UNIVERSITÀ DEGLI STUDI DI PALERMO

Dottorato in Ingegneria Chimica, Gestionale, Informatica e Meccanica
Indirizzo Ingegneria Informatica
Dipartimento di Ingegneria Chimica, Gestionale, Informatica, Meccanica
Settore Scientifico Disciplinare ING-INF/05

STRUCTURAL KNOWLEDGE EXTRACTION AND REPRESENTATION IN SENSORY DATA

IL DOTTORE
PIETRO COTTONE

IL COORDINATORE
PROF. SALVATORE GAGLIO

IL TUTOR
PROF. SALVATORE GAGLIO

IL CO-TUTOR
ING. MARCO ORTOLANI

CICLO XXVI
ANNO CONSEGUIMENTO TITOLO 2016

UNIVERSITÀ DEGLI STUDI DI PALERMO

Dottorato in Ingegneria Chimica, Gestionale, Informatica e Meccanica
Indirizzo Ingegneria Informatica
Dipartimento di Ingegneria Chimica, Gestionale, Informatica, Meccanica
Settore Scientifico Disciplinare ING-INF/05

STRUCTURAL KNOWLEDGE EXTRACTION AND REPRESENTATION IN SENSORY DATA

IL DOTTORE
PIETRO COTTONE
Pietro Cottone

IL COORDINATORE
PROF. SALVATORE GAGLIO
[illegible]

IL TUTOR
PROF. SALVATORE GAGLIO
[illegible]

IL CO-TUTOR
ING. MARCO ORTOLANI
[illegible]

CICLO XXVI
ANNO CONSEGUIMENTO TITOLO 2016

Thesis advisor: Professor Salvatore Gaglio
Thesis co-advisor: Professor Marco Ortolani

Pietro Cottone

Structural Knowledge extraction and representation in sensory data

Abstract

During the last decades the availability of increasingly cheaper technology for pervasive monitoring has boosted the creation of systems able to automatically comprehend the events occurring in the monitored area, in order to plan a set of actions to bring the environment closer to the user's preferences. These systems must inevitably process a great amount of raw data – sensor measurements – and need to summarize them in a high-level representation to accomplish their tasks. An implicit requirement is the need to learn from experience, in order to be able to capture the hidden structure of the data, in terms of relations between its key components. The availability of large collections of data, however, has increased the awareness that "measuring" does not seamlessly translate into "understanding", and more data does not entail more knowledge. Scientific literature documents a massive use of Statistical Machine Learning in almost all data analysis and data mining applications, aiming at minimizing the need for a-priori knowledge. A remarkable drawback of such algorithms, however, is their failure to effortlessly provide insight about the most significant features of the data, as they typically just provide optimal parameter settings for a "black-box".

In this thesis, it is claimed that *structure* is the key to handle the complexity of acquiring knowledge from unstructured data in real-life scenarios. A shift in perspective will allow to tackle with the unaddressed goal of representing knowledge by means of the structure inferred from the collected samples; more specifically, the suggestion is to state this process within the framework of formal languages and automata borrowing concepts and methods from Algorithmic Learning Theory. In this context, knowledge extraction may be turned into structural pattern identification, letting syntactic models *emerge* from data itself.

In order to prove the soundness of this proposal, three different case studies will be presented, exploiting statistical learning, syntactical methods and formal languages, respectively. The third approach will be particularly useful to highlight the advantage of building intrinsically recursive models, which give multi-scale – more natural – representations; as a result, the computational burden that characterizes the huge volume of data will be lessened. Moreover, the task of designing reliable and efficient automatic systems for knowledge extraction can be alleviated by using such human-understandable models.

Acknowledgments

I wish to thank everyone who helped me complete this thesis: without their continued encouragement and support, this work would have never been finished and it probably would have never started.

I would like to thank my Ph.D. advisor, Professor Salvatore Gaglio, for his guidance during these past three years. I am also very grateful to my Ph.D. co-advisor, Professor Marco Ortolani, for his scientific advice and knowledge and many insightful discussions and suggestions: his support in writing this thesis was invaluable.

I wish to express my sincere gratitude to Professor Giuseppe Lo Re, director of Networking and Distributed Systems research group, for his help and advice during my Ph.D. course study.

The present and past members of the AI and NDS research group have contributed immensely to my personal and professional time during these years: Marco, Orazio, Gabriele, Antonio, and all the other people I had the pleasure to work with.

Last but not least, I will forever be thankful to my family and my girlfriend for their understanding and endless love: without them none of this would be possible.

Contents

Detailed contents

Listing of figures

Listing of tables

Acronyms

ADL *Activities of Daily Living*. 30

ALT *Algorithmic Learning Theory*. 64, 72

AMB *Activity Model Builder*. 37, 42, 54

AmI *Ambient Intelligence*. 2, 32

ANN *Artificial Neural Network*. 4

ART *Activity Recognizer and Tracker*. 37, 45, 54

BECM *Building Energy and Comfort Management*. 32, 33, 36

CARE *Context Awareness in Residence for Elders*. 50

CASAS *Center for Advanced Studies in Adaptive Systems*. 34, 50

CDA *Confirmatory Data Analysis*. 2, 3

DFA *Deterministic Finite Automaton*. 71, 72, 76, 79, 80, 83, 90, 91, 93

DHMM *Discrete Hidden Markov Model*. 16

DVSM *Discontinuous Varied-order Sequential Miner*. 42, 43, 44

EBG *Explanation-Based Generalization*. 4

EBL *Explanation-Based Learning*. 4

EDA *Exploratory Data Analisys*. 2, 3

EDO *Energy Demand Optimizer*. 37, 48, 49, 60

EDS *Energy Demand Simulator*. 37, 46, 47, 60

EDSM *Evidence-Driven State Merging*. 81, 91

ETS *Evolving Transformation System*. 5

FSA *Finite State Automaton*. 70, 71, 85

GI *Grammatical Inference*. 65, 72, 74

GPS *Global Positioning System*. 92

GUHA *General Unary Hypotheses Automaton*. 3

HMM *Hidden Markov Model*. 16, 18, 20, 21, 22, 24, 45

KARD *Kinect Activity Recognition Dataset*. 24

KP *Knapsack Optimization*. 48, 49

LDA *Linear Discriminant Analysis*. 17

LOOCV *Leave-One-Out Cross Validation*. 24

MDL *Minimum Description Length*. 23, 40, 41, 42, 51, 53

NFA *Nondeterministic Finite Automaton*. 71, 72

NFL *No Free Lunch*. 7, 8

NN *Nearest Neighbour*. 17

PCA *Principal Component Analysis*. 22, 23

PCFG *Probabilistic Context-Free Grammars*. 85

POI *Point of Interest*. 84

PTA *Prefix Tree Acceptor*. 76, 81

SAIL *String mAtching with wIldcards and Length constraints*. 40, 41

SHT *Statistical Hypothesis Testing*. 2

SLT *Statistical Learning Theory*. 4

STAVIS *Trajectory Analysis and VIsualization System*. 85

SVM *Support Vector Machine*. 4, 16, 18, 20, 22

TM *Turing Machine*. 9, 69, 71, 72, 74

UA *Universal Automaton*. 78

VANET *Vehicular ad-hoc Network*. 84

WSAN *Wireless Sensor and Actuator Network*. 1

WSN *Wireless Sensor Network*. 1

We are drowning in information but starved for knowledge.

John Naisbitt

1 Introduction

In recent years, the availability of an ever-increasing number of cheap and unobtrusive sensing devices has piqued the interest of the scientific community about the need of novel methods for automatic comprehension of the environment based on the collection of raw data measurements.

Wide-area sensor infrastructures, *Wireless Sensor Network* (WSN) and *Wireless Sensor and Actuator Network* (WSAN) bear massive volumes of data with diverse features [1], which need to be efficiently handled and processed to extract relevant information, in order to provide predictive insights and support users in controlling the monitored environments. This unprecedented requirement has given rise to a new research field crossing several areas, such as machine learning, pattern recognition, statistics, expert systems, data visualization and high performance computing [2]. Researchers of this new field work closely with domain experts in order to create reliable models deeply rooted into the raw data provided by the complex set of monitoring sensors.

The applications of knowledge discovery from sensory data are numerous, ranging from energy grid monitoring to disaster prevention. However, one of the most common use scenarios is *Ambient Intelligence* (AmI), a new paradigm in Artificial Intelligence that aims at exploiting the information about the environment state in order to personalize it, adapting the environment to user preferences [3]. The personalization process should be transpatent to the user, thus the intrinsic requirement of any AmI system is the presence of pervasive sensory devices.

The majority of traditional data mining and machine learning approaches are not directly suitable to deal with the new challenges in knowledge extraction and representation fostered by sensory data analysis. So, the attention of researchers has been nudged towards representations able to capture relationships in data, highlighting hidden structures, in order to acquire accurate and general models, easy to be transferred across similar scenarios, and less tied to the specific settings of the sensor set that produced the data.

In this thesis, a general framework to extract and represent structural knowledge is presented, along with several methods and approaches to implement it. Moreover, representative case studies related to the main applications of sensory data are provided, thoroughly investigating the main issues of each scenario, and proposing an effective solution based on structural knowledge.

1.1 Knowledge discovery

The need for coupling semantics with a sequence of sensor readings can be expressed in the framework of knowledge discovery, which is well-known in literature. Thus, in this section, main issues related to this research area and state-of-the-art approaches proposed in scientific literature will be presented. Inferring knowledge from data is an open issue in Computer Science, and in particular in data mining [4]. In this context, defining what can be deemed as *interesting* knowledge is a hard problem, because it implies to find out what can be interpreted as an important information.

Historically, a first debate on the most profitable way to extract useful information (i.e, knowledge) from a data collection was opened by John Tukey [5]. In the seventies, he proposed the *Exploratory Data Analisys* (EDA), as opposite to the *Confirmatory Data Analysis* (CDA) or *Statistical Hypothesis Testing* (SHT), that was the standard approach in those years. In the EDA approach, data is analyzed with different techniques to summarize its characteristics. Unlike CDA, Tukey suggests to let hypotheses emerge from data itself, rather than using data only to test a-priori hypotheses. EDA is just an approach, not a set of techniques, i.e. a suggestion about how data analysis should be carried out and what its goals should be. Most of the techniques inspired to EDA use the power of graphical representation to reveal the structure of the data to the analyst, offering new and often unexpected insights. In other words, EDA empowers the analyst's natural pattern-recognition capabilities and was the seminal work of modern approaches to data mining and pattern recognition.

One of the contemporary and independent by developed research on the track of EDA is the so-called *General Unary Hypotheses Automaton* (GUHA) [6]. The aims are to describe all assertions which may be hypotheses, to verify each of such assertions and to find the "interesting" ones, based on collected data. These techniques systematically generate all interesting hypotheses with respect to the given data (hypotheses describing relations among properties of objects) via a standard computer system, and therefore represent a first attempt to formalize an automatic inductive approach. Formal logic is used to formulate hypotheses, coded as association of properties. Each object is represented by a row in a rectangular matrix, whose columns are properties of the object. By analysing this data structure, it is possible to discover dependencies between different properties. The whole process is composed by three steps: *preprocessing*, *kernel* and *post-processing*. In the first step, matrix is arranged in a form suitable for a quick hypothesis generation. In the *kernel* phase, hypotheses are generated and evaluated, while in the last step hypotheses are analyzed in order to interpret them.

It is crucial to note that the problem of letting structure and explanation emerge from data itself and not from a-priori hypotheses was central since the beginning of data analysis history, and has gained more relevance over the years, due to the ever-increasing size and heterogeneity that have characterized the data to analyze. Nowadays, the collected sensory data make it impossible to promote a-priori hypotheses to describe events of interest. The discussion between EDA and CDA approaches has renewed in machine learning. In fact, two different approaches have grown in importance: *inductive* and *deductive* learning. This distinction reflects the differences and goals already underlined by Tukey, with a special focus of attention to the learning matter. The inductive approaches state the learning problem as finding a hypothesis that agrees with the examples, preferring the most simple one.

One of the most famous and prolific inductive theory is the *Statistical Learning Theory* (SLT) [7]; its main goal is to analyze the problem of the inference, providing a framework that assumes statistical assumptions about the data generated by a phenomenon. It includes a variety of algorithms, such as instance-based learning, *Support Vector Machine* (SVM), *Artificial Neural Network* (ANN), etc.

Each of these approaches stresses different aspects of learning problem, but they all relieve the analyst and designer from formulating an a-priori hypothesis about data. On the other hand, their results are not useful to increase the knowledge regarding a particular problem because they can be considered as a black-box that can be applied on unseen data, but the model of the data they use is not human interpretable.

The deductive learning approaches constitute the other class of machine learning algorithms. For example, a method to infer general concepts from examples is known as *Explanation-Based Generalization* (EBG) [8]. This deductive approach explains why a training example is a member of the concept being learned. It relies on four main components: a goal concept, training example, domain theory and operational criterion; explanations are represented by Horn-clause inference rules arranged in proof trees. The aim of the system is just to generalize concepts from examples, describing them through high-level properties, while training examples are represented in terms of lower level features. Generalization is achieved through the manipulation of the so-called *domain theory* through *operational criteria*. The domain theory is made up of a set of inference rules and axioms about the domain of interest and it is used to demonstrate the validity of the example, whereas operational criteria indicate how a concept must be expressed to be recognized. A slightly different approach is that proposed by [9]. In this case the system is not only able to generalize a concept, but to check where a generalization fails for a particular example, so that the system can refine it. Therefore it is possible not only to infer a general concept, but also to check whether an example is coherent with that generalization, or why it is not; in other words, the system is able to learn. This approach is called *Explanation-Based Learning* (EBL). An evolution of the EBL is proposed in [10]. This approach tries to merge the old EBL engine, based on symbolic knowledge representation, with the statistical approach. The proposed system aims to take advantage of the robustness of statistical approach respect to real word problems, but at the same time it exploits the expressive power of symbolic knowledge representation.

An alternative approach to generalization uses formal languages, and is known as *syntactic pattern recognition* [11]. In these systems, concepts are decomposed into simpler parts and their description relies on a grammar. The problem of inferring knowledge is stated as the problem of design a learning machine for pattern recognition, where a pattern is a particular structure included into the grammar. The system infers a grammar from training examples and applies it on the new data, in order to verify if the string of terminal symbols belongs to the learned grammar. This kind of approach requires preliminary work by the designer in ontology domain definition, in order to identify the key elements of the representation. The major drawback with this methods is the high computational cost needed to infer grammars. Historically, these approaches have been considered as alternatives to statistical learning systems, but during the last decades many efforts have been made to unify statistical and syntactic pattern recognition (see [12]).

Other authors consider traditional approaches inadequate to cope with the complexity of managing knowledge and its evolution in complex phenomena. However, they believe that these scenarios cannot be modeled only by mathematical or statistical means. For example, *Evolving Transformation System* (ETS) is a formalism that tries to unify the syntactic and statistical pattern recognition, in order to create a new kind of class representation. The definition of *class*, according to the author, rests on the generative side: objects belonging to the same class share similar generative histories. In this context, a generative system is a nondeterministic system operating on actual entities and assembling them into larger entities (and eventually into class objects), guided by some hierarchical description of the class [13]. This kind of representation is focused on the problem of giving a structural representation to the data. Each object in this formalism is thought of as a temporal structural process and the representation of each element of a class evolve with the description of the class itself. ETS is a work in progress framework, limited by the lack of new mathematical instruments to deal with the complexity of a structural description.

In [14], Chazelle proposes a new vision to deal with phenomena arising from life sciences, stating that means used in physical science are not adequate. According to his work, algorithms are more suitable for these purposes, due to their rich and expressive language. Moreover, the author claims that some problems can take an enormous advantage from the novelties introduced by a new perspective, taking into account the peculiarities of complex non physical systems. In the case of sensory data used to investigate and predict human habits and behaviour, the complexity is very high, because of the high number of variables to include in the model. Chazelle introduces the *natural algorithms* to model these systems. This approach relies on the so-called *influence systems*, i.e. networks of agents that perpetually rewire themselves. These networks are specified by two functions: f and G; the function f calculates the position of an agent, taking as input the location of its neighbour agent, given by function G. The output of G is function of the state of the whole system, that is the position of all agents. In this approach, it is possible to note how the information travels through the system, in a way that separates its syntactic or structural component and its semantic. In other words, this method models complex systems exploiting equally qualitative and structural information.

1.2 Motivations and Goals

One main issue motivated the work described in this dissertation: how to handle the huge complexity implied in sensory data.

The main aim of a knowledge discovery system is to find regularities in data produced by a phenomenon, in order to obtain a model that can predict future data belonging to the same phenomenon.

In other words, the model is an abstraction in terms of features coming from data. There are many ways to achieve this goal, as the previous section has shown; however, all of them share a common feature: they all need the injection of an amount of a-priori knowledge about the phenomenon generating the data.

This is a very important point in the design of a knowledge-related system. The *No Free Lunch* (NFL) theorem states that [15]:

Theorem 1. *For all possible performance measure, no search algorithm is better than another when its performance is averaged over all possible discrete functions.*

It is a modern version of what Hume pointed out: *"Even after the observation of the frequent or constant conjunction of objects, we have no reason to draw any inference concerning any object beyond those of which we have had experience".* Indeed, all the algorithms perform well on a random selection of a sample set.

In other words, there is not a single best solution, suitable for all application scenarios and for all observations. During the last years, the research in machine learning has rapidly evolved towards unsupervised methods, that can be tuned by a limited set of parameters, giving general purpose algorithm to classify and recognize data. Thus, the knowledge of the designer should be coded in terms of feature selection and parameter tuning, in order to identify the best system, according to some metrics.

In [16], the generalization ability of an inference method is defined as:

$$Generalization = Data + Knowledge,$$

pointing out that a-priori knowledge is essential for the aim of generalization.

In the sensory data scenario, this process is very complex, because often a little or no a-priori assumptions can be pointed out about the nature of the data produced by a phenomenon; moreover, it often turns out that the traditional assumptions, which are at the basis of a plethora of machine learning approaches, are far from being preserved in the complex scenario of sensory data. Even the selection of an adequate set of features which effectively describes data is a very hard task.

For example, in the next chapters, the activity recognition problem from simple sensor readings will be presented; thinking about an a-priori model of an activity in terms of sensor readings (sensor activation, item sensor activations, noise level, etc) is close to impossible, and the task gets worse if a general enough model is required, that is able to deal with slightly different sensor sets or configurations.

Clearly, the best algorithm can be chosen by measuring the performance on training data; but the NFL theorem states that, without a restriction on the set of candidates, based on the possible phenomenon expected, it is probable to only get an overfitted algorithm.

Hence, in this thesis, the need to keep the designer in the loop of knowledge extraction is claimed; this is translated into the demand for methods and approaches that can deal with huge complexity in sensory data, providing human understandable models, to correctly encode the useful a-priori knowledge into the process and to give handy insights into nature of observed phenomenon, in terms of relevant data features.

Many approaches have attempted to deal with this complexity. In particular, many systems have been proposed in the area of Ambient Intelligence, which typically deals with sensor readings and their interpretation. For example, in [17], the authors suggest a three-tier paradigm for knowledge extraction. In particular, this paradigm cuts irrelevant details off from raw sensor readings, in order to obtain more refined data that can be analyzed by the reasoning module, at the top of this processing hierarchy. Methodologies borrowed from Statistical Learning Theory are used in [18] to cope with the complexity of large sensor reading dataset. According to the authors, user habits are coded into sensor readings, thus they can be inferred by analyzing sensory data and discovering relations between environmental conditions and user.

In this thesis an approach similar to knowledge extraction is proposed, but, focusing mainly on the structure of knowledge itself. Hierarchical or recursive models of knowledge can be the key to handle the issue previously presented, due to their divide-et-impera approach, able to limit the complexity at each level of representation.

Undoubtedly, this goal is very challenging, and many issues are to be addressed; some of those are related to theoretical open issues in computer science, so it is impossible to known if they are practically solvable. The basic idea is that data collected from sensors share an *underlying language*, i.e., it can be considered as generated by a particular language describing some phenomena. Similarly to what described in [19], it can be assumed that data are drawn from a process that can be modeled by a *Turing Machine* (TM). This means, according to Chomsky, that there is a language that can describe such data. Likely, this language is very complex and, moreover, data is corrupted by noise, so reconstructing the original language from data is a very challenging task.

The proposed framework is inspired to systems that extract knowledge from text corpora. Obviously, several changes have to be done in order to adapt these approaches to the context of sensory data. However, the general structure of the process remains the same. The framework presented in this thesis is composed by four steps:

1. individuating a set of basic properties (axioms) and features to discovery significant patterns;
2. discovery of relevant elementary patterns as terminology;
3. abstraction of patterns as concepts;
4. inferring hierarchical concept organisation;

Setting axioms In the first step, the key elements of the hierarchy are defined; it is the only phase of the system that requires human intervention. The analyst has to specify a description of the goal concepts in terms of general properties, like time, space or other very general features. The main difference from others approaches is how these features are described. For example, consider the user activity recognition task. A user activity may be defined as *a recurrent sequence of actions, that can be recursive decomposed in simpler subtasks*. Two approaches can translate this definition in features on data: the deductive, and the inductive one. According to the deductive approach, the definition is transformed into an abstract and general model, that hypothesises sensor reading interactions that identify executions of the same activity. In the inductive approach, the analyst translates the definition in terms of properties the sensors can measure, such as time duration; so, an activity is treated as a recurrent pattern in data, whose instances have similar structures and time durations. No attempts at generating a general activity model are made, but the model will emerge from data. The feature selection depends on the experience of the analyst, but it is a simpler task than the other approaches. Moreover, it is less prone to error and axioms chosen can be simpler checked.

Discovering terminology The second step faces the problem of finding data with the properties defined at the previous step. Several techniques can aid the most significant patterns to emerge, identifying the basic level of the hierarchy with the smallest, but more recurrent patterns in sensory data. A-priori based algorithms are an example of these techniques.

As an alternative, this step can be considered as a data fusion, i.e. the system associates data coming from different source and of different types. It is a very common approach (e.g., [3], [20]) and it is useful to process data in a multi-sensor context, in order to exploit relations between different sensor triggers.

Pattern abstraction Pattern abstraction allows to obtain a generalization from the instances of patterns present in data. The system addresses the problem of *synonyms*, grouping similar instances of the same patterns. At the end of this step, the system will be able to associate a model to each pattern and a classifier can be trained to recognize it.

Inferring concept hierarchy In this step, the system finds recurrent terminology structures. Several methods can be applied to recover the structure behind data. In this thesis, relationships have been modeled through Markovian models or with the use of grammars.

1.3 Contributions

The hierarchy presented in the previous section has been implemented with three different approaches, in the context of different application scenarios, representing very common goals in knowledge discovery from sensory data: namely, activity recognition, energy demand optimization and mobility model extraction. The selected application scenarios share some commons features and issues, such as heterogeneity in data and large amount of raw sensor readings.

The main contributions presented in this dissertation can be summarized as follows:

- Chapter 2 describes how the framework can be applied using tools by Statistical Learning; in other words, the identification and representation of the elements of the proposed approach are carried out through algorithms belonging to the Statistical Machine Learning. The chapter describes a system for recognizing human activities by exploiting the information encompassed in depth images acquired by the Kinect sensor. Activities are modeled as sentences built up from a posture vocabulary, extracted by classical Machine Learning algorithms. In particular, a clustering algorithm is used to highlight main postures revealed by the depth maps and a classifier is employed to generalize the models provided by clustering. The structure of an action is encoded by a probabilistic model that considers the sequence of postures in the action.
- In Chapter 3, structural knowledge extraction is handled by means of a syntactic approach, that takes advantages of classical tools of Data Mining and Machine Learning to recover the hidden structure behind the raw data. The design and implementation of a system for energy demand optimization taking advantage of the prediction of user activities is presented. The application scenario is that of activity discovery and recognition in smart homes, using a pervasive sensor network, made up by very simple sensors. A specific language was devised to describe activities and its main constituents emerge directly from data, without any predefined model, exploiting recursive nature of activities. The main idea is that, despite this variableness, a resilient recursive structure persists even if an activity is carried out from different subjects. Algorithms borrowed

from data mining are used to recover this hidden structure.

- Chapter 4 outlines how grammatical inference can be used to accomplish the task of structural knowledge extraction, completely exploiting the recursive nature hindered in data. In this chapter, a system for extracting user mobility models from fine-grained localization data is presented. The system is based on the idea of representing user mobility models with the use of formal languages, letting models themselves emerge from data and obtaining an actual language describing user mobility habits and behaviour at different scales. In particular, a set of regular languages are inferred from raw data, one for each level of spatial granularity, covering different levels of mobility behavior (from neighborhood to wide-area paths).

There is an increasing level of difficulty in the analysis of data of the three scenarios. The first one can be considered as a massive data scenario, but the nature of data is simpler to understand, thus feature extraction is easier than in the following applications. Postures are a very effective and intuitive model to express actions, so, the models coming from this representation can be easier to build.

The second application is strictly tied with the unstructured representation of raw data. In this application, relationships embedded in data are very counterintuitive, thus, the hierarchical approach allows to decompose the problem of activity models into simpler ones, that can be solved by mining frequent paths into data.

The third application is very hard to solve, due to the quantity of data. The dataset used is considered a big data source, thus traditional algorithms can not be applied in order to mine relevant patterns. Thus, the proposed multi-scale representation is an effective way to obtain some insights regarding the main data features.

1.4 Publications

Several parts of this dissertation have been published in various referred conference proceedings, journals and book chapters:

Journal articles

- P. Cottone, S. Gaglio, G. Lo Re, and M. Ortolani, "A machine learning approach for user localization exploiting connectivity data," *Engineering Applications of Artificial Intelligence*, 2016, accepted for publication
- P. Cottone, S. Gaglio, G. Lo Re, and M. Ortolani, "User activity recognition for energy saving in smart homes," *Pervasive and Mobile Computing*, vol. 16, Part A, pp. 156–170, 2015, DOI: `10.1016/j.pmcj.2014.08.006`

Book chapters

- P. Cottone, S. Gaglio, and M. Ortolani, "A structural approach to infer recurrent relations in data," in *Advances onto the Internet of Things*, ser. Advances in Intelligent Systems and Computing, S. Gaglio and G. Lo Re, Eds., vol. 260, Springer International Publishing, 2014, pp. 105–119, DOI: `10.1007/978-3-319-03992-3_8`

Conference proceedings

- P. Cottone, G. Maida, and M. Morana, "User activity recognition via Kinect in an ambient intelligence scenario," International Conference on Applied Computing, Computer Science, and Computer Engineering (ICACC 2013), vol. 7, 2014, pp. 49–54, DOI: `10.1016/j.ieri.2014.08.009`
- P. Cottone, S. Gaglio, G. Lo Re, and M. Ortolani, "User activity recognition for energy saving in smart homes," in *Sustainable Internet and ICT for Sustainability (SustainIT), 2013*, 2013, pp. 1–9, DOI: `10.1109/SustainIT.2013.6685196`
- P. Cottone, G. Lo Re, G. Maida, and M. Morana, "Motion sensors for activity recognition in an ambient-intelligence scenario," in *IEEE International Conference on Pervasive*

Computing and Communications Workshops (PERCOM Workshops),, 2013, pp. 646–651, DOI: `10.1109/PerComW.2013.6529573`

Talks

– P. Cottone, S. Gaglio, and M. Ortolani, *Grammatical inference for structural knowledge extraction*, Talk presented at the 4th Italian Workshop on Machine Learning and Data Mining (MLDM15.it), Palermo, Sep. 2015

– P. Cottone, S. Gaglio, and M. Ortolani, *Exploiting mobility models for sustainable urban transportation*, Talk presented at I-CiTies 2015 - CINI Annual Workshop on ICT for Smart Cities & Communities, Ferrara, Oct. 2015

Computers are useless. They can only give you answers.

Pablo Picasso

2 Statistical Learning for Activity Recognition by Postures

Knowledge extraction via Statistical Learning is the main topic of this chapter. In particular, it is shown how Statistical Learning can be used in order to identify the most important relations embedded in the raw data and then to model them.

The chosen application scenario is the problem of human activities recognition by exploiting depth images provided by Microsoft Kinect sensor [29].

Several issues have to be solved in order to get a reliable system:

- activity models should be invariant to people's different features (height, silhouette, etc);

- activities can be performed at different speeds;
- a comprehensive and representative set of predefined models can hardly be created only from a-priori information.

In this chapter, it is shown that activities can be modeled as sentences built up from a posture vocabulary. The main idea is to let this vocabulary emerge directly from data, without any predefined model. Postures are obtained as the most frequent configurations of the main joints of the human skeleton.

In order to preserve the pervasiveness of the system, the motion detection sensor provided by Kinect is coherently connected to a miniature fanless computer with reduced computation capabilities.

2.1 Human action recognition

During the last years, the issue of human action recognition has been addressed in several works.

In [30], the authors use a set of binary silhouettes as input of a framework based on *Hidden Markov Model* (HMM). An activity is described as a sequence of the poses of the person. The silhouettes are extracted from video images, thus this method lacks of flexibility since it requires a number of image processing steps (e.g., background removal, vector quantization, image normalization).

Two different recognition systems based on Silhouette features and *Discrete Hidden Markov Model* (DHMM) are presented in [31], [32]. The authors of [31] use Fourier shape descriptors, while in [32] the features are obtained by combining RGB and depth information. In both works, features classification is performed by SVM and the classified postures are considered as the discrete symbols emitted from the hidden states.

Several works [33], [34] address the problem of activity recognition by using intrusive sensors, e.g., wearable sensors. The release of the Kinect sensor allowed researchers to perform activity recognition in a unobtrusive way, i.e., by using depth and RGB information.

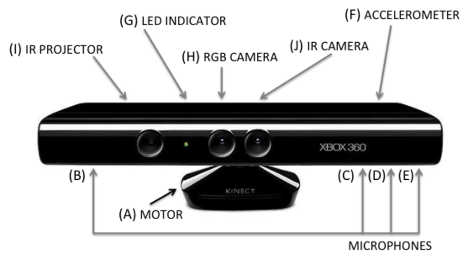


Figure 2.1: Kinect components.

In [35], salient postures are characterized as a bag of 3D points obtained from the depth map. Such postures represent the nodes in an activity graph that is used to model the dynamics of the activities.

A model for human actions called Actionlet Ensemble Model is presented in [36]. Human bodies are considered as a large number of kinematic joints and actions are characterized by the interaction of a subset of these joints. The authors introduced the concept of Actionlet as a particular conjunction of the features for a subset of joints. As there is an enormous number of possible Actionlets, a data mining approach is applied to discover the discriminative Actionlets. Then an action is represented as an Actionlet Ensemble, which is a linear combination of the Actionlets.

A supervised algorithm that use a dictionary of labeled hand gestures is presented in [37]. The authors use Kinect SDK to extract a sequence of skeleton-model parameters that represents the feature space. The covariance matrix of this space is used to discriminate the gestures and action recognition is performed by a *Nearest Neighbour* (NN) classifier.

A histogram based representation of human postures is presented in [38]. In this representation, the 3D space is partitioned into n bins using a spherical coordinate system. The authors built a model of human postures on 12 selected joints. Each joint position belongs to a bin with a certain level of uncertainty. The set of the vectors from the training sequences are reprojected using *Linear Discriminant Analysis* (LDA) and clustered into a K-postures vocabulary. The activities are represented as sequences of postures in the vocabulary and are recognized using HMM classifiers.

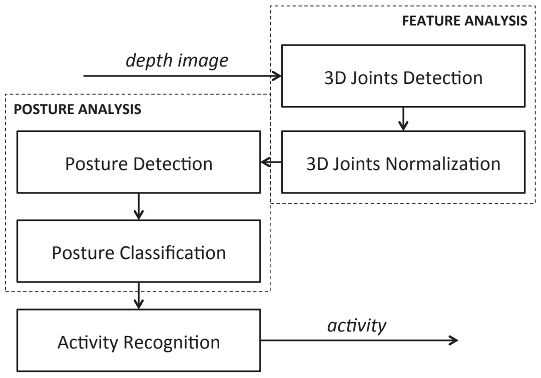


Figure 2.2: Activity recognition via depth images: an overview of the whole system.

2.2 Activity recognition via depth images

According to the considered scenario, Kinect represents the most suitable device both in terms of cost and functionalities since it is equipped with ten input/output components (see Fig. 2.1) that make it possible to sense the users and their interaction with the surrounding environment [29]. The Kinect sensor rests upon a base which contains a motor (Fig. 2.1-**A**) that allows for controlling the tilt angle of the cameras (30 degrees up or down). Three adjacent microphones are placed on the bottom of the device, in the right side (Fig. 2.1-**C**-**D**-**E**), while a fourth microphone is positioned on the left side (Fig. 2.1-**B**). A 3- axis accelerometer (Fig. 2.1-**F**) can be used for measuring the position of the sensor, while a led indicator (Fig. 2.1-**G**) shows its state. However, the core of the Kinect is represented by the vision system composed of: an RGB camera (Fig. 2.1-**H**) with VGA standard resolution (i.e., 640x480 pixels); an IR (Fig. 2.1-**I**) projector that shines a grid of infrared dots over the scene; an IR (Fig. 2.1-**J**) camera that captures the infrared light. Thanks to the factory calibration of the Kinect, it is possible to know the exact position of each projected dot against a surface at a known distance from the camera; this information is used to create depth images [*] of the observed scene that capture the object position in a three-dimensional space.

The system proposed in this chapter (see Fig. 2.2) aims at automatically inferring the activity

[*] A depth image is an image whose pixel values represent distances.

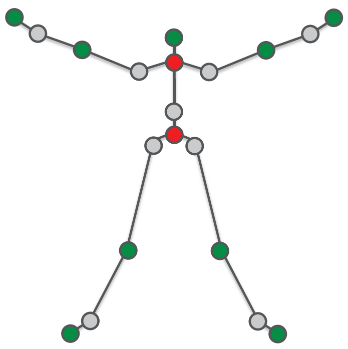


Figure 2.3: The 20 joints of the human body. Reference joints (red): *neck*, *hip center*. Selected joints (green): *head*, *elbows*, *hands*, *knees*, *feet*. Discarded joints (grey): *shoulders*, *wrists*, *spine*, *hips*, *ankles*.

performed by the user according to a set of known postures. Each posture is defined by the position of some body joints extracted by means of the OpenNI/NITE skeleton detection method. The set of detected joints is clustered by applying the K-Means algorithm in order to build a vocabulary of postures. The obtained "words" are validated by SVMs. Finally, HMMs are applied to model each activity as a sequence of vocabulary words.

2.2.1 Features Analysis

The OpenNI/NITE skeleton detection method performs real-time detection (i.e., to find the 3D coordinates) of 20 body joints (see Fig. 2.3). However, due to the sensitiveness of the IR sensor, some overlaying detected joints (e.g., hands touching other body parts) or occlusions (e.g., objects placed between the sensor and the user) may lead to significant errors.

For this reason, some redundant joints (i.e., wrists, ankles) have been discarded due to their closeness to other selected joints (i.e., hands, feet), while others (i.e., spine, neck, hip and shoulders) are not relevant for activity recognition. The selected joints are shown in green in Fig. 2.3, while the discarded ones in grey.

Moreover, since the distance of the skeleton joints from the hip depends on several factors (e.g., the users height, arm length, distance from the sensor), all feature vectors have been normalized according to the distance between the neck and hip center joints. A scale-independent representation of the body posture is then obtained by fixing the center of the reference coordinate system at the hip center and considering as x-direction the left-right hip axis. Reference joints are shown in red in Fig. 2.3.

2.2.2 Postures Analysis

Once the joints have been detected, a clustering algorithm is applied to quantize the number of observed joints configurations. Thus, the detected features are clustered into K classes (i.e., building a K-words vocabulary) by using the K-means algorithm. Each posture is then represented as a single word of the vocabulary and therefore each activity can be considered as an ordered sequence of vocabulary words.

In order to obtain a better statistical description of the content of each cluster, the output (i.e., the pairs features/cluster) of the K-means algorithm is used to train a multi-class SVM. SVMs are supervised learning models used for binary classification and regression. A multi-class SVM is a net of SVMs able to perform a multi-class classification [39].

Moreover, sequences of joints configurations are turned into the corresponding sequence of K-words, only postures transitions are considered: all repeated sequences of the same posture are merged. Thus, a more compact representation of the sequences is obtained, mitigating the problem of recognizing executions of the same activity performed with different time durations.

2.2.3 Activity Recognition

The issue of recognizing different sequences of postures referred to the same activity is addressed by means of a probabilistic approach. In particular, each action is modeled using a discrete HMM [40].

A HMM that has N states $S = \{S_1, S_2, \ldots, S_N\}$ and M output symbols $V = \{v_1, v_2, \ldots, v_M\}$ is fully specified by the triplet $\lambda = \{A, B, \pi\}$. The state transition probability distribution $A = \{a_{i,j}\}$ is

$$a_{i,j} = P\left[q_{t+1} = S_j | q_t = S_i\right], \quad 1 \leq i, j \leq N \tag{2.1}$$

where q_t is the actual state at time t.

The observation symbol probability distribution in state j, $B = \{b_j(k)\}$ is

$$b_j(k) = P\left[v_k \texttt{ at } t | q_t = Sj\right], \tag{2.2}$$

where $1 \leq j \leq N$ and $1 \leq k \leq M$.

And the initial state distribution $\pi = \{\pi_i\}$ is

$$\pi_i = P\left[q_1 = S_i\right], \quad 1 \leq i \leq N \tag{2.3}$$

Once each HMM has been trained on the posture sequences of each activity, a new (unknown) sequence is tested against the set of HMMs and classified according to the largest posterior probability, if such a probability rises above a prefixed threshold.

2.3 Experimental assessment: accuracy of the activity recognition

Activity recognition accuracy has been evaluated on the public MSR Action3D dataset [35] containing 20 actions: *high arm wave, horizontal arm wave, hammer, hand catch, forward punch, high throw, draw x, draw tick, draw circle, hand clap, two hand wave, side-boxing, bend, forward kick, side kick, jogging, tennis swing, tennis serve, golf swing and pickup & throw.* Every action is repeated 3 times by 10 different subjects.

During the training phase, it was noticed that the skeleton tracker heavily failed in correspondence of some particular actions or subjects, as reported by the authors of the dataset[†]. For this reason, the *"bend"* and *"side kick"* actions and the subject 4 have been removed. Thus, "filtered" dataset is reduced to 18 actions performed by 9 subjects.

[†]http://research.microsoft.com/~zliu/ActionRecoRsrc

Activity Set 1	Activity Set 2	Activity Set 3
Horizontal arm wave	High arm wave	High throw
Hammer	Hand catch	Forward Kick
Forward punch	Draw x	Jogging
High throw	Draw tick	Tennis swing
Hand Clap	Draw Circle	Tennis serve
Tennis serve	Two hand wave	Golf swing
Pickup & throw	Side boxing	Pickup & throw

Table 2.1: The three Activity Sets.

Three Activity Sets (ASs) have been obtained from the filtered dataset in similar way as done by [35] and [38]. Each Activity Set contains 7 activities, as shown in Table 2.1.

Since a number of solutions based on the SVM-HMM chain are presented in literature, each processing module has been individually tested to estimate its effect on the overall performance. For this reason, four different configurations have been used to assess the whole system:

1. *NONE configuration*: posture analysis is performed by applying only the K-means algorithm;
2. *PCA configuration*: a *Principal Component Analysis* (PCA) transformation on original data (i.e., joints positions) has been added to the feature analysis process in order to evaluate the impact of a reduced feature space on the system performance;
3. *SVM configuration*: posture classification is performed by means of a multi-class SVM classifier based on a RBF kernel with $\gamma = 1/n_f$ and regularization parameter $C = 1$, where n_f is the number of features considered;
4. *SVM_PCA configuration*: both PCA and SVM are employed.

The number of posture clusters (K) and HMM states (N) were obtained through a Grid Search [41] in the range $[10; 100]$ for K and $[3; 8]$ for N. For every node of the grid, the error of Leave One Out Cross Validation [42] was computed. For each of the three Activity Set, 188 action sequences

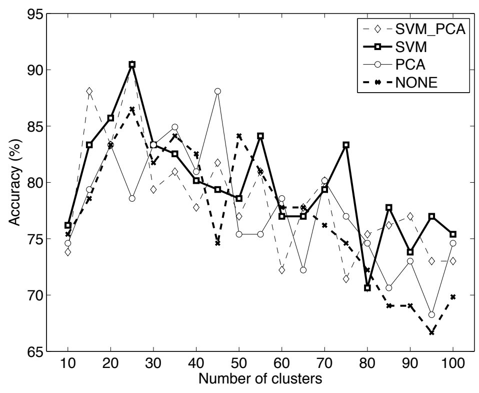


Figure 2.4: Comparison of the mean accuracy for the three Activity Sets according to the four system configurations.

Configuration	(K,N)	Accuracy
NONE	(25,4)	86.50%
PCA	(45,7)	88.09%
SVM	(25,5)	90.47%
PCA_SVM	(25,10)	90.47%

Table 2.2: Best mean accuracy obtained for each configuration.

were used for training and the remaining sequence was used for validation; each test set was repeated 10 times for every configuration. As a result of these experiments, the pair (K, N) minimizing the mean error on the three Activity Sets was chosen.

The results obtained for the best (K, N) pairs of each configuration are reported in Table 2.2. The reduction of the feature space, obtained by applying PCA on original data, decreased the system performances. This result is motivated by the preliminary selection of joints, demonstrating that original feature space does not contains correlated features.

A comparison of the accuracy measured with respect to the number of clusters is shown in Fig. 2.4. The best performances are obtained by *SVM* and *SVM_PCA*, both giving an overall mean accuracy of 90.47%. However, the results obtained by the *SVM configuration* showed a smaller variance, demonstrating that the former is preferable.

Such a result is confirmed by comparing *SVM* and *SVM_PCA* on different values of K, as showed

Action	Accuracy	Action	Accuracy
Horizontal arm wave	100%	Hand catch	71%
Hammer	100%	Two hand wave	100%
Forward punch	100%	Draw x	68%
Golf swing	83%	Draw tick	100%
Hand Clap	95%	Draw Circle	68%
Tennis serve	92%	High arm wave	83%
Pickup & throw	100%	Side boxing	83%
High throw	84%	Forward Kick	100%
Jogging	100%	Tennis swing	100%
Mean Accuracy 90.4%			

Table 2.3: Recognition rate of SVM system configuration

in Fig. 2.5. Moreover, according to the *Minimum Description Length* (MDL) principle [43], the model given by *SVM* is better than the *SVM_PCA* one, since the former use a smaller number of states (i.e., $N = 5$ versus $N = 10$) as shown in Table 2.2.

In table 2.3 are reported the mean accuracy values obtained by the *SVM configuration* for the whole set of considered activities.

The confusion matrices reported in Table 2.4 - 2.5 - 2.6 show classification errors related to the three activity datasets listed in Table 2.1. Please note that some activities are not correctly classified since they are considered as parts of more complex ones (e.g., *Hand catch* gesture is the beginning of *High arm wave*, *Draw tick* and *Two hand wave*).

Since the quality of existing public datasets is often poor, a new dataset was collected; it contains 8 activities (*Catch Cap, Toss Paper, Take Umbrella, Walk, Phone Call, Drink, Sit down, Stand up*), each performed 3 times by 10 different subjects. This dataset is an earlier version of *Kinect Activity Recognition Dataset* (KARD), described in [44]. Several tests have been performed on the 240 captured sequences to verify the accuracy and the robustness of the activity recognizer.

In particular, the *SVM* configuration was used for the system. The experimental tests started by

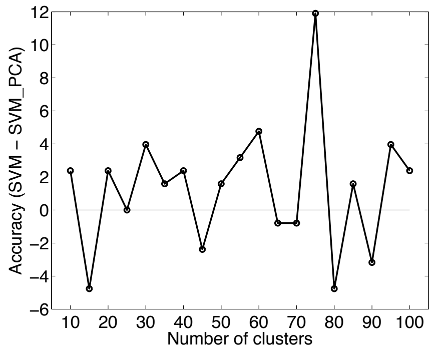


Figure 2.5: Difference of accuracy between the proposed system configuration *SVM* and *SVM_PCA*.

	1	2	3	4	5	6	7
1	100	-	-	-	-	-	-
2	-	100	-	-	5	10	-
3	-	-	100	-	-	-	-
4	-	-	-	84	-	-	-
5	-	-	-	16	95	-	-
6	-	-	-	-	-	90	-
7	-	-	-	-	-	-	100
8	-	-	-	-	-	-	-

Table 2.4: Confusion matrix of Activity Set 1. (1) Horizontal arm wave, (2) Hammer, (3) Forward punch, (4) High throw, (5) Hand Clap, (6) Tennis serve, (7) Pickup & throw, (8) Unknown.

applying a Grid Search approach to find out the best couple of values for the number of clusters K (i.e., the number of postures) and the number of the HMM states N. The value of each node of the grid has been computed as the mean rate of a *Leave-One-Out Cross Validation* (LOOCV) repeated ten times to overcome the randomness of the clustering algorithm. The best recognition rate is obtained with K = 39 and N = 5, with a mean accuracy of 95% and standard deviation of 2.45 between the different runs of the LOOCV.

Motivated by the results obtained over the whole dataset, the influence of choice of the training set on performances was investigated. For this reason, the whole dataset is divided into subsets and each subset is tested three times in a way similar to the one proposed in [35]:

	1	2	3	4	5	6	7
1	83	9	-	-	-	-	-
2	-	71	-	-	-	-	-
3	-	-	68	-	-	-	7
4	-	11	-	100	12	-	-
5	-	-	32	-	68	-	-
6	17	9	-	-	10	100	-
7	-	-	-	-	-	-	83
8	-	-	-	-	10	-	10

Table 2.5: Confusion matrix of Activity Set 2. (1) High arm wave (2) Hand catch, (3) Draw x, (4) Draw tick, (5) Draw Circle, (6) Two hand wave, (7) Side boxing, (8) Unknown.

	1	2	3	4	5	6	7
1	84	-	-	-	-	-	-
2	-	100	-	-	-	-	-
3	-	-	100	-	-	-	-
4	-	-	-	100	-	-	-
5	-	-	-	-	94	-	-
6	-	-	-	-	-	83	-
7	16	-	-	-	-	-	100
8	-	-	-	-	6	17	-

Table 2.6: Confusion matrix of Activity Set 3. (1) High throw (2) Forward Kick , (3) Jogging, (4) Tennis swing, (5) Tennis serve, (6) Golf swing, (7) Pickup & throw, (8) Unknown.

	Accuracy
1/3 Validation	93.75%
2/3 Validation	94.87%
Cross Subject Validation	90.98%

Table 2.7: Accuracy of the SVM configuration on our dataset.

- 1/3 Validation: 1/3 of the data captured for each subject is used for training, the remaining part is used for testing;
- 2/3 Validation: 2/3 of the data captured for each subject is used for training, the remaining part is used for testing;
- Cross Subject Validation: 1/2 of the subjects is used for training and the remaining part for testing.

Each of the above tests was repeated ten times, randomly choosing the sequences or subjects of the training and testing sets. The results of the three performed tests are shown in 2.7. The first two rows report accuracy values of 93.75% and 94.87% respectively, which are comparable to the mean accuracy of 95% obtained over the whole dataset. The most significant result is the one obtained by the cross subject test (bottom row) that aimed to measure the ability of the system in recognizing activities performed by new subjects. The achieved recognition rate of about 91% shows that the method proposed is able to capture a general model of the activity regardless to the user that performed it.

Non est ad astra mollis e terris via.

Lucius Annaeus Seneca

3 Syntactic Methods for Optimization of Energy Demand

Syntactic approaches have been widely investigated and used to analyze symbolic data. In this chapter, an application of syntactic method is presented, aiming at recognizing daily life activities performed by users in a smart home in order to minimize energy consumption by guaranteeing that peak demands do not exceed a given threshold.

The main idea behind this approach is to relieve the designer from the task of creating a detailed model for each activity to track, so, unlike previous proposals, this problem is addressed from an *algorithmic* perspective, rather than a *learning* one. A general high-level description of what may be regarded as an *activity* is all it is required, thus bypassing the difficulty of creating a reliable model of an activity in terms of sensory triggers or supposed interactions between users and their home appliances.

The resulting system should be able to work without an explicit human intervention, so a specific challenge is related to the *language* used to obtain a high-level, generalizable description of human behaviour, using only data coming from the measurements of a typical environmental sensor network and no specific knowledge of the particular sensor set.

Activities models are defined as *recursive structures* and identified by extracting relevant *events*, which, in this context, may be thought of as short and recurrent sequences of raw sensor readings. Hence, the designer is not forced to embed too specific knowledge into the system and may rather choose a description of events in terms of simple basic concepts, such as time duration or type of sensor measurement. It can be implicitly assumed that "ground" events are characterized by a short duration, and will directly correspond to readings; most of them will likely be not very meaningful for characterizing user activities, and their information content will not be apparent unless they are considered in a combination with other ground events, thus having the hidden structure of the activity progressively emerge.

Main focus has thus been on adaptiveness, and efforts have been specifically directed toward learning and prediction of user activities, as a first step towards an effective approach to energy saving.

3.1 Energy saving and user habits

User habits play a central role in household energy demand, thus recognizing activities carried out by the user should be an important part of every system aiming at optimizing energy consumption.

3.1.1 Activity recognition by simple sensors

In literature, several works have addressed the problem of activity recognition. Common proposals include (i) methods based on the use of logic, (ii) probabilistic methods, or (iii) methods based on common sense reasoning. In the context of logic-based methods, activity recognition consists in reconstructing the plan an agent is following, based on the observation of its actions, and the main difficulty lies in the hypothesis of rationality, which often does not hold when the agent is human, especially in the presence of illness or disability. The authors of [45] use reticular theory and a logic language for describing actions in order to detect non-standard behaviors; their system can generate new plans and provide explanation for unusual actions. In [46], event calculus is used to recognize activities and support users in performing the correct action at the right place and time. The significant advantage of using a logic language, such as event calculus, is the possibility to embed a-priori knowledge about the application domain, which reduces the need for annotations and allows for easy interpretation of the produced rules; the drawback, however, is the inability to deal with ambiguity, which arises when the system fails at detecting the on-going activity and can not even estimate the most likely one.

Probabilistic methods regard the sequence of events as a time series, and the goal is to determine the chain of hidden states which generated the observations. The probabilistic approach requires computing the sequence which maximizes the probability of the hidden states, given a particular set of observations. Several methods, such as Semi-Hidden Markov Models, Skip Chain Conditional Random Fields, and many others, have been applied to address the issue of activity recognition, as reported in [47]–[49]. Probabilistic methods require the availability of a large amount of labeled data in order to show acceptable performance; the need for annotation may be partially mitigated by hard-coding knowledge about how activities are typically carried on, e.g. by extracting it from the Web. In [50], for instance, a system whose purpose is to create a database of bits of common-sense knowledge is proposed; such data may be integrated in automated systems in order to augment their ability of interacting with the real world. Translation of sensory data into high-level abstractions is made by merging knowledge with information from the Web, and transforming the obtained data into clauses; the system then performs a statistical inference reasoning.

From a data mining perspective, activity discovery is often seen as the problem of detecting recurring patterns within a sequence of events; however, there are substantial differences between frequent itemsets detection, and discovery of patterns corresponding to activities. First of all, itemsets do not account for the ordering of the elements, which on the other hand is quite relevant during activity discovery; secondly, each itemset must not contain repetitions, whereas a pattern might do. In order to overcome such limitations, most proposals rely on the so-called *T-patterns* [51], and candidate itemsets are chosen according to criteria defining their meaningfulness within the event sequence. The authors of [52] use a variant of the *Apriori* algorithm [53] to discover sequences of events repeating with regular periodicity, besides patterns related to frequent activities. The system starts from elementary sequences and expands them to obtain longer ones, up to a maximum predefined size. Another approach, proposed in [54], relies on standard Apriori and considers the event sequence as a stream of incoming data; after identifying all sequences of predefined size and support, a transformation function maps them into models for activities. A hierarchical description is proposed for such models, and activities are divided into tasks and sub-tasks; the bottom of the hierarchy is represented by activities that cannot be further decomposed; activity recognition, as well as description, is carried on in a bottom-up fashion. A similar approach is described in [55], where the authors address the issue of broken or concurrent activities by considering *emerging patterns*, i.e. those patterns able to capture meaningful differences between two classes of data. Finally, an approach worth mentioning is proposed in [56], where *Activities of Daily Living* (ADL) are discovered by means of evolutionary techniques; the purpose is the creation of an Evolving ADL Library containing models for activities; the library evolves by learning additional models from new sequences.

3.1.2 Energy saving and BECM systems

The ever-increasing energy demand in recent years is becoming a major issue as it represents a possible drawback in our society's future development, where energy is arguably the single most valuable good. Current consumption trends are unsustainable from an environmental point of view, and efficient usage and overall energy demand reduction have become two major concerns of the international community and most governments, due to both economic and environmental motivations [57]. Namely, according to the classical market laws, those trends have caused a burst in energy price which eventually has attracted greater attention to the energy problem.

The periodical shortages in energy supply during the last century, led to the birth of new research areas, and considerable effort is being carried out to devise viable solutions to the energy issue, ranging from discovering new energy sources to raising people awareness. In this context, a steady attention has been devoted to energy saving in buildings, starting from the energy crises of the 1970s [58], [59].

User habits play a central role in household energy demand: an inefficient control of electric appliance and heating systems is a major energy waste source. Current literature about building automation, however, shows that building control is still mainly performed manually, as in the case of artificial lighting setting, powering appliances, or seasonal control of heating systems; additionally, automation in buildings has historically focused on narrow-scope tasks, such as lighting control with simple motion detection and a fixed timeout, or indoor climate control based on temperature and CO_2 level. On the other hand, user activities and behavior have a considerable impact on the amount of consumed energy in all kinds of buildings (i.e., residential, office, and retail sectors). Thus, the design of *Building Energy and Comfort Management* (BECM) [60] systems has grown to become a self-standing research area, in order to optimize energy use in home scenario. A significant amount of the energy dissipated in these areas can be saved by fine-tuning deployed devices and appliances according to actual user needs; for instance, many research efforts have been focused on proposing "smart thermostats" based on occupancy prediction, or on maximizing user comfort by providing appropriate artificial lighting, based on the activity carried on at a given moment.

This research area belongs to the greater field of AmI, however, while the general scope of AmI is to apply artificial intelligence techniques to transparently support users in their everyday activities, a BECM system can be defined more specifically as a control system that uses artificial intelligence and a distributed sensor network for monitoring a building in order to ensure efficient usage of the available energy sources. A system implementing this approach must be able to predict the users' course of actions, in order to cope with the issue of reducing energy consumption without negatively affecting the user experience. Keeping intrusiveness at a minimum is essential to promote this kind of systems and to allow acceptance by a broad target of users; in fact, their impact on energy consumption will be significant only if they are used at a large scale. Several studies (e.g. [61]) have shown that a user-centric optimization of energy consumption, with no perceivable effects on user comfort, can lead to significant energy saving. In other words, the primary goal of energy saving systems is to automatically adapt to user preferences; this suggested to follow the AmI paradigm, which requires minimizing user intervention, by "hiding" the system within the surrounding environment, while still enabling support to the users for their everyday-life activities.

Substantial research effort has been devoted to address the complex issues related to the design of a BECM system, and most proposals agree on the need for automated approaches to energy demand optimization; the presence of peaks in energy demand is often regarded as a symptom of a suboptimal scheduling of the use of electric appliances and the authors of [62]–[64], for instance, point out that even straightforward approaches, such as turning off unused devices, can be very effective in terms of energy saving. The challenging aspect of those proposals is their potential impact on user perception: if automated energy saving policies are so intrusive as to become a hindrance to the overall user experience, they might hardly be accepted from householders.

The key is to design a system capable of adapting to its users' needs is to correctly identify their activities. Several state-of-the-art proposals assume the availability of considerable *a priori* knowledge, which makes them often prone to overfitting. Results obtained by these systems depend on the particular features of the application scenario, and their activity models are fitted onto data, as opposed to "emerging" from data itself [20]; this may be a major issue, if the goal is the design of a fully adaptive and generalizable system. The system proposed in this chapter is partly inspired to the key ideas presented in [65] and [66]. The authors of [65], in particular, proceeding from a scenario characterized by scarcity of labeled data and uncertainty about activity granularity, showed that *formal grammars* are suitable to capture the inherent structure of activities. Their system, called Helix, initially generates a vocabulary combining unlabeled sensor readings, and attempts to incrementally merge them, by grouping similar activities into high-level ones. Grammar induction is used as a tool for heterogeneous sensor fusion in order to build up the structure of activities; each activity is regarded as a cluster in a multi dimensional space where the data streams coming from the different sensors present in the monitored area are represented; a hierarchical structure is then induced on this space, through statistical analysis. The authors of [66] focused on formalizing computational models for every-day human activities; they claim that global structural information about activities can be encoded by using a subset of their local event subsequences; hence, an activity is defined as a finite sequence of events, expressed in terms of the objects present in the observed environment, whose functionalities may be needed for the execution of a particular activity. An event is defined as a specific interaction between two or more objects in a finite duration of time, and a list of key objects for each environment needs to be provided as *a priori* knowledge. This approach does not need to rely on predefined activity models, whose creation is typically very challenging, rather it is pointed out that an analysis of continuous event subsequences suffices to discover and track every-day activities.

In order to test the effectiveness of activity recognition for energy saving, a data set including both power consumption and sensor measurement would be needed; however, despite the fact that data sets about activity recognition, as well as about power profiling have been independently collected, to the best of my knowledge none is available that encompasses both aspects. One of the data sets of the *Center for Advanced Studies in Adaptive Systems* (CASAS) project [67], for instance, contains readings from a power meter; however it provides information only about the overall consumption, which is not very useful in the context of activity recognition, where fine-grained energy monitoring is needed. Namely, aggregated information about energy consumption often leads to non optimal consumption control. Indeed, new systems have been developed to produce fine-grained energy reports, at an individual-device scale [68], although in the context of the new research area of "energy reporting", whose aim is that to guarantee a higher resolution in monitoring energy consumptions. In this context, a very promising data set, provided by the Smart* project [69], was collected by continuously gathering measurements from a wide range of sensors and meters placed in three different households;

however the sensor set should be significantly enriched before it may be profitably used for activity discovery and recognition. A natural alternative to gathering actual measurements consists in resorting to use synthesized ones; energy demand simulation, in particular, has been widely discussed in scientific literature. The authors of [70] discuss the use of models for end-use energy consumption; they point out that residential consumption represents a substantial part of energy demand in every countries, and suggest a partition of modeling techniques for residential energy consumption into two major classes; top-down, and bottom-up approaches. In the former case, no individual house energy profile is built, rather historic data is aggregated and analyzed to regress the energy model of the whole housing stock; on the contrary, in the latter case, energy consumption is estimated for a representative set of individual houses, and is later generalized to form the residential consumption model. For the purposes of this application, the bottom-up approach is more interesting; its main drawback is the need for detailed information about the home environment (the trend of common environmental measurements might need to be estimated, or simulated [71], [72]; supplier billing data, for instance, is private information, and typically it may be obtained only by disaggregation on the overall consumption); on the other hand, bottom-up techniques are often the only means to evaluate the impact of new systems or technologies, which are likely to lead to more effective power usage optimization.

Some modeling techniques for residential power consumption simulation are reviewed in [73]–[75]; those proposals share the idea that realistic energy usage simulation depends on three main factors: occupant behavior (i.e., activities), appliance models, and a model of energy consumption per activity. A slightly different approach, highlighting the importance of user activity simulation, was proposed in [75], where a Markov chain is used to simulate user presence and habits, modeled in terms of nine energy-hungry activities, such as for instance cooking, using a personal workstation, or simply being absent. The work presented in [74] performs energy demand simulation by summing up the contribution of each appliance in a dwelling, in a bottom-up fashion. The authors specifically focus on modeling user "active occupancy" and characterize an activity through a profile, storing its inception time, and duration; each activity profile is assigned to an appliance, strictly tying user presence to energy consumption; this choice also allows to model dependences and time correlations between appliances.

Besides detecting user activities, and linking them to a consumption profile, the ultimate task of a BECM system is the achievement of significant energy saving. In the past years, particular attention has been devoted to the specific issue of avoiding peaks in energy demand, which is a very complex issue, due to the high variability in user consumption demand and to the limited flexibility in scheduling in order not to negatively affect user experience; moreover, price policies adopted by providers are often insufficient to modify user habits and lower peak energy demand. In [76], [77], a demand-side load management system is proposed, suitable to be integrated in the future Smart Grid technology. The proposed system acts in real time, interacting with appliances and users, and adopts a layered structure, processing data coming from actual on-line consumption and schedule user requests, in order to balance electricity demand. Each appliance is modeled as a finite state machine, triggered by events generated by user or the balancing system. The core of the system is the admission control, that manages accesses to power resource and controls appliances. Its scheduling algorithm is heuristic-driven and finds a greedy solution; so the optimality of the solution is not guaranteed. The requests set is checked and, based on the state of appliances and the requested power, the system decides about its delivering. In [78], the authors propose a system to schedule only the so-called background loads, that is refrigerators, dehumidifiers, and so on. An algorithm inspired to the well-known Earliest Deadline First is used; the authors claim that scheduling non background loads may have an impact on user comfort, so they opt against controlling them. Finally, they introduce the concept of *slack*, that is the maximum amount of time a device can be disconnected from power, while still guaranteeing its performance; each load is assumed able to maintain an estimate of its remaining slack time. At fixed intervals, the algorithm checks the slack of each background load and gives priority to the one with the smallest slack; if a load reaches zero slack, then it is powered on, regardless of the increase in energy use. When the aggregated sum of background loads power reaches a prefixed threshold, no other loads are powered. Finally, in the approach presented in [79] the problem of shaving peaks in energy demand is formulated as a mixed integer linear program, in a mixed (i.e., renewable and non renewable) power source scenario. Authors aim at investigating the potential of a combined optimization approach that takes into account every possible kind of loads, namely shiftable, sliceable, stretchable ones, and so on. Each energy-demand task is characterized by a completion deadline, while each day is divided into equal time slices. The goal is to minimize the combined power of all slices. Some constraints are to be met; for instance, each device may be powered by only one source, and the amount of power needed by shiftable loads in each period is constant.

3.2 Learning User Habits for Energy Saving

Peaks of energy consumption can be shaved off p by tracking user activities in order to modify the functioning period of appliances that are not immediately useful for the current task; the approach aims to lower energy demand in the proximity of predicted peak loads so as to keep the overall consumption below a pre-set threshold. In order for the system to perform effectively and to be generalizable to previously unforeseen scenarios, it needs to capture and formalize the activities that actually account for user habits.

General *a priori* models of activities, appropriate to exemplify the behavior of any possible kind of user, are too complex to be realistically feasible. Designers are typically able to explain what an activity is in terms of the sensor set actually deployed, but they seldom succeed in describing how each activity can possibly be carried out by every user. Accurately discovering user activities and learning reliable models for them is however a very challenging task, so an initial preprocessing step is included, fulfilling two main goals: focusing future computation on the more interesting bits of data, and identifying events; hence, the original undistinguished stream of sensor readings can be translated in a more meaningful stream of events.

Figure 3.1 shows the overall architecture proposed in this chapter; energy consumption modeling is implemented by the *Energy Demand Simulator* (EDS) block, whereas energy saving algorithm, through peak load shaving, is represented by the *Energy Demand Optimizer* (EDO). The system core

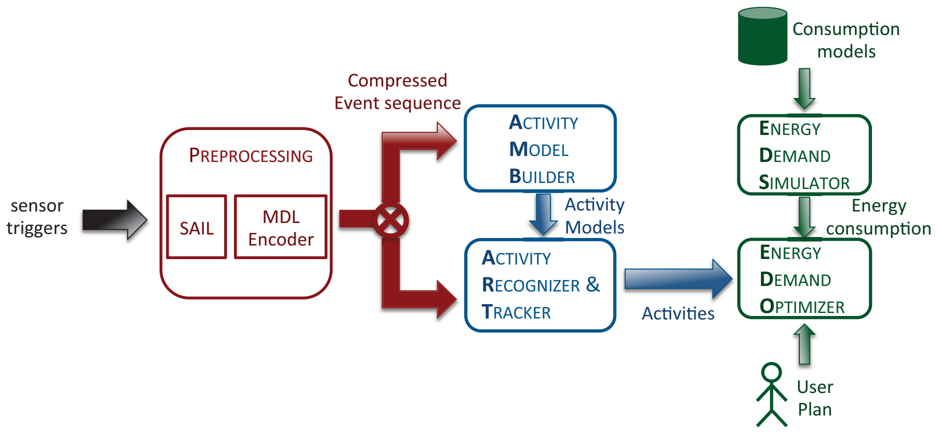


Figure 3.1: Energy demand optimization via activity recognition: overview of the whole system.

is represented by the *Activity Model Builder* (AMB) and *Activity Recognizer and Tracker* (ART), preceded by a *Preprocessing* block. The AMB is devoted to provide models of the most common user activities, which will be used by the ART module for on-line recognition; an optimal energy plan may thus by elaborated by the EDO module, on the basis of the energy demand provided by the EDS, the recognized and predicted activities and a user plan, containing the tasks to be executed in a given time interval.

In the following, the detailed descriptions for each of the mentioned modules are provided.

3.2.1 From sensor readings to a compressed event stream

A basic assumption is that a pervasive deployment of heterogeneous sensors is available over the monitored environment. In order to discover hidden relations between sensor triggers originated by different sources, a preprocessing step is needed; sensor readings can be merged to form templates for the most common events, which can be defined as significant frequently co-occurring triggers.

Performing an activity will generate a great number of sensor readings; for instance, breakfast preparation may involve proximity sensors (to the cupboard, to the oven, etc), item sensors (toaster, coffeemaker, taps), and environmental sensors (temperature, water flow), whose state may be represented by a *binary*, *discrete* or *continuous* variable, respectively. A *trigger* is defined as the pair composed

$\texttt{Ev}_1 \rightarrow \langle$`2.5 5.0`$\rangle$ `M13ON` $\langle$`1.25 1.5`$\rangle$ `M14ON` $\langle$`0.75 2`$\rangle$ `M13OFF` $\langle$`0.5 1.5`$\rangle$ `M14OFF`

$\texttt{Ev}_2 \rightarrow \langle$`2.5 5.0`$\rangle$ `F1OPEN` $\langle$`1 1.25`$\rangle$ `F1CLOSE` $\langle$`0.75 1.75`$\rangle$ `F1OPEN` $\langle$`0.75 2`$\rangle$ `F1CLOSE`

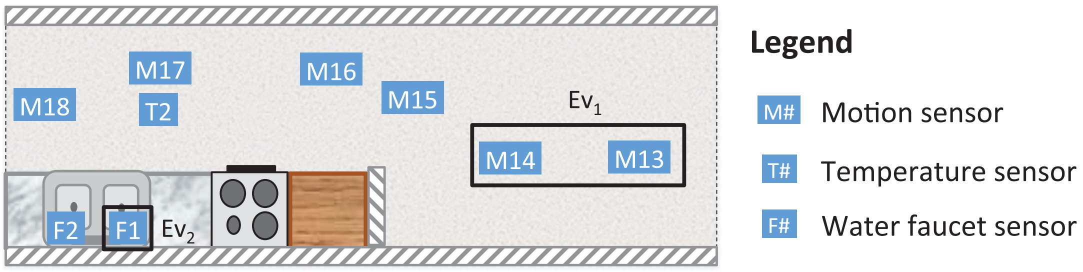


Figure 3.2: Two sample events extracted by the algorithm: $\texttt{Ev}_1$ captures the user walking toward the kitchen, while $\texttt{Ev}_2$ corresponds to using the kitchen faucet for washing. The maximum duration of events for template abstraction was set to 5s in both cases.

by sensor ID and sensor state.

Representative information must be extracted from a series of raw sensor triggers; to this end, a specific language is devsed, where an event is defined in terms of triggers according to the following syntax:

`EvID` $\langle$`durmin durmax`$\rangle$ `trigID` [, $\langle$`gapmin gapmax`$\rangle$ `trigID`]

According to this definition, each event is identified by the minimum and maximum expected duration of the whole sequence, an initial trigger followed by an optional sequence of triggers with intervening gaps of duration in the range `[gapmin, gapmax]`. An example of two events extracted by algorithm proposed in this chapter is shown in Figure 3.2.

Initially, the most frequent pairs of trigger occurrences are selected via a sliding window algorithm that filters out pairs whose duration would not satisfy search criteria; moreover, in order to select meaningful items, additional constraints are imposed applying a lower bound on the acceptable frequency:

$$\theta_{freq} = \mu_{freq} + 2 \cdot \sigma_{freq}$$

where mean μ_{freq} and standard deviation σ_{freq} are computed over the frequencies of all pairs.

Pairs of triggers may already be considered as elementary event templates, and may be expanded by iteratively adding more triggers to them. In order to discover the most frequent triggers comprised within each pair (if any), the conditional probability that a trigger falls within a given pair is exploited. Upon adding a new trigger to a sequence, the algorithm looks for the next possible value maximizing the updated conditional probability; addition of a trigger may reduce, but never increase the number of occurrences of a sequence in the overall trigger sequence, so the iterative procedure will terminate when such number falls below a preset value.

As a final step, all basic events made up of a single trigger are added to the newly found templates, thus producing a complete list of templates sorted by their relative frequency in the sequence.

Discovering the possible list of event templates enables for scanning previously unseen trigger sequences in order to identify the actual occurrences of events contained therein.

This step is accomplished by *String mAtching with wIldcards and Length constraints* (SAIL) [80], an on-line algorithm able to locate patterns as soon as they appear in the sequence, which was modified to account for representation of events and triggers.

Algorithmus 1 Extract Alphabet for event encoding.

```
Input: string $E$; int $n_{min}$, $n_{max}$
Output: alphabet $\Sigma$
 1: nlist $\Leftarrow$ extract_ngrams($E, n_{min}, n_{max}$)
 2: $\Sigma \Leftarrow \emptyset$
 3: while nlist $\neq \emptyset$ do
 4:     nlist $\Leftarrow$ sort(nlist)
 5:     ngram $\Leftarrow$ getfirst(nlist)
 6:     if get_obtainable_compression(ngram) $< \theta_{comp}$ then
 7:         return $\Sigma$
 8:     else
 9:         $\Sigma \Leftarrow \Sigma \bigcup \{$ngram$\}$
10:         nlist $\Leftarrow$ nlist $-$ $\{$ngram$\}$
11:         $E \Leftarrow$ delete($E$, ngram)
12:         nlist $\Leftarrow$ update(nlist, $E$, ngram)
13:     end if
14: end while
```

The use of SAIL transforms the *trigger* sequence into an *event* sequence, ready to be scanned to find frequent and relevant patterns, representing high-level activities. Information theory principles are used in order to keep this problem manageable, and to cope with the complexity of exploring the search space. Event sequence is compressed by lossy optimal coding so that events with low information content will be discarded; in other words, the most relevant patterns will be those that better describe the whole sequence, according to the MDL principle [81]; additionally, the compression of the event sequence allows for a decrease in the computational cost of later processing, thus coping with the exponential complexity of frequent event pattern mining.

Compression algorithm for activity discovery is inspired to arithmetic coding and entropy-based approaches. In order to find an optimal encoding for the event sequence E produced by SAIL, it is regarded as a string of symbols over the alphabet of event IDs. Algorithm 1 shows the pseudocode for the compression algorithm.

Borrowing the terminology from information theory, an n-gram is a subsequence of n contiguous items from a given string, so the goal is to translate the original sequence using a new alphabet whose symbols are the most significant n-grams in E. Line 1 of the algorithm extracts the list of n-grams of size between n_{min} and n_{max} together with their frequencies.

The algorithm then proceeds iteratively (lines 3-14). The n-grams are sorted according to the MDL principle: basically, each of them is viewed as a potential new symbol of the alphabet, and the length of string E is re-computed accordingly, using a binary encoding; the n-grams are sorted according to the degree of compression the can produce, and the n-gram producing the best compression is chosen (lines 4-5). In the following instructions, the frequencies of the remaining n-grams are updated, avoiding overlapping; the iteration stops when no n-gram is able to produce a compression rate above the chosen θ_{comp} threshold. Convergence is ensured since addition of an n-gram to the alphabet may only cause the frequencies of the remaining n-grams (hence, their potential compression rate) to decrease. The algorithm then returns the n-gram alphabet resulting in better encoding.

Once a shorter version E_R of the event sequence is obtained thanks to the new encoding, the most frequent patterns have to be discovered.

The entire preprocessing algorithm is shown in Figure 3.3.

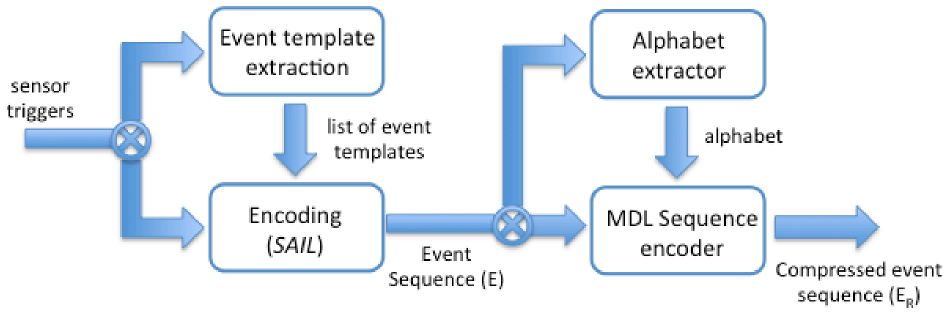


Figure 3.3: The preprocessing module.

3.2.2 Discovering, modeling, and tracking user activities

Activity discovery is formulated as a data mining problem, and frequent recurrent event patterns are regarded as instances of the yet unknown activities. Figure 3.4 depicts the process of information refinement underlying this approach: the system attempts to infer models for activities defined as *recursive structures* symbolically expressed in terms of a basic "alphabet"; the process starts by identifying relevant *events*, which, in this context, may be thought of as short and recurrent sequences of triggers, i.e. raw sensor readings. This bypasses the difficulty of creating a reliable model of an activity directly in terms of sensory triggers or supposed interactions between users and their home appliances.

Other proposals adopt a similar approach, but often rely on supervised algorithms, with the aim of looking for a translation of a predefined model of activity into data; however, explaining data through model established in advance implies some constraints and limitations: for instance, all users are supposed to carry out the same activities in a very similar way, and a great amount of data has to be collected and consistently labeled in order to create a sufficiently large training set.

In order to have activities naturally *emerge* from sensor observations, the AMB looks for recurrent structures; given the event sequence obtained from MDL encoding, the most frequent patterns have to be discovered.

AMB module is based on a modified version of *Discontinuous Varied-order Sequential Miner* (DVSM) [82], which is an Apriori-based iterative algorithm, relying on five main components: a can-

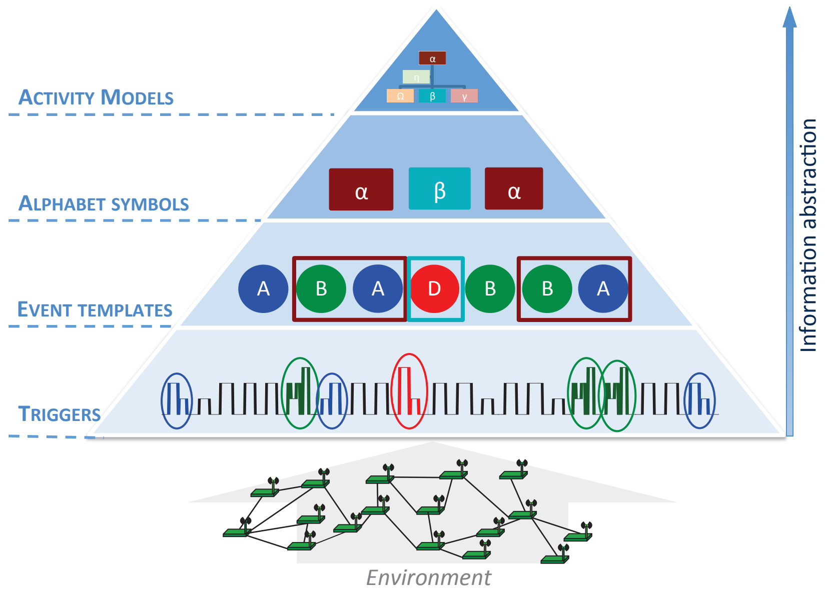
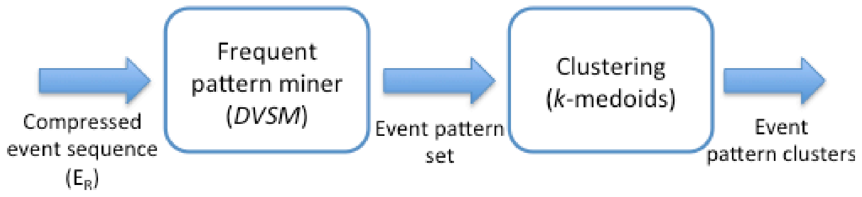


Figure 3.4: The process of activity discovery as an identification of recurrent structural patterns.

didate generation function, a pruning function, a candidate set, and a frequent pattern set. Initially, a candidate set is generated by considering the pruned set of all pairs of consecutive events in E_R. The idea of the algorithm is that each pattern in the candidate set is expanded at each iteration, according to a generation function. New patterns are checked against a pruning function, and only the ones surviving pruning are added to the new candidate set. Only those patterns whose expansions are all discarded (i.e. they are not "covered" by their expansions) will be part of the frequent pattern set. The algorithm stops when the candidate set is empty. The candidate generation function expands a pattern by adding the previous and the subsequent event in E_R, in order to create two new patterns. The pruning function is based on the MDL principle, and discards those sets of patterns unable to produce a sufficient compression rate for E, according to a predefined threshold.

In order to compute the compression rate, DVSM iteratively creates a hierarchical structure: at each step, variations of similar patterns in terms of the Levenshtein distance [83] are grouped together into general patterns. The compression rates of variations and general patterns are checked against


Compressed event sequence (E_R)
Frequent pattern miner (*DVSM*)
Event pattern set
Clustering (*k*-medoids)
Event pattern clusters
Figure 3.5: The AMB module.

two threshold values, C and C_v respectively:

$$\frac{1}{1+e^{A_v}} < C_v \qquad A_v = \frac{DL(D|a_i)*\Gamma_v(a_i)}{DL(D|a)*(1-\Gamma_g(a))} \tag{3.1}$$

$$\frac{1}{1+e^{A}} < C \quad A = \frac{DL(D)}{DL(a)+DL(D|a)*(1-\Gamma_g(a))} \tag{3.2}$$

where a is a general pattern, a_i one of its variations, $DL(\cdot)$ a measure of the description length and Γ is a continuity measure of the pattern, as in [82].

The final frequent pattern set returned by DVSM contains the most relevant patterns, which will be clustered into meaningful classes to obtain the discovered activities, by integrating temporal information with other features of interest, such as composition similarity, with an approach similar to [82]. This step is accomplished by k-medoids, a variant of the well-known k-means clustering, where representative points are bound to belong to the initial dataset; k-medoids uses a dissimilarity measure computed over all the possible pairs of points, giving it more robustness than traditional k-means measures with respect to noise and outliers [84]. Similarly to k-means, the number of partitions is a parameter chosen by the user.

The chosen dissimilarity measure reflects definition of pattern dissimilarity according to the T-pattern model, and consists of three components:

– *causality* is expressed by the order of the events in the pattern: earlier occurrences within the pattern may provide an explanation for occurrences found later on; therefore, the more dissimilar two patterns are with respect to the order of their events, the higher the probability that they represent instances of different activities. In this approach, causality is implemented by means of the Levenshtein distance;

- *critical intervals* deal with the relations between the distributions of components of a pattern; in other words, this measure considers the time distances between consecutive components. The corresponding function measures temporal information about the pattern element (time of day, duration, etc) and, clearly, the distance between two different components;

- the so-called *missing components*, i.e. the differences between the events present in two patterns, are determined based on the best pair of corresponding events between two patterns, if any.

In order to choose the best partitioning of the original pattern set, the algorithm is run multiple times with different initial random representative points. In the end, the partition that achieves the best overall dissimilarity measure among the obtained clusters is chosen. Such clusters constitute the so-called *discovered* activities, i.e. activities emerging from collected data.

The software modules involved in activity discovery and modeling are represented in Figure 3.5.

In the last phase, the features of the obtained clusters are encoded into models representing the discovered activities. An approach based on boosting was adopted; HMMs [40] are used to describe activities: an HMM is trained for each discovered activity, using the corresponding cluster set as training set. In the recognition phase, a window of fixed size is slid over the input events, and an activity label is assigned to the last event in the window, according to the HMM that achieves the higher posterior probability in correspondence to that event.

Once models for activities are available, the ART may process the incoming stream of sensor triggers, convert them into event sequences, and use a sliding window on them in order to recognize the current activity; the label assigned to the last element of the window is that of the activity corresponding to the HMM that maximizes the posterior probability.

3.2.3 Optimizing Energy Demand by Peak Shaving

Energy demand optimization is based on the assumption that recognizing user activities automatically and non disruptively for the inhabitants of the monitored environment is the key to effective energy demand optimization; to the best of my knowledge, no comprehensive dataset is available to date with details about power profiling and the corresponding information about user activities. However a few repositories have been created in the context of pervasive monitoring for activity recognition via simple, off-the-shelf sensors; such publicly available datasets were used and enriched with synthetic information about energy demand.

For the purposes of the present discussion, the overall energy demand of a smart home is characterized by identifying its main sources, from a user's point of view; energy consumption may thus be seen as the sum of three different components: a *baseline* demand, the (activity-driven) *user loads*, and what is called the *schedulable loads* (see *Energy Demand Simulator* (EDS) block in Figure 3.1).

3.2.4 Simulating energy consumption

The *baseline* consumption is generated by all appliances operating in background, such as heaters, dehumidifiers, freezers, refrigerators, and so on. Most loads belonging to this class can be shifted in time, getting a better execution order from an energy saving point of view; moreover, price forecast could be considered in order to minimize costs. Definition of baseline loads is inspired to the works by [78], and [76]. All such appliances are somewhat transparent to the end user, who does not perceive their presence and does not make an explicit scheduling plan for them. Moreover, they may be assumed to always have an impact on energy demand, as they account for essential services, or are necessary to guarantee a minimal comfort level. The baseline load profile can be modeled by considering a typical usage in an ordinary house. Once the set of baseline appliances is defined, their consumption is predictable according to the most common consumption profile, which may be obtained by referring to well established references. In particular, the study described in [85] was followed and the energy profile for a few common appliances was built, matching their respective loads to the previous taxonomy. For instance, the energy demand profile for baseline loads was inferred from the typical use of the corresponding appliances; a daily demand curve was generated based on the data provided by [85], and was parameterized to produce a set of standard daily usages.

User loads are a byproduct of the current user activity; microwave ovens, TV sets, computers represent typical examples of devices belonging to this class. EDS follows the approach proposed in [78], where the authors choose to leave out all those appliances that can be scheduled by the user in a predefined fashion (e.g. dishwasher); on the other hand energy demand due to user loads is likely unpredictable, hence very difficult to cope with, in order to prevent a negative impact on peak demands. The energy demand due to each activity is simulated by combining the effects of some randomly chosen devices that can be possibly turned on during its execution. For example, cooking may require the use of different appliances (e.g stove, as opposed to microwave oven), so different instances of the same activity may result into very different energy consumption profiles. In order to account for this peculiarity, only simulated consumption is considered as due to a random selection of devices from the set of all the appliances related to that activity. The coupling between appliances and activities was defined a priori; moreover, for each device activation, a random duration is chosen, by simulating the use of the same appliance in different executions of the same activity.

Schedulable loads, the third component of the proposed energy model is obtainable by analyzing a plan provided by the user. It includes all the appliances that are characterized by long-lasting tasks, as compared to normal user activities. Washing machines and tumble dryers are typical examples of this kind of appliances, similarly to "burst loads" in the terminology proposed by [76].

Finally, user plan is considered, which is a predefined list of tasks; for each of them, the user needs to provide two intervals defining the acceptable ranges for the beginning and ending time for the task; moreover, a priority is associated to every task, expressing its importance in the user's opinion. Time intervals associated with tasks may possibly take into account price forecasting, in order to minimize energy costs. The plan also takes into account dependencies between tasks, thus preventing the execution of meaningless chains of tasks. For example, a user might want the `tumble dryer` task executed only after the `washing machine` one; furthermore if, for any reason, `washing machine` was not executed, then neither `tumble dryer` should be. The idea of including a user plan might be profitable in other contexts as well; for instance, in a scenario where energy cost minimization is required, priorities can be chosen according to dynamic price strategies.

3.2.5 Peak shaving

Optimization of energy demand is implemented by the EDO block in Figure 3.1; it is focused on peak avoidance, considering the estimates of the baseline, user and schedulable loads. Energy optimization is regarded as a variant of the *Knapsack Optimization* (KP) [86], a theoretical approach that has already been applied to several practical fields. KP belongs to the integer combinatorial optimization domain, and encompasses a set of problems in the field of integer linear programming. It is known to be an NP-complete problem, and it has been widely studied due to its possible applications, ranging from financing to resource distribution; it has also found application in the context of energy optimization [64]. Given a set of objects, characterized by a *volume* and a *value*, the KP aims at selecting the best subset of objects that maximizes the total value, while maintaining the overall volume below a pre-set threshold (which is termed the *capacity* of the knapsack).

In this context, the main goal of the system is to estimate the current energy usage, and to predict its short-term trend in order to check that it is compatible with the activity the user is performing; the system then tries to rearrange loads generated by the appliances, in order to avoid exceeding a pre-set threshold for the overall demand, while satisfying user requirements, and completing the planned tasks. The underlying assumption is that the total energy consumption can be parted into two main components, namely the predictable consumption and the unpredictable one. The first component includes all the baseline loads simulated by the EDS module, as well as the schedulable loads due to the user plan; both components are intrinsically predictable. On the other hand, user loads generated by the current activities are hardly predictable, unless a short term prediction is considered by taking advantage of user activity recognition. Figure 3.6 shows a sample of a breakdown of energy demand in

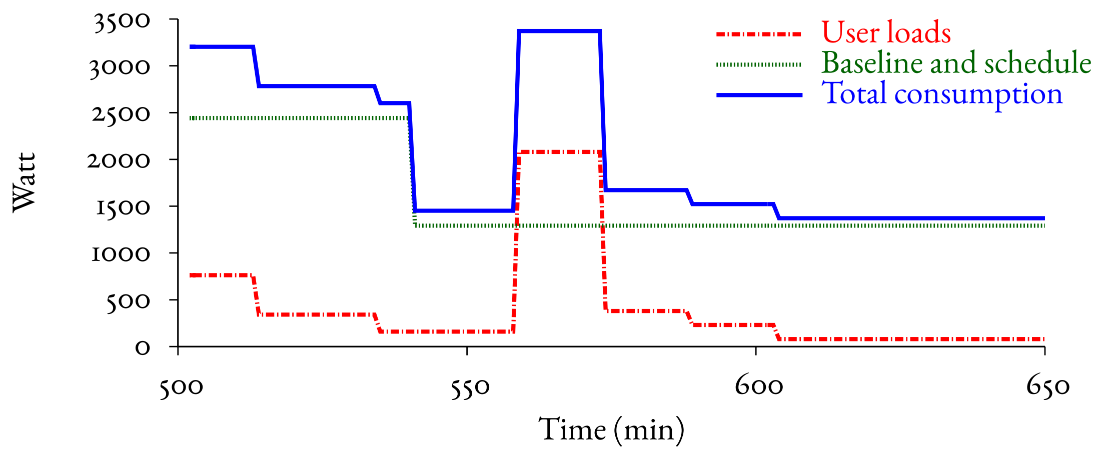


Figure 3.6: Example of a breakdown of energy demand in terms of baseline, schedulable and user loads.

the proposed scenario. The constraint represented by the pre-set threshold is thus further narrowed by an amount corresponding to the estimate of the consumption due to user loads; hence, predictable loads energy consumption is rearranged in order to meet the more restrictive threshold.

This functionality is provided by the block named EDO in Figure 3.1, which represents a software module accepting the following inputs: the current estimated energy consumption, the predicted user activities, and the user plan. When the predicted short-term energy use exceeds the pre-set threshold, EDO module attempts to select the minimum necessary amount of devices to be temporarily turned off so as to satisfy the energy use constraint, while respecting the provided priorities. Once the predicted load falls within the limit, the system attempts to restore the device; another option is to look for another device to turn off in order to trade for the reactivation of the old one.

In order to take user requirements into account, the proposals of [78], [79] is followed, providing a *slack* time for each baseline appliance; this piece of information is used to prevent the optimizer from turning a device on and off too quickly, which would cause a degradation in the overall performance, or even a possible failure. In the end, the deactivation time for a device is minimized, causing as little inconvenience as possible for the users.

Time is split into fixed-size slices; for each slice, the system selects the optimal set of devices to turn on in order to meet the energy consumption constraint, and match the user plan as closely as possible;

as already mentioned, the optimal selection of devices is formulated as a KP, to be solved at each time slice.

The capacity of the knapsack is defined as:

$$E_k = E_T - \hat{E}_U, \tag{3.3}$$

where $\hat{E}_U$ is the estimated maximum energy of user load component for the considered time slice, and E_T is the pre-set consumption threshold. The maximum consumption value is stored for each activity instance, together with the timestamps of its beginning and end time; for each new instance, a probability distribution parameterized over the initial time of the activity is recomputed; a similar approach is used for the baseline estimation, based on a whole day prediction.

The function to be maximized is expressed as:

$$\sum_{i=1}^{n} v_i x_i, \tag{3.4}$$

where the summation is taken over all the appliances generating baseline and schedulable loads. The integer variable x_i is defined as:

$$x_i = \begin{cases} 1 & \text{if the device is turned ON} \\ 0 & \text{if the device is turned OFF} \end{cases} \tag{3.5}$$

The coefficient v_i indicates the priority of the task, corresponding to the user-defined one for the schedulable appliances and to a function of the slack value for the baseline loads.

The constraint to meet is:

$$\sum_{i=1}^{n} E_i x_i \leq E_k, \tag{3.6}$$

where E_i is the consumption of the device, according to its consumption model.

3.3 Experimental assessment of the peak shaver system

In order to assess the performance of the whole system, two reference scenarios were considered, according to activity recognition or energy saving tasks; in the former case, events generated by sensors deployed in a smart home environment was analyzed, where each sequence of triggers was labeled according to the activities performed by the user, whereas for the latter the system was assumed able to control a predefined set of appliances, and an energy consumption demand was simulated, according to a realistic energy use profile. In particular, three public datasets were used to measure the accuracy of the system: *adlnormal* [87], and *aruba* [88] (both from the CASAS project), and the one called *kast* [47] from the *Context Awareness in Residence for Elders* (CARE) project. All datasets are annotated, i.e. their sensor trigger sequences are labeled with the activity the user was performing in correspondence to that portion of data: the so-called *actual* activities; however, the three datasets are very different with respect to the set of employed sensors and to the way the data was collected; their descriptions are reported in Table 3.1.

3.3.1 Evaluation of the MDL event encoder

The MDL encoder represents the core of the preprocessing step, and its main goal is to reduce the "uncertainty" inherently present in data so that the subsequent modules of the system may focus only on the most significant information. Thanks to the new encoding, dissimilarities among event patterns are magnified, so that they get scattered throughout the ideal representation space, which ultimately results in more easily distinguishable activities.

The effects of user activities are observable by the system only in terms of the effects they produce on the environment, so an activity might be abstractly modeled as a stochastic source of sensor triggers; more specifically, an activity is regarded as a source of alphabet elements (the n-grams selected by the MDL encoder) and compute its emission probability. Telling different activities apart is only possible if each element can be associated to the correct source; this task becomes more manageable as the source emission probability distributions are most different from each other.

Table 3.1: The datasets used for testing the system.

Dataset	**Features**	**Activities**	**Sensors**
adlnormal	20 users (one at a time), about 6,000 sensor readings, 100 activity instances	5 activities (*Telephone use*, *Hand Washing*, *Meal Preparation*, *Eating and Medication Use*, *Cleaning*)	motion sensors, analog sensors for monitoring water and stove burner use, as well as software sensors (VOIP), and contact switch sensors on phone book, cooking pot and medicine container
aruba	1 user, about 6,000 sensor readings (out of 1,600,000 total), 120 activity instances (6,471 total)	11 activities: *Meal Preparation*, *Relax*, *Eating*, *Work*, *Sleeping*, *Wash Dishes*, *Bed to Toilet*, *Enter Home*, *Leave Home*, *Housekeeping*, *Resperate*	binary sensors: motion sensors and door closure sensors (temperature sensors were also present, but they were not used by the proposed system)
kast	1 user, 2,120 sensor readings and 245 activity instances spanning 28 days	7 activities (characterized by different time duration and different frequency): *Leave house*, *Toileting*, *Showering*, *Sleeping*, *Preparing breakfast*, *Preparing dinner* and *Preparing a beverage*	14 binary sensors deployed in the house, placed on doors, cupboards, refrigerator and a toilet flush.

The assessment of this module was thus carried out by comparing the statistical properties of the different activities, in terms of probability distribution of their basic elements. Temporal information was purposely disregarded at this step, as it does not carry additional significant information in this context. Different instances of the same activity can be very dissimilar in terms of their temporal unfolding, depending on how specific users perform them, but the relevant information content consists in their respective subtask composition, regardless of the exact duration and consequentiality[*]. The statistical properties of an activity, thought of as a stochastic source, might reasonably be considered invariant and distinctive of the activity itself.

Hellinger distance was chosen as a measure of dissimilarity between different activities [89]. This is a f-divergence measure, which quantifies the difference between two probability distributions $P^{(i)}$

[*]For example, *Cooking* will likely involve a set of tasks such as opening the cupboard, grabbing a pot, and switching on the stove burner, but their duration and exact sequence may vary among different instances of this activity.

Table 3.2: Comparison of Hellinger distance with original triggers and after MDL encoding.

	adlnormal		***kasteren***		***aruba***	
	Original	**Encoded**	**Original**	**Encoded**	**Original**	**Encoded**
Mean	0.6116	0.6931	0.8360	0.8533	0.9007	0.9024
Max	0.9423	0.9793	1	1	1	1
Min	0.2483	0.1668	0.3190	0.3076	0.1666	0.1600

and $P^{(j)}$:

$$h(P^{(i)}, P^{(j)}) = \frac{1}{\sqrt{2}} \left\| \sqrt{P^{(i)}} - \sqrt{P^{(j)}} \right\|_2, \tag{3.7}$$

where:

$$\sqrt{P^{(i)}} = \left(\sqrt{p_1^{(i)}}, \sqrt{p_2^{(i)}}, \cdots, \sqrt{p_m^{(i)}}, \cdots \sqrt{p_n^{(i)}} \right) \tag{3.8}$$

is a unit vector in 2-norm, $p_m^{(i)}$ is the probability that the i^{th} activity 'emits' the m^{th} symbol, and n is the cardinality of the encoding alphabet.

By definition, the Hellinger distance is symmetric and satisfies the triangle inequality, so it is a proper distance, which induces a metric space. This metric space was used to get a quality measure of the preprocessing; namely, if Hellinger distance was computed for every pair of activities, both before and after preprocessing, it would expect that a useful encoding imply a larger distance on average in the latter case.

Tests show that an improvement in Hellinger distance was achieved for every dataset, with an increase as high as 8% as compared to the original representation in the case of *adlnormal*, demonstrating the effectiveness of the MDL encoder. For this dataset, the average Hellinger distance computed between all the ten pairs of the five considered activities is 0.6116, when only activation triggers are considered as suggested by [82]; after MDL encoding, it increases up to 0.6931. Table 3.2 summarizes the results of the tests. Table 3.3 shows how much the Hellinger distance matrix differs, for each couple of activities of *adlnormal*, with and without applying the MDL encoding; element (i, j) of this matrix is the difference of the Hellinger distance between activity i and activity j in the two cases; obviously, it is a strictly triangular matrix. The obtained results show a significant improvement, in

Table 3.3: Confusion matrix of the difference of Hellinger distance between original triggers and MDL encoding.

	Telephone use	***Hand Washing***	***Meal Preparation***	***Eating/med. Use***	***Cleaning***
Telephone use	0	0.1572	0.0370	0.1465	0.0996
Hand Washing	0.1572	0	0.1151	0.1407	-0.0815
Meal Preparation	0.0370	0.1151	0	0.0528	0.1102
Eating/med. Use	0.1465	0.1407	0.0528	0	0.0375
Cleaning	0.0996	-0.0815	0.1102	0.0375	0

terms of a higher Hellinger distance, for most of the activity description dissimilarities. The original encoding outperformed MDL encoding only for the (*Hand Washing*, *Cleaning*) pair, probably due to the extreme similarity of the two activities. Figure 3.7 shows a detailed comparison of the two activities, in terms of distribution of triggers and alphabet elements; the *Hand Washing* activity clearly shows how the MDL encoder succeeds in compressing the statistical description of the activity; however, this eventually resulted in an increased similarity to the *Cleaning* activity. On the other hand, the most significant improvement was obtained for the (*Hand Washing*, *Telephone use*) pair, likely because encoding is able to emphasize the difference in terms of the predominant set of subtasks; these two activities indeed involve very different sensor sets, as they are carried out in different areas of the house and with different tools.

Similar results were obtained for the other considered datasets, with an overall increase in the Hellinger distance, except for those pairs composed by only a very reduced set of significant elements.

3.3.2 Testing the AMB and ART modules

In order to assess the ability of ART and AMB to correctly identify patterns of events, their performances were tested against the 3 datasets, with varying compression thresholds for the *DVSM* module

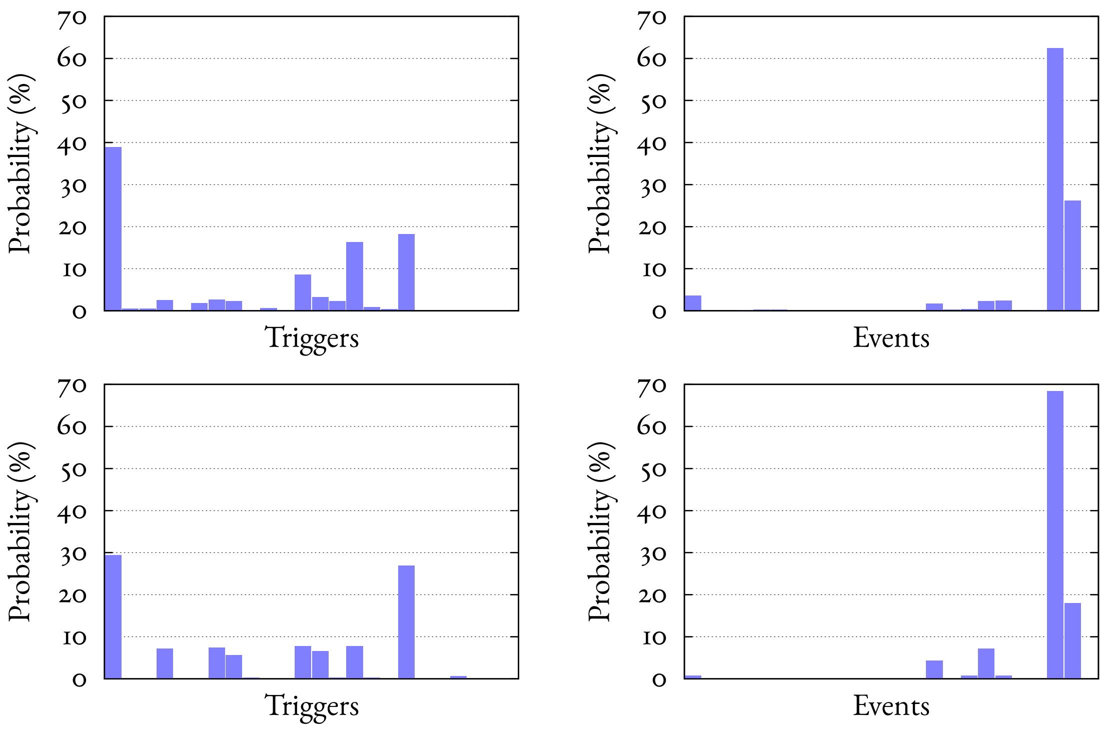


Figure 3.7: Hellinger distance: difference between original triggers (left column) and MDL encoding (right column) for *Cleaning* (top row) and *Hand Washing* (bottom row).

(i.e. C influencing general patterns, and C_v for variations, see Eq. (3.1) and (3.2) on p. 44). Figure 3.8 shows that the algorithm performs similarly in all cases, but the resulting number of patterns is very threshold-dependent. Higher values for both thresholds increase the number of discovered patterns, up to a saturation point; the best performance is obtained with *adlnormal*, arguably due to the fact that test users were instructed to simulate daily actions by following a preset script. A bad choice of thresholds may result in failing to discover any patterns at all, as is the case with $C_v = 0.38$ for *aruba*. The results show that appropriate values of C and C_v allow *DVSM* to prune most of the less meaningful patterns, also in combination with the preprocessing and encoding steps, that purge the input trigger sequence from non-significant data.

Performance of k-medoids algorithm in producing meaningful classes of activities was also tested, in terms of the goodness of its clustering. To this end, the same metrics as in [82] was used, namely:

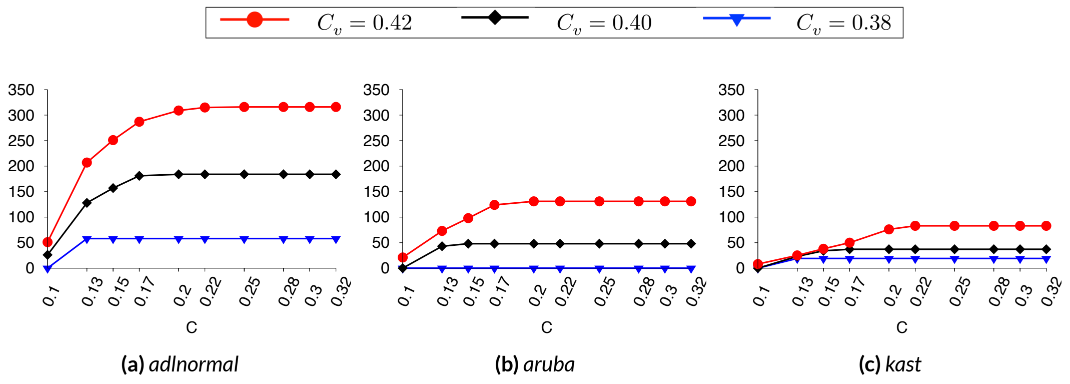
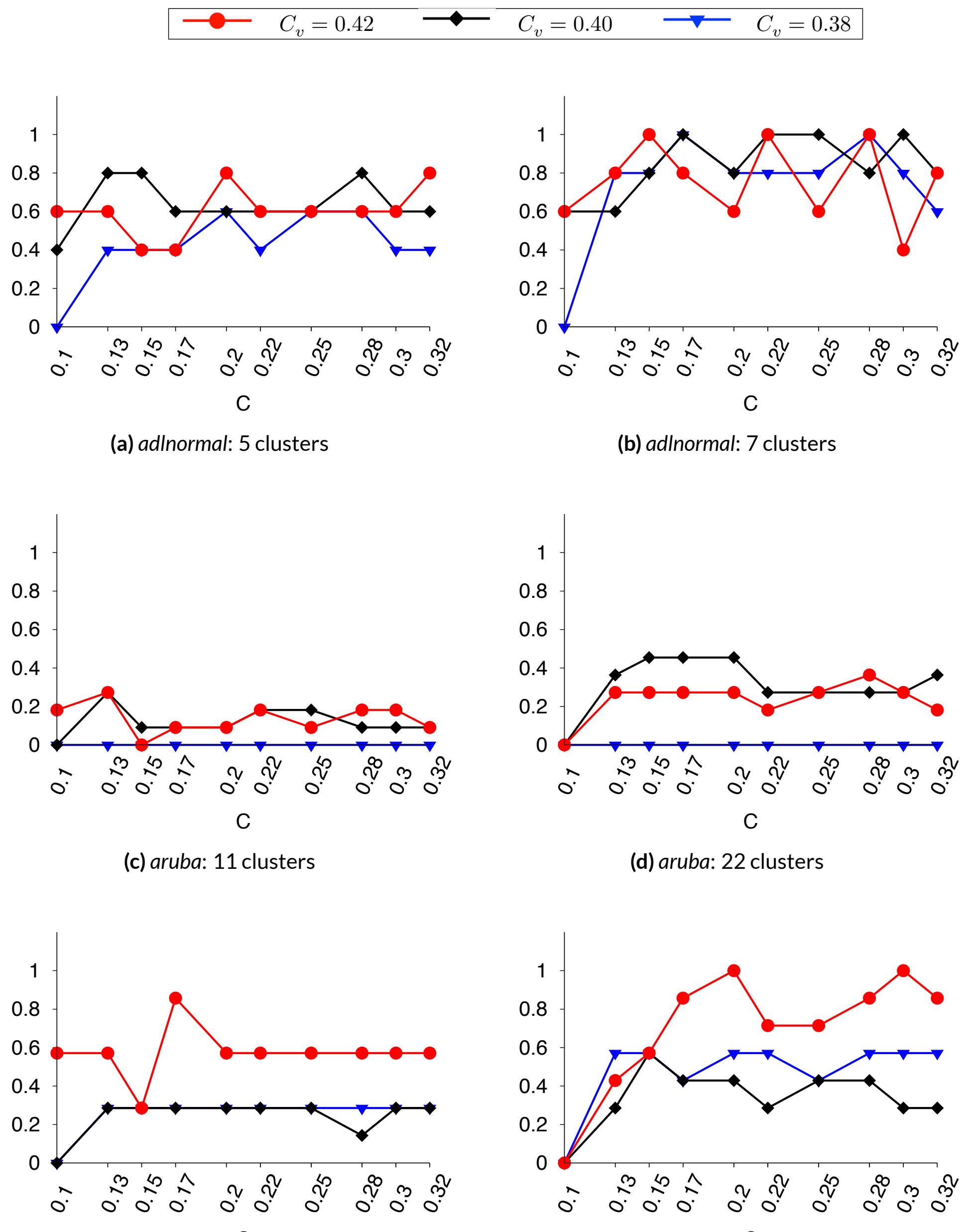


Figure 3.8: No. of extracted patterns as a function of the compression threshold C, parameterized on C_v.

- q_1: the ability to identify activities, computed as the ratio between the number of actual labels assigned to the *discovered* cluster representatives, and the total number of *actual* activities;
- q_2: the ability to assign correct labels to the extracted patterns with respect to actual activities, computed as the fraction of patterns actually belonging to the activity assigned to the cluster medoid, per each cluster.

The obtained results are shown in Figures 3.9 and 3.10 for different values of C, C_v; in order to assess the influence of the chosen number of clusters (k) on the selected metrics, this parameter was initially set equal to the number of actual activities for each dataset, and then increased it. The results show that q_1 is more sensitive to k than to the thresholds C and C_v, and higher values of k cause an increase in q_1, as is particularly evident in *adlnormal*. The worst performance is obtained on *aruba*, due to the presence of many unlabelled triggers, reflecting the fact that actual activities poorly correspond to the user's normal life; this is also highlighted by the results for q_2 on the same dataset, which show that when the cluster does represent an actual activity, its patterns are labeled in the correct way. For the other datasets, q_2 confirms the results from q_1, and shows good performance on accuracy in classification. The number of patterns does not influence this metric as much as it does for q_1, suggesting that increasing the number of clusters improves the "coverage", but not the quality of produced clusters.

Finally, the accuracy of the HMM-based activity recognizer was assessed, with respect to discovered and actual activities.



(a) *adlnormal*: 5 clusters

(b) *adlnormal*: 7 clusters

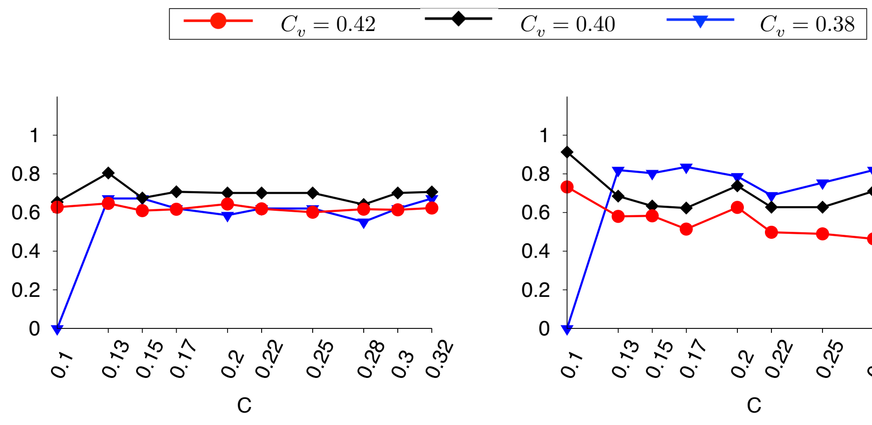
(c) *aruba*: 11 clusters

(d) *aruba*: 22 clusters

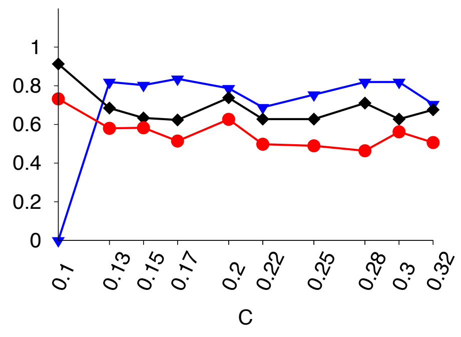
(e) *kast*: 7 clusters

(f) *kast*: 11 clusters

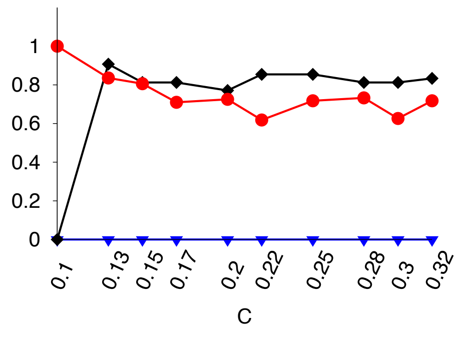
Figure 3.9: Metric q_1 for the 3 different datasets as a function of the compression threshold C e C_v and number of clusters.



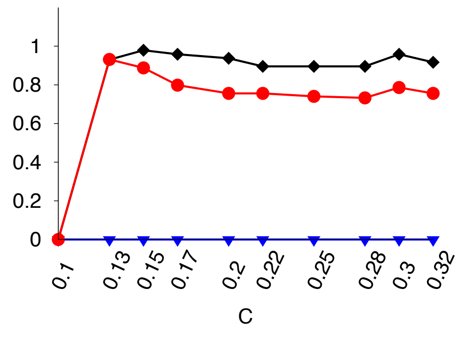
(a) *adlnormal*: 5 clusters



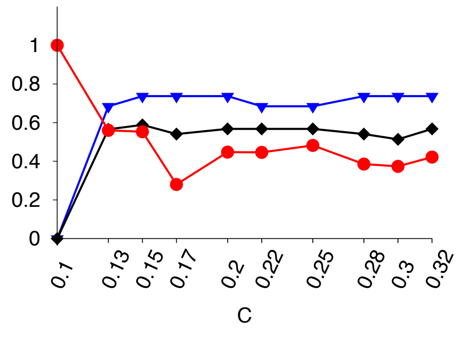
(b) *adlnormal*: 7 clusters



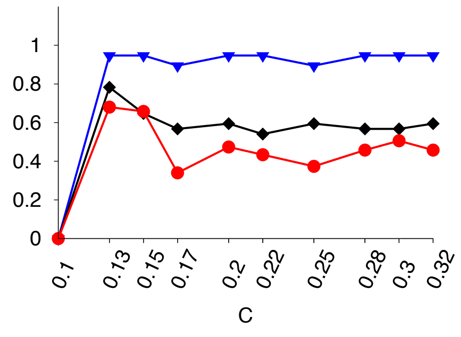
(c) *aruba*: 11 clusters



(d) *aruba*: 22 clusters



(e) *kast*: 7 clusters



(f) *kast*: 11 clusters

Figure 3.10: Metric q_2 for the 3 different datasets as a function of the compression threshold C e C_v and the number of clusters.

The best values for setting the HMM parameters, i.e. the number of hidden states (N), and the size of the sliding window (w), were computed; to this end, a grid search was used, with $N \in [3; 15]$ and $w \in [3; 15]$, and computed the accuracy of the system at each point in the grid. Two separate tests were conducted, aimed at the recognition accuracy of actual and discovered activities, respectively. Results for the best configuration of parameters with respect to actual activities are shown in Table 3.4, where the corresponding value for discovered activities is also shown. As expected, better results are achieved for actual activities in *adlnormal*, due to better correspondence between actual and discovered activities. The achieved accuracy is very high, confirming the capacity of the method of building reliable models. The results obtained for the *aruba* and *kast* show that the proposed recognition system is able to create models of discovered activities with no assumption regarding the particular scenario. On the other hand, results on actual activities in these dataset suffer from the poor correspondence between discovered activities and actual activities. The setting for parameters N and w is also dependent on the specific dataset; such values need to be carefully chosen with respect the data at hand, as they basically represent how different activity definitions are mirrored into the corresponding datasets.

3.3.3 Energy consumption optimization by peak load shaving

The lack of a sufficiently rich dataset to measure the effects of real-time user activity recognition on energy usage optimization motivated us to generate synthetic data to assess the performance of the EDO block. Simulation takes advantage of the typical home appliance profiles, as documented in [85]; additional profiles were generated by using the models proposed in [74]. With such information fed into the EDS block, two energy demand curves can be computed in order to compare the performance

Table 3.4: Best results in recognition accuracy.

	C	C_v	k	N	w	**Actual**	**Discovered**
adlnormal	0.17	0.38	7	4	12	0.95	0.98
aruba	0.30	0.42	11	6	3	0.66	0.92
kast	0.13	0.40	11	6	3	0.55	0.97

Table 3.5: Correspondence between activities and the relative appliances.

Activity	Appliances
Meal Preparation	`Hobs`, `Stove`, `Microwave oven`, `Kettle`
Cleaning	`Vacuum cleaner`
Eating and Medication Use	`Coffeemaker`
Telephone use	`Lamp`
Hand Washing	`Instantaneous Water Heater`

Table 3.6: List of appliances associated to schedulable and baseline loads.

Load category	Appliances
Schedulable	`Washing machine`, `Tumble dryer`, `Dishwasher`
Baseline	`Refrigerator`, `Electric heating`, `Freezer`, `Air conditioner`, `Circulation Pump`

obtained without the intervention of the EDO block with the one resulting from the inclusion of the energy optimizer. In the former case, the EDS was tuned to simulate a typical domestic usage, considering the actual sequence of activities in a fashion similar to [74], [75]. In the latter case, the optimized energy demand is computed by following the indications of the EDO block about toggling the appliances on and off.

Tests were conducted by considering the *adlnormal* dataset so as to build an energy profile for each of the five tracked activities (namely, *Telephone use*, *Hand Washing*, *Meal Preparation*, *Eating and Medication Use*, and *Cleaning*). Table 3.5 indicates the subset of appliances involved in their execution. *Adlnormal* activities often span a short interval, hence the simulation makes use of time slices of appropriate length (2 minutes in this case).

As regards the user plan, it was assumed that the appliances whose use was suitable to be scheduled were those reported in the first row of Table 3.6, while the baseline loads were simulated according to the devices reported in the second row. The dependencies between different tasks was also coded, where applicable; for instance, the use of the `Tumble dryer` is only admissible after the `Washing machine` task has been completed. In the experiments, the pre-set threshold for limiting peaks in energy demand was set to 3 kW; in order to solve the knapsack problem, the capacity may thus be recomputed at each time step by subtracting the predicted energy demand due to the user activity from such threshold (see Def. 3.3 on p. 50).

Figure 3.11 shows some significant examples of the outcome of the peak shaving algorithm. The reported charts are representative of the cases where energy demand was successfully maintained below the pre-set threshold; overall, the system managed to reduce the number of unacceptable peaks by about 30%, on synthetic data; however, there were circumstances when the excess of energy demand could not be avoided due, for instance, to the combined effect of the user plan and the requirements of the current activity or, much less frequently, to a wrong prediction of the activity recognition module.

The two charts shown in the topmost row illustrate a common situation when the operating time of some baseline appliances is delayed until the overall load falls below the given threshold (see the shadowed are in the charts).

A different behavior is shown in the middle row, where the original loads are presumably due to appliances for which a considerable *slack* time was provided; the final effect is that the system is allowed to give priority to the energy constraint at the expense of slightly bending the requirements of the users, who experience a delay in the services offered by baseline appliances; basically over-threshold loads are immediately switched off, and their re-activation (if any) falls beyond the currently shown window.

Finally, the last row shows a specific instance of the action of the optimizer on schedulable loads. Those are typically characterized by long activation times; for instance, one such load is present for about 50 minutes (from time 1050 to 1100 in the left chart). The right chart shows the action of the optimizer resulting in an immediate re-scheduling of the critical loads, which are temporarily removed; this is followed by an additional deactivation at time t_1, and some loads appearing again at time t_2. However, it is evident from the chart that, at time t_2, some loads start "competing" for re-activation, thus producing an oscillation in the optimizer behavior; this is likely due to their relatively similar priorities, or simply to an intrinsic "bursty" consumption (which is typical of some appliances, such as `kettle`).

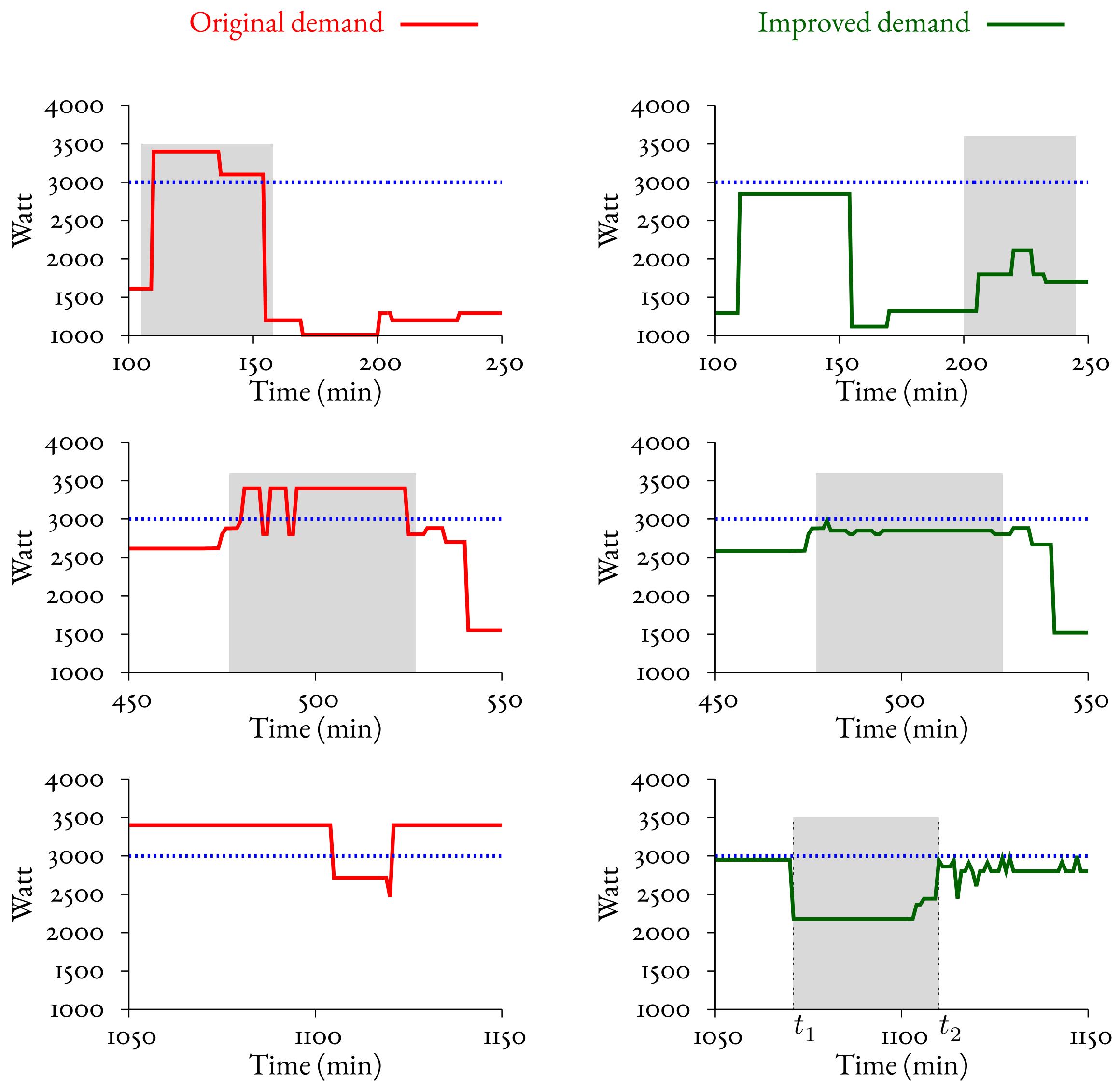


Figure 3.11: Comparison of original energy demand, and the one obtained after applying the proposed approach.

To iterate is human, to recurse divine.

L. Peter Deutsch

4 Formal Languages for Mobility Models

The high flexibility of algorithms based on Statistical Learning is also their weakest point. Considering analysis of very large and unstructured data, performance of these algorithms, in terms of knowledge extraction, is very hard to assess, because they can not give real insights about the most significant features of data; indeed, they are encoded as "black-boxes", i.e., the set of parameters used to tune the learning algorithms. Thus, only parameters to adapt known hypotheses to the data are available: relations can be found only if they are supposed to exist. Hypotheses are represented by class of functions and operators, therefore parameters can hardly be understood in terms of original data and choosing between several models that fit the data comparably well is rather impossible.

A shift in perspective may be of help to tackle with the unaddressed goal of representing knowledge by means of the *structure* inferred from the collected samples; more specifically, concepts and methods borrowed from *Algorithmic Learning Theory* (ALT), which relies on formal languages and automata, can be very useful in knowledge extraction. Unlike its statistical counterpart, ALT does not require any specific constraints on the statistic properties of the available data, and it rather relies on formal languages and automata theory. Its most interesting peculiarity is that the obtained knowledge is syntactically driven, hence intrinsically "structural".

In this framework, knowledge extraction may be formulated in terms of *Grammatical Inference* (GI) [90], an inductive process able to select the best grammar (according to a metric) that is consistent with the samples, according to the learning model known as *identification in the limit* [91]. Unlike statistical approaches, data is not encoded into a vectorial space, rather it is regarded as strings generated by an unknown grammar [92].

GI can be successfully applied in order to get relevant insights about the hidden structure embedded in large collections of data, enabling the user to ask and answer to new kinds of questions, taking advantage of the generative models obtained by the inductive process. Indeed, grammars are very informative if used to explain the relations between different subsets of samples. Moreover, thanks to their recursive nature, grammars are also able to perform multi-scale analyses, finding out what the most recurrent relations at different granularities of data are.

All these concerns are also central in data mining, whose main goal is to highlight the most important characteristic relations in data, in order to predict future trends; thus, GI can be the right tool to enable a deeper understanding of large collections of data, characterized by counter-intuitive and hard to guess relations.

In order to highlight the potential of the suggested approach, grammatical inference, and more specifically inference of regular languages [93], is applied to the problem of inferring mobility models. In this context, the availability of generative and multiscale models allows to simulate and predict changes in user habits, according to variations in viable paths.

4.1 Knowledge and formal languages

Three main forms of knowledge can be identified, according to [94]: declarative, procedural and structural.

- *declarative*: it expresses the awareness about some items or events or concepts*. It is the knowledge of the *"knowing that"*, i.e., it allows to identify and describe an item or a concept, but it does not enable to use them.
- *procedural*: it describes how learners use or apply declarative knowledge; it is the knowledge of *"knowing how"*.
- *structural*: it mediates the translation of declarative into procedural knowledge and facilitates the application of the latter; it is the knowledge of how concepts within a domain are interrelated; it is the knowledge of *"knowing why"*.

Some researchers consider structural knowledge as part of the declarative one [96], but the existence or nature of structural knowledge is not undermined by this assumption [94].

Structural knowledge is different from *structured* knowledge. Structured knowledge typically refers to a description through entities and relationships: the focus is on how knowledge itself is organized. On the other hand, structural knowledge deals with the type of knowledge to be acquired, rather than the way it is organized: the emphasis is on the organization and structure of the objects of the analysis.

Formal languages are the best tool to represent, organize and process structural knowledge, because they provide a representation focused on the description of the relations between their elements. A formal language is a set (finite or infinite) of sentences, each finite in length and made up of a finite set of elements [97].

*In this context, *concept* refers to a class of equivalence that can be described through a finite set of assertions; moreover, an effective procedure to classify it must exist [95].

This definition, as Chomsky suggests [97], can be easily adapted to natural languages, but also sentences drawn by a formalized mathematical system can be considered as a language. In the context of this thesis, under some restrictions, it is claimed that the sensory data collected during an observation of an event can be considered as produced by an hidden language, that acts as model of the event itself.

In a recent work [98], some researchers discovered that the understanding of a connected speech gives rise to the concurrent tracking of different timescales, in order to identify abstract linguistic structures at different hierarchical levels. Different neural processing timescales suggest a grammar-based internal construction of the linguistic structure. Thus, it is a clear clue that grammar-based representations are effective and efficient methods to represent and handle complexity in knowledge extraction process.

4.1.1 Formal language representation

Two different descriptions can be associated to a language: *generative* and *recognition-based*.

Generative According to this description, a language corresponds to the set of strings *generated by a grammar*. A grammar is a formal system able to transform an input through a set of predefined rules. The different abilities of the adopted set of transformation rules induce a hierarchy of generative grammars [99]. Generative description is appealing to humans, because it is intuitive and compact, but their straightforward implementation is inefficient.

Recognition-based description In this description, a language is considered as the set of strings accepted by an automaton. An automaton is a formal system that accepts a sentence as input and determines if the sentence belongs to a language. Recognition-based descriptions are appealing to machines, because automata are formal, compact, low-level machines that can be implemented easily and efficiently, but hardly understandable for a user.

This dual description of a language is a key factor in the scenario of sensory data analysis; indeed, there are two major issues with this kind of data: the difficulty to visualize it and the difficulty to think about targets of knowledge extraction in terms of raw data.

The first problem is common to every application related to the analysis of great quantity of data. Nowadays, big data is very common, due to the great number of cheap devices able to provide a steady stream of measures; often, this requires an organization of data in a high-dimensional space, inducing several issues caused by the curse of dimensionality. So, data needs to be observed from the right point of view, picking only its significant attributes.

In the past, this process was carried out by an expert of the application domain, who selected the most relevant attributes and validated the models drawn by data analysis, according to his knowledge and expertise. This approach is not feasible anymore for big data, because it is impossible to check all the choices of attributes. Moreover, in sensory data scenario, at the beginning of the analysis there are no clues about what the best candidate features are. Thus, an efficient tool able to suggest a set of possible data representations in a human understandable form is essential for an effective data analysis.

The second problem is strictly tied with the first. Models obtained by traditional Machine Learning algorithm are described by the set of their parameter values, that are hardly correlated with the original representation of data. Human validation of models is therefore infeasible, especially between models that performs equally well on test data. Moreover, this kind of models does not provide any new insight in data, while often this is the true aim of the analysis. On the other hand, traditional approaches are very efficient to cope with the computational burden implied by big data analysis, while traditional symbolic approaches cannot be employed because of their high computational costs.

Therefore, an approach that provides the efficiency of automata and the representation power of grammars can be the key to overtake the hurdles in the analysis of great quantity of sensory data.

4.1.2 Grammars

A generative grammar is a quadruple [100], [101]:

$$G = (\Sigma, V, S, P)$$

where:

- Σ is a set of *terminal symbols* called alphabet; strings of the language are made up of these symbols;
- V is a set of so-called *nonterminals*, *variables* or *syntactic variables*; V is disjointed from Σ;
- S is the *start symbol*, i.e., the nonterminal representing the language being defined.
- P is a finite set of *productions* or *rules*, that is an ordered set of pairs of strings. Each production takes the form:

  $$\Psi \rightarrow \Theta$$

  with $\Psi \in (\Sigma \cup V)^* V (\Sigma \cup V)^*$ and $\Theta \in (\Sigma \cup V)^*$. Ψ is called *head* of the production, while Θ *body*. This means that string of elements Ψ can be replaced by, or rewritten as, string of elements Θ.

Increasing restrictive conditions on productions define a hierarchy of grammars [99], [101]:

Type-0 grammars are also called *unrestricted rewriting systems*, because their productions are not restricted by any limiting condition. The automaton that accepts the language of these grammars is the TM.

Type-1 grammars does not contain any production whose application reduces the length of the resulting string. Productions of these grammars have the form:

$$\alpha_1 A \alpha_2 \rightarrow \alpha_1 \Psi \alpha_2$$

with $\alpha_1, \alpha_2, \Psi \in (\Sigma \cup V)^*$ and $A \in V$.

Grammars of this type are a subset of the type-0 grammars. Automata that accept language of these grammars are called *Linear Bounded Automata*.

Type-2 grammars allow only rules whose heads are limited to one nonterminal and bodies can not take on the empty string (λ) as value. Productions have this form:

$$A \to \Psi$$

with $\alpha_1, \alpha_2, \Psi \in (\Sigma \cup V)^* - \lambda$ and $A \in V$. These grammars are a subset of type-1 grammars. Automata that recognize the language of these type-2 grammars are called *Push Down Automata*.

Type-3 grammars have productions with the form:

$$A \to \Omega B \qquad \text{or} \qquad A \to B\Omega$$

with $A \in V$, $B \in (V \cup \{\lambda\})$ and $\Omega \in \{\Sigma - \lambda\}$. These grammars are a subset of type-2 grammars. Moreover, they produce the so-called *regular languages*, that can be recognized by *Finite Automata*.

The language generated by a grammar G is the set of strings Σ^* that can be obtained from the start symbol S, applying productions:

$$L(G) = \{x \in \Sigma^* : S \overset{G_*}{\Rightarrow} x\}$$

where $\overset{G_*}{\Rightarrow}$ is the reflexive and transitive closure of the production rules $\overset{G}{\Rightarrow}$.

It is worth noticing that a grammar can produce only a language, while a language can be generated by several grammars.

4.1.3 Automata

For the aims of this thesis, it is sufficient to describe *Finite State Automata* (FSAs), because they have enough expressive power to handle regular languages, that are used in the current approach. Nevertheless, the results obtained could be adapted to more powerful languages.

A FSA is a limited version of the TM; it is an abstract machine that can be in only one *state*, drawn by a finite set, at a time. The transition from one state to another one is triggered according to a function whose inputs are the current state and the input of the FSA; this function is called *transition function*.

There are two main types of FSAs: the *Deterministic Finite Automata* (DFAs) and the *Nondeterministic Finite Automata* (NFAs).

A NFA is a system:

$$A = (\Sigma, Q, q_\lambda, \mathbb{F}_\mathbb{A}, \mathbb{F}_\mathbb{R}, \delta),$$

where:

Σ is a set of symbols called *alphabet*;

Q is a finite set of *states*;

$q_\lambda \in Q$ is the *start state* or *initial state*;

$\delta : Q \times \Sigma \to Q$ is the transition function whose inputs are Q and a symbol of Σ and returns a subset of Q as output.

$\mathbb{F}_\mathbb{A} \subseteq Q$ e $\mathbb{F}_\mathbb{R} \subseteq Q$ are sets of state called respectively *accepting* and *rejecting* states.

DFA is a system very similar to the previous one:

$$A = (\Sigma, Q, q_\lambda, \mathbb{F}_\mathbb{A}, \mathbb{F}_\mathbb{R}, \delta),$$

where the only difference with the previous definitions regards the transition function δ. In this case, it returns only one state of Q.

NFA and DFA are equivalent: a DFA can be always transformed into a NFA, preserving the language recognized and viceversa. NFAs are more compact with respect to DFAs; it can be shown that, in the worst case, 2^n states are needed by a DFA equivalent to a NFA with n states.

Similarly to grammars, a DFA recognizes only one language, but a language is recognized by several DFAs. Among all the DFAs that recognize a regular language, the *minimal canonical automaton* has a particular importance. It is unique for each language (but for a renaming of the states) and has a central role in the process of grammatical inference, because all this process can be turned in a search for the minimal canonical automaton.

4.1.4 Inferring a language

This section analyzes the inference of a language from a set of its samples, focusing on regular languages.

As stated in [90], inferring, or better *identifying* a language is the main concern of GI, that is the process of searching for a hidden grammar by little information available, often only a set of strings.

GI is contained in the wider framework of ALT, a mathematical framework to study machine learning problems and algorithms [102]. ALT is based on the concept of *learning in the limit*: increasing the number of samples, the learning algorithm should identify the correct hypothesis on every possible data sequence consistent with the problem space. This idea is a non-probabilistic equivalent of statistical consistency, where the learner can fail on data sequences whose probability measure is 0.

Central objects of ALT are TMs, thus grammatical inference through DFAs can be declined in this framework. *Language learnability models* are one of the most relevant concepts in ALT.

A language learnability model has three main components:

1. a *definition of learnability*: it states what learning a language means;

2. a *method of information presentation*: how the learner is instructed during the learning process;

3. a *naming relation*, which assigns names to languages: the “learner” identifies a language by stating one of its names.

In this context, learnability corresponds to the identification in the limit principle.

A presentation is a function $\phi : \mathbb{N} \rightarrow$ X, where X is some set. The set of all possible presentations for a class of languages $\mathcal{L}$ is indicated by Pres($\mathcal{L}$). A presentation can be considered as an enumeration of the elements in some set X; this set can be a set of strings drawn from Σ^*, but, in a broader sense, it is a sequence of information of some type that guides the identification of the target language.

A *presentation mode* describes what valid presentations are and the way in which the set X is created. Moreover, presentations indicate languages of $\mathcal{L}$; in other words, a function can be defined from Pres($\mathcal{L}$) to $\mathcal{L}$: Y : Pres($\mathcal{L}$) $\rightarrow \mathcal{L}$. There are two main modes of presentation for a language L:

- from *text*: a sequence of strings $(x_1,\ x_2,\ \dots)$ belonging to the language L is provided; every string of L appears at least once in the sequence. This presentation is known also as *positive* presentation.

$$T(L) = \{\phi : \mathbb{N} \rightarrow \Sigma^* : \phi(\mathbb{N}) = L\};$$

- from *informant*: the learner is supplied with strings marked as *positive* (belonging to the language L) or *negative* (not in L). This kind of presentation is known as *complete*.

$$I(L) = \{\phi : \mathbb{N} \rightarrow \Sigma^* \times \{0,\ 1\} : \phi(\mathbb{N}) = L \times \{1\} \cup \overline{L} \times \{1\}\},$$

where $\overline{L}$ indicates the complement of L with respect to Σ^*.

Grammars are the chosen representation for the languages, thus the naming function is a surjective function with the grammar set as domain and the set of languages as codomain: $\mathbb{L} : \mathcal{G} \rightarrow \mathcal{L}$.

Using the previous definitions, a learning algorithm can be defined as [90]:

Definition 1. A *learning algorithm* $\mathcal{A}$ is a function whose inputs are the first n elements of a presentation and whose output is a grammar:

$$\mathcal{A} : \{\phi_i : i \in \mathbb{N}, \phi \in \text{Pres}(\mathcal{L})\} \to \mathcal{G}$$

4.1.5 Identification in the limit

A more formal definition of the identification in the limit is the following [91]:

Theorem 2. *Let:*

- $\phi = \text{Pres}(\mathcal{L})$ *be a presentation of the languages in* $\mathcal{L}$*;*
- ϕ_n *be the set of the first* n *strings from* T.

The class of languages $\mathcal{L}$ *is learnable by algorithm* $\mathcal{A}$ *if:*

$$\forall L \in \mathcal{L}, \forall \phi \in \text{Pres}(\mathcal{L}), \quad \exists m : \mathbb{L}(\mathcal{A}(\phi_n)) = L, \ \forall n \geq m$$

Theorem 2 states that a learner can identify a language in the limit if, after a number of presented strings, *its hypothesis no longer changes*. Applying Theorem 2, it can be shown that every TM can be identified in the limit by another Turing-complete Machine by enumeration.

With a slight abuse of notation, a presentation of a grammar G can be defined as $\text{Pres}(G) = \text{Pres}(\mathbb{L}(G))$. Moreover, taking advantage of Definition 1, the learnability in the case of GI can be expressed as follows [90]:

Definition 2. The class of $\mathcal{G}$ is identifiable in the limit from $\text{Pres}(\mathcal{G})$ if there exists a learning algorithm $\mathcal{A}$ such that:

$$\forall G \in \mathcal{G}, \forall \phi \in \text{Pres}(\mathcal{G}), \quad \exists m : \mathbb{L}(\mathcal{A}(\phi_n)) = \mathbb{L}(G) \quad \text{and} \quad \mathcal{A}(\phi_n) = \mathcal{A}(\phi_m), \ \forall n \geq m$$

It is worth noting that Definition 2 implies that a grammar equivalent to the target grammar G is learned. Moreover, the learning algorithm $\mathcal{A}$ does not change its output anymore from a given point on. In some context, a *behaviorally correct identification* is enough; in this identification, the learner can change the grammar with an equivalent one; in this case the condition $(\forall m \geq n, \mathcal{A}(\phi_n) = \mathcal{A}(\phi_m))$ can be discarded.

4.1.6 Identification and regular languages

Gold showed in [91] that a class of *super-finite languages*[†] can not be identified from a text presentation. The class of regular languages is super-finite, thus regular languages can not be inferred only from positive examples; in other words, a set of strings belonging to the target language is not sufficient to learn it.

Some limitations about learning with a presentation from an informant also exist, as Gold pointed out [91]:

Theorem 3. *The whole class of recursive languages can not be identified in the limit from a complete presentation.*

However, in the same work, Gold showed that:

Theorem 4. *The class of primitive recursive languages can be identified in the limit by a complete presentation.*

This class of languages contains also the regular language class, therefore a regular language can be identified in the limit from a complete presentation of examples.

4.1.7 Inference as a search

Given a complete presentation $I = I_+ \cup I_-$, the minimum canonical automaton consistent with I exists and is unique, as showed in [100]. Thus, the inference problem can be turned into a search for this automaton; but Gold showed that finding the minimum consistent automaton with a set of

[†]A super-finite language class is a class that contains all finite languages and at least one infinite language.

samples is a NP-hard problem. Therefore, some heuristics are needed to carry out this search in an efficient way.

The search space can be sketched through the following basic elements:

- *Initial node*: an “acceptable” DFA;
- *Successor function*: pairwise state merging;
- *Target*: minimum automaton that is consistent with the samples I.

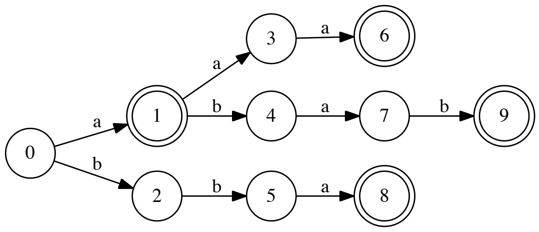


Figure 4.1: $PTA(I_+)$ with $I_+ : \{a, aaa, abab, bba\}$

In [103], Dupont describes this search space as a boolean lattice. The initial node of this space is the so-called *Prefix Tree Acceptor* (PTA), a tree automaton accepting only the positive examples I_+. A PTA is shown in Figure 4.1.

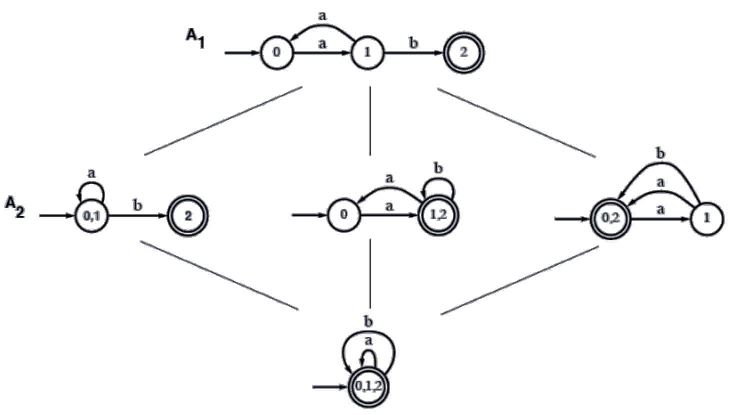


Figure 4.2: Example of pairwise merging operation: from automaton A_1, the three automata of the second line are obtained.

The merging operation is a partition of the set of states of the original automaton. Formally, if $A_1 = (\Sigma, Q, q_\lambda, \mathbb{F}_\mathbb{A}, \mathbb{F}_\mathbb{R}, \delta)$ is the original automaton and π is a partition of the set of its states, then the obtained automaton applying π is called quotient automaton: $A_2 = A_1/\pi = (\Sigma', Q', q'_\lambda, \mathbb{F}'_\mathbb{A}, \mathbb{F}'_\mathbb{R}, \delta')$ and its elements are defined as follows:

- $\Sigma' = \Sigma$
- $Q' = Q/\pi = \{B(q,\pi) | q \in Q\}$
- $\mathbb{F}'_\mathbb{A} = \{B \in Q' | B \cap \mathbb{F}_\mathbb{A} \neq \phi\}$
- $\delta' : Q' \times \Sigma \to 2^{Q'} : \ \forall B, B' \in Q', \ \forall a \in \Sigma, B' \in \delta'(B,a) \iff \exists q, q' \in Q, q \in B, q' \in B'$ and $q' \in \delta(q,a)$

where $B(q,\pi)$ denotes the unique element, or block, of π containing q. The states of Q in the same block B of the partition π are said to be merged together.

The set of successors of an automaton is generated by pairwise merging operations: two states of the original automaton are merged, giving a new automaton with a number of states decreased by one, as shown in Figure 4.2.

The pairwise merging operation is also known as *derivation operation*. Suppose that $P(A)$ is the set of all the possible partitions of the set of states of the automaton A and let π_1 and π_2 two items of $P(A)$. The partition π_2 is said to be directly derived from π_1 if:

$$\pi_2 = \{B_{1j} \cup B_{1k}\} \cup (\pi_1 \backslash \{B_{1j}, B_{1k}\}),$$

for some j, k between 1 and the number of blocks in π_1, with $j \neq k$.

The derivation operation defines a partial order relation $\preceq$ on $P(A)$, whose transitive closure will be indicated with $\ll$. Thus, if $\pi_1 \ll \pi_2$, then, as extension, $A/\pi_1 \ll A/\pi_2$. By construction of the quotient automaton, the property of language inclusion holds [103]:

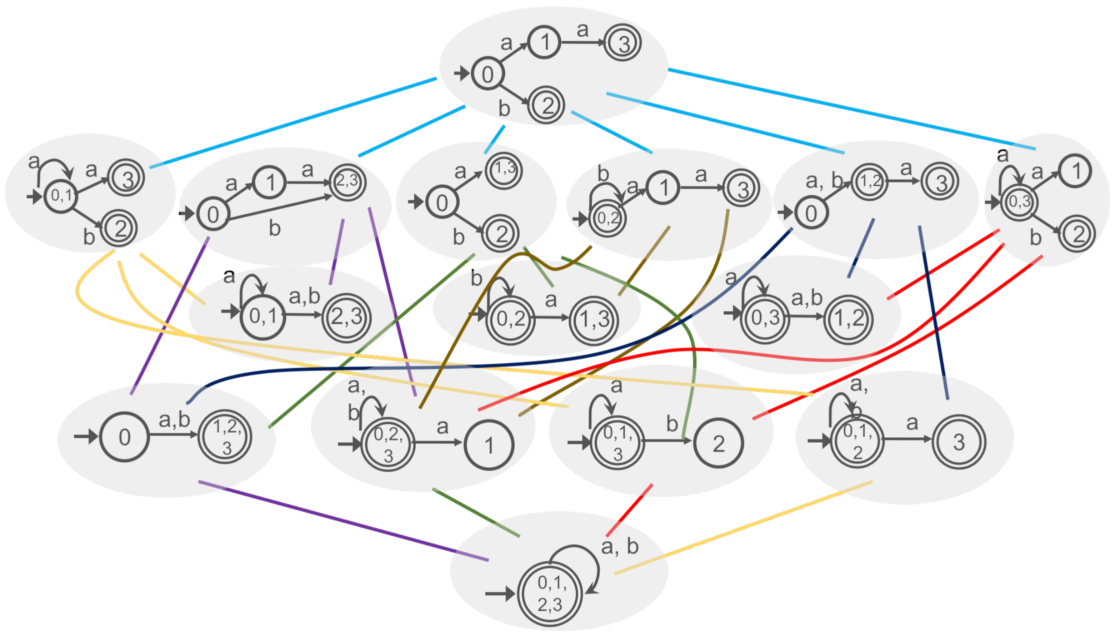


Figure 4.3: Example of boolean lattice.

Theorem 5 (Property of language inclusion). *Let $P(A)$ the set of all partitions of the states of an automaton A and $\pi_i \ll \pi_j$, with $\pi_i, \pi_j \in P(A)$. Then, the language identified by the quotient automaton $L(A/\pi_i)$ is included in the language of $L(A/\pi_j)$:*

$$A/\pi_i \ll A/\pi_j \quad \textit{if} \quad L(A/\pi_i) \subseteq L(A/\pi_j).$$

Applying the merging operator has two possible consequences: in the first case, only the number of states is decreased and the recognized language is preserved; in the second case, the reduction in the number of states is followed by a change in the language recognized by the resulting automaton; indeed, the language accepted is more general, properly including the original one.

The set $P(A)$, along with the partial order relation, defines the boolean lattice $Lat(A)$. The nodes of this lattice are the quotient automata, obtained by applying merging operations included in $P(A)$ to the automaton A. The deepest node in $Lat(A)$ is the *Universal Automaton* (UA), that accepts all the strings defined over an alphabet Σ, i.e., $L(UA) = \Sigma^*$. An example of a boolean lattice is shown in Figure 4.3.

The inference of regular languages, provided a presentation from an informant, can be turned into the search for an automaton $A' \in Lat(PTA(I_+))$, given the additional hypothesis of structural completeness of I_+, that can be defines as [103]:

Definition 3. A I_+ sample set is said to be structural complete with respect to an automaton A, if :

1. every transition of A is used by at least a string in I_+;
2. every state in $\mathbb{F}_{\mathbb{A}}$ is the final state of at least a string in I_+.

Under these conditions, the following theorem can be demonstrated [103]:

Theorem 6. *Let I_+ a structural complete sample with respect to the minimal automaton A accepting a regular language L; then A belongs to $Lat(PTA(I_+))$.*

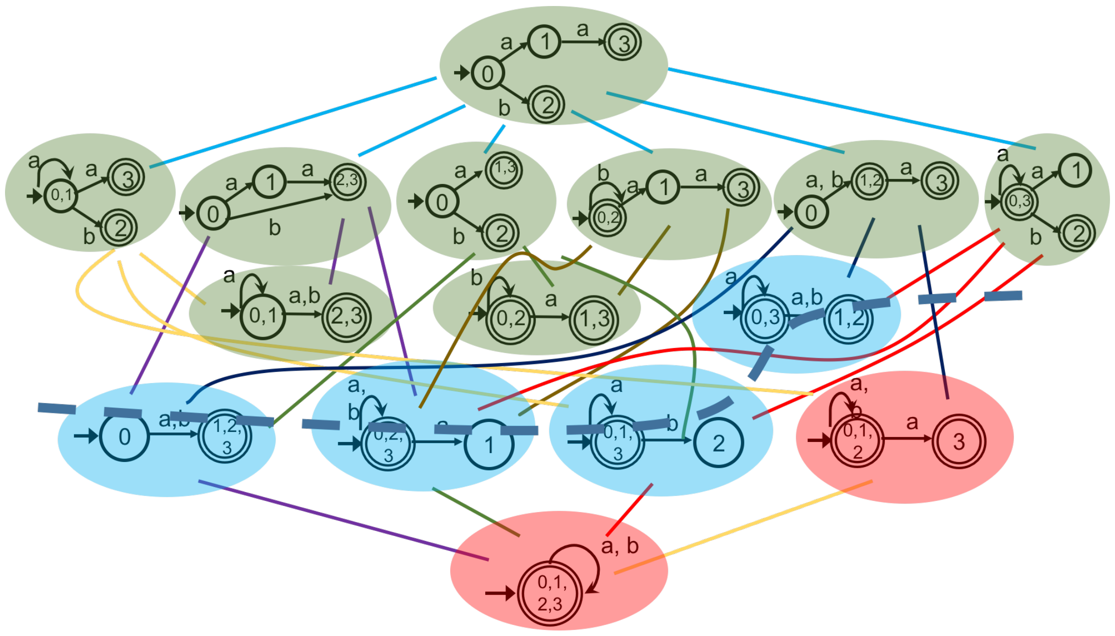


Figure 4.4: Boolean lattice decomposition: admissible (in green), inadmissible (in red) and border-set (in blue) automata.

The definition of minimal DFA consistent with the sample set I can be expressed using the elements of boolean lattice, in terms of the so-called *Border Set*; but to define border set, *antistring* and *automaton at maximal depth* are being defined:

Definition 4. The antistring of a lattice of automata is the set of automata whose elements are not related with any other element of antistring by a $\preceq$ relation.

Definition 5. An automaton is at maximal depth in a lattice if there is no automaton A' that can be derived from it such that $A' \cap I_{\_} = \emptyset$.

Definition 6. The Border Set $BS_{PTA}(I_+, I_{\_})$ is he set of automata of $Lat(PTA(I_+))$ of which each element is at a maximum depth.

Thus, the border set establishes the limit of generalization in the search process under the control of negative samples $I_{\_}$. So, the minimum DFA consistent with I is the smallest automaton of the border set, i.e., the deepest one. Moreover, the border set parts the lattice into two main subsets: admissible automata A_A, i.e., $A_A \cap I_{\_} = \emptyset$, and inadmissible ones A_I, i.e, $A_I \cap I_{\_} \neq \emptyset$. Figure 4.4 shows a decomposition of lattice according to this classification.

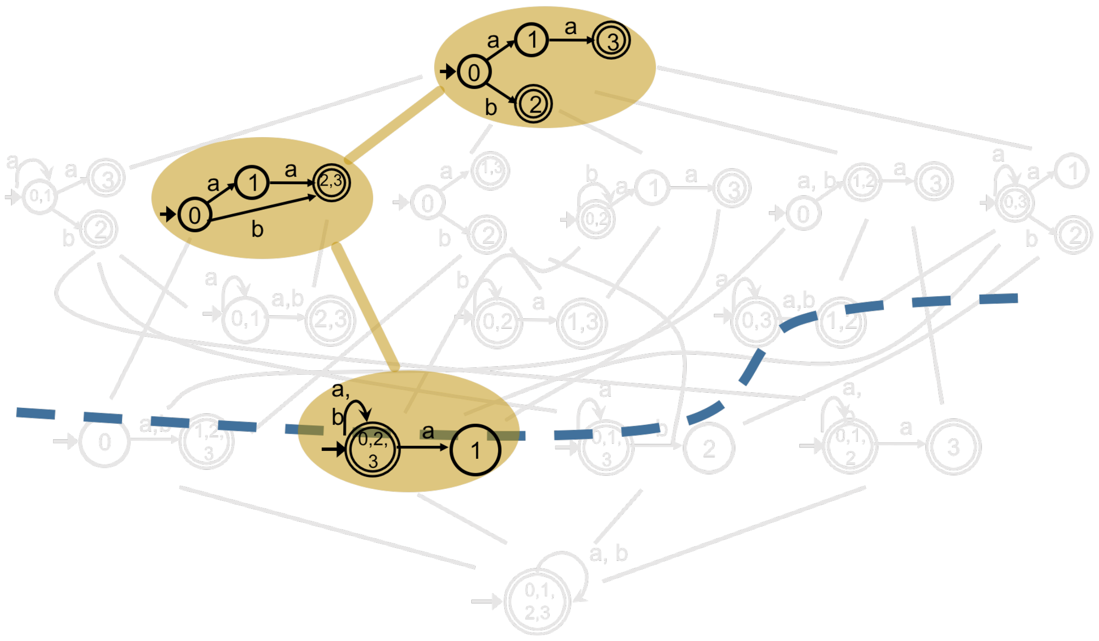


Figure 4.5: Sketch of a search in the boolean lattice.

Theorem 6 guarantees that the target of the search belongs to the boolean lattice. Unfortunately, this space is too large to be searched extensively. Indeed, the number of automata in the lattice generated by an initial PTA with n states is given by the *Bell number*:

$$\omega(n) = \sum_{p=0}^{n-1} \binom{n-1}{p} \omega(n-1),$$

with $\omega(0) = 1$.

Therefore, some approaches have been proposed to carry out the search in the boolean lattice in an efficient way, starting from the PTA towards the minimal automaton, as shown in Figure 4.5.

Evidence-Driven State Merging (EDSM) algorithm represents a state-of-the-art algorithm to perform a search in boolean lattice, and detailed description can be found in [93].

In Algorithm 2, the psuedo-code of EDSM, as presented in [90], is shown.

EDSM is employed as inference algorithm in the approach proposed by this thesis. EDSM is an iterative algorithm in the *blue-red* framework, introduced to reduce the number of comparisons for merging options, in order to choose the most promising one. Red nodes represent already identified node, while blue nodes are the current options for merging.

At the beginning of the algorithm, the root node is marked as red, while its children as blue. At each iteration, the algorithm tries to merge a blue node with a red one; if no merge is possible, its colour is changed to red (the node is *promoted*); then, its uncoloured children are marked as blue.

Basically, the main steps of the algorithm can be sketched as follows:

1. Given a structural complete sample set I, create $PTA(I_+)$ and mark the root node as red and its children as blue.

2. Compute a score for each couple of red and blue nodes.

3. If there exists a blue node that can not be merged with any red one, promote the blue node, mark its uncoloured children as blue and go back to step 2.

4. If there is no blue node to promote, chose the merge with the highest score and go back to step 2.

Algorithmus 2 EDSM

```
Input: $I = I_+ \cup I_-$
Output: $A = (\Sigma, Q, q_\lambda, \mathbb{F}_\mathbb{A}, \mathbb{F}_\mathbb{R}, \delta)$
 1: $A \leftarrow$ PTA($I_+$)
 2: Red $\leftarrow \{q_\lambda\}$
 3: Blue $\leftarrow \{q_a : a \in \Sigma$ and $I_+ \cap a\Sigma^* \neq \emptyset\}$

 4: while Blue $\neq \emptyset$ do
 5:     promotion $\leftarrow$ false
 6:     bs $\leftarrow -\infty$
 7:     for $q_b \in$ Blue do
 8:         if not promotion then;
 9:             atleastonemerge $\leftarrow$ false;
10:             for $q_b \in$ Red do
11:                 $sc \leftarrow$ COUNT(MERGE($q_r, q_b, A$), $I_+, I_-$);
12:                 if $s > -\infty$ then
13:                     atleastonemerge $\leftarrow$ true;
14:                 end if
15:                 if $s > bs$ then
16:                     $bs \leftarrow s$;  $\overline{q_r} \leftarrow q_r$;  $\overline{q_b} \leftarrow q_b$;
17:                 end if
18:             end for
19:             if not atleastonemerge then
20:                 PROMOTE($q_b, A$)
21:                 promotion $\leftarrow$ true;
22:             end if
23:         end if
24:     end for
25:     if not promotion then
26:         Blue $\leftarrow$ Blue $\{\overline{q_b}\}$;
27:         $A \leftarrow$ MERGE($\overline{q_r}, \overline{q_b}, A$)
28:     end if
29: end while

30: for all $x \in I_+$ do
31:     $\mathbb{F}_\mathbb{A} \leftarrow \mathbb{F}_\mathbb{A} \cup \{\delta(q_\lambda, x)\}$
32: end for
33: for all $x \in I_-$ do
34:     $\mathbb{F}_\mathbb{R} \leftarrow \mathbb{F}_\mathbb{R} \cup \{\delta(q_\lambda, x)\}$
35: end for
36: return $A$
```

The function COUNT (at line 11 of Algorithm 2) returns the number of strings that would end in the same state, if q_r and q_b were merged; the function returns $-\infty$ if the merge makes the automaton inadmissible, i.e., an element of I_- has been accepted or an element of I_+ has been rejected. The pair with the highest score is chosen.

4.2 The language of paths

A mobility model is a concise and meaningful representation of past and future mobility behaviors.

Nowadays, location data is easy to collect, due to availability of a wide set of common devices, such as smartphones or tablets, that can provide a great quantity of positioning data. The discovery of meaningful information from this huge amount of data is an open issue.

Several works [104]–[106] revealed that human spatial trajectories are highly predictable; thus, regular languages can be an adeguate tool to capture and compress regularities in an effective representation. This motivates for employing regular languages to describe users' mobility patterns, identifying the "language of paths". Moreover, regular languages are selected among all the other classes of the Chomsky hierarchy, because the inductive process for this class of languages is very efficient.

In the following, an approach to infer and represent user mobility models via regular languages is described. The first step of this process is to translate paths into a symbolic representation and it is accomplished by geohash encoding. Then, geohash representation of paths is used in order to build up a hierarchy of DFAs, representing a hierarchical mobility model that describes user habits at different scales.

This model has several interesting applications:

- *Mobility pattern recognition*: the model can recognize trajectories compatible with usual user behavior.
- *Future trajectory prediction*: the model can infer plausible trajectories based on structural properties and considerations.

- *Anomaly detection*: the model can detect changes in user habits.
- *Point of Interest (POI) extraction*: the model can suggest interesting place, in a user-centric fashion.
- *Synthetic samples*: the model can generate realistic trajectories to simulate user behavior.

In next sections, a detailed description of this approach is provided, along with a test about its recognition abilities.

4.2.1 Human mobility models

In the last years, the steady diffusion of positioning system have generated large volumes of mobility data, giving raise to the research field of movement data analysis. The main aim of this new research area is to find tailored solutions to mine movement data and get significant insights in frequent patterns travelled by users, in order to predict their future movements. Movement data analysis has adapted techniques borrowed from several data mining approaches, originally studied for transactional-based systems.

Thus, a wide literature has been accumulated on the topic of mobility models and their applications; for example, the topic of mobility models is crucial in the analysis and simulation of opportunistic networks, that are based on opportunistic contacts for peer-to-peer message forwarding [107].

Tourism is another application scenario for mobility models [108], because trajectory data contains sequences of locations that are frequently visited, that are very valuable in the identification of POIs.

Human mobility models have also an application in social sciences, because their analysis can explain and provide a better understanding of social phenomena. In [109], authors propose an analysis of data gathered from monitoring of users freely moving in a university campus, aiming at getting new insights of the life in the campus.

Vehicular ad-hoc Network (VANET) is a further application scenario for mobility models. The prohibitive cost of deploying and implementing system for VANET pushes towards the creation of realistic simulators of vehicular movements, exploiting mobility models to get reliable results from simulations [110].

Two main types of mobility models emerge from this literature [111]: synthetic and trace-based models.

Synthetic models are often aimed to the automatic generation of mobility traces, based on graph models [112], or using vehicle profiles [110]. They offer a straightforward mathematical framework to experiment and test with mobility behavior. A survey of this type of models can be found in [111].

Trace-based models are created by real traces, and are therefore more accurate. Their main drawback is the need of a large amount of positioning data, collected during a sufficient period of time, to make them reliable [113].

Traces are obtained from characterization of urban spaces [114], or are collected by smart devices [115].

Previous works have investigated mobility data mining to extract grammar models. The authors of [116] use *Probabilistic Context-Free Grammars* (PCFG) to model network observations; they propose a new inference algorithm and a definition for PCFGs, oriented toward mobility data. In [117], FSAs were used to model mobility behaviors. Authors propose two approaches: in the first, the alphabet is made up of the "status" of the user and the states of the automaton are the locations (e.g., at home, at work); in the second one, the role of locations and "status" are switched. Locations were inferred through unsupervised learning algorithms, mining the most visited places; "status" categories are extrapolated from temporal sequences of movements.

An approach based on grammar induction to analyze spatial trajectories was investigated in [118]. A grammar induction algorithm, called *mSEQUITUR*, was proposed; it is able to obtain a grammar rule set from a trajectory for motif generation. Moreover, the authors present the *Trajectory Analysis and VIsualization System* (STAVIS), a trajectory analytical system that derives trajectory signatures and allows to extract relevant information from them, using a grammar inference algorithm.

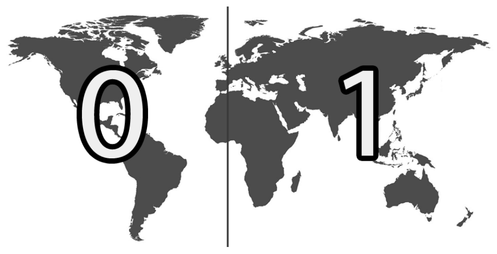


Figure 4.6: Areas corresponding to first bit of geohash string.

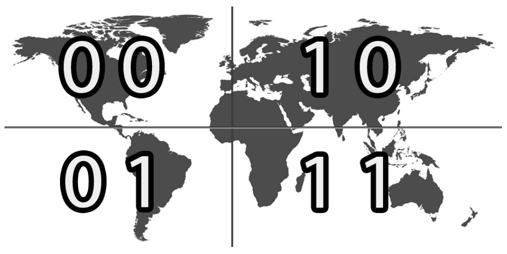


Figure 4.7: Areas associated with the first two bits of a geohash string.

4.2.2 A hierarchical encoding: Geohash

Geohash is an encoding system developed by Gustavo Niemeyer for geographical coordinates. It assigns a hash string to each (latitude, longitude) pair; originally, it was developed to provide a smart and easy representation of URLs, but then it has been widely used to store spatial coordinates into databases [119]. Geohash is based on a hierarchical spatial data structure that recursively subdivides world into "buckets" of grid shape; unlike coordinate systems, it does not actually represents a point, rather a bounding area in which the point is restricted.

The geohash algorithm partitions the space using a grid composed by 32 cells, arranged in 4 rows and 8 columns; each cell can be recursively divided into 32 cells, providing a hierarchical structure that corresponds to a recursive quadtree. Geohash representation marks each cell with an alphanumerical character from its alphabet, made up of 32 symbols, i.e., *{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, b, c, d, e, f, g, h, j, k, m, n, p, q, r, s, t, u, v, w, x, y, z}*. The alphabet symbols are associated to each cell adopting the Z-order, allowing an easy computation of the next cell character by switching some bits.

In the geohash string, bit in even positions encode the longitude information, while those in odd positions encode the latitude. For example, in the first phase of the encoding, the whole world is parted into two rectangle, according to the longitude, as showed in Figure 4.6.

The next bit is obtained through a longitudinal partition of the rectangles obtained by the previous step, as showed in Figure 4.7.

This process can be iterated until the desired spatial accuracy is obtained. The length of the binary string must be a multiple of 5 to allow its conversion to a sequence of symbols from geohash alphabet. Indeed, each symbol is associated with a 5-bit code, thus, the binary string can be partitioned into substring of 5 bits, that are replaced by corresponding symbols [120]. Table 4.1 shows the correspondence between the binary string $11000101100101100010111110111111100$ and its geohash *sqc2zgw* counterpart. Clearly, every geohash string identifies a particular cell in the hierarchical representation.

s	q	c	2	z	g	w
11000	10110	01011	00010	11111	01111	11100

Table 4.1: Geohash string and its binary representation

Therefore, the obtained representation is based on the principle of gradual degradation: the longer the geohash string, the smaller the area. Table 4.2 shows the size of the area identified by a geohash code with respect to its length. It is worth noting that extending a geohash string by a character decreases the area of the identified cell of a factor $2^5 = 32$ with respect to the original one.

Geohash length	$\approx$ Covered Area km^2
1	16.000.000
2	500.000
3	15.000
4	500
5	15
6	0.5
7	0.02

Table 4.2: Area covered by a cell with respect to the length of its geohash encoding string.

Inclusion property is a notable property of geohash encoding: it is always possible to add a character to a geohash string, obtaining a new string that identifies a cell contained into the original one. For example, the coordinates (38.120281, 13.357278) identify a point included inside the *sqc2zg* cell, but also inside *sqc2zgw* or *sqc2zgwk*.

Locality property is another property of geohash: strings with common prefix mark contiguous cells. Thus, it is very simple to check if two cells are neighbors. The converse is not always true: two cells could be next to each other even if they do not share a common prefix.

4.2.3 Mobility models as automata

In this thesis, it is claimed that mobility models can be successfully represented by languages, specifically regular languages.

In the previous sections, the main elements needed for language inference have been outlined. In this section, a description of mobility models as languages and of a method to infer them from mobility data are provided.

In the proposed approach, data is provided as *movement tracks* [108]:

Definition 7 (Movement track)**.** The movement track is the temporally ordered of spatial-temporal position records captured by a positioning device during the whole lifespan of the user observation. Each record contains a position and the instant of the capture. There are no two records with the same instant value.

Movement tracks are the raw data collected from a positioning system monitoring user movements. They have to be turned into *trajectories* [121] to be used, in order to filter out noise, and to estimate other movement features, such as speed and direction. *Paths* are the true aim of the analysis:

Definition 8 (Path)**.** A path is the portion of a trajectory between two relevant points in time or space dimensions.

Paths reveal user behavior and highlight relevant places where the user spends most of his time. Knowing these places is crucial in many applications, and they are fundamental in comparing habits of several users or in recognizing anomalies or changes in their routines.

Paths have a *multiscale* nature: significant information can be extracted by observing data at different scales. For example, usual path of the user could be the route from home to the workplace,

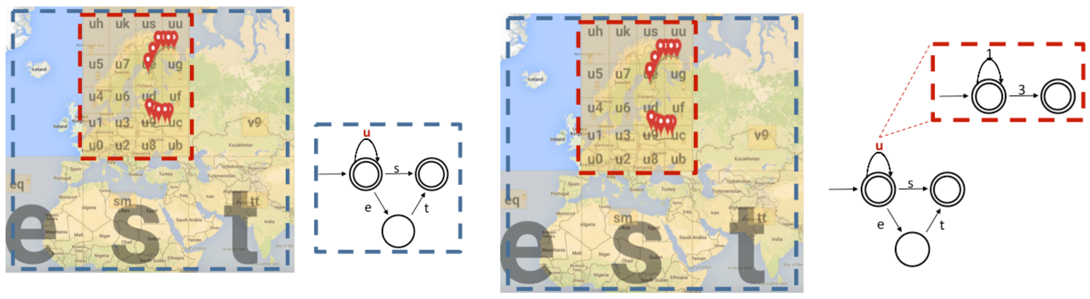


Figure 4.8: From trajectories to a hierarchy of DFAs: given the DFA of the cell containing the sub-cell "*u*" (figure on the left), a more detailed model can be built up inferring the DFA of the language "u" (figure on the right).

because it recurs almost every day; but it can be very different in scale for different users. Consider a user whose workplace is in another city with respect to his home: his paths crosses a wide area, compared to the same type of path in the case of workplace and home are in the same city; so, it compares at different scale of data in the two cases. Moreover, frequent paths of the same user can have different scales: a user can daily move across two cities to get to his workplace, but everyday he also moves, for example, from his workplace to the place where he has lunch, that is probably close to his workplace.

Moreover, paths share the same recursive structure illustrated for human activities in the previous chapters. Thus, a path can be decomposed into simpler paths, that are composed by even simpler paths, and so on. Therefore, all the considerations about the analysis and representation of recursive structures hold even in this scenario.

Trajectories are "geohashed", turning each pair of coordinates into the correspondent geohash string. So, trajectories are sequences of strings, whose alphabet is the set of the geohash symbols. Encoded with this representation, trajectories can be analyzed at different spatial scale: once the required precision is fixed, it sufficient to recover the correspondent length of geohash string and truncate every string of each trajectory at that length.

The user mobility model is decomposed following his behavior related to every cell of geohash encoding: a regular language is learned for each cell of the geographical area crossed by user movements, starting from the highest level of granularity. Then, a regular languages is inferred for each sector in which each element of the grid can be decomposed, as showed in Figure 4.8. The process ends when the level of cells representing the required accuracy is reached.

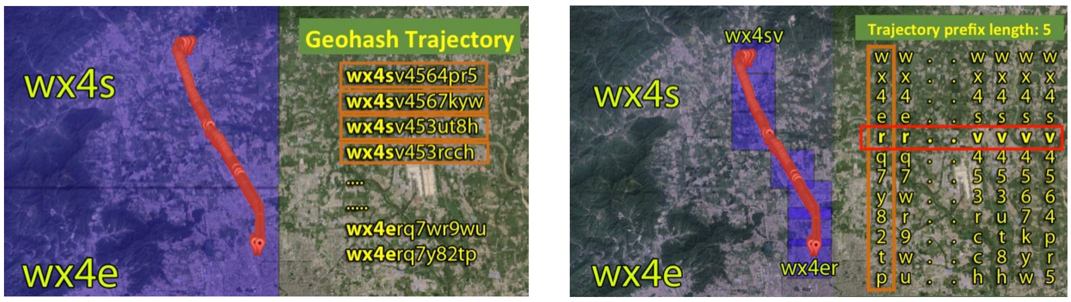


Figure 4.9: Extracting mini trajectories from trajectories: the image on the left shows a trajectory in its geohash encoding; each string (inside the orange rectangle) corresponds to a location. The image on the right shows how subsequences with the same prefix (arranged in columns) originate mini trajectories: the fifth element of each strings is concatenated to obtain a mini trajectory (marked with the red box).

Mini trajectories are a basic element of mobility models construction; they describe planar movements that take place in each cell. Indeed, inside each cell, the mobility model is described by sequences of contiguous movements among sub-cells, producing a correspondent sequence of geohash symbols, i.e., the sub-cell symbols. Mini trajectories can be obtained for each cell by considering all the contiguous subsequences of strings inside each trajectory that share the prefix corresponding to the cell. For each element of the subsequence, only the symbol of the sub-cell is considered, thus the subsequence is turned into a string (see Figure 4.9); after recovering all the strings related to the cell, the needed information to infer a regular language is obtained.

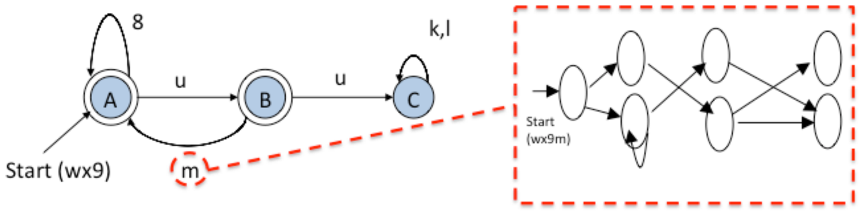


Figure 4.10: Hierarchichal structure of DFAs: each transition can be substituted with the correspondent DFA, obtaining an enhanced automaton.

The inference process provides a DFA that reflects the described structure. Once a DFA has been learned for a particular cell of the geohash representation, the transition function represents user movements inside that cell: each transition stands for user moving to a particular sub-cell. The behavior of the user inside the sub-cell is described by the regular language corresponding to the symbol of that cell. This representation allows a simple navigation between the different spatial scales of the model; indeed, to increase resolution and get a more detailed model, it is sufficent a "hierarchical" navigation through the pool of automata, substituting to each symbol the correspondent language, i.e DFA, obtaining a more complex and detailed automaton (Figure 4.10); this is equivalent to concatenating a new symbol to the geohash string, and inspecting the movements of a new level of detail. The new automaton, identified by the built prefix, encodes details about the movements in an area increasingly smaller and detailed.

In the previous sections, it has been shown that a regular language can be inferred only with a presentation from an informant; thus, to obtain the mobility models for a user, a set of examples of his paths are not enough. The proposed approach considers the symmetric difference between the set of trajectories of other users and trajectories of current user as the negative sample set. This set represents viable routes chosen by other users, which have not been traversed by the current user, so it can be considered as negative sample for the language that represents mobility habits of the current user.

Given the mini-trajectory sets of negative and positive route samples, the correspondent regular language is inferred by the EDSM algorithm.

The whole inference process, from the set of trajectories to the final pool of DFAs, is framed into a system, made up of three independent modules:

- **Mini Trajectory Database Manager**: it converts the ellipsoidal coordinate to geohash strings. For every prefix of variable length from 0 to 6, it stores a record in a database. For each prefix (one for every cell crossed by at least a relevant number of trajectories), it computes the set of mini-trajectories for all users. Thus, mini trajectories are searchable by user or cell.
- **Inference Processor**: it requests data to compute the mobility model of a user and executes EDSM. At the end of the inference process, it returns the hierarchical pool of DFAs represent-

ing the mobility model of the user.

- **Mobility Model Handler**: it exploits some features of the model, such as the ability to recognize if a path belongs to a user, or to produce a numbers of synthetic paths generated through a mobility model.

Each module was designed and implemented focusing mainly on efficiency and independence, aiming at obtaing self-contained systems.

4.3 Experimental assessment of mobility model extraction

The proposed approach has been tested on the data provided by the *Geolife* dataset [122], collected at Microsoft Research Asia; the huge volume of data classifies it as big data source. It is a collection of time-stamped points (latititude, longitude and altitude), monitoring spatial behaviors of 182 users for 5 years, gathering spatial and temporal information about their movements. The majority of trajectories are located in China, near Beijing; but there are also some trajectories from the USA and Europe. More than $17,000$ trajectories are contained the database, for a total amount of approximately $50,000$ hours of tracked routes. *Global Positioning System* (GPS) logger and smartphones acted as acquisition devices, providing a high density sampling rate ($1 \sim 5$ seconds in time, and $5 \sim 10$ meters in space) for more than 90% of the data. Different kinds of movements were monitored, related to daily activities: going and coming back home from workplace, entertainment activities, such as shopping or riding a bike or walking.

The assessment was conducted considering all the users in the dataset and all the trajectories. The whole dataset was parted into training and test set, in order to assess the generalization ability of the Inference Processor.

Two experiments were conducted to test the accuracy of the proposed approach, fixing the ratio between training and test set respectively at 80/20 and 60/40 for each user. The string length for geohash encoding was set to 7, corresponding to a precision of 153m. A mobility model was inferred for the user from training data, and then its generalization ability was assessed in recognizing trajectories from the test set.

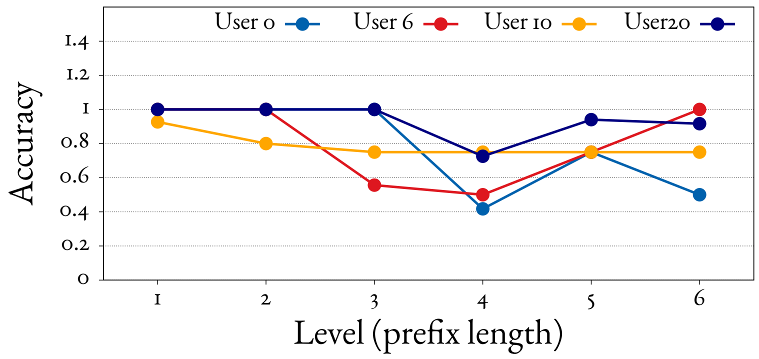


Figure 4.11: Accuracy at six different levels of granularity for four users (80% training, 20% test).

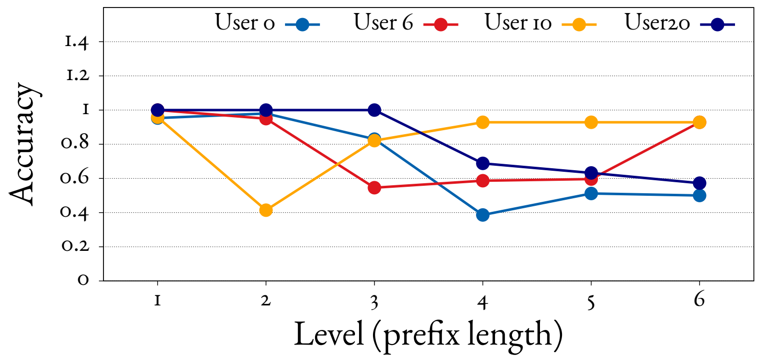


Figure 4.12: Accuracy at six different levels of granularity for four users (60% training, 40% test).

Results for 4 representative users are reported in Figures 4.11, 4.12 and show that the proposed approach provides a high rate of accuracy at all spatial scales, and its performances are not influenced by the resolution of the trajectories.

Figures 4.13, 4.14, 4.15 shows some examples of inferred DFAs; they demonstrate that mobility models encoded through automata can be very simple to understand and capture the most relevant characteristics of the data, despite its huge complexity.

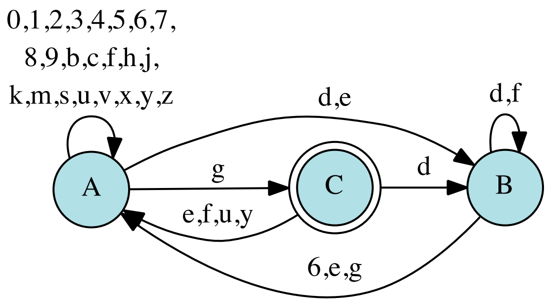


Figure 4.13: One of the DFA included in the mobility model of user 6 (prefix lenght 2, 60% training, 40% test)

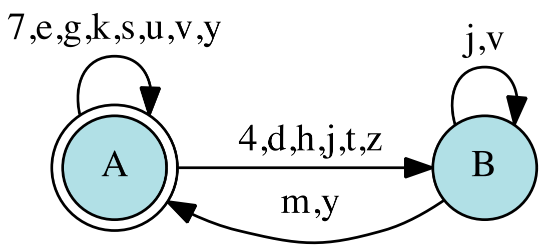


Figure 4.14: One of the DFA included in the mobility model of user 6 (prefix lenght 4, 60% training, 40% test)

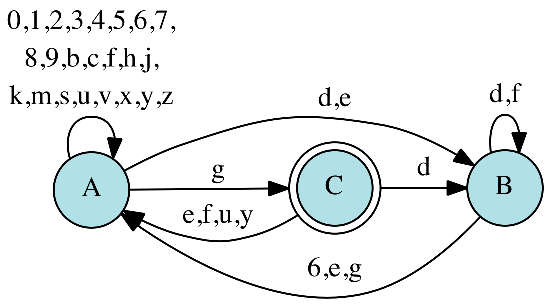


Figure 4.15: One of the DFA included in the mobility model of user 6 (prefix lenght 6, 60% training, 40% test)

In theory, theory and practice are the same. In practice, they are not.

Albert Einstein

5 Conclusion

This thesis has described an approach to cope with the complexity of knowledge representation and extraction in sensory data; this is a very challenging and complex task, due to the great amount of available raw data and to its heterogeneity. Moreover, well-established approaches are not directly suitable for this scenario, because knowledge discovery raises new challenges due to the huge complexity hidden in data. One of the main issues is the difficulty to introduce the right amount of a-priori knowledge into the system; hence, algorithms can only be tuned through a set of parameters, barely correlated with original data.

Hence it is claimed here that a structural approach to knowledge extraction and representation is the key to enhance and improve the quality of the obtained models. Structural representations can provide more human-understandable models, aiding the designer of the system in tuning and improving the process of knowledge extraction, and obtaining more generalizable representations, as compared to classical approaches.

Three different case studies, which implement this idea, have been presented, each one exploiting a different approach based on statistical learning, syntactical methods and formal languages respectively.

It has been shown that different techniques can be successfully employed in order to recover the structure hidden behind raw data, and that this kind of representation can be very effective for managing the issues related to sensory data.

Among all the implemented approaches, the third one, based on techniques belonging to Algorithmic Learning Theory, has demonstrated a clear advantage with respect to the others, especially evident when considering the produced models. Algorithms manipulating formal languages, by being intrinsically recursive, can give a more natural representation of multi-scale models, which are more suitable for the analysis of sensory data, as they allow to ease the computational burden that characterizes the huge volume of data involved.

Moreover, such tools as Grammatical Inference pave the way to the realization of a new class of promising systems, able to alleviate the task of designing reliable and efficient automatic systems for knowledge extraction.

References

[1] A. R. Ganguly, J. Gama, O. A. Omitaomu, M. M. Gaber, and R. R. Vatsavai, *Knowledge Discovery from Sensor Data*, 1st. Boca Raton, FL, USA: CRC Press, Inc., 2008 (cit. on p. 1).

[2] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From data mining to knowledge discovery in databases," *AI Magazine*, vol. 17, no. 3, pp. 37–54, 1996 (cit. on p. 1). DOI: 10.1609/aimag.v17i3.1230.

[3] A. De Paola, M. La Cascia, G. Lo Re, M. Morana, and M. Ortolani, "User detection through multi-sensor fusion in an AmI scenario," in *Proceedings of the 15th International Conference on Information Fusion*, Published by the IEEE Computer Society, 2012, pp. 2502–2509 (cit. on pp. 2, 10).

[4] Q. Yang and X. Wu, "10 challenging problems in data mining research," *International Journal of Information Technology & Decision Making (IJITDM)*, vol. 5, no. 04, pp. 597–604, 2006 (cit. on p. 2).

[5] J. W. Tukey, *Exploratory Data Analysis*. Addison-Wesley, 1977 (cit. on p. 3).

[6] P. Hájek and T. Havránek, *Mechanizing hypothesis formation: Mathematical foundations for a general theory*, ser. Universitext. Berlin, New York: Springer-Verlag, 1978 (cit. on p. 3).

[7] V. N. Vapnik, *The nature of statistical learning theory*. New York, NY, USA: Springer-Verlag New York, Inc., 1995 (cit. on p. 4).

[8] T. M. Mitchell, R. M. Keller, and S. T. Kedar-Cabelli, "Explanation-based generalization: A unifying view," *Machine Learning*, vol. 1, no. 1, pp. 47–80, 1986 (cit. on p. 4). DOI: 10.1023/A:1022691120807.

[9] G. Dejong and R. Mooney, "Explanation-based learning: An alternative view," *Machine Learning*, vol. 1, no. 2, pp. 145–176, 1986 (cit. on p. 5). DOI: 10.1023/A:1022898111663.

[10] G. DeJong, "Toward robust real-world inference: A new perspective on explanation-based learning," in *Proceedings of the 17th European conference on Machine Learning*, ser. ECML'06, Berlin, Germany: Springer-Verlag, 2006, pp. 102–113 (cit. on p. 5).
DOI: 10.1007/11871842_14.

[11] K. S. Fu, *Syntactic Methods in Pattern Recognition*, ser. Mathematics in science and engineering. New York: Academic, 1974, vol. 112 (cit. on p. 5).

[12] W.-H. Tsai and K.-S. Fu, "Attributed Grammar - a tool for combining syntactic and statistical approaches to pattern recognition," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 10, no. 12, pp. 873–885, 1980 (cit. on p. 5).
DOI: 10.1109/TSMC.1980.4308414.

[13] L. Goldfarb, "Representation before computation," *Natural Computing*, vol. 9, no. 2, pp. 365–379, 2010 (cit. on p. 6).
DOI: 10.1007/s11047-009-9135-y.

[14] B. Chazelle, "Natural algorithms and influence systems," *Commun. ACM*, vol. 55, no. 12, pp. 101–110, Dec. 2012 (cit. on p. 6).
DOI: 10.1145/2380656.2380679.

[15] D. H. Wolpert, "The lack of a priori distinctions between learning algorithms," *Neural Comput.*, vol. 8, no. 7, pp. 1341–1390, Oct. 1996 (cit. on p. 7).
DOI: 10.1162/neco.1996.8.7.1341.

[16] O. Bousquet, S. Boucheron, and G. Lugosi, "Introduction to statistical learning theory," English, in *Advanced Lectures on Machine Learning*, ser. Lecture Notes in Computer Science, O. Bousquet, U. von Luxburg, and G. Rätsch, Eds., vol. 3176, Springer Berlin Heidelberg, 2004, pp. 169–207 (cit. on p. 7).
DOI: 10.1007/978-3-540-28650-9_8.

[17] A. De Paola, S. Gaglio, G. Lo Re, and M. Ortolani, "An ambient intelligence architecture for extracting knowledge from distributed sensors," in *Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human*, ACM, 2009, pp. 104–109 (cit. on p. 8).
DOI: 10.1145/1655925.1655945.

[18] A. Augello, M. Ortolani, G. Lo Re, and S. Gaglio, "Sensor mining for user behavior profiling in intelligent environments," in *Advances in Distributed Agent-Based Retrieval Tools*, Springer, 2011, pp. 143–158 (cit. on p. 8).
DOI: 10.1007/978-3-642-21384-7_10.

[19] S. Gaglio, G. Lo Re, and M. Ortolani, "Cognitive meta-learning of syntactically inferred concepts," in *BICA*, A. V. Samsonovich and K. R. Johannsdottir, Eds., ser. Frontiers in Artificial Intelligence and Applications, vol. 233, IOS Press, 2011, pp. 118–123 (cit. on p. 9).
DOI: 10.3233/978-1-60750-959-2-118.

[20] A. De Paola, S. Gaglio, G. Lo Re, and M. Ortolani, "Multi-sensor fusion through adaptive Bayesian networks," in *AI*IA 2011: Artificial Intelligence Around Man and Beyond*, ser. Lecture Notes in Computer Science, vol. 6934, Springer Berlin Heidelberg, 2011, pp. 360–371 (cit. on pp. 10, 33).
DOI: 10.1007/978-3-642-23954-0_33.

[21] P. Cottone, S. Gaglio, G. Lo Re, and M. Ortolani, "A machine learning approach for user localization exploiting connectivity data," *Engineering Applications of Artificial Intelligence*, 2016 (cit. on p. 13), accepted for publication.

[22] P. Cottone, S. Gaglio, G. Lo Re, and M. Ortolani, "User activity recognition for energy saving in smart homes," *Pervasive and Mobile Computing*, vol. 16, Part A, pp. 156–170, 2015 (cit. on p. 13).
DOI: 10.1016/j.pmcj.2014.08.006.

[23] P. Cottone, S. Gaglio, and M. Ortolani, "A structural approach to infer recurrent relations in data," in *Advances onto the Internet of Things*, ser. Advances in Intelligent Systems and Computing, S. Gaglio and G. Lo Re, Eds., vol. 260, Springer International Publishing, 2014, pp. 105–119 (cit. on p. 13).
DOI: 10.1007/978-3-319-03992-3_8.

[24] P. Cottone, G. Maida, and M. Morana, "User activity recognition via Kinect in an ambient intelligence scenario," International Conference on Applied Computing, Computer Science, and Computer Engineering (ICACC 2013), vol. 7, 2014, pp. 49–54 (cit. on p. 13).
DOI: 10.1016/j.ieri.2014.08.009.

[25] P. Cottone, S. Gaglio, G. Lo Re, and M. Ortolani, "User activity recognition for energy saving in smart homes," in *Sustainable Internet and ICT for Sustainability (SustainIT), 2013*, 2013, pp. 1–9 (cit. on p. 13).
DOI: `10.1109/SustainIT.2013.6685196`.

[26] P. Cottone, G. Lo Re, G. Maida, and M. Morana, "Motion sensors for activity recognition in an ambient-intelligence scenario," in *IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops),*, 2013, pp. 646–651 (cit. on p. 13).
DOI: `10.1109/PerComW.2013.6529573`.

[27] P. Cottone, S. Gaglio, and M. Ortolani, *Grammatical inference for structural knowledge extraction*, Talk presented at the 4th Italian Workshop on Machine Learning and Data Mining (MLDM15.it), Palermo, Sep. 2015 (cit. on p. 14).

[28] ——, *Exploiting mobility models for sustainable urban transportation*, Talk presented at I-CiTies 2015 - CINI Annual Workshop on ICT for Smart Cities & Communities, Ferrara, Oct. 2015 (cit. on p. 14).

[29] Z. Zhang, "Microsoft Kinect sensor and its effect," *IEEE MultiMedia*, vol. 19, no. 2, pp. 4–10, Apr. 2012 (cit. on pp. 15, 18).
DOI: `10.1109/MMUL.2012.24`.

[30] J. Yamato, J. Ohya, and K. Ishii, "Recognizing human action in time-sequential images using hidden Markov model," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '92),* Jun. 1992, pp. 379–385 (cit. on p. 16).
DOI: `10.1109/CVPR.1992.223161`.

[31] M. P. V. Kellokumpu and J. Heikkila, "Human activity recognition using sequences of postures," in *In Proceedings of the IAPR Conference on Machine Vision Applications*, 2005, pp. 570–573 (cit. on p. 16).

[32] M. Tang, "Recognizing hand gestures with Microsoft's Kinect," Department of Electrical Engineering, Stanford University, Tech. Rep., Mar. 2011 (cit. on p. 16).

[33] S. J. Preece, J. Y. Goulermas, L. P. J. Kenney, D. Howard, K. Meijer, and R. Crompton, "Activity identification using body-mounted sensors—a review of classification techniques," *Physiological Measurement*, vol. 30, no. 4, R1, 2009 (cit. on p. 16).

[34] L. Bao and S. Intille, "Activity recognition from user-annotated acceleration data," in *Pervasive Computing*, ser. Lecture Notes in Computer Science, A. Ferscha and F. Mattern, Eds., vol. 3001, Springer Berlin Heidelberg, 2004, pp. 1–17 (cit. on p. 16). DOI: 10.1007/978-3-540-24646-6_1.

[35] W. Li, Z. Zhang, and Z. Liu, "Action recognition based on a bag of 3D points," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Jun. 2010, pp. 9–14 (cit. on pp. 17, 21, 22, 25). DOI: 10.1109/CVPRW.2010.5543273.

[36] J. Wang, Z. Liu, Y. Wu, and J. Yuan, "Mining actionlet ensemble for action recognition with depth cameras," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2012, pp. 1290–1297 (cit. on p. 17). DOI: 10.1109/CVPR.2012.6247813.

[37] K. Lai, J. Konrad, and P. Ishwar, "A gesture-driven computer interface using Kinect," in *Proceedings of the IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI)*, Apr. 2012, pp. 185–188 (cit. on p. 17). DOI: 10.1109/SSIAI.2012.6202484.

[38] L. Xia, C.-C. Chen, and J. Aggarwal, "View invariant human action recognition using histograms of 3D joints," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Jun. 2012, pp. 20–27 (cit. on pp. 17, 22). DOI: 10.1109/CVPRW.2012.6239233.

[39] B. Scholkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA, USA: MIT Press, 2001 (cit. on p. 20).

[40] L. Rabiner and B. Juang, "An introduction to hidden Markov models," *IEEE ASSP Magazine*, vol. 3, no. 1, pp. 4–16, Jan. 1986 (cit. on pp. 20, 45). DOI: 10.1109/MASSP.1986.1165342.

[41] S. S. Rao, *Engineering Optimization: Theory and Practice, 3rd Edition*. Wiley-Interscience, 1996 (cit. on p. 22).

[42] S. Arlot and A. Celisse, "A survey of cross-validation procedures for model selection," *Statistics Surveys*, vol. 4, pp. 40–79, 2010 (cit. on p. 22). DOI: 10.1214/09-SS054.

[43] P. D. Grünwald, *The Minimum Description Length Principle (Adaptive Computation and Machine Learning).* The MIT Press, 2007 (cit. on p. 24).

[44] S. Gaglio, G. Lo Re, and M. Morana, "Human activity recognition process using 3-D posture data," *IEEE Transactions on Human-Machine Systems*, vol. 45, no. 5, pp. 586–597, Oct. 2015 (cit. on p. 24).
DOI: 10.1109/THMS.2014.2377111.

[45] A. B. B. Bouchard and S. Giroux, "A logical approach to ADL recognition for alzheimer's patients," in *Proceedings of the 4th International Conference on Smart homes and health Telematic (ICOST'06)*, IOS press, 2006, pp. 1–8 (cit. on p. 30).

[46] L. Chen, C. Nugent, M. Mulvenna, D. Finlay, X. Hong, and M. Poland, "Using event calculus for behaviour reasoning and assistance in a smart home," in *Proceedings of the 6th international conference on Smart Homes and Health Telematics*, ser. ICOST '08, Ames, IA, USA: Springer-Verlag, 2008, pp. 81–89 (cit. on p. 30).
DOI: 10.1007/978-3-540-69916-3_10.

[47] T. van Kasteren, A. Noulas, G. Englebienne, and B. Kröse, "Accurate activity recognition in a home setting," in *Proceedings of the 10th international conference on Ubiquitous computing*, ser. UbiComp '08, Seoul, Korea: ACM, 2008, pp. 1–9 (cit. on pp. 30, 51).
DOI: 10.1145/1409635.1409637.

[48] S. P. Rao and D. J. Cook, "Predicting inhabitant action using action and task models with application to smart homes," *International Journal on Artificial Intelligence Tools*, vol. 13, pp. 81–100, 2004 (cit. on p. 30).

[49] D. Hao Hu, S. J. Pan, V. W. Zheng, N. N. Liu, and Q. Yang, "Real world activity recognition with multiple goals," in *Proceedings of the 10th international conference on Ubiquitous computing*, ser. UbiComp '08, Seoul, Korea: ACM, 2008, pp. 30–39 (cit. on p. 30).
DOI: 10.1145/1409635.1409640.

[50] W. Pentney, A.-M. Popescu, S. Wang, H. Kautz, and M. Philipose, "Sensor-based understanding of daily life via large-scale use of common sense," in *Proceedings of the 21st national conference on Artificial intelligence - Volume 1*, ser. AAAI'06, Boston, Massachusetts: AAAI Press, 2006, pp. 906–912 (cit. on p. 30).

[51] M. Magnusson, "Discovering hidden time patterns in behavior: T-patterns and their detection," *Behavior Research Methods, Instruments, & Computers*, vol. 32, no. 1, pp. 93–110, 2000 (cit. on p. 31).
DOI: 10.3758/BF03200792.

[52] P. Rashidi and D. J. Cook, "Keeping the intelligent environment resident in the loop," in *Proceedings of the IET 4th International Conference on Intelligent Environments*, Jul. 2008, pp. 1–9 (cit. on p. 31).
DOI: 10.1049/cp:20081170.

[53] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," *SIGMOD Rec.*, vol. 22, pp. 207–216, 2 Jun. 1993 (cit. on p. 31).
DOI: 10.1145/170036.170072.

[54] B. Chikhaoui, S. Wang, and H. Pigot, "A frequent pattern mining approach for adls recognition in smart environments," Los Alamitos, CA, USA: IEEE Computer Society, 2011, pp. 248–255 (cit. on p. 31).
DOI: 10.1109/AINA.2011.13.

[55] T. Gu, Z. Wu, X. Tao, H. K. Pung, and J. Lu, "Epsicar: An emerging patterns based approach to sequential, interleaved and concurrent activity recognition," in *Proceedings of the IEEE International Conference on Pervasive Computing and Communications*, Mar. 2009, pp. 1–9 (cit. on p. 31).
DOI: 10.1109/PERCOM.2009.4912776.

[56] J. A. Iglesias, P. Angelov, A. Ledezma, and A. Sanchis, "Human activity recognition in intelligent home environments: An evolving approach," in *Proceedings of the 2010 conference on ECAI 2010: 19th European Conference on Artificial Intelligence*, Amsterdam, The Netherlands, The Netherlands: IOS Press, 2010, pp. 1047–1048 (cit. on p. 31).

[57] B. Bose, "Global warming: Energy, environmental pollution, and the impact of power electronics," *Industrial Electronics Magazine, IEEE*, vol. 4, no. 1, pp. 6–17, 2010 (cit. on p. 31).
DOI: 10.1109/MIE.2010.935860.

[58] J. Hollander and J. Harris, *Improving Energy Efficiency in Buildings: Progress and Problems*. American Council for an Energy-Efficient Economy, 1980 (cit. on p. 32).

[59] D. Landsberg and R. Stewart, *Improving Energy Efficiency in Buildings*. State University of New York Press, Jun. 1980 (cit. on p. 32).

[60] A. De Paola, G. Lo Re, M. Morana, and M. Ortolani, "An intelligent system for energy efficiency in a complex of buildings," in *Proceedings of the 2nd IFIP Conference on Sustainable Internet and ICT for Sustainability*, 2012 (cit. on p. 32).

[61] J. V. Paatero and P. D. Lund, "A model for generating household electricity load profiles," *International Journal of Energy Research*, vol. 30, no. 5, pp. 273–290, 2006 (cit. on p. 33).
DOI: 10.1002/er.1136.

[62] M. Erol-Kantarci and H. Mouftah, "Wireless sensor networks for domestic energy management in smart grids," in *Proceedings of the 25th IEEE Biennial Symposium on Communications (QBSC)*, 2010, pp. 63–66 (cit. on p. 33).
DOI: 10.1109/BSC.2010.5473004.

[63] M. Milenkovic and O. Amft, "Recognizing energy-related activities using sensors commonly installed in office buildings," in *Proceedings of the 3rd International Conference on Sustainable Energy Information Technology (SEIT-2013)*, vol. 19, 2013, pp. 669–677 (cit. on p. 33).
DOI: 10.1016/j.procs.2013.06.089.

[64] O. Sianaki, O. Hussain, and A. Tabesh, "A knapsack problem approach for achieving efficient energy consumption in smart grid for end-users' life style," in *Proceeding of the IEEE Conference on Innovative Technologies for an Efficient and Reliable Electricity Supply (CITRES)*, 2010, pp. 159–164 (cit. on pp. 33, 48).
DOI: 10.1109/CITRES.2010.5619873.

[65] H. Peng, P. Wu, J. Zhu, and J. Zhang, "Helix: Unsupervised grammar induction for structured activity recognition," in *Proceedings of the 11th IEEE International Conference on Data Mining (ICDM)*, Dec. 2011, pp. 1194–1199 (cit. on p. 33).
DOI: 10.1109/ICDM.2011.74.

[66] R. Hamid, S. Maddi, A. Johnson, A. Bobick, I. Essa, and C. Isbell, "A novel sequence representation for unsupervised analysis of human activities," *Artificial Intelligence*, vol. 173, no. 14, pp. 1221–1244, Sep. 2009 (cit. on pp. 33, 34).
DOI: 10.1016/j.artint.2009.05.002.

[67] C. Chen, D. J. Cook, and A. S. Crandall, "The user side of sustainability: Modeling behavior and energy usage in the home," *Pervasive and Mobile Computing*, vol. 9, no. 1, pp. 161–175, Feb. 2013, Special Section: Pervasive Sustainability (cit. on p. 34).
DOI: 10.1016/j.pmcj.2012.10.004.

[68] Y. Kim, T. Schmid, Z. M. Charbiwala, and M. B. Srivastava, "Viridiscope: Design and implementation of a fine grained power monitoring system for homes," in *Proceedings of the 11th International Conference on Ubiquitous Computing*, ser. Ubicomp '09, Orlando, Florida, USA: ACM, 2009, pp. 245–254 (cit. on p. 34).
DOI: 10.1145/1620545.1620582.

[69] S. Barker, A. Mishra, D. Irwin, E. Cecchet, P. Shenoy, and J. Albrecht, "Smart*: An open data set and tools for enabling research in sustainable homes," in *Proceedings of the 2nd KDD Workshop on Data Mining Applications in Sustainability (SustKDD)*, Beijing, China, Aug. 2012, p. 6 (cit. on p. 34).

[70] L. G. Swan and V. I. Ugursal, "Modeling of end-use energy consumption in the residential sector: A review of modeling techniques," *Renewable and Sustainable Energy Reviews*, vol. 13, no. 8, pp. 1819–1835, 2009 (cit. on p. 35).
DOI: 10.1016/j.rser.2008.09.033.

[71] A. De Paola, G. Lo Re, F. Milazzo, and M. Ortolani, "Predictive models for energy saving in wireless sensor networks," in *Proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, Jun. 2011, pp. 1–6 (cit. on p. 35).
DOI: 10.1109/WoWMoM.2011.5986204.

[72] A. Lalomia, G. Lo Re, and M. Ortolani, "A hybrid framework for soft real-time WSN simulation," in *Proceedings of the 13th IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications (DS-RT)*, IEEE, Oct. 2009, pp. 201–207 (cit. on p. 35).
DOI: 10.1109/DS-RT.2009.30.

[73] M. Muratori, M. C. Roberts, R. Sioshansi, V. Marano, and G. Rizzoni, "A highly resolved modeling technique to simulate residential power demand," *Applied Energy*, vol. 107, pp. 465–473, 2013 (cit. on p. 35).
DOI: 10.1016/j.apenergy.2013.02.057.

[74] I. Richardson, M. Thomson, D. Infield, and C. Clifford, "Domestic electricity use: A high-resolution energy demand model," *Energy and Buildings*, vol. 42, no. 10, pp. 1878–1887, 2010 (cit. on pp. 35, 60, 61).
DOI: 10.1016/j.enbuild.2010.05.023.

[75] J. Widén and E. Wäckelgård, "A high-resolution stochastic model of domestic activity patterns and electricity demand," *Applied Energy*, vol. 87, no. 6, pp. 1880–1892, 2010 (cit. on pp. 35, 61).
DOI: 10.1016/j.apenergy.2009.11.006.

[76] G. Costanzo, A. Kosek, G. Zhu, L. Ferrarini, M. Anjos, and G. Savard, "An experimental study on load-peak shaving in smart homes by means of online admission control," in *Proceedings of the 3rd IEEE PES International Conference and Exhibition on Innovative Smart Grid Technologies (ISGT Europe)*, Oct. 2012, pp. 1–8 (cit. on pp. 36, 46, 47).
DOI: 10.1109/ISGTEurope.2012.6465658.

[77] G. Costanzo, J. Kheir, and G. Zhu, "Peak-load shaving in smart homes via online scheduling," in *Proceedings of the 20th IEEE International Symposium on Industrial Electronics (ISIE)*, Jun. 2011, pp. 1347–1352 (cit. on p. 36).
DOI: 10.1109/ISIE.2011.5984354.

[78] S. Barker, A. Mishra, D. Irwin, P. Shenoy, and J. Albrecht, "Smartcap: Flattening peak electricity demand in smart homes," in *Proceedings of the IEEE International Conference on Pervasive Computing and Communications (PerCom)*, Mar. 2012, pp. 67–75 (cit. on pp. 36, 46, 47, 49).
DOI: 10.1109/PerCom.2012.6199851.

[79] Y. Xu, D. Irwin, and P. Shenoy, "Incentivizing advanced load scheduling in smart homes," in *Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings*, ser. BuildSys'13, Roma, Italy: ACM, 2013, pp. 1–8 (cit. on pp. 36, 49).
DOI: 10.1145/2528282.2528292.

[80] Chen, Gong, Wu, Xindong, Zhu, Xingquan, Arslan, Abdullah, He, and Yu, "Efficient string matching with wildcards and length constraints," *Knowledge and Information Systems*, vol. 10, no. 4, pp. 399–419, Nov. 2006 (cit. on p. 40).
DOI: 10.1007/s10115-006-0016-8.

[81] J. Rissanen, "Minimum description length principle," in *Encyclopedia of Machine Learning*, C. Sammut and G. Webb, Eds., Springer US, 2010, pp. 666–668 (cit. on p. 41).
DOI: 10.1007/978-0-387-30164-8_540.

[82] P. Rashidi, D. J. Cook, L. B. Holder, and M. S. Edgecombe, "Discovering activities to recognize and track in a smart environment," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, pp. 527–539, Apr. 2011 (cit. on pp. 42, 44, 53, 55).
DOI: 10.1109/TKDE.2010.148.

[83] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Soviet Physics Doklady*, vol. 10, no. 8, pp. 707–710, 1966 (cit. on p. 43).

[84] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*. Orlando, FL, USA: Academic Press, Inc., 2006 (cit. on p. 44).

[85] R. Stamminger and U. B. L. Fakultät, *Synergy Potential of Smart Domestic Appliances in Renewable Energy Systems*, ser. Schriftenreihe der Haushaltstechnik Bonn. Shaker, 2009 (cit. on pp. 46, 47, 60).

[86] Martello, Silvano, and P. Toth, *Knapsack problems: Algorithms and computer implementations*. New York, NY, USA: John Wiley & Sons, Inc., 1990 (cit. on p. 48).

[87] D. J. Cook and M. Schmitter-Edgecombe, "Assessing the quality of activities in a smart environment," *Methods of Information in Medicine*, vol. 48, no. 5, pp. 480–485, 2009 (cit. on p. 51).

[88] D. J. Cook, "Learning setting-generalized activity models for smart spaces," *IEEE Intelligent Systems*, vol. 27, no. 1, pp. 32–38, Jan. 2012 (cit. on p. 51).
DOI: 10.1109/MIS.2010.112.

[89] E. Hellinger, "Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen," *J. Reine Angew. Math.*, vol. 136, pp. 210–271, 1909 (cit. on p. 52).

[90] C. de la Higuera, *Grammatical Inference: Learning Automata and Grammars*. New York, NY, USA: Cambridge University Press, 2010 (cit. on pp. 65, 72–74, 81).

[91] E. M. Gold, "Language identification in the limit," *Information and Control*, vol. 10, no. 5, pp. 447–474, 1967 (cit. on pp. 65, 74, 75).

[92] D. Angluin, "Queries and concept learning," *Machine Learning*, vol. 2, no. 4, pp. 319–342, Apr. 1988 (cit. on p. 65).
DOI: 10.1007/BF00116828.

[93] K. J. Lang, B. A. Pearlmutter, and R. A. Price, "Results of the Abbadingo One DFA learning competition and a new Evidence-Driven State Merging Algorithm," in *Proceedings of the 4th International Colloquium on Grammatical Inference*, ser. ICGI '98, London, UK, UK: Springer-Verlag, 1998, pp. 1–12 (cit. on pp. 65, 81).
DOI: 10.1007/BFb0054059.

[94] D. H. Jonassen, K. Beissner, and M. Yacci, *Structural knowledge: Techniques for representing, conveying, and acquiring structural knowledge*. Psychology Press, 1993 (cit. on p. 66).

[95] R. S. Michalski, "Concept learning," *Encyclopedia of artificial intelligence*, vol. 1, pp. 185–194, 1987 (cit. on p. 66).

[96] A. Mitchell and M. Chi, "Measuring knowledge within a domain," *The representation of cognitive structure*, pp. 85–109, 1984 (cit. on p. 66).

[97] N. Chomsky, *Syntactic structures*. Walter de Gruyter, 2002 (cit. on pp. 66, 67).

[98] N. Ding, L. Melloni, H. Zhang, X. Tian, and D. Poeppel, "Cortical tracking of hierarchical linguistic structures in connected speech," *Nature Neuroscience*, vol. 19, no. 1, pp. 158–164, Jan. 2016 (cit. on p. 67).
DOI: 10.1038/nn.4186.

[99] N. Chomsky, "Three models for the description of language," *IRE Transactions on Information Theory*, vol. 2, no. 3, pp. 113–124, 1956 (cit. on pp. 67, 69).

[100] J. E. Hopcroft, R. Motwani, and J. D. Ullman, *Introduction to Automata Theory, Languages, and Computation*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2006 (cit. on pp. 68, 75).

[101] W. Levelt, *An Introduction to the Theory of Formal Languages and Automata*. John Benjamins Pub., 2008 (cit. on pp. 68, 69).

[102] S. Jain, *Systems that learn : An introduction to learning theory*, ser. Learning, development, and conceptual change. Cambridge, Mass. MIT Press, 1999, A Bradford book. (cit. on p. 72).

[103] P. Dupont, L. Miclet, and E. Vidal, "What is the search space of the regular inference?" In *In Proceedings of the Second International Colloquium on Grammatical Inference (ICGI'94*, Springer Verlag, 1994, pp. 25–37 (cit. on pp. 76, 77, 79).
DOI: 10.1007/3-540-58473-0_134.

[104] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi, "Understanding individual human mobility patterns," *Nature*, vol. 453, no. 7196, pp. 779–782, 2008 (cit. on p. 83).
DOI: 10.1038/nature06958.

[105] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási, "Limits of predictability in human mobility," *Science*, vol. 327, no. 5968, pp. 1018–1021, 2010 (cit. on p. 83).
DOI: 10.1126/science.1177170.

[106] Y. Chon, H. Shin, E. Talipov, and H. Cha, "Evaluating mobility models for temporal prediction with high-granularity mobility data," in *Proceedings of the 2012 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, IEEE, 2012, pp. 206–212 (cit. on p. 83).
DOI: 10.1109/PerCom.2012.6199868.

[107] D. Karamshuk, C. Boldrini, M. Conti, and A. Passarella, "Spot: Representing the social, spatial, and temporal dimensions of human mobility with a unifying framework," *Pervasive and Mobile Computing*, vol. 11, pp. 19–40, 2014 (cit. on p. 84).
DOI: 10.1016/j.pmcj.2013.07.011.

[108] D. C. Renso, D. S. Spaccapietra, and D. E. Zimnyi, *Mobility Data: Modeling, Management, and Understanding*. New York, NY, USA: Cambridge University Press, 2013 (cit. on pp. 84, 88).

[109] F. Meneses and A. Moreira, "Large scale movement analysis from wifi based location data," in *Proceedings of the 2012 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Nov. 2012, pp. 1–9 (cit. on p. 84).
DOI: 10.1109/IPIN.2012.6418885.

[110] F. Karnadi, Z. H. Mo, and K.-c. Lan, "Rapid generation of realistic mobility models for VANET," in *Proceedings of the IEEE Conference on Wireless Communications and Networking*, Mar. 2007, pp. 2506–2511 (cit. on p. 85).
DOI: 10.1109/WCNC.2007.467.

[111] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *Wireless communications and mobile computing*, vol. 2, no. 5, pp. 483–502, 2002 (cit. on p. 85).
DOI: `10.1002/wcm.72`.

[112] R. Calegari, M. Musolesi, F. Raimondi, and C. Mascolo, "CTG: A connectivity trace generator for testing the performance of opportunistic mobile systems," in *Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering*, ACM, 2007, pp. 415–424 (cit. on p. 85).
DOI: `10.1145/1287624.1287684`.

[113] M. Kim, D. Kotz, and S. Kim, "Extracting a mobility model from real user traces," in *INFOCOM*, vol. 6, 2006, pp. 1–13 (cit. on p. 85).
DOI: `10.1109/INFOCOM.2006.173`.

[114] S. Jiang, J. Ferreira Jr, and M. C. Gonzalez, "Discovering urban spatial-temporal structure from human activity patterns," in *Proceedings of the ACM SIGKDD international workshop on urban computing*, ACM, 2012, pp. 95–102 (cit. on p. 85).
DOI: `10.1145/2346496.2346512`.

[115] S. Hoteit, S. Secci, S. Sobolevsky, C. Ratti, and G. Pujolle, "Estimating human trajectories and hotspots through mobile phone data," *Computer Networks*, vol. 64, pp. 296–307, 2014 (cit. on p. 85).
DOI: `http://dx.doi.org/10.1016/j.comnet.2014.02.011`.

[116] S. C. Geyik, E. Bulut, and B. K. Szymanski, "Grammatical inference for modeling mobility patterns in networks," *IEEE Transactions on Mobile Computing*, vol. 12, no. 11, pp. 2119–2131, 2013 (cit. on p. 85).
DOI: `10.1109/TMC.2012.184`.

[117] Z. Koppányi, "Individual human mobility modeling with probabilistic automata," in *Proceedings of the Conference of Junior Researchers in Civil Engineering*, 2012 (cit. on p. 85).

[118] T. Oates, A. P. Boedihardjo, J. Lin, C. Chen, S. Frankenstein, and S. Gandhi, "Motif discovery in spatial trajectories using grammar inference," in *Proceedings of the 22Nd ACM International Conference on Information & Knowledge Management*,

ser. CIKM '13, San Francisco, California, USA: ACM, 2013, pp. 1465–1468 (cit. on p. 85).
DOI: 10.1145/2505515.2507820.

[119] Z. Balkić, D. Šoštarić, and G. Horvat, "Geohash and UUID identifier for multi-agent systems," in *Proceedings of the 6th KES International Conference on Agent and Multi-Agent Systems: Technologies and Applications*, ser. KES-AMSTA'12, Dubrovnik, Croatia: Springer-Verlag, 2012, pp. 290–298 (cit. on p. 86).
DOI: 10.1007/978-3-642-30947-2_33.

[120] J. A. Dam, "A web-based weather service for wind sports," PhD thesis, Aarhus Universitet, Datalogisk Institut, 2009 (cit. on p. 87).

[121] Y. Zheng and X. Zhou, *Computing with Spatial Trajectories*, 1st. Springer Publishing Company, Incorporated, 2011 (cit. on p. 88).

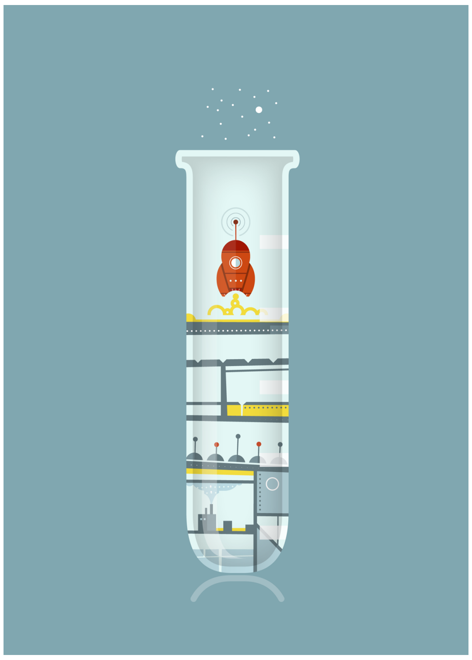
[122] Y. Zheng, L. Liu, L. Wang, and X. Xie, "Learning transportation mode from raw gps data for geographic applications on the web," in *Proceedings of the 17th International Conference on World Wide Web*, ser. WWW '08, Beijing, China: ACM, 2008, pp. 247–256 (cit. on p. 92).
DOI: 10.1145/1367497.1367532.

Declaration

I herewith declare that I have produced this thesis without the prohibited assistance of third parties and without making use of aids other than those specified; notions taken over directly or indirectly from other sources have been identified as such. This thesis has not previously been presented in identical or similar form to any other Italian or foreign examination board.

The work described in this thesis was conducted from January 2013 to December 2015 under the supervision of my advisor Prof. Salvatore Gaglio and my co-advisor Prof. Marco Ortolani at the University of Palermo.

Palermo
January 15, 2016



This thesis was created using LaTeX 2_ε, originally developed by Leslie Lamport and based on Donald Knuth's TeX. The typesettings software used the XeLaTeX, the BibTeXand the fontspec package. The template of this document was based on the Dissertate class, by Jordan Suchow and can be found online at github.com/asm-products/Dissertate. The body text is set in 11 point Egenolff-Berner Garamond, a revival of Claude Garamont's humanist typeface. The above illustration, *Science Experiment 02*, was created by Ben Schlitter and released under CC BY-NC-ND 3.0.