Networks

RNN are a special type of Neural Networks where each column of the input is a node that connects to the next one. Hence, it forms a sequence. This approach has been breaking some of the state of the art methods for problems like text translation. [CvMG+14] Even for time series events *RNN* are efficient. [CET12]

Actually, the main reason for the proficiency of *RNN* in data that comes in series is derived from its design. Starting with an example, imagine that a person is reading a book and a character is introduced. Later on, the author uses a pronoun (e.g., he or she). Now, the reader, which is aware of the context, is able to realise that the author is referring to that specific character. In its essence, *RNN* emulate this with cells. A cell has a state (the context) and gates capable of controlling both past and recent memories. Thus the name Long Short-Term Memory. *RNN* is often represented as in figure 5.3. Moreover, in this figure is not shown the gates previously introduced because current approaches to implement them vary. In this work, these gates are implemented in 2 different ways with Long Short-Term Memory (*LSTM*) and Gated Recurrent Unit (*GRU*). *LSTM* is represented in figure 5.4 and the gates have a memory unit to control the flow of information. On the other hand, *GRU*, figure 5.5, does not have this memory unit and has fewer parameters which makes it able to train faster (require less data). *GRU* is often seen as an improvement of *LSTM*, but *LSTM* with large data tends to lead to better results. [CGCB14]

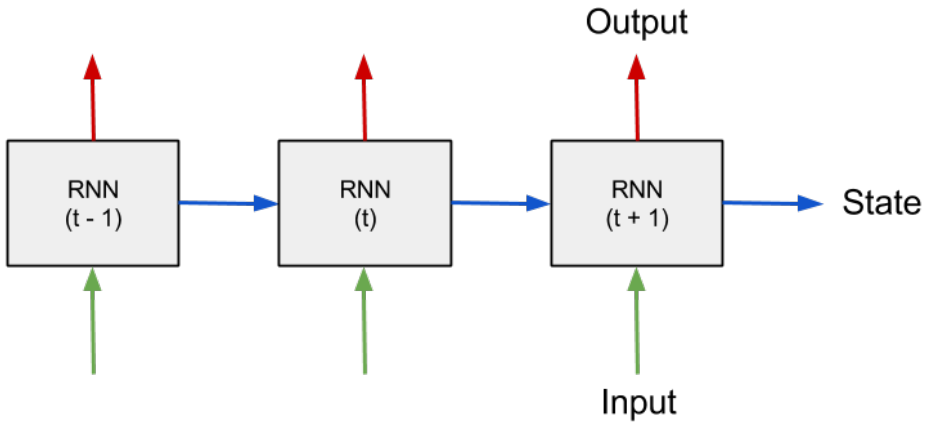

Figure 5.3: RNN architecture [Hal16]

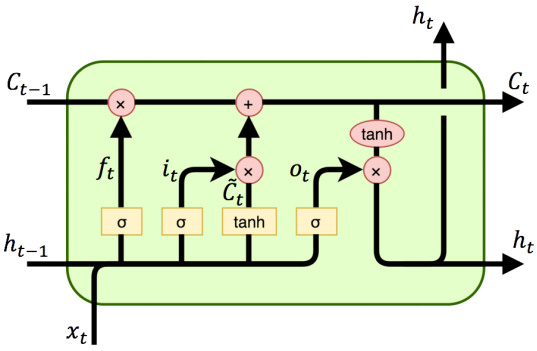

Figure 5.4: LSTM architecture [Cha17]

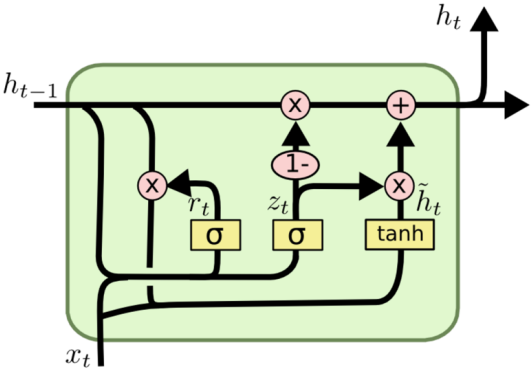

Figure 5.5: GRU architecture [Cha17]

Specifically to this work, *RNN* are then applied to the challenges in smart-homes. The activities are already timely sorted, so it was a natural choice. Again, the implementation is publicly available on GitHub[9]. The code uses Keras[10] to abstract the *RNN* implementation and provide an API to build the model. Also, Keras neatly provides constructors that allow to easily change the implementation from *LSTM* to *GRU*, because both are *RNN* algorithms. Thus, both were tested with the same architecture: dropout at 0, window size equal to 5, cells cleared at the end of each batch (where a batch is 1 day), and 128 neurons for each of the 1 layer. The rest of the variables are the default.

Regarding the input, it was given the column ACTIVITY in table 4.1, but with a small tweak. It was applied the sliding window technique with size equal to 5 (this is the window size referred before) and the next entry to the window is the output of that window. Mathematically, this window effect can be expressed as (ACT is used as shorthand for ACTIVITY):

$$\forall i \in [0, |ACT| - 6] : Input = [ACT_i, ACT_{i+1}, ACT_{i+2}, ACT_{i+3}, ACT_{i+4}, ACT_{i+5}]; Output = [ACT_{i+6}]$$

[9] https://github.com/antonio-ramadas/upis/blob/master/sourcecode/Algorithms/RNN.py

[10] https://keras.io

Actually, the process to achieve this values was trial and error by watching similar graphs to the ones in figure 5.10. It is worth referring that validation data is used to measure the performance of the model at the end of each epoch. This data is one single day that has been removed from training data. Hence, it simulates the test set. This is a common practice across the community to avoid that the model overfits to the data. Specifically about the graphs, according to F1, both figures 5.6 and 5.8 show that both models fit the training data perfectly. However, in all cases, this score is not reflected on validation data, because the score consistently stays below 0.3 - figures 5.7 and 5.9. One way to fix this situation is by providing more data (i.e., a larger dataset), but the current dataset does not allow this. Lastly, the contrast of these results with table 5.5 is a consequence of the model being measured against the test set, and not the validation set as it used in figure 5.10.

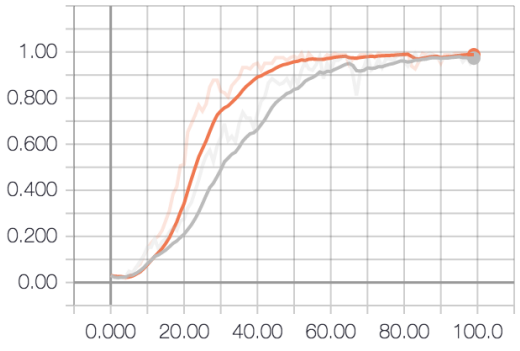

Figure 5.6: Model performance on training data of *MIT* 1

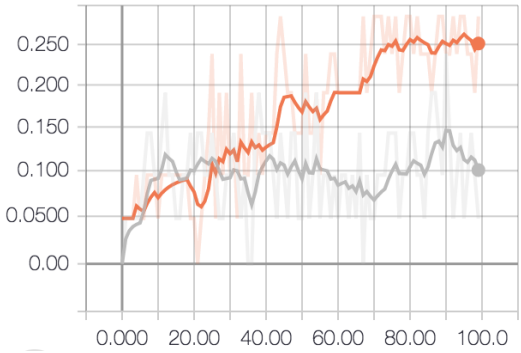

Figure 5.7: Model performance on validation data of *MIT* 1

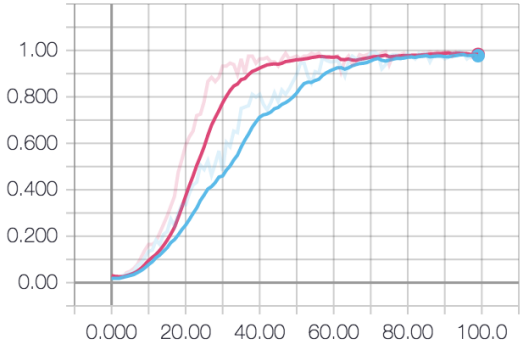

Figure 5.8: Model performance on training data of *MIT* 2

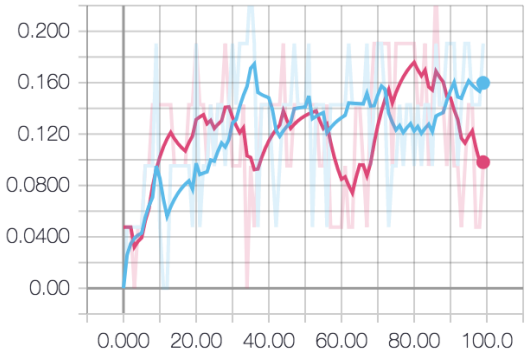

Figure 5.9: Model performance on validation data of *MIT* 2

Figure 5.10: Colors orange and pink are *GRU*, grey and blue are *LSTM*; performance measured with the metric F1; y-axis is the F1 result and the x-axis is the epoch

Regarding the performance, table 5.5 shows the result of the metrics applied to the test dataset. There, it is possible to observe that *GRU* is consistently the best performant with F1 at 0.2882 and 0.4452 for *MIT* 1, and 2, respectively. *GRU* was introduced as an improvement over *LSTM* to the research community and it is known to require fewer data to achieve similar results. Here, because the dataset is small for this type of algorithms, *GRU* is more efficient.

To conclude, when this algorithm is reported to the community, they usually have very lengthy datasets, e.g. hundreds of thousands of inputs entries instead of the current few hundreds ($\pm$200 total entries for each *MIT* 1, and 2). Also, the architecture here is very simple with small layers to avoid overfitting. **It is evident that more data is required for this approach, but current results already hint that *RNN* is a promising the state of the art method in smart-homes.**

Table 5.5: Performance of different metrics for an average of 10-fold (in bold are the best F1 results)

			Datasets	
			MIT 1	*MIT* 2
RNN	*LSTM*	F1	0.1806	0.1238
		Precision	0.2026	0.1362
		Recall	0.1927	0.1401
	GRU	F1	**0.2882**	**0.4452**
		Precision	0.3014	0.4737
		Recall	0.3123	0.4604

5.2.5 Overall Discussion

So far, sections 5.2.1, 5.2.2, 5.2.3 and 5.2.4 provided a detail explanation of both the algorithms and results. The analysis made allows the extraction of the best results on each section to be compared to each other. Table 5.6 holds this logic for the F1 metric, the one more robust of the three. There, it is possible to separate the methods into 2 categories.

The first, with the worst results, captures Random Forest and Reinforcement Learning. Their scores are below 0.1 which is very undesirable. **Anyway, Reinforcement Learning, the one that required assumptions to be applied, is about 80% worse than Random Forest.** This shows that this algorithm is far from state of the art, and that also requires a more elaborated thought process.

The second, with the best results, capture Naive Bayes Multiple and Recurrent Neural Network. On *MIT* 1, they are almost the same with a difference of just 0.0219 (out of curiosity, this delta is bigger than the score of Reinforcement Learning). On *MIT* 2, Recurrent Neural Network is more than twice the better (0.1867 and 0.4452). **Despite *RNN* losing in one of the datasets, it shows a greater power of generalisation when compared to all the other methods.**

Table 5.6: The best F1 metric for each type of algorithm on *MIT* 1, and 2

	MIT 1	*MIT* 2
Naive Bayes Multiple	**0.3101**	0.1867
Random Forest	0.0886	0.0821
Reinforcement Learning	0.0190	0.0171
Recurrent Neural Network	0.2882	**0.4452**

5.3 Conclusions

Throughout this section, it was given two insights: how the traditional evaluation process was customised for the current context - section 5.1 -, and how the methods were implemented and their benchmark - section 5.2. Specifically, from this last part, it was concluded that *RNN* **is the one with greatest generalisation power despite still having room for improvement** (i.e., providing a larger dataset).

Chapter 6

Conclusions and Future Work

This chapter concludes the document. It extracts the main points achieved throughout the chapters - section 6.1 -, concludes the work - section 6.2 -, and provides the topics which lead to future work to improve the classification in smart-homes - section 6.3.

6.1 Summary of Work

With the continuous expansion of the Internet of Things industry, devices now serve as collectors and as actuators. Therefore, the research community has been releasing proposals that make use of these devices to empower a smart home by predicting the user behaviour. As a consequence, the inhabitant can save energy while maximising his comfort. There are two classification levels: actions and activities. In this work has been chosen to predict activities due to its binary nature (an activity is being done or not; there is nothing in between) and it also removes the possible granularity an action may have (e.g., level of brightness of a lamp).

Regarding the current state of the art - chapter 2 -, it is possible to conclude that current technologies and architectures already allow giving intelligence to smart homes by deploying smart devices. With the capture of this smart devices, it is possible to extract the data necessary for smart algorithms to help the user with his actions/activities. Hence, current methods make use of the data captured in datasets created by learning environments. Also, these methods vary on the approach taken, but all yield the same output. However, comparing these is obstructed by the multiplicity of techniques used by each author to evaluate the performance. As such, most of the sources of the dataset vary and the metrics used are few times the same.

Therefore, current work aims to contribute by performing a comparative evaluation of multiple machine learning algorithms to detect user activities. While achieving this, there are 2 outcomes. The first is an analysis of the dataset chosen and the other is the publication of the implementation of the 4 methods studied.

The dataset chosen is from the Massachusetts Institute of Technology because it is labelled (each sensor used has its activity associated) and it is widely used by the research community despite its short length (just 2-weeks). As such, this choice eases the comparison with methods from other proposals - chapter 5.

On top of the dataset, the methods used are Naive Bayes, Random Forest, Reinforcement Learning and *RNN*. Through 3 different metrics (F1, Precision, and Recall), it was concluded that Reinforcement Learning was the worst performant of the four. On other hand, *RNN* did not consistently reveal the top results, but overall was the one with most generalisation power.

6.2 Conclusions

In section 5.2.5, it has been concluded that *RNN* have the greatest generalisation power (overall had the best results). However, it was not disclosed situations in which this results may be affected and even inverted.

In fact, one possible important factor to cause the disruption would be a different dataset. The research community aims to create a system that works in a general environment because not all houses have the same architecture. However, the dataset used only has two different architectures. Other datasets do have different architectures than the learning environment used by *MIT*, but most only have 1 house. Hence, a more detailed comparative study would use more datasets, but this work did not make that, because different datasets have different input entries. Consequently, it would be necessary to create a specific parser for each dataset and reduce the information of all to the common one (i.e., one dataset has binary devices and the rest has non-binary devices). Logically, this requires a great amount of time on dataset analysis and implementation.

Another possible disruption may be caused by the replacement of the dataset from *MIT* with a lengthier one. *MIT* lasts 2-weeks and happens in April. So, there are two probable problems. The first is that 2-weeks hardly allow inferring the activities of the inhabitant throughout the whole year (i.e., the other 50 weeks). The second is that April is a single month in a year. Thus, it is difficult to generalise the inhabitant activities in more distant months like August or November, because in these months the weather or is traditionally warmer, and invites outside activities (i.e., not inside the smart-home) and use of other devices (e.g., A/C), or it is often colder, and the inhabitant may stay more at home and use more heating devices. Thus, the method studied against *MIT* should, probably, be more accurate in April than the rest of the year. Hence, an inference of the generalisation of the method may be not correct.

To overcome this possible disruption, some proposals used simulated datasets which can be larger and configurable (e.g., different house architectures and a different number of devices). However, as mentioned in section 2.2.8, they are challenged by the complexity of human interactions with the environment. Therefore, they tend to provide datasets which are too synthetic.

Finally, because the dataset may be improved, the results of the methods may not keep the performance currently achieved and either be improved or diminished. Nonetheless, current results allow to derive some conclusions and create hypothesis for future study.

6.3 Future Work

This work identified 5 challenges for future work in smart-homes - section 3.1.

During the course of development, it was also uncovered that binary devices are no longer the monopoly of the industry. Hence, a more updated dataset is required to better replicate a real-world scenario. Additionally, a larger dataset is extremely appreciated to provide a bigger test suit for future methods. Another important factor is that it is difficult to extract good solutions from a 2-week dataset that perform well, for instance, throughout 1 year, because users may stay more at home during the winter than during the summer. Hence, they may perform a different set of activities.

It is also worth highlighting that the methods studied could not be automatically triggered, i.e. they were required to be invoked to yield an output. So, there is an opportunity to research on a technique that is more aware of the environment and capable of triggering itself to yield an output with a certain degree of quality (basically, be aware of time variable).

Finally, more specifically to *IoT*, it would be interesting to have *IoT* devices coordinating among themselves to act autonomously without having to rely on a server that is running the learning component (e.g., running *RNN* to make classifications).

References

[AF94] James F. Allen and George Ferguson. Actions and Events in Interval Temporal Logic. *Journal of Logic and Computation*, 4(5):531–579, 1994.

[ALS12] Avinash Achar, Srivatsan Laxman, and P. S. Sastry. A unified view of the apriori-based algorithms for frequent episode discovery. *Knowledge and Information Systems*, 31(2):223–250, 2012.

[Ash09] Kevin Ashton. That 'internet of things' thing. *RFID Journal*, Jun 2009.

[BFSL+15] Kevin Bouchard, Dany Fortin-Simard, Jeremy Lapalu, Sebastien Gaboury, Abdenour Bouzouane, and Bruno Bouchard. Unsupervised Spatial Data Mining for Smart Homes. *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, pages 1433–1440, 2015.

[Bor05] Christian Borgelt. Keeping things simple. In *Proceedings of the 1st international workshop on open source data mining frequent pattern mining implementations - OSDM '05*, pages 66–70, New York, New York, USA, 2005. ACM Press.

[BY15] Serge Thomas Mickala Bourobou and Younghwan Yoo. User activity recognition in smart homes using pattern clustering applied to temporal ANN algorithm. *Sensors (Switzerland)*, 15(5):11953–11971, 2015.

[CCJ17] Wen-hui Chen, Po-chuan Cho, and Yong-lin Jiang. Activity Recognition Using Transfer Learning. *Sensors and Materials*, 29(7):1, 2017.

[CCTK13] Diane J Cook, Aaron S Crandall, Brian L Thomas, and Narayanan C Krishnan. CASAS: A Smart Home in a Box. *Computer*, 46(7):62–69, jul 2013.

[CET12] Qing Cao, Bradley T. Ewing, and Mark A. Thompson. Forecasting wind speed with recurrent neural networks. *European Journal of Operational Research*, 221(1):148–154, 2012.

[CGCB14] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. pages 1–9, 2014.

[Cha17] Isaac Changhau. LSTM and GRU – Formula Summary, 2017.

[CHGY03] Diane J. Cook, Manfred Huber, Karthik Gopalratnam, and Michael Youngblood. Learning to Control a Smart Home Environment. *Innovative Applications of Artificial Intelligence*, 2003.

[CKO13] Sung Joon Choi, Eun Woo Kim, and Song Hwai Oh. Human Behavior Prediction for Smart Homes Using Deep Learning. *22nd International Symposium on Robot and Human Interactive Communication*, 1(dataset 2):173–179, 2013.

[CLH17] Yi-ting Chiang, Ching-hu Lu, and Jane Yung-jen Hsu. A Feature-Based Knowledge Transfer Framework for Cross-Environment Activity Recognition Toward Smart Home Applications. *IEEE Transactions on Human-Machine Systems*, 47(3):310–322, jun 2017.

[CvMG+14] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. 2014.

[CZ16] Mung Chiang and Tao Zhang. Fog and IoT: An Overview of Research Opportunities. *IEEE Internet of Things Journal*, 3(6):854–864, 2016.

[DBNA15] Paul Daugherty, Prith Banerjee, Walid Negm, and Allan E Alter. Driving unconventional growth through the Industrial Internet of Things. *Accenture*, pages 1–20, 2015.

[FR12] Hongqing Fang and Jinjin Ruan. An improved position prediction algorithm based on active lezi in smart home. *Proceedings - 2012 International Conference on Computer Science and Service System, CSSS 2012*, pages 1733–1736, 2012.

[FU08] Strategic Business Insights (Firm) and National Intelligence Council (U.S.). *Disruptive Civil Technologies: Six Technologies with Potential Impacts on US Interests Out to 2025 : Biogerontechnology, Energy Storage Materials, Biofuels and Bio-based Chemicals, Clean Coal Technologies, Service Robotics, the Internet of Things*. National Intelligence Council, 2008.

[GBMP13] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, 29(7):1645–1660, 2013.

[GMKS17] O Garcia-Morchon, S Kumar, and M Sethi. State of the Art and Challenges for the Internet of Things. pages 1–56, 2017.

[HA15] Marwa Hassan and Mirna Atieh. Action Prediction in Smart Home Based on Reinforcement Learning. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 8456, pages 207–212. 2015.

[Hal16] Erik Hallstrom. How to build a Recurrent Neural Network in TensorFlow (1/7), 2016.

[HFH15] Md. Mahmud Hossain, Maziar Fotouhi, and Ragib Hasan. Towards an Analysis of Security Issues, Challenges, and Open Problems in the Internet of Things. *2015 IEEE World Congress on Services*, pages 21–28, 2015.

[JC07] V Jakkula and Dj Cook. Learning temporal relations in smart home data. *International Conference on Technology and Aging*, 2007.

[LBB+13] Jérémy Lapalu, Kevin Bouchard, Abdenour Bouzouane, Bruno Bouchard, and Sylvain Giroux. Unsupervised Mining of Activities for Smart Home Prediction. *Procedia Computer Science*, 19(Ant):503–510, 2013.

[LHSS10] O. León, J. Hernández-Serrano, and M. Soriano. Securing cognitive radio networks. *International Journal of Communication Systems*, 23(5):633–652, 2010.

[LXZ15] Shancang Li, Li Da Xu, and Shanshan Zhao. The internet of things: a survey. *Information Systems Frontiers*, 17(2):243–259, 2015.

[MCB+15] James Manyika, Michael Chui, Peter Bisson, Jonathan Woetzel, Richard Dobbs, Jacques Bughin, and Dan Aharon. The Internet of Things: Mapping the value beyond the hype. Technical Report June, McKinsey Global Institute, 2015.

[MI13] Mohd Marufuzzaman and Mamun Bin Ibne Reaz. Hardware simulation of pattern matching and reinforcement learning to predict the user next action of smart home device usage. *World Applied Sciences Journal*, 22(9):1302–1309, 2013.

[MMST16] Julien Mineraud, Oleksiy Mazhelis, Xiang Su, and Sasu Tarkoma. A gap analysis of Internet-of-Things platforms. *Computer Communications*, 89-90:5–16, 2016.

[MSDC12] Daniele Miorandi, Sabrina Sicari, Francesco De Pellegrini, and Imrich Chlamtac. Internet of things: Vision, applications and research challenges. *Ad Hoc Networks*, 10(7):1497–1516, 2012.

[Nat15] United Nations. World Population Ageing. Technical Report (ST/ESA/SER.A/390, 2015.

[Nes18] Nest. Nest Learning Thermostat, 2018.

[Ras14] Katharina Rasch. An unsupervised recommender system for smart homes. *Journal of Ambient Intelligence and Smart Environments*, 6(1):21–37, 2014.

[RDD+17] António Ramadas, Gil Domingues, João Pedro Dias, Ademar Aguiar, and Hugo Sereno Ferreira. Patterns for Things that Fail. In *Pattern Languages of Programs*, pages 1–10, Vancouver, Canada, 2017.

[Rey87] Craig W. Reynolds. Flocks, herds and schools: A distributed behavioral model. *ACM SIGGRAPH Computer Graphics*, 21(4):25–34, 1987.

[RRD06] Nirmalya Roy, Abhishek Roy, and Sajal K. Das. Context-aware resource management in multi-inhabitant smart homes: A nash H-learning based approach. *Proceedings - Fourth Annual IEEE International Conference on Pervasive Computing and Communications, PerCom 2006*, 2006:148–158, 2006.

[RT17] Biljana L. Risteska Stojkoska and Kire V. Trivodaliev. A review of Internet of Things for smart home: Challenges and solutions. *Journal of Cleaner Production*, 140:1454–1464, 2017.

[SNJ15] Jonathan Synnott, Chris Nugent, and Paul Jeffers. Simulation of smart home activity datasets. *Sensors (Switzerland)*, 15(6):14162–14179, 2015.

[SS13] P Sukanya and Gayathri K S. An Unsupervised Pattern Clustering Approach for Identifying Abnormal User Behaviors in Smart Homes. *IJCSN International Journal of Computer Science and Network ISSN*, 2(3):2277–5420, 2013.

REFERENCES

[Tan10] Lu Tan. Future internet: The Internet of Things. *2010 3rd International Conference on Advanced Computer Theory and Engineering(ICACTE)*, pages V5–376–V5–380, 2010.

[TIL04] Emmanuel Munguia Tapia, Stephen S. Intille, and Kent Larson. Activity Recognition in the Home Using Simple and Ubiquitous Sensors. pages 158–175, 2004.

[WAD15] Andrew Whitmore, Anurag Agarwal, and Li Da Xu. The Internet of Things—A survey of topics and trends. *Information Systems Frontiers*, 17(2):261–274, 2015.

[WB14] Gerald Wagenknecht and Torsten Ingo Braun. *Internet of Things, Smart Spaces, and Next Generation Networks and Systems*, volume 8638. 2014.

[WC07] Spokane Way and Diane J Cook. Using Temporal Relations in Smart Environment Data for Activity Prediction. *Proc of the 24th International Conference on Machine Learning (ICML '07)*, (October):4, 2007.

[WF15] Felix Wortmann and Kristina Flüchter. Internet of Things: Technology and Value Added. *Business and Information Systems Engineering*, 57(3):221–224, 2015.

[WRS+17] Shaoen Wu, Jacob B. Rendall, Matthew J. Smith, Shangyu Zhu, Junhong Xu, Honggang Wang, Qing Yang, and Pinle Qin. Survey on Prediction Algorithms in Smart Homes. *IEEE Internet of Things Journal*, 4(3):636–644, 2017.

[XHL14] Li Da Xu, Wu He, and Shancang Li. Internet of things in industries: A survey. *IEEE Transactions on Industrial Informatics*, 10(4):2233–2243, 2014.

[ZFYF17] Tongda Zhang, Wensi Fu, Jinchao Ye, and Martin Fischer. Learning movement patterns of the occupant in smart home environments: an unsupervised learning approach. *Journal of Ambient Intelligence and Humanized Computing*, 8(1):133–146, 2017.

[ZPOL97] Mohammed Javeed Zaki, Srinivasan Parthasarathy, Mitsunori Ogihara, and Wei Li. New Algorithms for Fast Discovery of Association Rules. *3rd Intl Conf on Knowledge Discovery and Data Mining*, 20(651):283–286, 1997.

[ZQ15] Yongjun Zhang and Tingting Qi. Research on Mining Association Behavior of Smart Home Users Based on Apriori Algorithm. (Meita):817–821, 2015.