CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF NOTATION

This list indexes the key notation appearing in the equations, algorithms, figures, legends or explanations throughout this thesis.

a	A scalar (integer or real)
$\mathbf{a}$	A vector
α	A behavioral pattern
A	A set
$\|A\|$	The cardinality of set A
$\|A\| \times \|B\|$	A matrix with $\|A\|$ rows and $\|B\|$ columns
$x_d^{(i)}$	A sensor event of sequence $x^{(i)}$
$x^{(i)}$	A sequence of sensor events $x_0^{(i)}, x_1^{(i)}, \ldots, x_d^{(i)}$
$\mathcal{X}$	A collection of sequences $x^{(0)}, x^{(1)}, \ldots, x^{(m)}$
$\mathcal{Y}$	A vector of scores, one for each category in classification task
$\mathbb{K}(G,M,I)$	A formal context of *FormalConceptAnalysis*
c	A formal concept
$\mathfrak{B}(G,M,I)$	All the formal concepts of the formal context (G,M,I)
$\underline{\mathfrak{B}}(G,M,I)$	The concept lattice of the formal context (G,M,I)
$\underline{\mathfrak{B}}(\mathbb{K})$	The concept lattice of the formal context $\mathbb{K}$
$n_i \in \underline{\mathfrak{B}}$	A node of the Hasse diagram $\underline{\mathfrak{B}}(\mathbb{K})$
$\text{ext}(c)$	The extent of the formal concept c
$\text{int}(c)$	The intent of the formal concept c
$a \gg b$	A constant a is much greater than b
$\preceq$	The hierarchical or partial order of formal concept analysis
$\prec_j$	A possible permutation of data indicating a possible execution order
$a \prec b$	An execution order indicates that a executes before b
$\curvearrowright$	A transition of inference in the knowledge retrieval process
$\circlearrowleft_{Infimum}$	A rollback operation from the Infimum in the knowledge retrieval process

LIST OF ACRONYMS

This list provides a concise reference describing the acronyms appearing in this thesis.

AI	Artificial Intelligence
AmI	Ambient Intelligence
ANN	Artificial Neural Networks
AR	Activity Recognition
BN	Bayesian Network
CPT	Conditional Probability Table
CRF	Conditional Random Field
DAG	Directed Acyclic Graph
DBN	Dynamic Bayesian Network
DM	Data Mining
FCA	Formal Concept Analysis
GMM	Gaussian Mixture Model
HCI	Human-Computer Interaction
HDS	Half-Duplex Search
HMM	Hidden Markov Model
ICT	Information and Communication Technologies
IoT	Internet of Things
KNN	K-Nearest Neighbors
ML	Machine Learning
NBC	Naive Bayes Classifier
RDF	Resource Description Framework
RF	Random Forest
RMSD	Root-Mean-Square Deviation
SSE	Sum of Squared Error
SVM	Support Vector Machine
WSN	Wireless Sensor Network

ABSTRACT

With the advancement of information and communication technology, sensors, actuators or other computational elements can be embedded seamlessly in the daily objects of our lives. These components can make our lives smarter by generating an intelligent living environment called smart home. Information indicating environmental changes can be integrated from many sources and exchanged in such an environment through wireless communications. Smart homes attempt to create a human-centered environment that let all kinds of components work cooperatively to make residents lives more comfortable, and allow the environment to respond adaptively to various requests. They are also be expected to autonomously acquire contextual information under the premise of ensuring privacy to guarantee the safety of residents and improve their experience in that environment.

As a prerequisite for all above functionalities, activity recognition is an important part of smart home applications. It greatly affects the appropriateness and accuracy of intelligent assistance and preventive interventions. However, modeling and understanding human behaviors involve many tasks, each of which may affect the final recognition results. First, the collected sensor data is massive and continuous with various data types. How to filter noise, extract useful behavioral patterns and manage discovered knowledge are a thorny issue at the

preprocessing stage. Second, because of various lifestyles and other factors, there are often many different behavioral patterns that describe the same activities. Moreover, different activities may also have similar patterns. In addition, some composite activities can be performed in a continuous, concurrent or interleaved manner. These factors increase the uncertainty and complexity of activity recognition problem. Third, if there are multiple residents in a smart home, it is difficult to determine exactly who triggered some sensor events or which activity a sensor data belongs to. Fourth, how to detect abnormal data and normal one as well as the moments they occur are also very difficult.

The purpose of this thesis is to establish a knowledge-driven activity inference engine based on formal concept analysis to extract useful behavioral patterns and model human behaviors from massive sensor data. All explored inferences are represented as nodes in a lattice structure knowledge base. Using partially observed data as a query condition, we propose a new lattice search algorithm to incrementally retrieve the most probable inference in order to recognize ongoing activities and predict subsequent behaviors. Furthermore, abnormal behavioral patterns are successfully detected to avoid activity failures or severe consequences. More complex situations, such as composite and multi-resident activity recognition can also be addressed by the extension modules of the inference engine. Finally, we use an incremental lattice construction algorithm to strengthen the inference engine to avoid retraining the whole model when new training data with new features are available. Compared with recently published research, our method avoids the interventions of domain experts in building a knowledge base and can achieve competitive results in the benchmark datasets with or without unbalanced distribution.

Keywords: Activity Recognition, Anomaly Detection, Data Mining, Formal Concept Analysis, Ambient Intelligence

RÉSUMÉ

Avec l'avancement des technologies de l'information et de la communication, des capteurs ou d'autres composants informatiques peuvent être intégrés de manière transparente aux objets quotidiens de notre vie. Ces composants peuvent rendre nos vies plus intelligentes en générant un environnement intelligent appelé maison intelligente. Les informations et les données indiquant les changements de l'environnement peuvent être intégrées à partir de nombreuses sources et échangées dans un tel environnement par les communications sans fil. Les maisons intelligentes tentent de créer un environnement concentré sur humains qui permet à toutes sortes de composants de travailler en coopération pour rendre la vie des résidents plus confortable et permettre à l'environnement de répondre de manière adaptative aux diverses demandes. Ils sont également censés acquérir des informations contextuelles en manière autonome afin de garantir la sécurité des résidents et d'améliorer leur expérience dans cet environnement.

Pour réduire le fardeau des familles et de la société, la communauté scientifique considère les environnements intelligents comme une solution prometteuse pour aider les personnes âgées à vivre de manière autonome avec dignité et bien-être. Les données sensorielles indiquant les changements environnementaux et le comportement humain devraient être recueillies par les

réseaux de capteurs sans fil dans les maisons intelligentes. Après avoir compris les situations en temps réel et les activités en cours, les maisons intelligentes peuvent fournir une assistance proactive si nécessaire pour aider les personnes âgées à mieux accomplir leurs activités. De plus, si certains résidents ont tendance à se comporter de manière anormale en raison de leur déficience cognitive, les maisons intelligentes peuvent détecter ces anomalies, évaluer leurs menaces, les avertir et prendre des mesures préventives ou des interventions pour éviter d'autres conséquences graves.

Comme condition préalable à toutes les fonctionnalités ci-dessus, la reconnaissance d'activité est une partie importante des applications de maison intelligente. Cela affecte grandement la pertinence et l'exactitude de l'assistance intelligente et des interventions préventives. Cependant, la modélisation et la compréhension des comportements humains impliquent de nombreuses tâches, dont chacune peut affecter les résultats de la reconnaissance finale. Premièrement, les données collectées sur les capteurs sont massives, hétérogènes et continues. Comment filtrer les données de bruit, extraire les modèles comportementaux utiles et leur gestion des connaissances sont un problème épineux au stade du prétraitement. Deuxièmement, en raison de divers modes de vie et d'autres facteurs, il peut y avoir de nombreux modèles de comportement différents qui décrivent les mêmes activités. De plus, différentes activités peuvent également avoir des tendances similaires. De plus, certaines activités composites peuvent être réalisées de manière continue, simultanée ou entrelacée. Ces facteurs augmentent l'incertitude et la complexité du problème de reconnaissance d'activité. Troisièmement, s'il y a plusieurs résidents dans une maison intelligente, il est difficile de déterminer exactement qui a déclenché certains événements de capteurs ou à quelle activité appartiennent les données d'un capteur. Quatrièmement, comment détecter des données anormales et normales ainsi que les moments où elles se produisent sont également très difficiles.

Le but de cette thèse est d'établir un moteur d'inférence d'activité basé sur la connaissance

basé sur l'analyse conceptuelle formelle pour extraire des modèles comportementaux utiles et modéliser des comportements humains à partir de données de capteurs massives et hétérogènes. Toutes les inférences explorées sont représentées sous la forme de nœuds dans une base de connaissances de la structure en treillis. En utilisant des données partiellement observées comme condition de requête, nous proposons un nouvel algorithme de recherche sur réseau pour récupérer de façon incrémentielle l'inférence la plus probable afin de reconnaître les activités en cours et de prédire les comportements subséquents. De plus, des modèles comportementaux anormaux dus à des erreurs cognitives sont détectés avec succès pour éviter des échecs d'activité ou des conséquences graves. Des situations plus complexes, telles que la reconnaissance d'activité composite et multi-résident peuvent également être adressées par les modules d'extension du moteur d'inférence. Enfin, nous utilisons un algorithme de construction de réseau incrémental pour renforcer le moteur d'inférence afin d'éviter de recycler l'ensemble du modèle lorsque de nouvelles données d'entraînement avec de nouvelles fonctionnalités sont disponibles. Par rapport à la recherche publiée récemment, notre méthode évite les interventions des experts du domaine dans la construction d'une base de connaissances, et peut atteindre des résultats compétitifs dans les jeux de données de référence avec ou sans distribution déséquilibrée.

Mots clés: Reconnaissance d'activité, détection d'anomalies, exploration de données, analyse de concept formel, intelligence ambiante

CHAPTER 1

GENERAL INTRODUCTION

As the introductory part of the whole dissertation, this chapter first introduces the technical background and research orientation of the thesis, and outlines the reason why we choose sensor-based activity recognition as the thesis topic. Then, in Section 1.3 to 1.4, we present our hypothesis and prospective techniques to address the problems raised in the previous sections. Next, in Section 1.5, we summarize the issues that may be confronted during the research process, including the obstacles in design and the inherent challenges of the research itself. After that, in Section 1.6, we present the objectives of our research. Finally, in Section 1.7, we give a brief indication about how the thesis will be organized.

1.1 ADVENT OF INFORMATION AGE

Since the late 1950s, the shift from mechanical and analogue electronic technology to digital electronics has led to the third industrial revolution due to the growth and popularity of digital computers and digital recording. This revolution marks the beginning of the information age, which is redefining many aspects of modern life around the world.

Originally, computers were used only in the military field during World War II [1], but today,

computers and their derivatives are becoming more and more common due to the evolution from transistors to integrated circuits, and their size is getting smaller and smaller. In addition, computers with appropriate software can solve a variety of problems. Because of the lower cost of personal computers and their increasing popularity [2], computers are no longer independent individuals, but are interconnected through the Internet to form a huge network. Such a network makes information easier to access. Not only computers, but also various mobile, even wearable devices can connect to the Internet. Computers are now used as control systems for a wide variety of industrial and consumer devices.

In the early stages of their development, computers were used only as a computational tool to liberate humans from heavy computing tasks, only for simple calculations. With the rapid advances in technology, the next generation of computers will always be able to significantly surpass their previous generation in performance, which is called *Moore's law*. At the same time, computers have also been greatly improved in the field of information communication and storage. In the 1950s, Alan Turing first introduced his famous *Turing test* [3], which has a profound impact on the development of artificial intelligence, a new discipline of computer science. Since then, people were no longer satisfied with computers that solely focus on mechanical calculation, but hope that the future computer can have the ability to automatically learn, reason, recommend, predict, identify and make decisions like human beings.

Now, the information age is changing our society in every aspect of life, and creating a new and efficient economy. It affects the business models, commerce and market structure by reducing the importance of distance and the informational barriers. The workspace and labor market are no longer limited by the geographical constraints. With the help of powerful computers, people have been freed from handling numerous tasks artificially. Highly repetitive and predictable work with a high frequency is gradually being replaced by the automation of information age [4].

1.2 SPRING OF ARTIFICIAL INTELLIGENCE

In 1956, *artificial intelligence* (AI) was formally established as an academic discipline during the Dartmouth workshop. After that, it has gone through a series of boom-bust cycles. Because of the half-century efforts, it has become a prosperous field with many practical applications and active research subjects [5]. AI has already several mature capabilities for perceiving, understanding, self-learning, and reasoning. Advances in AI technology have opened up broader markets and new opportunities in the domains such as health, finance, communication, education, energy, manufacturing and logistics, etc [6].

AI is one of the newest fields in science and engineering, it is committed to build intelligent systems and to learn how to improve system performance by the use of experience. It encompasses a huge variety of subfields such as natural language processing, knowledge representation, pattern recognition, automated reasoning, *machine learning* (ML) and *data mining* (DM), etc. Moreover, AI is also an interdisciplinary field which is inspired by other disciplines: philosophy, mathematics, economics, neuroscience, psychology, computer engineering, linguistics, control theory and cybernetics [7]. Technological progress of computer science in the fields of big data, algorithmic development and processing power have made the performance, accessibility, and costs of AI more favorable than ever before [6].

AI systems are designed to evaluate, categorize and learn received data, and then output inferences concerning insight, decision or conclusion. Today, the great success of academic and industrial research in speech recognition [8, 9, 10, 11], image processing [12, 13], medical diagnosis [14, 15, 16, 17], and game AI [18, 19, 20] has triggered another new wave of AI. Almost all the famous universities and science & technology giants in the world have increased their investments in AI research [6, 21]. At the same time, many counties have treated AI research as a national priority or a national strategic goal [22] and have constantly

raised their research and development budgets. More and more companies such as NVIDIA, Intel, Qualcomm and Samsung are developing machine-learning chips to enable real-time applications in *Internet of things* (IoT) devices [23]. Among branches of computer science, AI is the only field to attempt to build intelligent systems that will function autonomously in complex, changing environments [7]. Therefore, it serves as the preferred solution for more and more practical problems. AI has become ubiquitous and ambient in our personal lives. Many industries are gradually turning into the AI-driven ones.

1.3 AMBIENT INTELLIGENCE

Ambient Intelligence (AmI) is a paradigm of AI that supports the design of next generation of intelligent systems and introduces innovative means of communication among human beings, machines and living environments [24, 25]. It is a prospective solution for intelligent living assistance that takes advantage of cutting-edge technologies to improve habitual supports [26]. With huge commercial prospects and rapid development of *information and communication technologies* (ICT) in recent years, smart environments have become a very active research topic. As a promising intervention for intelligent living assistance, smart environments, also known as one of the most successful applications of AmI, is to support residents by providing appropriate assistance while carrying out activities.

As an emerging interdisciplinary domain, in addition to advanced data analysis techniques, AmI also incorporates multiple cutting-edge technologies such as *Internet of things* (IoT) and *wireless sensor network* (WSN), etc [24, 27, 28]. Recent advances in these techniques present unprecedented opportunities to research and develop intelligent living environments. They embed computer intelligence into home devices that allow electronics, software and actuators to connect, interact and exchange data. They can also provide a convenient way to

measure home conditions and monitor home appliances [29].

AmI applications usually have the following characteristics: firstly, they are aware of environmental changes. Secondly, with the support of computational units, they can rapidly respond to a variety of requirements in a short time. Thirdly, they can provide better personal interactive experiences concerning context awareness. Context-aware systems offer entirely new opportunities for application developers and for end users by gathering context data and adapting systems behavior accordingly [30].

AmI utilizes IoT to build a network of objects embedded with measurable electronic components like sensors, radio frequency identification (RFID) tags/readers, power analyzer or actuators to gather data continuously from the smart environments [31, 32]. Target objects include home appliances, household furniture, and the other daily commodities. In recent years, considerable attention has also been paid to wearable devices, to collect user's behavioral information or vital signs [28]. These ubiquitous electronics make it possible to achieve real-time monitoring and avoid risks at the earliest stages.

A *wireless sensor network* (WSN) can be defined as a network of sensor nodes, which are spatially distributed and work cooperatively to communicate information gathered from the monitored fields through wireless links [33]. In home care, sensor nodes can help to monitor residents in a smart home in order to guarantee their safety and independence. However, the gathered data are usually large-scale and chaotic. It is very hard to directly use it. At this time, effective data analysis processing models are critical for parsing the behavioral data of residents. Moreover, AmI could seamlessly integrate sensors, processing, and interfaces such as touchscreen, speech processing, assisted social robots or any other advanced HCI technologies with daily activities [34]. Ideally, AmI needs to be sensitive to the needs of residents. Real-time situations will be analyzed and appropriate feedbacks or interventions

will be given out.

Besides, due to heterogeneous components, AmI continuously produces large-scale data. Such output data can involve environmental changes (positions, movements, temperature, and pressure, etc.), and consumption (energy or resources) [6, 27]. It is usually temporal and sequential, even unstructured and chaotic. Without advanced and effective data analysis methods, it is not possible to analyze such numerous data. As a consequence, machine learning and data mining, two subfields of artificial intelligence, are indispensable to automatically interpret, infer and understand the current situation, for the purpose of responding real-time requirements of residents [25, 26].

1.4 SMART ENVIRONMENTS

Considering the advantages above, our future living environments will become more and more intelligent [35], AmI shows great potential to offer personal assistance services for people who cannot live independently [36, 37]. With the help of home automation and ubiquitous computing, new generations of smart homes will be devoted to providing dynamic, intelligent, suitable and considerate personal services to their residents.

Therefore, understanding the true intention of a resident has significant effects in ensuring high-quality services for real-time assistance. Hence, activity recognition is the minimum requirement, and prediction is the ultimate objective. Many intelligent applications in reality often use the speech recognition technology to identify information such as user instructions to obtain an user's motivation in advance, and then provide services. In our case, the information are obtained by behavioral analysis. Accurate activity recognition is necessary for intervention and behavioral monitoring. Furthermore, activity prediction is often more practical in preventing serious situations.

1.5 OVERVIEW OF ACTIVITY RECOGNITION

AmI covers a wide range of AI research topics. However, the most important one is the human activity recognition and behavior understanding. The ultimate objective is to recognize human behaviors and understand real-time situations within a smart environment, in order to predict next behaviors, provide proactive services, detect abnormal activities and thereby prevent undesirable consequences [38, 39]. Activity recognition is a sort of empirical science, which involves the observations and hypothesis of human behaviors [7]. It analyzes massive data gathered from heterogeneous data sources to recognize different behavioral patterns describing specific activities of interest [40]. According to different types of data sources, the solutions of activity recognition can be broadly classified as two categories: vision-based activity recognition and sensor-based activity recognition.

The vision-based activity recognition uses RGB/depth cameras to capture image or video sequence. The captured information indicates the real-time positions of moving objects or the latest states of monitored objects. Each image (or each frame of a video sequence) is a set of pixel values. According to information entropy [41], the vision-based activity recognition captures more details about living environments than the sensor-based one. Thus, it has better performance in AmI applications [42, 43, 44]. In contrast, efficient image processing, machine learning and pattern recognition algorithms [45, 46, 47, 48] have to be used to handle large-scale pixel values. The characteristics of pixel values with known patterns which resemble existing images are compared and analyzed. Also, because vision-based activity recognition directly acquires highly sensitive personal information, the trade-off between privacy and excellent performances has always been controversial [49].

Besides, rather than use the natural characteristics of data, the vision-based activity recognition takes more time in the preprocessing phase. Derived features have to be generated

from the pixel values to detect desired portions or shapes. As a consequence, more AmI applications have chosen the sensor-based activity recognition.

For the sensor-based activity recognition, it has some advantages such as a smaller amount of data to be analyzed, fewer controversies about privacy, and more accurate ability to capture environmental conditions. Non-intrusive sensors like electromagnetic contacts, motion detectors, radio-frequency identification readers/tags, power analyzers, smart plugs and pressure mats are used to collect diverse measures of current states within a smart environment (e.g. distances, motions, environmental changes, usages of household appliances, energy consumptions, etc.).

In this thesis, we only take into account the sensor-based activity recognition except for wearable accelerometers [39] and mobile phone sensors [50, 51]. This is because not everyone can accept their ways to gain data. In Appendix A, we discuss in detail the infrastructure design of experimental sensor-based testbeds, and those non-intrusive sensors used in smart environments.

A general architecture of any AmI system is defined in [52]. As shown in Fig. 1.1, in the first place, massive data are monitored and collected from smart environments at the preparing stage. In the second place, by using data-driven, knowledge-driven or hybrid approaches, raw data are processed and segmented from continuous data. After that, human activities are inferred and recognized by mixed activity inferences to guide and provide assistance or intervention. In the third place, advanced HCI technologies can ensure that the assistance and interventions are fed back to residents in various ways. Likewise, we use a similar architecture to capture sensor data from smart environments, infer ongoing activities, provide assistance, and use multi-modal interaction to assist or intervene residents in the completion of activities.

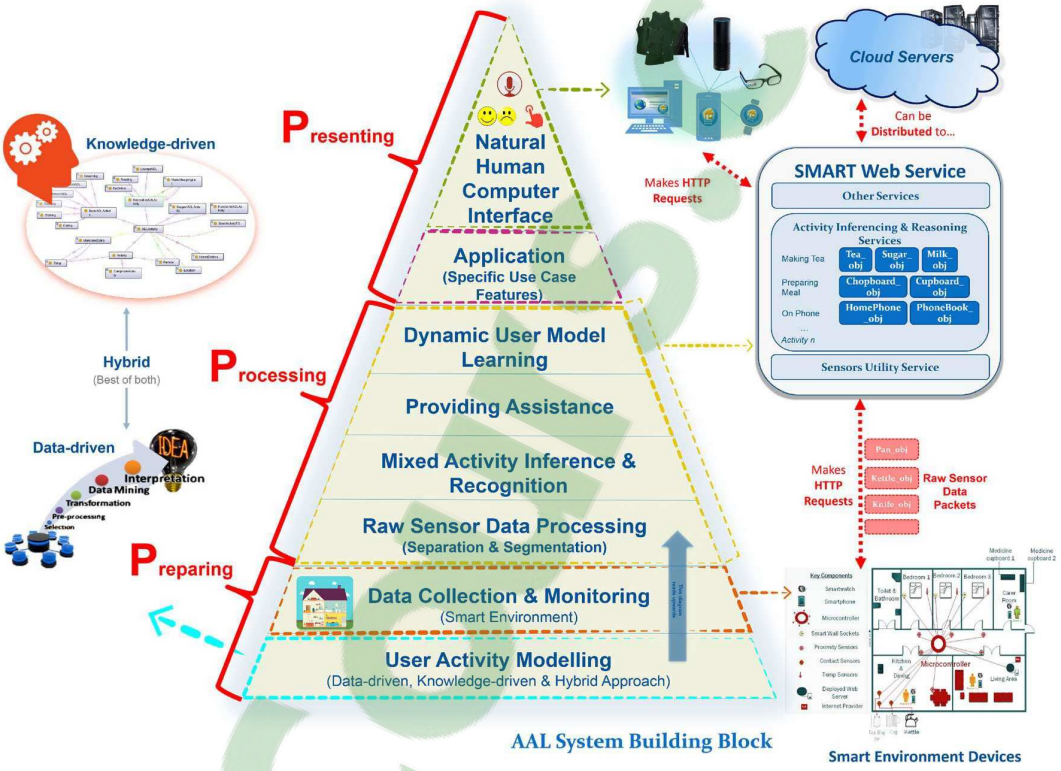


Figure 1.1: General architecture of an AmI system with different stages [52]

1.5.1 SIGNIFICANCE OF ACTIVITY RECOGNITION

In order to provide relevant feedback or assistance to residents, almost all AmI applications need to understand the current situation within a smart environment as soon as possible, especially the behavioral information of residents [53]. Understanding the current situation can also determine if residents have difficulty completing their activities [54]. On the contrary, inaccurate activity prediction and recognition will mislead residents, and lead residents to lose trust in the proposed suggestion. Furthermore, residents have to spend more time to correct or cancel inappropriate assistance.

Indeed, the most effective way is to directly inform the AmI applications what is the real intention of residents. Nevertheless, most of the time, it is impractical to allow residents to communicate their intentions directly with the applications. Thus, as one of the most important prerequisites, activity recognition takes responsibility of mining, translating and understanding the real intentions indirectly behind a series of observable behaviors. Moreover, modeling human behavior and understanding behavioral patterns involve a number of tasks [53] and each of them can affect the final recognition results.

1.5.2 ABSTRACTION OF ACTIVITY RECOGNITION PROBLEMS

In computer science, abstraction is a modeling process that removes minor, unnecessary or irrelevant reality details in order to focus on other details of interest. This is essential when building appropriate models, designs, and implementations for a specific purpose [55]. A good abstraction can improve the generalization of constructed models as well. For this reason, we present the formal definitions of activity recognition problems.

Data mining can be applied to multiple data types such as temporal data, sequential data,

spatial data, and multimedia data, etc. These types bring new challenges about how to mine patterns that carry rich structures and semantics. However, the form used in the AmI issues depends on the adopted sensors. Let $\mathcal{X} = \{x^{(0)}, x^{(1)}, ..., x^{(m)}\}$ be a collection of captured sequences of sensor data. Each sequence $x^{(i)}$ in $\mathcal{X}$ is a sequential description of human behaviors, called an "instance" or a "sample". Inside a sequence, smaller data fields that represent characteristics or natures of a sample in certain points of view, are called "attributes" or "features", represented as $x_d^{(i)}$. If there are d different attributes existing in the sample space $\mathcal{X}$, then they constitute a d-dimensional attribute space (or universe of attributes) at the same time.

The literature of data mining and formal concept analysis to be introduced later trends to use the term *attribute*, while statisticians prefer the term *variable*. Pattern recognition professionals commonly use the term *feature*, and we do here as well. Every attribute has a feature value. It can be either an enumerable or a discrete value such as nominal (categorical), binary or ordinal [56]. In contrast, the numeric values are usually quantitative and continuous, represented in an integer or real format. Based on the attribute space or universe of attributes, any sequence of sensor events $x^{(i)} = \{x_0^{(i)}, x_1^{(i)}, ..., x_d^{(i)}\}$ can be transformed into a d-dimensional feature vector $\mathbf{a_i} = [a^d]$, where $a \in \{0, 1\}$. For example, if the universe of attributes is equal to $M = \{m_0, m_1, m_2, m_3\}$, and an observed sequence of sensor events is $x_0 = \{m_1, m_0, m_3, m_1\}$, then the feature vector is equal to $[1, 1, 0, 1]$.

Meanwhile, datasets are made up of samples. Pair $(x^{(i)}, y^{(i)})$ is called the *i*-th *training sample* if sequence $x^{(i)}$ is labeled by the ground truth $y^{(i)}$ for training. In fact, the sensor-based activity recognition is a multi-class classification that learns regularity from a training dataset $\mathcal{D} = (\mathcal{X}, \mathcal{Y}) = \{(x^{(0)}, y^{(0)}), (x^{(1)}, y^{(1)}), ..., (x^{(i)}, y^{(i)})\}$, where $y^{(i)} \in \mathcal{Y}$ and $| \mathcal{Y} | \geq 2$. The objective of an activity recognition system is to build a mapping $f : \mathcal{X} \to \mathcal{Y}$ from the input space $\mathcal{X}$ (i.e. sensor data) to the output space $\mathcal{Y}$ (i.e. inferred activity labels).

1.5.3 DATA GRANULARITY

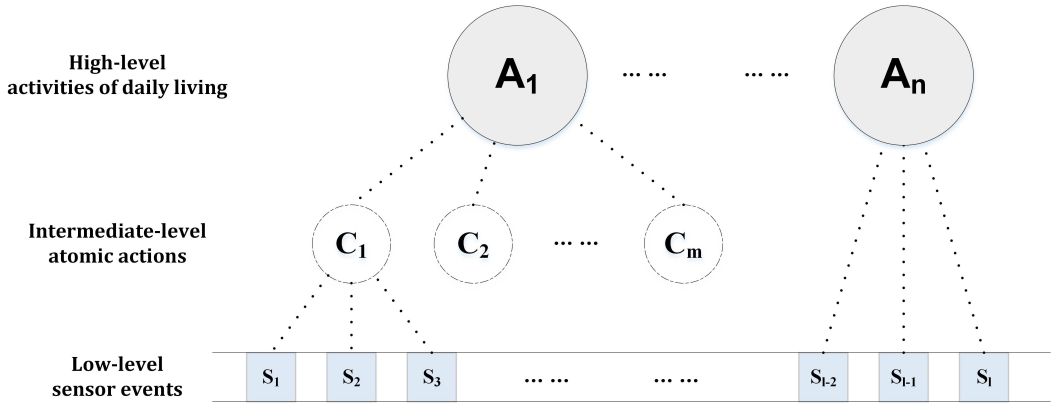


Figure 1.2: Multiple data granularity in smart homes

In the research of ubiquitous computing, the data produced by the facilities in a smart environment can be divided into three layers according to different granularity [45, 57]. Each layer represents a type of behavioral data. As shown in Fig. 1.2, they are low-layer *sensor data*, intermediate-layer *atomic actions* and high-layer *activities of daily living*. Based on facts, their interrelationships can be defined as follows: each atomic action (hereafter referred as *action*) is the smallest meaningful behavioral unit describing a short-term intention of residents. An action is transitory and indivisible, and can be detected by one or more sensors. At the same time, an activity consists of multiple actions. Each activity indicates a real long-term intention of residents.

There are two main many-to-many mappings among them. Figure. 1.2 indicates such mappings. The first one is from low-layer sensor data to high-layer activities (i.e. $S_l \Rightarrow A_n$). The second one is from intermediate-layer actions to high-layer activities (i.e. $C_m \Rightarrow A_n$). Fine-grained elements are located at relatively lower layers (e.g. S_l or C_m for A_n). Each coarse-grained element is composed of one or more fine-grained elements. For instance, an activity "prepare dinner (A_1)" consists of several actions like "take out something from a re-

frigerator (C_1)" and "preheat an oven (C_2)". Furthermore, both C_1 and C_2 can also be detected and represented by one or more sensor data (S_l). Both of the mappings will be validated by our proposed method.

The LIARA datasets, which are used in our experiments and will be introduced in Appendix A, are sequences of actions labeled with timestamps. These actions are obtained by the previous research of LIARA laboratory. In the work of Fortin-Simard et al. [58], the topological relationships among the objects attached by RFID tags are analyzed to infer actions done by a resident. In the work of Belley et al. [59, 60], the load signatures of appliances are extracted to identify the power-consuming actions related to electrical devices. Thus, in the following chapters, we ignore the mapping $S_l \Rightarrow C_m$, and directly use the above results of previous studies to recognize complex human activities from the sequence of actions, that is the mapping $C_m \Rightarrow A_n$. It is worth mentioning that any C_m could belong to more than one activities A_n in some complex scenarios.

The CASAS datasets described in Appendix A, which are a collection of benchmark datasets used in our experiments, contain the sequences of sensor data labeled with the information about the ground truth, such as performer ID, activity ID, and timestamps, etc. Consequently, we directly validate our proposed method by mapping low-level sensor data to high-level activities, that is the mapping $S_l \Rightarrow A_n$.

For activity recognition task, higher layers of representation amplify discrimination and suppress irrelevant variations. Coarse-grained behavioral units have a stronger semantic representation and differentiation ability than fine-grained ones, and the correlations among them are clearer. This is the reason that recognizing activities by coarse-grained actions [61, 62] have better results than the one by fine-grained sensor data [57]. For example, sensor data can appear in several sequences describing different irrelevant activities due to weak semantic

representation and differentiation.

1.5.4 CHALLENGES

For the scientific community, human activity recognition has always been a serious task [63]. AmI applications bring us new challenges about how to explore useful patterns from behavioral data having sequential structures and rich semantics. Therefore, in the following subsections, we investigate some key factors that can greatly affect the accuracy of activity recognition.

Mining Massive Data

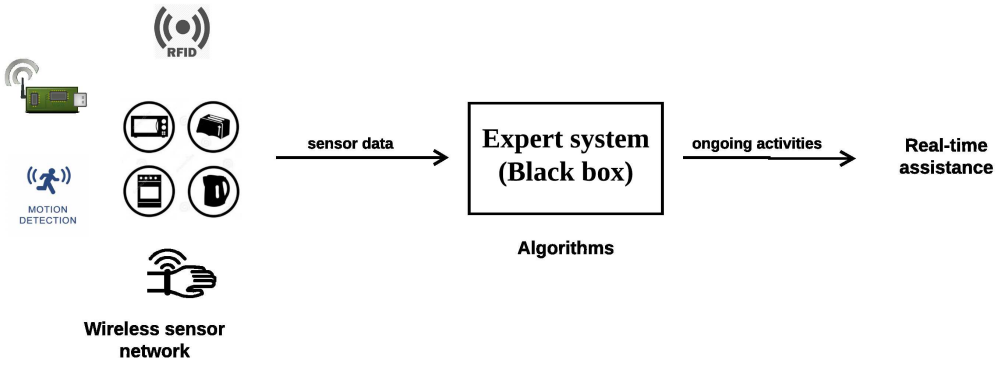


Figure 1.3: Real-time assistance in smart homes.

Smart environments are designed to monitor and record situations that occur in the living environments all the time. Nowadays, more and more household appliances and daily necessities are integrated seamlessly with wireless networks as intelligent components in smart environments. Due to lack of uniform specifications widely used and accepted by industry, these intelligent components made by different manufacturers may produce disparate data with various types or structures [27, 64]. Related solutions [65, 66] are still in the start-up stage. Therefore, captured data that recorded in a log system are usually massive, unlabeled

and continuous with various data types [67, 68].

As shown in Figure 1.3, the data type of captured data can be discrete, continuous, nominal (categorical), binary, ordinal or numeric values that describe continuous changes about environmental states in a smart environment [69]. Analyzing and mining such data is an important need for AmI applications. Massive data should be turned into knowledge by efficient knowledge representation and management techniques.

Moreover, determining the boundaries between activities, which means the beginning and the end of a sequence describing an activity in a data flow, is another challenging problem [52]. To obtain the best results of activity recognition, most of the methods choose data segmentation to roughly classify the data segments by activities.

Generally, there are two common ways to identify the boundaries between activities. A resident may take a break to perform the next activity after completing an activity. Thus, one is to differentiate data segments by identifying longer time intervals between data segments describing different activities. The other is from the perspective of the different semantic gaps between different activities.

In other scenarios, where composite activities [57] are involved, not only the boundaries of data segments describing different activities are difficult to be identified, but also the sequences or fragments describing multiple activities are mixed, that makes it difficult for a model to identify which data belong to which activity. What's more, a piece of data fragment may belong to multiple collaborative activities when a collaborative task is completed in a multi-resident scenario. When there is more than one resident in the same monitored zone, it is also difficult to identify who triggered the sensor events by non-intrusive sensors, without labeled data.

Various Categories of Behavioral Patterns

A behavioral pattern can be understood as a set of sequential and temporal data that contains a sequence of characteristics describing a particular activity. It can also be treated as a permutation of a set of characteristics under specific constraints. If a characteristic is repeatable, optional, and its position in the sequence can be variant, the number of permutations, in other words, the number of behavioral patterns describing the same activity, will be infinite.

For the reason of varied living habits, personal preferences or the other external factors, an activity can have multiple different lifestyle patterns to describe itself. Even if having almost the same constituent data, two patterns could be totally dissimilar due to repetitive or optional data, and their different execution orders.

For example, if a resident keeps staying in a certain area, the movements will be frequently captured by several motion sensors. Another example, in the process of preparing a cup of coffee, you can add milk first, and then add sugar, or conversely (flexible execution order), or without adding milk (optional action) due to different personal tastes.

Additionally, according to the number of residents and different ways of human-object interactions, activities can also be classified as basic, composite and multi-resident ones [70]. For the basic activities, the sensor data collected in a period only describes a single activity. In other words, the boundaries of behavioral patterns are precisely segmented by activities. However, it is the most basic situation and unrealistic in reality. Most of the time, a resident performs activities in composite ways, such as sequential, interleaving and concurrent [52]. If there is more than one resident, the situation will be more complicated. This is because each resident may perform basic or composite activities, and it is also difficult to identify who triggered which sensors.

Temporal and sequential sensor data with human behavioral information collected in a fixed time interval can be referred as *a sequence of sensor events*. Each sensor event corresponds to a feature. Because of varied living habits or other external factors, an activity may be described by diverse behavioral patterns having different optional features. Even if having the same sets of features, two patterns may be completely different due to different orders and unavoidable repetition of certain sensor events. Thus, activity possessing j different sets of features can derive N_i different patterns, and $N_i \gg j$. To simplify various activity recognition and anomaly detection, we formally define a variety of behavioral patterns.

Single Patterns Single patterns are the simplest form among numerous behavioral patterns. All data captured during a fixed time interval describe only one activity. Although all data is related to only one resident, the recognition task is still a complicated task. This is because there may be a variety of behavioral patterns that describe the same activity.

```
Pattern 1: abcdefgh            (standard pattern)
Pattern 2: abacadaeafagah      (recurrent sensor reading ‘a’)
Pattern 3: eadcfgbh            (flexible execution order)
Pattern 4: eadcfgb             (without optional sensor reading ‘h’)
Pattern 5: aedcfgbhi           (with optional sensor reading ‘i’)
```

Figure 1.4: Different behavioral patterns describing the same ADL.

Figure 1.4 illustrates an example about the diversity of behavioral patterns. Suppose that each letter in the five patterns indicates a sensor reading generated or affected by a human behavior. Although some of these patterns are dissimilar in their compositions, they may also describe the same activity (e.g. prepare a cup of coffee).

For instance, if Pattern 1 is a standard pattern that most people follow to prepare a cup of coffee, then Patterns 2 to 5 indicate other four deviations. Compared to the other sensor readings that indicate meaningful behaviors, some meaningless ones like motion sensor data

may recur in a pattern (e.g. reading “a” in Pattern 2). In another case, the orders of two or more readings may be reversed due to lack of order constraints (e.g. in Pattern 1 and 3, the orders of “c” indicating “take out a coffee cup from cabinet” and “d” indicating “take out a spoon from cabinet”). In fact, different people usually have different ways to carry out an activity, their habits and personal preferences are reflected as optional behavioral data in patterns [68, 71].

As a consequence, the variety of human behaviors makes it difficult to recognize corresponding behavioral patterns by conventional similarity-based [56, 72], frequency-based [73, 74] and case-based [56, 75] data mining methods. The reason is that the number of variations that describe the standard pattern of an activity is theoretically infinite, and it is impossible to cover all possible situations due to the repetition and optional data. The unbalanced distributions of patterns in a training dataset can cause high rates of misdetection, especially the false alarm rate.

Composite Patterns We present some composite patterns in this paragraph. In the following definitions, we assume that each pattern is performed by only one resident, which means that the patterns belonging to the multi-resident scenario are not considered here.

1. Sequential Pattern: The sequential mode is a typical composite pattern in which activities are performed one after another in a sequential way without interweaving. Figure 1.5 illustrates such an example. There are 4 steps in task *a* followed by 3 steps in task *b*. For instance, a resident may prepare a cup of coffee after preparing a sandwich. In addition, each activity is independent, and there are few shared behaviors between two successive activities.

2. Interleaved Pattern: In the interleaved pattern, the behaviors of different activities are interwoven with pauses. As shown in Fig 1.6, a resident may temporarily suspend cur-

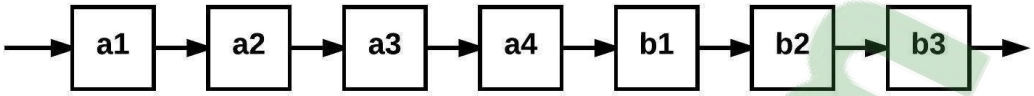


Figure 1.5: Example of the sequential pattern of single-resident activity recognition

rent ongoing activity and begin to do another one, the suspended one will be completed later. In fact, when an ongoing activity needs to wait for processing, a resident usually carries out another activity during the waiting time (e.g. while waiting for cooking spaghetti, a resident may start to prepare a cup of coffee). In other words, a resident may frequently schedule or plan his/her behaviors among different activities. Furthermore, some behaviors belonging to different, but similar activities may be shared in some cases.

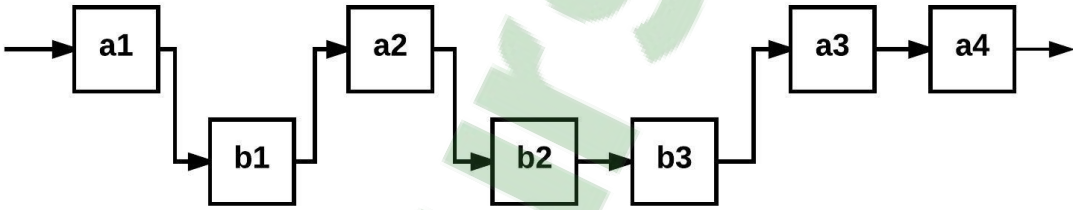


Figure 1.6: Example of the interleaved pattern of single-resident activity recognition

3. Concurrent Pattern: In the concurrent pattern, a single resident may perform multiple activities at the same time. As a result, many behaviors are shared or interwoven among different activities. Although these patterns are similar to the interleaved ones, the biggest difference is that different behaviors can be done at the same time (see a_2 and b_1 in Fig. 1.7). For example, because there is no order inversion, a person can make a phone call while cooking.

   In fact, the composite patterns appear more frequently in reality than the single ones. In addition to the various behavioral patterns mentioned earlier, the composite patterns focus more on classifying unbounded and mixed sensor data. In other words, it is

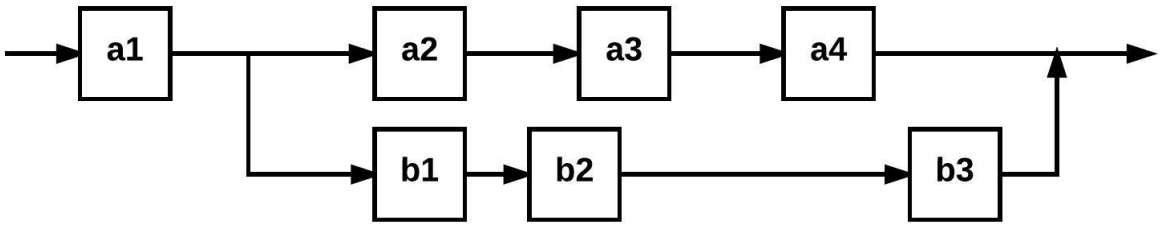


Figure 1.7: Example of the concurrent pattern of single-resident activity recognition

necessary and important to roughly determine each sensor data belongs to which unrecognized activity before it is processed to activity recognition process.

Multi-resident Patterns Compared with the single-resident activity recognition, recognizing activities in the multi-resident scenario is equally important. People usually live with the other family members, such as their parents, spouses and children, so that there will be more than one resident in a smart environment. Thus, the number of residents in a smart home is usually more than one. At this time, the behaviors of different activities performed by different residents may be captured by the sensor network at the same time, and will be mixed together. There are two common kinds of multi-resident behavioral patterns:

1. Parallel Pattern: In the parallel pattern, many residents perform more than one activity at the same time. It is the multi-resident version of concurrent activities. Their patterns look like the one shown in Fig. 1.7, however, activities are carried out by different residents (indicated as “P1” and “P2”, similarly hereinafter, see Fig. 1.8).

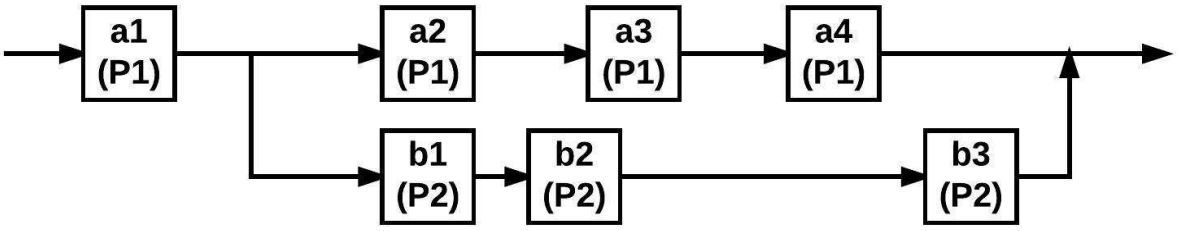


Figure 1.8: Example of the parallel pattern of multi-resident activity recognition

2. Collaborative Pattern: In the collaborative pattern, residents work together in a collaborative way to finish the same goal. As shown in Fig. 1.9, an activity can be implemented by more than one resident. For example, two residents cooperated in preparing a dinner, and both of them are involved in the preparation of each dish. For activity a, behaviors a_1, a_3 are performed by P_1 and a_2, a_4 are performed by P_2.

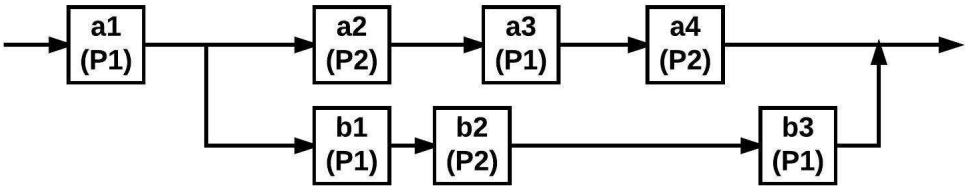


Figure 1.9: Example of the collaborative pattern of multi-resident activity recognition

The analysis of each sequential and temporal pattern is essential to help us find the regularity of data in different scenarios. More detailed information about recognizing activities is described in the next chapters.

1.6 OBJECTIVES OF THESIS

With the help of advanced HCI technology, intelligent assistance can be reflected in multi-modal interactions such as synthetic voice, image, video or text modality. Many IT vendors have increased the investment of research and development, in order to design their own smart home devices and applications, such as Google home, Google assistance, Siri, and Cortana etc. They also provide rich APIs that allow researchers to develop their personalized smart device. Once such assistance is needed to guide or warn residents, a multi-modal message can be sent to corresponding interactive devices like information terminals or wearable devices, to prompt the next step. In some extreme situations like forgetting to turn off the stove, preventive interventions such as shutdown could be executed to avoid further severe consequences.

AmI integrates a variety of sensors to understand human behaviors. As a promising solution, smart homes attempt to support residents by providing appropriate assistance, health, and safety monitoring [76]. In order to achieve this goal, in this thesis, we propose a prototypical inference engine based on graphical models to dynamically analyze human behaviors from ubiquitous sensor data. The thesis includes work on knowledge representation, pattern recognition, and anomaly detection. As a consequence, we define the following objectives:

- how to represent and manage knowledge information
- how to predict and recognize various human activities
- how to formally define and detect errors
- how to ensure the robustness of the constructed model

In the following subsections, we discuss each objective in details, such as their descriptions, significances, the roles in AmI applications and our expectations.

1.6.1 KNOWLEDGE REPRESENTATION AND MANAGEMENT

Good knowledge representation can facilitate the representation and management of discovered knowledge. Compared with other non-graphical models such as decision trees, association rule learning or K-means clustering, graphical ones can better represent the state transitions and context-aware features of sensor data. This is because these transitions or features can be represented as edges or nodes. In this thesis, we use an innovative graphical model to represent, organize and retrieve useful patterns in sensor data. The hierarchical relations between sensor data, behaviors and human activities are reflected. In [61, 77], we propose a lattice model based on formal concept analysis to represent and manage binary relations among hierarchal behavioral data.

From raw data to the discovered knowledge used for inference, there are several critical processes. After acquiring data from disparate data sources in a pervasive sensor network, massive sensor data have to be sorted according to their timestamps in order to convert them into sequential and temporal data. After that, the feature selection will filter irrelevant features and choose the most representative ones to build models. Some optional operations, such as pruning, can remove redundant data, or complement missing values by default ones. Finally, we use data mining algorithms to extract knowledge from the data and build the knowledge base.

Moreover, through a good knowledge management, we hope that the constructed knowledge base can be reused for other similar smart homes with similar infrastructure design, and can be extensible with new and homogeneous knowledge and scenarios.

1.6.2 REAL-TIME ACTIVITY RECOGNITION

In AmI applications, one of the most important preconditions for appropriate assistance is to understand the current context of residents [38], in other words, the real requirements of residents. In the broader sense, context awareness uses observed sensor data to abstract information about the current situation. Context-aware systems are able to adapt their operations to the current context without user intervention and thus aim at increasing usability and effectiveness by taking environmental context into account. It is desirable that services react specifically to environment attributes and adapt their behavior according to the changing circumstances as context data may change rapidly [30].

Each assistance offered by the smart environments should satisfy user's real needs, otherwise, it will increase the burden of residents to correct the mistakes. Historical data is a great treasure for data analysis. Most of them contain valuable information including regular patterns

or useful cases. At the same time, they are usually difficult to be used directly to solve practical problems due to the lack of efficient knowledge discovery and retrieval strategies. Thus, it is essential to choose an effective representative form to index, organize and retrieve unstructured information [35]. Because of the periodicity and regularity of human behaviors caused by the habits and preferences of residents, it is possible to discover and analyze behavioral patterns in smart environments by means of data analysis techniques, such as data mining and pattern recognition. Instead of short-term intentions (i.e. actions) describing instantaneous human behaviors or human-object interactions, long-term intentions (i.e. activities) are more meaningful and have enough intervals to provide follow-up assistance.

The nature of AmI requests that the adopted recognition algorithm itself cannot spend too much time to process continuous data. Furthermore, conventional solutions mainly focus on the finished activity recognition, which means that they only analyze entire sensor data describing an activity. Although their performances are encouraging, the main drawback is also evident: appropriate assistance cannot be provided on time, in other words, only a few assistance can be provided after an activity has been done. In this case, the related advice is useless. For example, the dosages and recommended directions should be provided before a resident takes dietary supplements. Thus, tips should be provided before taking supplements.

Consequently, we hope that the proposed method will be able to handle partially observed data and give reasonable candidates about ongoing activities. With the increase in observed data, the scope of potential candidates should be reduced.

1.6.3 ACTIVITY PREDICTION

As studied in [38], another important precondition for appropriate assistance is the anticipatory capability. It allows a system with the predictive capability to produce a timely and

useful response.

In some cases, especially in the context of AmI, recognizing a completely finished activity may not be helpful because no assistance was provided during execution. Compared with activity recognition task, activity prediction is required to infer the most possible ongoing activity using limited observed data. The performance of smart homes can be greatly improved if it enables to predict an ongoing activity as early as possible according to cumulative observed data.

1.6.4 DEFINITION AND DETECTION OF ABNORMAL BEHAVIORS

Summarizing common human abnormal behaviors, we will analyze their regular patterns and features from captured data streams. Those patterns having the anomaly in the execution order, completeness and composition parts will be extracted. Some errors are related to the composed behaviors, such as irregular repetition or omission. The others are related to the order constraints, or the semantic difference of data.

After formally defining characteristics of each abnormal behavioral pattern, for each error, we will design a custom-built extension module to detect similar abnormal patterns in the experiments. In some specific cases, weights will also be used to control detection sensitivities.

1.6.5 ROBUSTNESS

In software engineering, the robustness of a system refers to the ability that handles exceptions or erroneous inputs during execution. For machine learning or data mining algorithms, it refers to the performance of dealing with the datasets with noisy data or missing values [78]. A dataset with noisy data means that its data contain errors. They can be of two types: inaccurate attribute values or incorrect class labels. They can make the algorithms have poor

classification accuracy on unseen examples.

In terms of our research issues, collected sensor data in smart environments is usually unreliable. This is because the data usually comes from a variety of unreliable sources, which makes it difficult to guarantee the integrity and correctness of data. Frequent sampling and accidental triggering can result in redundant data and inaccurate attribute values. Some sensors may also fail and cause missing values. Sometimes the ground truth is ambiguous, especially in the multi-resident scenario. It is difficult to determine exactly who triggered certain sensor events or an ongoing action belongs to which activity. At this time, the data annotation usually depends on the subjective decisions of observers in an experiment. Thus, incorrect class labels may be assigned.

As a consequence, we hope that our system can get rid of the difficulties caused by data quality and maintain stable accuracy in complex and changeable smart environments. Moreover, we also hope that the system can make a reasonable inference on the examples with unseen patterns.

1.7 THESIS FRAMEWORK

This thesis proposes an innovative activity inference engine to address the aforementioned objectives. It tries to avoid specifying the required knowledge through domain experts. Our proposed solution considers the ontological correlations among interested activities. At the same time, it allows smart environments to learn knowledge automatically from experience, such as historical data, and to understand the context inside a smart environment in terms of conceptual hierarchy.

The remainder of the thesis is organized as follows: In Chapter 2, we introduce the summary of recent research about data mining technique applied to AmI. We classify the data mining

algorithms into graphical and non-graphical categories according to the structures of their built models. In addition, we compare and analyze their performances in the AmI scenarios. In the end, because of the better representation of dynamic state transitions, we are more inclined to use graphical models to solve AmI problems.

In Chapter 3, we introduce the theoretical basis of our research, a mathematical theory called formal concept analysis. It is used as an efficient tool to represent and manage discovered knowledge. It is composed of five major components, which are respectively responsible for extracting features, reformulating captured data, maximizing similarity among patterns, merging and encapsulating similar patterns as inferences, sorting inferences for fast information retrieval, and visualizing the discovered knowledge. Besides, how to apply formal concept analysis into ambient intelligence and the role of each component in activity recognition are described in detail. Moreover, in order to overcome the natural limitation of formal conceptual analysis in dynamic search, a new lattice search algorithm is proposed to retrieve the inferences in a graphical knowledge base incrementally. These studies have been summarized as a conference paper [61]. So far, we establish an embryonic model for activity prediction and recognition. Furthermore, to improve the prediction accuracy when only a few data are available in the recognition process, we propose an ontological clustering approach to further cluster discovered inferences. For example, a more general inference like "prepare something to drink" will be prompted to residents instead of a precise inference like "prepare a cup of coffee". This part of the research has been summarized as an article [77] published in the Journal of Ambient Intelligence and Humanized Computing.

After accurately predicting and recognizing human behaviors, another important AmI application is the anomaly detection and composite activity recognition. When a resident has a tendency to make abnormal behaviors, corrective suggestions or interventions may be provided in an appropriate moment. Chapter 4 consists of two parts. In the first part, the study is

devoted to more complicated activity recognition. Unlike the basic activity recognition that the data collected in a period of time only describes one activity, in reality, a resident normally performs more activities concurrently, intermittently or successively with more complicated patterns. As a reaction to such an issue, we propose an extended search strategy to identify these specific patterns in the lattice knowledge base. This study has been published in the Journal of Reliable Intelligent Environments [57].

In the second part, through analyzing the behaviors produced by volunteers, we formally define several abnormal behavioral patterns and propose self-built modules to detect those anomalies. This study has been published as a conference paper [62].

After taking into account all the complicated scenario of single resident activity recognition, Chapter 5 discusses the multi-resident activity recognition. In this case, each collected data no longer has a unique trigger source. It may be produced by one or more residents. Moreover, an activity can be completed in collaboration with multiple residents. Besides another specific multi-resident search strategy, to identify cooperative activities with highly similar patterns, we propose transition matrices to represent the context of collected data. This research has been published in the Journal Neurocomputing [79].

An optional extension, incremental learning, is developed in Chapter 6. In order to avoid retraining the entire model when new training data or features are available, we improve an efficient algorithm of lattice construction to adapt to the smart environments that constantly change its infrastructure design. This research will be submitted soon as a journal article [80].

In Appendices A, we present basic infrastructure designs of typical sensor-based smart homes. We introduce the datasets collected from different scenarios to solve different AmI problems, such as basic activity recognition, composite activity recognition, multi-resident activ-

ity recognition, and anomaly detection. We briefly introduce several common methods to measure the model performance in terms of generalization and recognition in Appendix B.

CHAPTER 2

LITERATURE ON DATA MINING APPLIED FOR AMBIENT INTELLIGENCE

As mentioned in Chapter 1, the issues about recognizing activities in sensor-based smart environments can be regarded as a problem about mining behavioral patterns in sequential and temporal data flow. Data mining focuses on discovering interesting patterns from data in various applications, and furthermore, developing effective, efficient and scalable tools [56].

In fact, we can make an effective prediction because we have accumulated a lot of experience, and through the use of experience, we can make effective decisions in new situations. Data mining is such a sub-discipline of artificial intelligence that focuses on the automatic summarization and induction of useful information from historical data. It is also an essential process of knowledge discovery that extracts data patterns (see Fig. 2.1).

Because of the fast development of powerful data collection and storage tools, people live in smart environments where vast amounts of data are collected daily. However, numerous captured data have far exceeded our human ability to handle with them without powerful tools. Such a dilemma has been described as “data rich but information poor” situation [56]. Moreover, the manual knowledge extraction and discovery with the intervention of domain experts are prone to biases as well as errors, and is extremely costly and time-consuming. As a consequence, we must find ways to automatically analyze the captured data containing behavioral

information, characterize trends, discover interesting patterns and flag data fragments having anomalies.

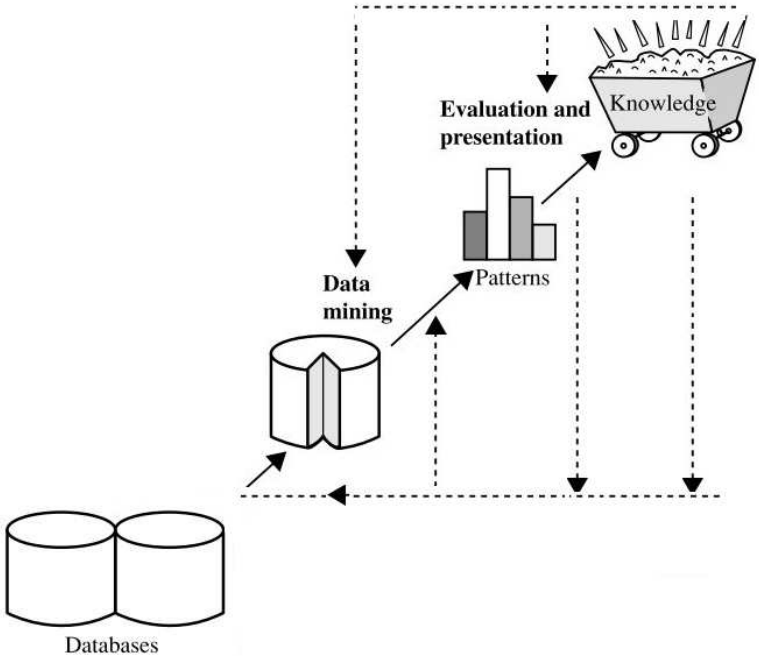


Figure 2.1: Data mining, a step of knowledge discovery [56].

Therefore, this chapter will focus on the core methods of data mining and machine learning, and their applications in activity recognition.

2.1 MACHINE LEARNING VERSUS DATA MINING APPROACHES

Data mining is a practical learning technique that turns a large collection of data into knowledge [56]. In other words, it extracts implicit, previously unknown and potentially useful information from large-scale data for further inference. Besides knowledge discovery, it also involves the efficient data management and analysis. The main objective is to automatically seek and sift regularities and representative patterns from databases. Discovered knowledge will be used to make accurate decisions on the future data [81]. Similarly, the goal of machine learning is to develop methods that can automatically detect patterns in huge data reposito-

ries, and then to predict future data. Some classic problems in machine learning are highly related to data mining, but differs slightly in terms of its emphasis [56, 73, 82]. Most machine learning approaches are inclined to use the systematic application of probabilistic reasoning to explore the best prediction given some data in a precise and quantitative manner [46, 83].

Compared with machine learning, data mining focuses on discovering unknown knowledge from raw data and forecasting what will happen in new situations. It concentrates more on data features, statistical correlations, data similarity, dissimilarity, semantic relationships as well as relational characteristics to discover useful patterns [56]. For the machine learning technique, it prefers to apply calculus, linear algebra, and probability theory for quantification and manipulation of uncertainty [46].

In this thesis, we prefer to use data mining to recognize human behaviors, because the captured sensor data have rich contextual, semantic and relational features. These features have better distinguishable abilities to classify different activities.

2.2 DATA-DRIVEN VERSUS KNOWLEDGE-DRIVEN APPROACHES

In the early stages of AI development, due to more limited computer hardware calculation and data processing ability, several AI projects have sought to hard-code knowledge about the world in formal languages [5]. These systems used logical inference rules or ontologies to reason cases automatically. However, the biggest drawback is that those AI systems have to devise enough confident and accuracy rules to describe the world.

Compared with data-driven approaches, knowledge-driven ones have several advantages [73]. First, knowledge representation is easier to be understood and interpreted by researchers and domain experts [84]. This is because knowledge-driven approaches have sought to hard-code knowledge about the world in formal languages using logical inference rules [5]. Second,

classification results can easily be explained. Third, knowledge-based models can be easily extended by new domain knowledge.

However, domain knowledge or datasets produced by experts are often expensive, inflexible or simply unavailable. Even in the ideal case, they may impose a ceiling on the performance of systems trained in this manner [85]. Thus, we wish to find out a solution that allows expert systems to operate in complex sensor networks where human expertise is lacking. As a consequence, the expert system provides lookahead inferences to narrow down the search for high-probability ongoing activities.

In many domains, especially in the applications involving complex pattern analysis, interpretable models are more desirable [84]. Domain experts prefer transparent predictive models rather than black-box ones [86], because the former ones make easier to find out the key factors involving the performance of models and then improve it.

2.3 SUPERVISED VERSUS UNSUPERVISED APPROACHES

Data mining approaches can be categorized into two main tasks based on whether training data has been labeled: supervised learning and unsupervised learning. The task of supervised learning can be described as follows: given a training set of N input-output pairs

$$(\mathbf{a_1}, y_1), (\mathbf{a_2}, y_2), ..., (\mathbf{a_N}, y_N) \tag{2.1}$$

where each $\mathbf{a_i}$ is an input vector of features and the corresponding y_N is a label information.

Supervised learning can be further subdivided into two categories: *classification* and *regression* [83, 87]. In the supervised learning, the goal is to predict the value of y on unseen instances on the basis of each input vector [46, 88]. If the desired output y is one of a finite

number of discrete categories, the task is called *classification* problem, and *y* means the label of the target class. If *y* consists of one or more continuous variables, the task is called *regression*, and *y* is the value of the target variable to predict.

On the contrary, unsupervised learning does not rely on explicit label information, and its goal is to discover some inherent density estimation [46, 89] or distribution information [88] in the data. Compared with supervised learning, unsupervised one is nearer to human learning and more widely applicable. This is because unlabeled data is easy and cheap to acquire, it does not require a human expert to manually label the data.

2.4 GRAPHICAL MODELS

From the point of view of model visualization, commonly used data mining approaches can be divided into two categories: graphical and non-graphical models. This is also the point of entry for detailing each classic method.

Graphical models can provide a concise description about the structure of constructed models. From the perspective of data analysis, they have several advantages. First, graphical models are more suitable for representing dependencies relations between sequential, spatial or temporal data [90]. Second, the changes in states over time, such as transitions and shifting, are more easily described. Third, most graphical models are in the form of directed or undirected graphs, thus, they are homogeneous and easier to combine with other ones to produce new improved models.

2.4.1 BAYESIAN NETWORK

A *Bayesian network* (BN), also known as *belief network* or *causal network*, is a probabilistic graphical model that represents discrete or continuous variables as well as their conditional

dependencies via a *directed acyclic graph* (DAG) [82, 91]. Each node in the DAG corresponds to a random variable, and pairs of nodes are connected by the arrows that represent probabilistic dependences and causal knowledge. An arrow from node x to node y indicates that x is a parent of y and y is a descendant of x [56]. Each variable x_i has a corresponding *conditional probability table* (CPT) specifying the conditional distribution $P(x_i|Parents(x_i))$ that quantifies the effect of the parents [7], where $Parents(x_i)$ are the parents of x_i [56]. Equation 2.2 shows how a BN can be used to answer probability of evidence queries [56, 82, 92].

$$P(x_1, x_2, ..., x_d) = \prod_{i=1}^{d} P(x_i \mid Parents(x_i)) \tag{2.2}$$

where $P(x_1, x_2, ..., x_d)$ is the joint probability of a particular combination of evidences $X = (x_1, x_2, ..., x_d)$, and the values for $P(x_i \mid Parents(x_i))$ correspond to the entries in the CPT of x_i.

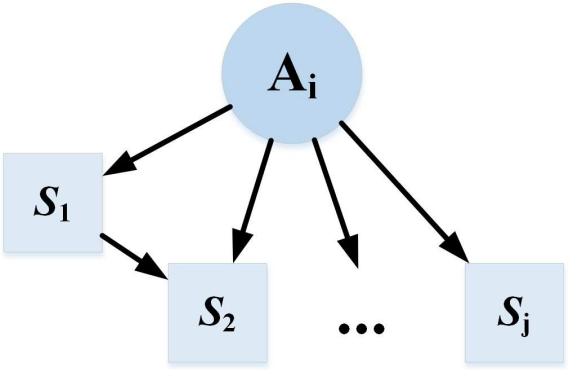


Figure 2.2: Bayesian network for sensor-based activity recognition

Thus, a BN $\langle G, \Theta \rangle$ is defined by two components, where G represents the directed acyclic graph, and Θ represents the set of CPTs that quantitatively describes conditional dependencies of each variable. Figure. 2.2 gives an example of the sensor-based activity recognition model using Bayesian network [93]. The symbol A_i denotes human activities, and s_1 to s_j denote sensor data. All related causal constraints are described by arrows. Therefore, the

network topology, also known as the layout of nodes and arcs, can be constructed by human experts or inferred from training data by several algorithms [92]. Experts must specify conditional probabilities for the nodes involving direct dependencies [56]. For the activity recognition problems, an activity may involve many internal or external factors, those dependencies among many activities, features and sensor data are difficult to be defined and specified by domain experts. In addition, it is also difficult to accurately measure the conditional probability that indicates the direct influence of one variable on another.

Another extension of the Bayesian network, the *dynamic Bayesian network* (DBN), is a Bayesian network that represents a temporal probability model widely used in the time-series analysis [7]. It consists of a series of time slices that record the snapshots representing the state of all variables at a certain time [90]. For simplicity, we assume that the variables and their links in a DBN are exactly reduplicated from slice to slice in a first-order Markov process [7]. As shown in Fig. 2.3, the nodes $Z_t = \{X_t, Y_t\}$ in a DAG represent random variables and the arcs direct the dependencies between variables [90]. The graphical structure encodes a set of conditional independent relations between the variables.

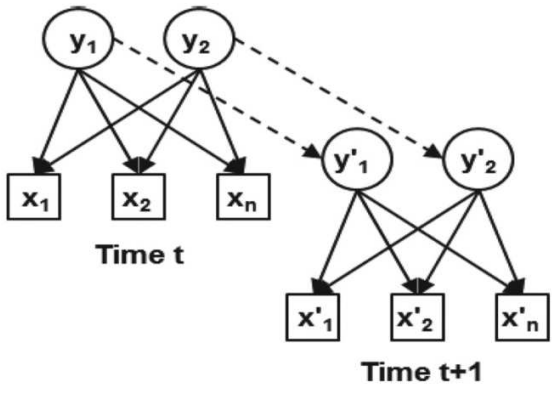


Figure 2.3: Dynamic Bayesian network applied to activity recognition problems [70]

To solve activity recognition problems, $Y_t = \{y_t, y'_t\}$ denotes activities, and $X_t = \{x_t, x'_t\}$ denotes sensor data. Thus, a DBN is defined to be a pair $(B_1, B_{\rightarrow})$, where B_1 is a BN defining

the prior $P(Z_1)$, and $B_{\rightarrow}$ is a two-slice temporal BN defining $P(Z_t \mid Z_{t-1})$ by means of a DAG as follows [94]:

$$P(Z_t \mid Z_{t-1}) = \prod_{i=1}^{n} P(Z_t^i \mid Parents(Z_t^i)) \tag{2.3}$$

where Z_t^i is the i'th node at time t, which can be a component of X_t or Y_t, and $Parents(Z_t^i)$ are the parents of Z_t^i in the DAG. Similarly, the joint probability is given by Equation 2.4:

$$P(Z_{1:T}) = \prod_{t=1}^{T} \prod_{i=1}^{N} P(Z_t^i \mid Parents(Z_t^i)) \tag{2.4}$$

where $Z_{1:T}$ indicate T time-slices. The inferences algorithms of DBN are summarized as exact and approximate inferences [91, 94].

Nazerfard et al. [95] proposed an activity prediction model using the Bayesian network with a two-step inference process to predict the next activity and its behaviors. In [96], Liu et al. presented a Bayesian network-based probabilistic generative framework to characterize the structural variabilities of complex activities. Kasteren and Krose [93] carried out activity recognition in a DBN to model the temporal aspects of activities. The dynamics of sensor data are taken into account by a *k*-observation history matrix.

2.4.2 HIDDEN MARKOV MODELS

Hidden Markov Model (HMM) can be represented as the simplest DBN, it is a probabilistic model composed of hidden and observable variables [97]. In our AmI issues, captured sensor data represent observable variables, and the activities to be recognized refer to the hidden variables. A sequence of observable sensor data $X = \{x_t\}_{t=1}^{T}$ and a sequences of activities $Y = \{y_t\}_{t=1}^{T}$ to decoder. In HMM, hidden variables (i.e. activities to be recognized) are linked

as a chain and governed by a Markov process. Observable sensor data are independently generated given the hidden state, which form a sequence.

The elements and the mechanism of HMM are listed below. There are two assumptions based on the Markov properties to simplify the inference process. One is that each state y_t depends only on its predecessor state y_{t-1}. Another one is that each observable variables x_t depends only on the current state y_t.

The modeling of the AmI issues via HMM are made according to three probability distributions: the distribution $p(y_1)$ over initial states, the transition distribution $p(y_t \mid y_{t-1})$, and the observation distribution $p(x_t \mid y_t)$.

The most probable inference is inferred by the maximum joint probability $p(x,y)$. The labels of activity class for observations are not only dependent on the observations, but also dependent on the adjacent states.

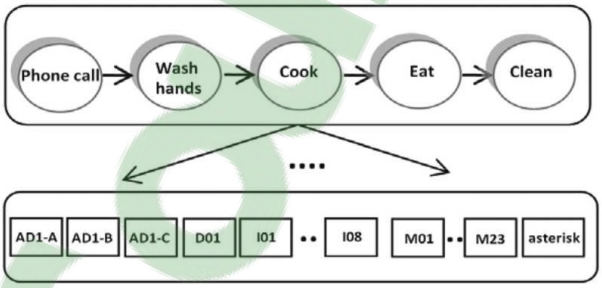


Figure 2.4: Representation of a global HMM [70]

As one of the most efficient technique interpreting sensor data at the early stage of AmI development, several solutions have achieved excellent results. Van Kasteren et al. [98] proposed a two-layer hierarchical model using the hierarchical hidden Markov model to cluster sensor data into clusters of actions, and then use them to recognize activities. Another Markov-based technique is called Markov decision process that analyzes collected continual observations and makes decisions based on the state of environment [99]. Chiang et al. [100] adopt two

graphical models, parallel HMM (PHMM) and coupled HMM (CHMM), to identify activities in a multi-resident environment. Benmansour et al. [70] developed an HMM-based combined label (CL-HMM) and a linked HMM (LHMM) to compare their performances against the PHMM and CHMM methods.

As shown in Fig. 2.4, HMM could infer the most possible hidden activities through observable sensor data. However, the inferences obtained by HMM are hidden to the knowledge experts and hard to explain when the results are unreasonable. Furthermore, the model should be totally retrained when new unseen knowledge enriches current knowledge base. Datasets with unbalanced data or unstable distribution can affect the classification results [101]. Because a classifier can be heavily biased toward the majority class, or the learned conditional dependence structures between random variables are unstable in reality.

2.4.3 CONDITIONAL RANDOM FIELD

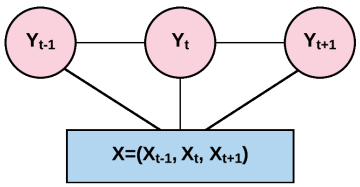


Figure 2.5: Simple CRF model [102]

Conditional random fields (CRFs) are one of the most popular discriminative probabilistic models for sequential data processing [70]. A CRF is an undirected graphical model which is used to label an observation sequence X by selecting the label sequence Y that maximizes the conditional probability $P(Y|X)$. Avoiding the limitation of HMMs, CRFs do not require the independence assumptions on the observations, thus there is no wasted effort on modeling the observations [90]. Figure 2.5 gives an example of CRFs graphical structure and Fig. 2.6 shows that how CRFs model and represent different activities [102]. The illustration shown in

Fig. 2.7 depicts another example of how a CRF is applied for activity recognition. Activities are represented as hidden states and the sensor readings correspond to the observations [70].

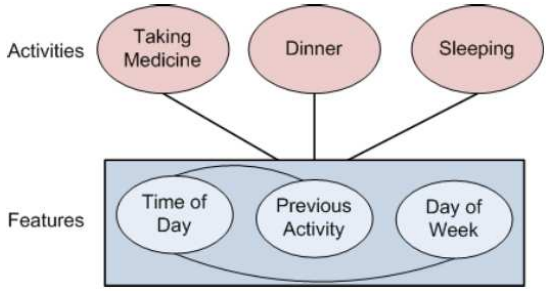


Figure 2.6: CRF model representing different activities [102]

In the work of Nazerfad et al. [102], CRFs are successfully used for activity recognition. They tested their model on the CASAS Cairo-14 dataset, which includes the activities performed by two residents and a pet, and gave an average accuracy as high as 91% for all activities. The comparison between CRFs and HMMs has demonstrated that the former has better performance for some specific activities. CRFs have also been applied to multiple-resident AR problem in smart homes cooperating with decomposition inference [103]. They achieved an average accuracy as high as 58.41% on the CASAS Kyoto-4 multi-resident dataset. The multiple-resident problem is decomposed into sub-problems using single-resident models. Under the assumption about the negligible influence of interactions between residents, single-resident models are used to infer the activities of each person by single-resident activity sequences. Yin et al. [104] developed a novel spatio-temporal event detection algorithm in large-scale sensor networks based on a dynamic CRF model. They tested their method on their own datasets containing both real and synthetic data. The performance is higher than other three baselines (precision 88.2%, recall 93.8% and F1-score 87.6%).

Although CRFs are flexible enough in terms of feature selection, the most evident disadvantage is the high computational complexity in the training stage. This fact makes them more difficult to retrain the models when new training data samples become available. Furthermore,

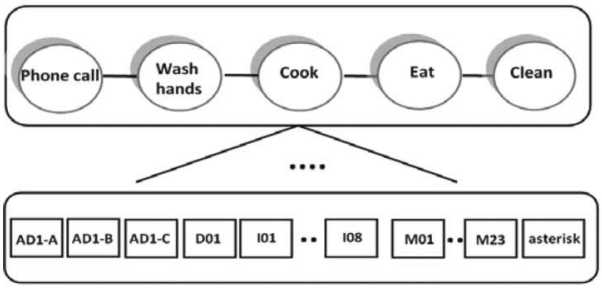


Figure 2.7: Representation of a global linear-chain CRF [70]

CRFs can not work with unknown observations, which means that it is difficult to apply them to the anomaly detection.

2.5 NON-GRAPHICAL MODELS

For non-graphic models, their visualization is no longer based on the graph structure. Their decisions are usually based on statistical correlations, data similarity, and dissimilarity. However, the state changes and transitions in variables are difficult to describe with these models.

2.5.1 DECISION TREES

A decision tree is a flowchart-like tree structure, where each internal non-leaf node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node holds a class label [56]. Unlike most other techniques, decision trees often generate understandable rules [105]. As an expanded work on concept learning systems, Quinlan firstly developed a well-known decision tree algorithm named *ID3* [106]. He later presented an improved version of ID3, known as *C4.5* [107, 108], to handle both continuous, discrete and missing-values attributes. Another famous variant about decision tree is called *CART* that describes the generation of binary decision trees.

The performance of decision tree-based activity recognition models was experimentally mea-

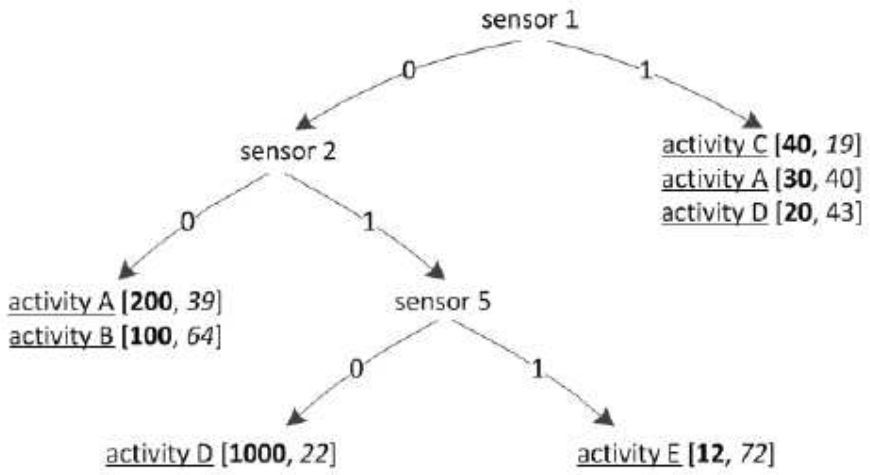


Figure 2.8: Decision tree used to recognize activities by sensor data [109]

sured by Ravi et al. [110]. Maurer et al. [111] have employed decision trees to learn the logical description of the activities. In the work of Prossegger and Bouchachia [109], as shown in Fig. 2.8, an incremental decision tree algorithm was proposed to model activities in a multi-resident context. Leaf nodes were augmented and allowed to be multi-labeled. Another application was demonstrated by Fan et al. [112]. Various behavioral features are extracted and later modeled by the ID3 decision tree.

Compared with learning-based approaches, rule-based decision trees are more readable and comprehensive. Bao et al. [113] tested various machine learning approaches to recognize activities from user-annotated acceleration data, and concluded that C4.5 decision tree received the highest recognition accuracy. Chen et al. [114] proposed a heterogeneous feature selection approach using J48 decision tree to create a classification model.

However, it is difficult for rule-based decision trees to achieve real-time classification due to incomplete information. Moreover, most of them do not have the capacity to consider sequential constraints.

2.5.2 ASSOCIATION RULE LEARNING

Association rule learning is a rule-based classification. Rules are represented in the "condition-conclusion" logic form. The learning is about finding interesting rules between features in relational data. The discovery process is generally related to the occurrences of particular features appearing in the dataset. In order to select interesting rules, two important indications called *support* and *confidence* are used to measure the degrees about significances and interests. The most common approaches for mining frequent patterns are *FP-growth* and *FP-tree* approaches [56].

Association rule-generation is a two-phase process. The first phase determines all the frequent patterns at a given minimum support level. Frequent patterns satisfy a downward closure property, according to which every subset of a frequent pattern is also frequent. The second phase extracts all the rules from these patterns [115]. The discovery of association can help in many decision-making processes such as expert systems in various domains. In the AmI issues, activities are composed of essential constituent actions or sensor events, thus, these essential data are definitely in the frequent patterns while applying association rule learning techniques and guide the recognition process.

Chikhaoui et al. [116] introduced an activity recognition method based on the frequent pattern mining technique. A mapping function calculating the matching degrees between training behavioral patterns and test data was proposed to recognize activities. In this research, activities were decomposed into tasks and subtasks. In another research, Rashidi et al. [117] discovered frequent patterns and their variations from event sequences. Considering the discontinuous property and the varied orders of behavioral patterns, the Levenshtein distance [118] was used to define a similarity measure between the already-discovered frequent patterns and a new one extended by prefix and suffix.

However, the biggest problem of association rule learning is that its concept ignores the context of data, which means that the sequential order and the temporal factor are not considered in this approach.

2.5.3 ENSEMBLE METHODS

The appearance of ensemble methods is to improve the accuracy of the classification task. Ensemble methods combine multiple learned classifiers for creating an improved composite classification model [56]. In contrast to ordinary learning approaches constructing one learner from training data, they try to construct and combine a set of learners [88]. As shown in Fig. 2.9, ensemble methods generate a group of classifiers $M_1, ..., M_k$. Given a new tuple to classify, each classifier votes for the class label of that tuple. The ensemble combines those votes to return the best class prediction [56].

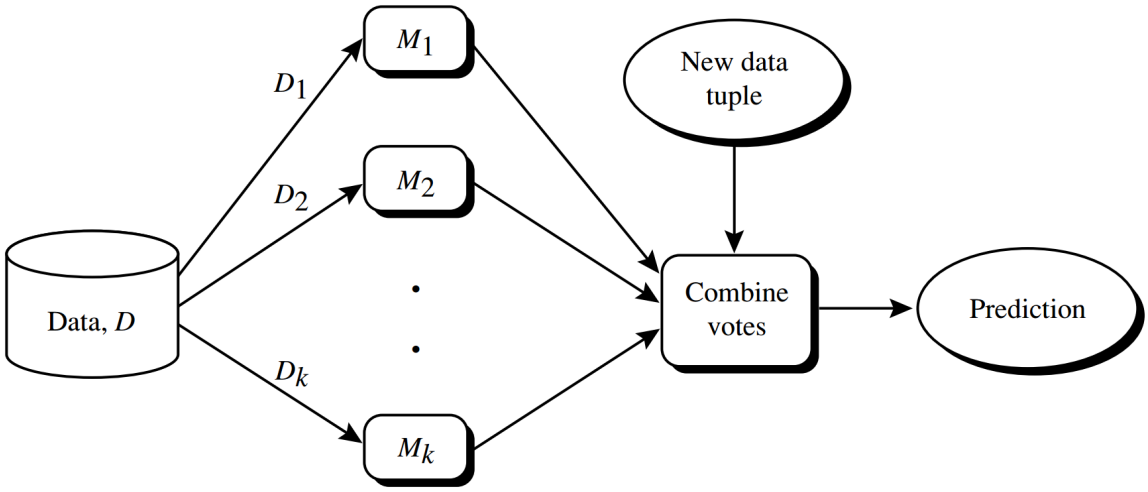


Figure 2.9: Ensemble methods generate multiple classifiers $M_1, ..., M_k$ for voting [56]

Jurek et al. [119] explored a cluster-based ensemble method, which models activities as collections of clusters built on different subsets of features. A classification process is performed by assigning new data with numeric and binary values to its closest cluster from each collection. The final prediction is made based on the class labels of the selected clusters. Another

study [120] designed an ensemble learning algorithm integrating several independent random forest classifiers based on different sensor feature sets to build a more stable, more accurate and faster classifier for human activity recognition.

Krawczyk's comparative study [121] used a weighted Naive Bayes classifier and a weighted combination to form a committee of simpler and diverse learners. In another investigation [122], a template-based multiple classifiers fusion using *k*-NN was proposed to enhance recognition rate through the ensemble framework. Generally, the performance of the ensemble classifier is better than those single classifiers [119, 120], however, ensemble methods are usually computationally expensive.

2.5.4 K-MEANS CLUSTERING

K-means clustering is a widely used unsupervised learning algorithm. It attempts to generate *k* clusters in a dataset, where *k* is a hyperparameter determined by data scientists [123]. Same to other clustering algorithms, its objective aims for high intra-cluster similarity and low inter-cluster similarity [56].

Suppose objects in a dataset D are partitioned into the k clusters $C = \{C_1, ..., C_k\}$. The center of all the objects that make up a cluster is called the centroid of the cluster, represented as μ_i. It can be defined as the mean of the objects assigned to the cluster, see Equation 2.5.

$$\mu_i = \frac{1}{n_i} \sum_{x_j \in C_i} x_j \tag{2.5}$$

where $n_i = \mid C_i \mid$ is the number of objects in cluster C_i, x_j is the point in multidimensional space representing a given object.

The quality of cluster C_i can be measured by the *sum of squared error* (SSE) between all

objects in C_i and the centroid μ_i [56, 124], defined as Equation 2.6:

$$SSE(C) = \sum_{i=1}^{k} \sum_{x_j \in C_i} dist(x_j, \mu_i)^2 \tag{2.6}$$

where $dist(x_j, \mu_i)$ is the Euclidean distance between the point x_j and the centroid μ_i in cluster C_i. The goal is to find the clustering C^* that minimizes the *SSE* score:

$$C^* = \arg\min_{C} \{SSE(C)\} \tag{2.7}$$

However, finding the optimal clustering is NP-hard in general Euclidean space even for $k = 2$. To overcome the prohibitive computational cost, k-means partitioning algorithms using greedy iterative approaches are often used in practice [56, 124].

Additionally, k-means algorithm is usually applied for preprocessing or subtasks of AmI problems, such as data labeling [125], data annotation [126] or clustering deviations [127]. On one hand, the predefined hyperparameter k is difficult to be precisely determined for these problems. On the other hand, some data may belong to multiple clusters at the same time, thus, the k-means algorithm cannot well distinguish similar activities and just cluster them as deviations.

2.5.5 K-NEAREST NEIGHBORS

K-nearest neighbors (KNN) is a classical supervised learning algorithm, it is also an example of instance-based learning that all learning is essentially based on instances [73]. KNN is a lazy learner that simply stores each given training instance and waits until an observation to classify is available.

Given a positive k and an observation x_0, the KNN classifier first identifies the closest k training instances to x_0, represented by N_0. A commonly used distance metric is Euclidean distance. KNN estimates the conditional probability for class label $y = j$ by the following equation [128]:

$$P(y = j \mid x_0) = \frac{1}{K} \sum_{i \in N_0} I(y_i = j) \tag{2.8}$$

where $I(y_i = j)$ is an indicator variable that equals one if the class label y_i equals j and zero if $y_i \neq j$.

A comparative study [119] demonstrated that KNN is an efficient and effective algorithm with excellent results, but not robust to imbalanced datasets or noisy data [129]. However, as an extension [130] or the basic classifier of ensemble methods [122], KNN can improve the performance of classification.

2.5.6 SUPPORT VECTOR MACHINE

Support vector machine (SVM) is a classic method for the classification of both linear and nonlinear data. It uses a nonlinear mapping to transform training data into a higher dimension [56]. Thus, all the transformed data in a sufficiently high dimension is separable by a linear optimal hyperplane. SVM finds this hyperplane using support vectors and the maximum margin [56]. Geometrically, the margin is defined by the support vectors and corresponds to the shortest distance between the closest data points to a point on the hyperplane [131]. The SVM solution with the maximum margin hyperplane offers the best generalization ability. SVMs can be used for numeric prediction as well as classification [56]. An SVM classifier

attempts to maximize Equation 2.9:

$$L_p = \frac{1}{2}\|\vec{w}\| - \sum_{i=1}^{t} \alpha_i y_i (\vec{w} \cdot \vec{x}_i + b) + \sum_{i=1}^{t} \alpha_i \tag{2.9}$$

where t is the number of training examples, and α_i are non-negative numbers and the Lagrange multipliers, L_p is called the Lagrangian. $\vec{x}_i$ are training vectors with associated class labels $y_i \in \{+1, -1\}$. The hyperplane is defined by the vectors $\vec{w}$ and constant b [56, 131].

SVMs have been proved that they are much less prone to overfitting than other methods [56]. Some AmI applications using SVMs were described in [95, 132, 133]. SVMs are usually used for binomial classification, However, activity recognition is a multi-class classification problem. Thus, the multi-class classification has to be transformed into a set of binomial classifications. Alternatively, extended multi-class SVMs are proposed by [134, 135]. SVMs are also integrated with other methods [136, 137]. Although SVMs are highly accurate, their training time can be extremely slow.

As a short summary, for the AmI problems, graphical models have natural advantages in the aspect of representing dynamic changes of variable states. However, most probabilistic inferences are sensitive about the datasets with imbalanced or unstable distributions. For non-graphical models, their decisions are usually based on statistical correlations, data similarity, and dissimilarity. Their performances are limited by data with high similarity and complex scenarios with concurrent, parallel or cooperative activities. Moreover, they can not construct a unified framework that is suitable to solve various AmI problems. For this reason, we propose an inference engine based on formal concept analysis (FCA) theory in the following chapter that constructs a graphical knowledge-based model. It combines the advantages of both graphical and non-graphical algorithms. Its independent design about knowledge representation and inference can separate the inference logic and knowledge modeling. Thus, each

solution of AmI problem can be abstracted as an independent module and all such modules can be grouped as a unified inference engine.

The algorithms discussed in this chapter are only part of methods used for activity recognition. More specific methods for particular AmI problems will be given in the following corresponding chapters. Activity recognition and related AmI issues are dynamic problems that describe behavioral or environmental changes due to human activities. For this reason, dynamic graphical models can better describe such state transitions or changes than the other ones. However, traditional probabilistic models rely on reliable transition probabilities and emission matrices which depend on a large amount of training data having stable probability distributions. For knowledge-driven models, the domain knowledge is hard to be defined automatically and has to be personalized with significant artificial costs. As a consequence, the proposed FCA-based model can automatically mine inference rules from data without human expert interventions. As one of the homogeneous graphical models, it is possible to combine the FCA-based model with the others to improve the performance.

CHAPTER 3

FORMAL CONCEPT ANALYSIS AND ACTIVITY RECOGNITION OF BASIC HUMAN ACTIVITIES

Formal concept analysis (FCA) is a mathematical theory based on conceptual hierarchies [138, 139]. It is an efficient solution for discovering, expressing and organizing knowledge from a large number of unstructured data [140, 141]. FCA activates the mathematical thinking of conceptual data analysis and knowledge discovery, especially the extraction of potentially interesting regularities from the initial data [142]. With its help, the heterogeneous correlations existing between two sets, the target classes of interest and the observed data, can be unified as homogeneous binary relations. FCA was first introduced in the early 1980s by Rudolf Wille [143], and now it is widely used in various domains such as knowledge discovery [142, 144], ontology engineering [145, 146], information retrieval [147], recommendation system [148, 149], semantic annotation [139] and data visualization [150, 151, 152], etc. It provides an efficient way to store, retrieve, and organize information.

FCA is an inductive learning method that summarizes regularities and rules from concrete examples without giving any preamble to guide how to generate them. Unlike black-box models, its learning process is more transparent. In the training phase, after extracting features from specific examples, FCA firstly clusters similar target classes sharing the same

ontological features, then encapsulates them as inferences, and finally orders them for quick retrieval. The above processes can be achieved by any FCA lattice construction algorithm [153, 154]. In the recognition phase, we can regard the recognition process as a continuous search of the patterns adapted to the hypothesis in the training space. In the case of sensor-based activity recognition, the target classes of interest are the activities to recognize, and the ontological features are the captured sensor data. The set of inferences based on the observed data is a set of precomputed rules. Considering continuous observations at different stages, an FCA-based model can infer the possible activities incrementally.

In order to achieve these goals, we extend the static formal concept analysis and introduce an innovative pattern recognition system, which can be used to search for specific patterns inside a constructed lattice knowledge base, so as to recognize and predict current activities quickly and accurately. We introduce a generative model, which yields inferences on the basis of partially observed data. The recognition process begins with self-inference, without any supervision or intervention from domain experts. Our goal is to design recognizable and predictive models that are as accurate as the top-level activity recognition algorithms, but are highly interpretable and convincing.

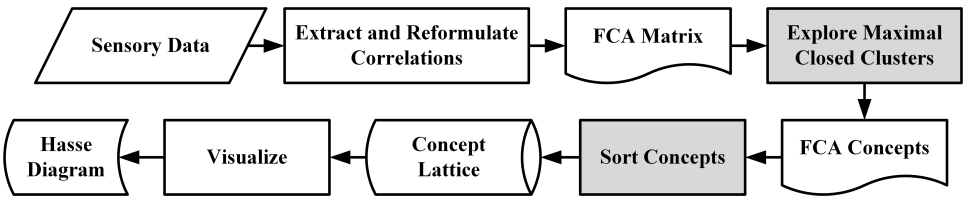


Figure 3.1: Overview procedure of FCA learning

As with other data mining methods, the process of obtaining formal lattice from raw data is called “learning” or “training”. Figure. 3.1 depicts the overview procedure of FCA learning. It represents how to construct the Hasse diagram, a visual knowledge base, from sequential sensor data. First of all, the binary relations between activities and sensor data are extracted

from the captured data stream. The extracted relations are represented by an FCA binary matrix. It is usually implemented by an ad-hoc script, according to the original format of captured sequences. Then, any lattice construction algorithm introduced in Section 3.3 will explore all the maximal clusters through FCA matrix, and sort them by their partial orders. They are the key processes in the FCA modeling and are highlighted in gray in the figure.

The remaining part of the chapter is structured as follows: Section 3.1 presents the relations between FCA and the other data mining theories. Section 3.2 introduces the components of FCA, and how each component coordinates with the others. Then, various algorithms for lattice construction are outlined in Section 3.3. After that, how to infer human activities by using FCA is introduced in Section 3.4. An innovative lattice search algorithm is proposed. It is also the core algorithm of the FCA-based model to retrieve appropriate inferences according to a series of continuous observations. An ontological clustering method is also proposed to further cluster FCA concepts in order to improve predictive accuracies. Finally, in Section 3.5, a candidate assessment is proposed to measure the pertinence of each inference in order to refine results. The primary results recognizing basic activities and relative discussions are introduced in Section 3.6 and Section 3.7. It is worth mentioning that these works of this chapter were published in [57, 61].

3.1 RELATIONS WITH OTHER THEORIES OF DATA MINING

As an independent mathematical theory, FCA is different from traditional data mining methods, but they are closely related. It is more like the fusion of these methods. In the following subsections, we compare it to these methods in order to clarify their similarity and use classical data mining terminology to explain it.

3.1.1 RELATIONS WITH ASSOCIATION RULE LEARNING

As described in Section 2.5.2, association rule learning is a rule-based data mining approach for discovering interesting relations between variables in large databases. Association rules are usually required to satisfy user-specified minimum support and confidence at the same time. Although discovering frequent itemsets is a prerequisite to generate association rules, finding all of them in a large database is also computationally expensive. Similar to some well-known frequent itemsets mining algorithms such as Apriori [155] and FP-Growth [156], FCA can help to explore frequent itemsets by a predefined threshold in order to provide intermediate data for the rule generation [157].

Additionally, FCA can generate similar "condition-conclusion" pairs instead of association rules to infer activities. We call these rough pairs as inferences, which are encapsulated in a data structure called formal concepts. More information about using these inferences to recognize activities are discussed in Sections 3.4.1 and 3.4.2.

3.1.2 RELATIONS WITH CASE-BASED REASONING

Case-based classification, or case-based reasoning, is an approach of summarizing and reusing old similar experiences to understand and solve new situations [75]. It is also a unified approach of knowledge representation, classification, and learning. It usually integrates cases as distributed subunits within an indexable knowledge structure to match similar cases later.

A typical case-based reasoning is normally a four-step process [158]. The first step named *retrieve* retrieves relevant solutions from memory cases to solve a given problem. The second step named *reuse* maps the solution from previous cases. The third step named *revise* tests the found solution in the real world, and revises again if necessary. The final step named

retain keeps the past experience as a new case in memory for the future retrieval.

Our FCA-based method is such a case-based reasoning that organizes the past patterns for identifying current ones. Compared to the hierarchical indexing structures used in the case-based reasoning, we adopt a more readable and comprehensive lattice structure to manage and infer knowledge in real-time.

3.1.3 RELATIONS WITH ONTOLOGY

Ontologies are formal definitions of types, properties, and interrelationships between existing entities in a particular domain of interest. Its objective is to build a shared understanding that enables people, organizations and software systems to communicate well with each other [159]. Shared understanding represents detailed descriptions such as individuals, classes, attributes, relations, restrictions, rules, axioms and events about a common set of scenarios in a domain. However, defining important concepts and terms within a domain is guided by enough brainstorming, collaborations and domain expertise [159].

Due to the complexity and heavy workload of building ontologies, much research focuses on the ontology engineering, which investigates the methods and methodologies for building and managing ontologies by tools and formal languages [160]. The purpose of both ontologies and FCA is to model concepts by evaluating the similarities among individuals. Therefore, some research has also applied the FCA theory to build domain ontologies from data [145, 161].

3.1.4 RELATIONS WITH DATA CLUSTERING

Data clustering is the process of grouping a set of data objects into multiple subsets called clusters. Without specific labeled information, clustering can be considered a concise model.

The basic problem of data clustering is broadly defined as follows: given a set of data points, divide them into groups as similar as possible [74]. Grouped objects within a cluster are similar to each other, yet very dissimilar to objects in other clusters. Dissimilarities and similarities are often assessed by distance or density measures [156].

The FCA theory can be regarded as a special clustering approach that data items are grouped according to their similarity in ontology. On the one hand, similarity metrics are essential for data clustering [74]. On the other hand, the FCA theory does not clearly define its own metrics. However, from their similarities, we are still trying to establish the relationship with the conventional clustering approaches, which puts a theoretical foundation for our innovative research. Alternatively, data clustering serves as a preprocessing step for other algorithms, such as classification. This is because a cluster of data objects can be treated as an implicit class [156].

FCA is a mixture of learning by observations and by examples. First, the process of constructing an FCA-based model is done in an unsupervised way, because the label information of each formal concept does not exist. Second, the internal objects of each automatic clustered formal concept are treated as the label information about patterns to infer ongoing activities. From this point of view, FCA is also supervised that learns inferences from labeled examples.

The FCA theory is closely related to two clustering methods: hierarchical clustering and conceptual clustering. They all build a hierarchy of clusters, which may be browsed for taxonomy, semantic insights and visualization [124, 162].

In Table. 3.1, the synonyms about different theories discussed in this subsection are summarized. It is to make the FCA theory more understandable.

Table 3.1: Synonyms about Different Theories of Data Mining.

Association Rules	Ontology [163]	Clustering	FCA	AR Scenario	Real World
rules	classes	clusters	formal concepts	inferences/groups	semantic definitions
individuals	instances	data objects	objects	activities (labels)	entities of interest
attributes	properties	features	attributes	sensor/behavioral features	descriptors and properties of different natures
conditions	rules	-	intent	partial observed data	requisite states of affairs
conclusion	consequents	-	extent	possible activities	consequence of proposition

3.2 COMPONENTS OF FORMAL CONCEPT ANALYSIS

In order to construct an efficient knowledge base from input data, the internal FCA components cooperate with each other. In this section, we introduce the key FCA components and their roles in knowledge base construction and knowledge inference.

As shown in Fig. 3.1, from the raw input data to the final knowledge base, there are three intermediate results: formal context, formal concepts and concept lattice, and three processes: reformulation, clustering, and sorting. Firstly, raw input data is represented and reformulated into a structured form called the formal context, which is a data structure that reorganizes sequential and temporal data to a machine-readable format. Secondly, formal concepts are explored from the formal context through a pair of concept-forming operations. Thirdly, these formal concepts can be sorted and linked with each other according to the partial order in mathematics. The sorted set of formal concepts is called concept lattice, which is also a graphical knowledge base.

To illustrate the relationship between the FCA components and the AmI problems, we make the following assumptions: behavioral patterns are sequences of sensor data captured in some time intervals, and captured sensor data are ordered by their timestamps.

3.2.1 FEATURE EXTRACTION BY FORMAL CONTEXT

To analyze temporal and sequential behavioral patterns, first of all, correlations between target classes and features are extracted from data and reformulated into a specific data structure named **formal context**. Formal context $\mathbb{K}(G,M,I)$ is triplet consisting of two disjoint sets, G and M, and their Cartesian product set $I = G \times M$. It can be represented and visualized by a $|G| \times |M|$ matrix. The elements in set G are formally called *objects*, which represent coarse-grained target classes of interest (i.e. activities to recognize). The ones in set M are called *attributes*, which represent fine-grained observable features (i.e. captured sensor data). If $g \in G$ is correlated with $m \in M$, the correlation can be written as gIm [138].

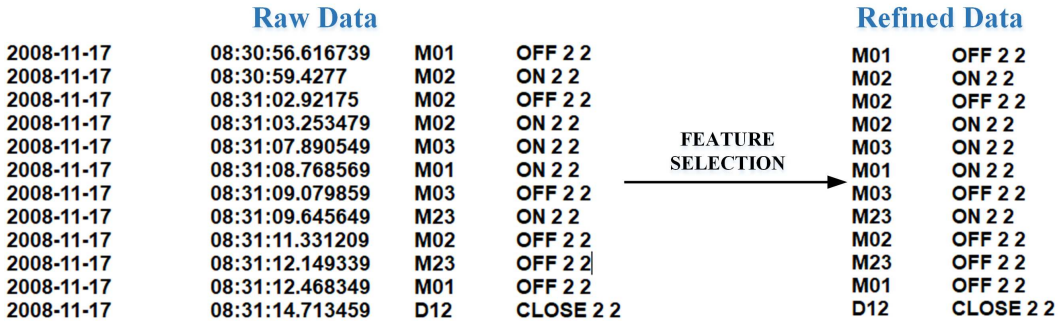


Figure 3.2: Feature extraction

Because of the limitation of the triplet structure of formal context, first of all, the most representative features should be selected from the input data. Normally, the captured data in a smart environment for supervised learning usually has several essential data fields: timestamps, sensor ID, sensor value and a label indicating the ground truth. As shown in Fig. 3.2, we refine the input data and only keep the fields of sensor IDs, sensor values and labels.

To extract and reformulate correlations from sensor data, if the sensor data m_j appears in a pattern describing an activity g_i, it means $g_i I m_j$, then a cross will be filled in the row g_i and column m_j in the binary matrix. Fig. 3.3 shows a concrete example which is generated

Simplified CASAS Activities [164]		m_1: D07	m_2: D11	m_3: D12	m_4: D13	m_5: D14	m_6: I04	m_7: M04	m_8: M06	m_9: M07	m_{10}: M09	m_{11}: M13	m_{12}: M17	m_{13}: M23
Fill medication dispenser	g_1	×					×						×	
Hang up clothes	g_2			×						×	×			×
Move furniture	g_3							×						×
Read magazine	g_4								×	×	×			
Water plants	g_5								×	×				
Sweep floor	g_6			×					×	×	×	×	×	×
Play checkers	g_7		×										×	
Prepare dinner	g_8										×	×		
Set table	g_9			×							×			×
Read magazine	g_{10}	×											×	
Pay bills	g_{11}					×							×	
Pack picnic food	g_{12}								×	×				
Retrieve dishes	g_{13}				×				×	×	×	×		
Retrieve dishes	$g_{13'}$								×	×	×	×		
Pack picnic supplies	g_{14}	×									×		×	
Pack and bring supplies	g_{15}			×		×				×	×		×	×

Figure 3.3: Binary matrix representing the relations between activities g_i and sensor events m_j.

from a simplified version of the CASAS benchmark dataset [164]. In this simplified example, fifteen activities are described by thirteen non-intrusive sensors passively capturing human behaviors in a smart apartment. It is worth mentioning that g_{13} and $g_{13'}$ are two different behavioral patterns implementing the same activity "Retrieve dishes".

Pruning Since our predictive model is entirely learned from pervasive sensors, in order to enhance the generalization capability and improve modeling efficiency, in the feature selection phase, we propose two optional pruning processes to filter the useless attributes from a formal context. The first pruning is global. The attributes that have extremely high or low occurrences should be removed from the context to avoid overfitting. This is because the attributes with extremely high occurrences among activities have very limited ability to differentiate different activities. Similarly, the ones with extremely low occurrences are usually identified as noisy or meaningless data. This is because their ability to distinguish between different activities may be related to their occurrences, not to the semantic correlations be-

tween activities.

The second pruning is local. Training data can be first grouped by homogeneous activities according to their labels, and then a pruning operation is used to filter redundant correlations (i.e. crosses in the matrix) with extremely low occurrences in a group. In our previous research [62], attributes were divided into two categories: essential and optional. Essential attributes mean that they are indispensable for an activity, in other words, they appear in all the patterns describing the same activity. Optional attributes usually represent personal preferences, and they do not appear in each pattern. Therefore, in a group that contains all the patterns describing the same activity, the correlations with low occurrences are considered to be noisy data.

3.2.2 SIMILARITY MAXIMIZATION BY CONCEPT-FORMING OPERATIONS

In data mining, especially in data clustering technique, similarity metrics are essential to generate clusters [74]. To exploit useful information from an FCA matrix and cluster similar target classes sharing the same feature variables, the FCA theory defines its own metrics to maximize similarity. Items in the same cluster have high similarity because they share some of the same ontological features.

In the FCA theory, the similarity is measured by a pair of metrics, so-called the *concept-forming operators*.

For a subset $G_1 \subseteq G$, we define

$$G_1' := \{m \in M \mid \text{for all } g \in G_1,\ gIm\} \tag{3.1}$$

as a closure operation to find out the common features $G_1' \subseteq M$ shared by all the objects in

G_1. Conversely, for $M_1 \subseteq M$, we define

$$M_1' := \{g \in G \mid \text{for all } m \in M_1,\ gIm\} \tag{3.2}$$

as another operation to find out all the objects $M_1' \subseteq G$ sharing the common features M_1 [138].

Using the two operators at the same time, FCA can generate stable closures, named class-feature pairs, to cluster correlated classes and features for maximizing their dependency and the similarity. For AmI problems, the operator 3.1 can find out the common sensor data shared by a set of activities, and the operator 3.2 can reveal which activities have the given set of observations (sensor data).

For instance, as shown in Fig. 3.3, if $\{m_3 m_{10}\}$ are observed, according to $\{m_3 m_{10}\}' = \{g_2 g_6 g_9 g_{15}\}$, the most possible ongoing activities are g_2, g_6, g_9 and g_{15}. However, such a class-feature pair is not stable due to $\{g_2 g_6 g_9 g_{15}\}' = \{m_3 m_{10} m_{13}\}$. The stable one $\{m_3 m_{10}$ $m_{13}\}' = \{g_2 g_6 g_9 g_{15}\}$ is called formal concept.

3.2.3 CLUSTER REPRESENTATION BY FORMAL CONCEPT

Given the training data, FCA partitions behavioral patterns into distinctive groups based on the different features shared among those patterns. Similar to data clustering, the different features used to partition patterns are called centroids. Therefore, patterns in the same group share similar behavioral characteristics.

Let us come back to our activity recognition scenario. In order to infer ongoing activities from given observable sensor data, FCA first clusters similar patterns according to different centroids, and encapsulate these class-feature pairs in itemsets. Moreover, to ensure the reliability of inferences, FCA only uses the itemsets that simultaneously satisfy the two concept-

forming operations. The satisfied itemsets are so-called formal concepts.

Formal concept $c := (G_1, M_1)$ is a closure itemset under the limitation of the concept-forming operations, where $(G_1')' = (M_1)' = G_1$. G_1 is called the *extent* of c, written as ext(c). Likewise, M_1 is called the *intent* of c, written as int(c) [138], which is also treated as the centroid of a cluster [74, 81]. The space of all the formal concepts is denoted by $\mathfrak{B}(G, M, I)$. The process that enumerates $\mathfrak{B}(G, M, I)$ is done by lattice construction algorithms (see Section 3.3).

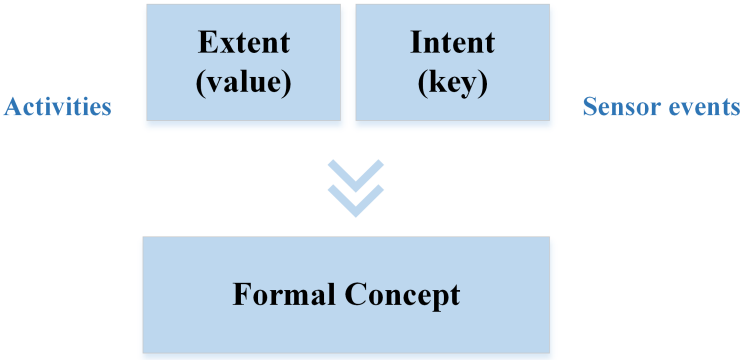


Figure 3.4: Key-value structure of formal concept

As shown in Fig. 3.4, each formal concept has a key-value structure that consists of two parts. The extent is the value part that indicates the labels of patterns, also used as inferred results in the inference process. And the intent is the key part that represents common features, also indicates the observed data in the inference process. A concept c clusters similar patterns ext(c) based on their common features described in the int(c). Furthermore, if $\alpha \subset$ int(c) is an observed sequence, the elements in the ext(c) indicate inferred activities given the observed data α.

$$\{\ \underbrace{g_2 g_6 g_9 g_{15}}_{\text{possible ongoing activities}}\ ,\ \overbrace{m_3 m_{10} m_{13}}^{\text{current observed data}}\ \}$$

Consider the above example, ext$(c) = \{g_2 g_6 g_9 g_{15}\}$ and int$(c) = \{m_3 m_{10} m_{13}\}$. As described in Section 3.2.2, the sensor events in int(c) exist in all the patterns of activities in ext(c). Therefore, if current observed data are m_3, m_{10} and m_{13}, the scope of possible ongoing activities should be g_2, g_6, g_9 or g_{15}. Therefore, based on such key-value tuple structure of itemsets, FCA-based models can infer the ongoing activities of residents according to partially observed sensor data.

3.2.4 CLUSTER INDEXING BY FORMAL CONCEPT LATTICE

After the generation of concepts clustering similar patterns by different centroids (i.e. feature variables), lattice construction algorithms automatically index all the discovered concepts according to a mathematical order called the *partial order* [138]. The objective is to efficiently manage and construct a graphical knowledge base to quickly retrieve inferences.

Formal concept lattice $\underline{\mathfrak{B}}$ is an ordered version of $\mathfrak{B}(G,M,I)$. All the concepts in $\mathfrak{B}(G,M,I)$ are ordered by a predefined partial order $\preceq$ indicating hierarchical relations between two concepts [138].

Suppose (G_1,M_1) and (G_2,M_2) are two concepts, (G_1,M_1) is called the *subconcept* of (G_2,M_2) if either $G_1 \subseteq G_2$ or $M_2 \subseteq M_1$, written as $(G_1,M_1) \preceq (G_2,M_2)$. The symbol $\preceq$ is named as the *hierarchical order*. Meanwhile, (G_2,M_2) is the *superconcept* of (G_1,M_1). It is worth pointing out that the subconcept and the superconcept of a concept are not unique in $\mathfrak{B}(G,M,I)$ due to the existing transitive relation.

For instance, three concepts, $\{g_6 g_8 g_{13} g_{13'}, m_{10} m_{11}\}$, $\{g_6 g_{13} g_{13'}, m_8 m_9 m_{10} m_{11}\}$ and $\{g_{13}, m_4, m_8 m_9 m_{10} m_{11}\}$, are discovered from the matrix in Fig. 3.3. As shown in Equation (3.3), the

last two concepts are the superconcepts of the first one.

$$\begin{aligned}\{g_6 g_8 g_{13} g_{13'}, m_{10} m_{11}\} &\preceq \{g_6 g_{13} g_{13'}, m_8 m_9 m_{10} m_{11}\} \\ &\preceq \{g_{13}, m_4 m_8 m_9 m_{10} m_{11}\}\end{aligned} \tag{3.3}$$

The relations among concepts having different centroids are established and linked by the hierarchical order. Thus, a lattice $\underline{\mathfrak{B}}$ can be visualized as a graphical model.

3.2.5 KNOWLEDGE VISUALIZATION BY HASSE DIAGRAM

In mathematics, a finite partially ordered set can be depicted by a Hasse diagram. In our case, a formal lattice $\underline{\mathfrak{B}}$ can also be visualized as an undirected graph, such as the one shown in Fig. 3.5. Each node refers to a discovered concept, and partial orders are represented by edges, which are also named Galois connections [138].

As can be seen from Fig. 3.5, concepts are organized by different levels. There are two special nodes in a Hasse diagram: the topmost one $\{G, \varnothing\}$ named *Supremum* and the lowermost one $\{\varnothing, M\}$ named *Infimum*. They separately represent the initial and the final states of the recognition process.

3.3 LATTICE CONSTRUCTION

The lattice construction plays an essential role in the FCA applications. It can quickly start from a context $\mathbb{K}(G, M, I)$ to efficiently enumerate all the concepts $\mathfrak{B}(G, M, I)$, and order them by the partial order. Compared with brute-force ways, a lattice construction algorithm can be more efficient to complete the time-consuming sorting and combination operations.

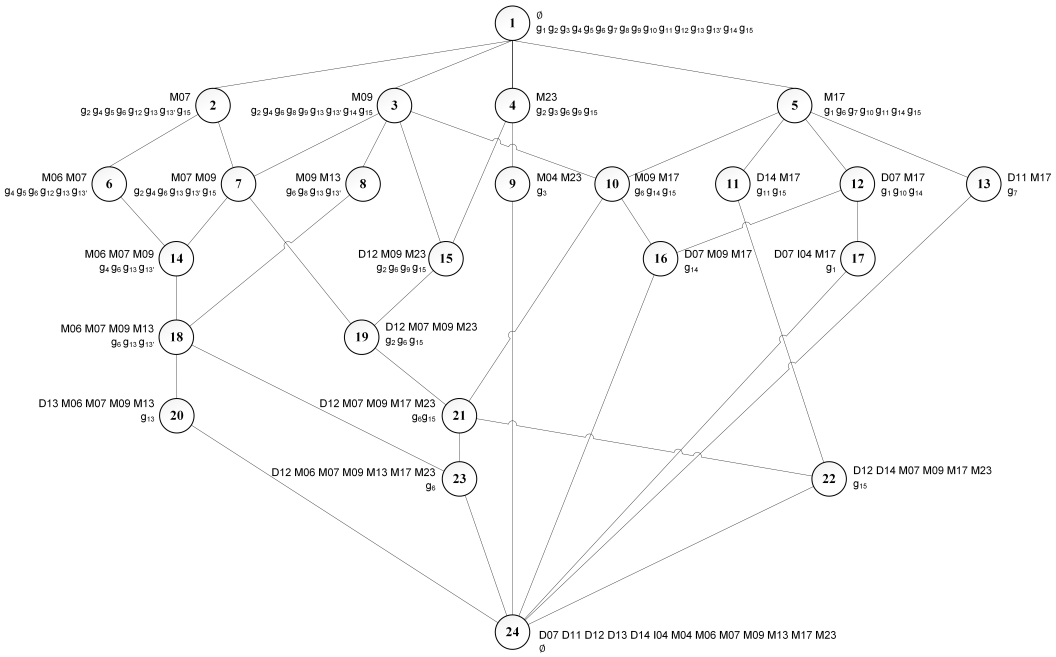


Figure 3.5: Hasse diagram of the binary matrix shown in Fig. 3.3

The time complexity also drops from $O(|\ G\ |!\ |\ M\ ||\ L\ |)$ to $O(|\ G\ |^2|\ M\ ||\ L\ |)$, where $|\ L\ |$ is the size of lattice [154]. There are two main types of algorithms: batch algorithms and incremental algorithms [154, 165]. Their difference is that the batch ones have to load and deal with the whole training data at the same time, but the incremental ones can update a lattice once new data are available. However, some incremental algorithms also sacrifice their efficiency in exchange for functional extensions.

For the batch algorithms, they can still be divided into three subtypes: descending, ascending and enumeration algorithms [166]. For the descending ones, a lattice is built from the Supremum, such as the typical Bordat algorithm [167]. On the contrary, the ascending ones build a lattice from the Infimum, such as Chein algorithm [168]. The enumeration ones enumerate all the nodes of a lattice by a certain order, such as the Ganter's algorithm [169] using lexicographical order.

For applications based on the FCA models, no matter which lattice construction algorithm is

used, there is no effect on the application itself. This is because all the construction algorithms generate the same lattice with the same structure. However, for activity recognition, the most suitable construction algorithms are the incremental ones. Because continuous new data will be captured and used to update the existing model, and the sensor layouts for smart environments can be modified if needed. All these requirements will change the structure of the current lattice. For incremental algorithms, the cost of frequent updating is much lower than the one of retraining. This is because for incremental algorithms, only a few parts of lattice may be modified, not the entire structure. However, for the other algorithms, the lattice should be reconstructed from scratch.

3.4 APPLICATIONS IN SMART ENVIRONMENTS

As shown in Fig. 3.1, in the training phase, correlations are first extracted from the sequences of captured sensor data, and then saved into an FCA matrix. In the matrix, the first column indicates "*activity with pattern id*" and the rest indicates "*correlations*" between patterns and sensor data. If a pattern contains some sensor data, we can affirm that the pattern itself has binary relations with the data. Correlations are represented as crosses in the matrix.

As a result, implicit ontological correlations are revealed by FCA. Once different patterns describing the same activity are clustered together, most of their internal attributes are aggregated by formal concepts due to their similarity in ontology. This is because an activity is usually associated with some particular locations and constant interactive items. For example, the behavioral patterns involving preparing coffee will always interact with coffee cups. Another example is that the patterns about preparing dinner always involve some fixed positions in a kitchen. Therefore, the related correlations in the FCA matrix are clustered together and generate a formal concept in the visualization.

3.4.1 STATIC INFORMATION RETRIEVAL

Once the Hasse diagram is built, the next step is to use efficient algorithms to retrieve knowledge encapsulated inside concepts from the graphical structure. The concept lattice can represent knowledge in a very simple and effective way. Through its hierarchical structure, relevant inferences are well indexed for efficient retrieval. From the top to down in a Hasse diagram, the scope of inferred results shrinks when more data are observed.

If we treat observed data as query conditions and retrieve them within all the concepts with the key-value structure, suitable inference results may be obtained from the value parts of certain concepts. However, as a static information retrieval method, it cannot guarantee that suitable results are returned each time according to the observations. If all the data observed during a period of time is used as query conditions for retrieving inferences in the lattice knowledge base, due to mixed noisy data or irrelevant one (data belonging to different activities), the returned result is likely to be a null value. For this reason, we propose another continuous retrieval algorithm to avoid null inference.

3.4.2 CONTINUOUS INFERENCES

Figure. 3.6 illustrates the principle of continuous FCA inference for activity recognition. The scope of inferred possible activities (e.x. g_i in the ext(c)) decreases when more and more sensor data (e.x. m_j in the int(c)) are observed. As shown in Fig. 3.6, possible activities are gradually refined to g_{14}, when observed data are extended from M09 to D07M09M7. Thus, the real-time activity recognition task can be transferred into a diagram search problem. Each time the model infers possible activities by locating the most relevant concept insides the Hasse diagram. To locate the most relevant one according to the observed data, we need an efficient inference retrieval algorithm. For this reason, we propose a diagram search algo-

rithm called *Half-Duplex Search (HDS) algorithm*. It can be treated as an algorithm using observed data as query conditions to search for the most optimal key-value formal concept with the best corresponding value. It is the basic algorithm used in our published research [57, 61, 62]. It consists of two parts: the top-down search described in Algorithm 1 can quickly locate an intermediate concept with the value satisfying the query conditions, and the bottom up search described in Algorithm 2 can further find the most optimal one through the intermediate concept. Each search starts from the previous position p ($p = 0$ in the initial stage of recognition) where the last inference was located.

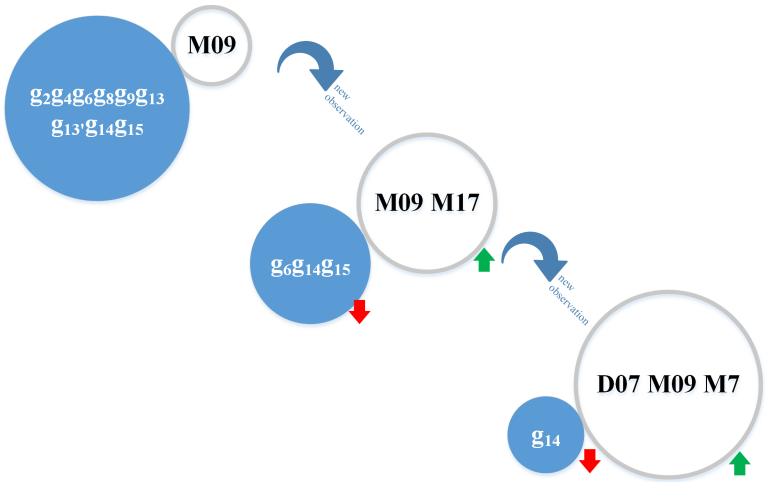


Figure 3.6: Continuous inference for activity recognition

The HDS algorithm only provides the basic function that retrieves suitable inference quickly and incrementally. For one resident performing simple activities in smart homes, we can directly use it to recognize activities without complex patterns [61]. For more complex scenarios, other auxiliary search strategies are required. Besides, the choice of these strategies is also affected by the number of residents. For example, once there are more than one resident in a smart home, they may perform parallel or cooperative activities. We propose a specific strategy to distinguish their highly similar behavioral data.

For more complex situations such as composite activities or multi-resident activities, their

Algorithm 1: Top-down search of HDS algorithm

Data: previous position *p*, sequence α.
Result: first met concept containing α.

```
1  begin
2      fifo ← node[p]
3      while fifo do
4          if fifo[0] not visited then
5              mark as visited
6              if α ⊆ fifo[0].intent then
7                  return fifo[0]
8              else
9                  add fifo[0].successors into fifo
10         remove fifo[0] from fifo
11     end
12 end
```

Algorithm 2: Bottom up search of HDS algorithm

Data: located position *p*, sequence α.
Result: topmost concept containing α.

```
1  begin
2      fifo ← node[p].predecessors
3      S ← ∅
4      while fifo do
5          if fifo[0] not visited then
6              mark as visited
7              if α ⊆ fifo[0].intent then
8                  add fifo[0].predecessors into fifo
9                  S ← S ∪ fifo[0]
10         remove fifo[0] from fifo
11     end
12     return arg min_{s∈S}(| s.intents |)
13 end
```

auxiliary search strategies pay more attention to the analysis of the behavioral characteristics shown in Section 1.5.4. In [62], we propose another solution for detecting errors. We summarize six common errors [58, 170] and their typical abnormal behavioral patterns in Section 4.

3.4.3 ONTOLOGICAL CLUSTERING

In the initial stage of activity execution, the accuracy of identification and prediction is not as accurate as in other periods due to the small amount of observational data. Moreover, some semantically similar activities with almost the same subsequences, especially those with multilevel inheritance relations, may confuse predictions at early stages.

The purpose of this subsection is to automatically create an alternative level on the basis of the multiple data granularity presented in Fig. 1.2 for integrating similar target variables of interest, reducing semantic gaps between two layers, and enhancing data interpretation. Figure 3.7 illustrates such a structure: the intermediate layer is an alternative abstraction of some clustered target variables of interest.

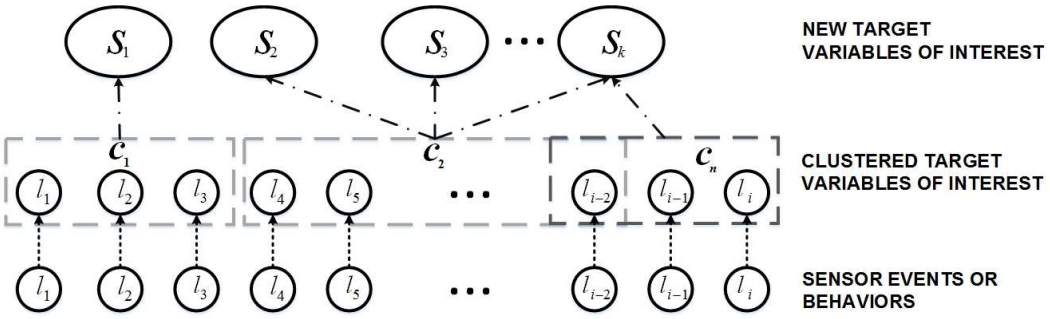


Figure 3.7: Alternative level created by ontological clustering

In Section 3.2.5, we concluded that the fewer data were observed, the more ambiguous inferred results there are. Instead of seeking precise predictions by few observed data at the

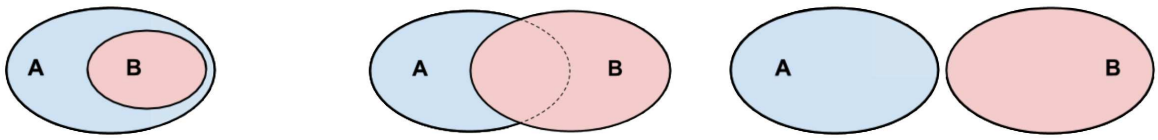
early stages, approximate predictions are more useful in our case.

For example, if there are three observed actions, BoilWater, TakeOutSpoon, and TakeOutMilk, it is difficult to precisely predict which one is being done, maybe PrepareCoffee or PrepareMilkTea. However, due to the irrelevant action BoilWater for the behavioral patterns relating to make instant oatmeal, we can at least determine that the ongoing activity is related to preparing something to drink. Therefore, the system may pay more attention to the cognitive assistance and preventive interventions for preparing something to drink, rather than the ones about preparing something to eat.

As a potential solution, our objective is to cluster target variables of interest according to their semantic similarities. Each new cluster is a more general semantic definition that can be renamed on the basis of their common semantic features. The research of Formica [161] has demonstrated that there are some shared characteristics between ontologies and FCA theories (see Table. 3.1). Consequently, we propose an ontological clustering method based on FCA to improve our predictions in the early stages.

Ontological Similarity Metric

To generate ontological clusters, first of all, we need to define a metric to evaluate semantic similarity among target variables of interest. As shown in Fig. 3.8, there are three possible semantic relations between two patterns, which are related to the number of shared features.



(a) inherited **(b) semantically similar** **(c) independent**

Figure 3.8: Semantic relations between two activities

Suppose that A and B are two patterns. The first relation is called *inherited*. It is true if and

only if a pattern is the subset of another one. In Fig. 3.8a, A contains all the features of B, referred as $B \subset A$, called A is inherited from B. Such a relation is very common in reality due to the *multilevel inheritance* caused by diverse living habits and personal preferences. For instance, PrepareCoffeeWithSugar (A_0) is inherited from PrepareBlackCoffee (A_1) because of $A_1 \subset A_0$.

The second one is called *semantically similar*. It is true if and only if two patterns have partial common parts among their features. In Fig. 3.8b, A and B have a partial intersection, referred as $A \cap B \neq \emptyset$. No matter how few the common features are, two semantically similar objects have always semantic similarity.

The third one is called *independent*, which means that two patterns are mutually independent. In Fig. 3.8c, A has no common feature shared with B, referred as $A \cap B = \emptyset$.

Because of the limitation of shared features, some newly clustered target variables of interest cannot be easily renamed, but it will not affect their generation. The construction of ontological clusters is the process enumerating those patterns mutually having inherited or semantically similar relations.

There are a wide variety of methods that can be used to address the clustering problems. The objective is to maximize the similarity of objects in a cluster and simultaneously maximize the dissimilarity among clusters. Distance-based and density-based algorithms are the two most common categories, especially the distance-based one. The former is desirable because of the simplicity and ease of implementation in a wide variety of scenarios [74]. In our case, each clustered target variable has inherited or semantically similar relations with others. Like classical distance-based clustering algorithms [171], in the final clusters, ontological clustering is also required to find out the clustroids which are the closest on average to the other patterns in their clusters. In practice, these clustroids are the commonly shared features of

those patterns. However, there are also some special differences. One of them is that patterns from different clusters are relatively dissimilar, which means there are overlaps among clusters of target variables.

Our ontological clustering further discovers the target variables of interest having inherited or semantically similar relations on the basis of the current Hasse diagram. The process of ontological clustering based on the FCA can be summarized as follows:

1. Select relevant features (attributes) and prune the noisy or irrelevant ones [74].
2. Initially define each indexed target variable of interest as an independent cluster by itself.
3. Define a metric to measure similarity.
4. According to the predefined minimal threshold of ontological similarity, repeatedly merge two nearest clusters into one (see Algorithm 3).

In our clustering algorithm, patterns g_i in a cluster $A \subset G$ share the same attributes (clustroid). In other words, all the objects sharing the same clustroid should be merged in a cluster. The cardinality of clustroid should be greater than the predefined threshold t_0 (see Equation 3.4).

$$\left| \bigcap_{i=1}^{n} g_i' \right| > t_0, \quad g_i \in A \subset G \tag{3.4}$$

where g_i' are the attributes of g_i obtained by the concept-forming operation defined in Section 3.2.2.

Furthermore, the merger based on a fixed threshold is not sufficient due to various cardinalities of clustroids in different clusters. Thus, the percentage threshold should be better to

Algorithm 3: ontological clustering algorithm

Data: start position sp, Hasse diagram $diag$, threshold t_1.
Result: topmost superconcept containing α.

```
1  begin
2      fifo ← diag[sp].successors
3      S ← ∅
4      while fifo do
5          if fifo[0] not visited then
6              mark as visited
7          if fifo[0].extent.len < fifo[0].children.extents.len then
8              cluster ← fifo[0].extent
9          similar ← True
10         foreach o in fifo[0].extent do
11             n_0 ← fifo[0].intent.len
12             N ← o′ .len
13             if n_0/N < t_1 then
14                 similar ← False
15         end
16         if similar then
17             cluster ← fifo[0].extent
18         remove fifo[0] from fifo
19         clusters.add(cluster)
20     end
21     return clusters
22 end
```

evaluate the ontological similarities in different clusters. On the basis of Equation 3.4, we propose another metric as:

$$\frac{\left|\bigcap_{i=1}^{n} g_i'\right|}{\max\left|g_i'\right|} > t_1, \quad g_i \in A \subset G \tag{3.5}$$

where the numerator is the commonly shared attributes among internal patterns, which is also the clustroid of a cluster. The denominator is the cardinality of the maximal set of observed attributes among sequences describing g_i.

In fact, Equation 3.4 is as same as the definition of the concept-forming operation 3.1. As a consequence, every concept in a Hasse diagram is an ontological cluster with a dynamic threshold.

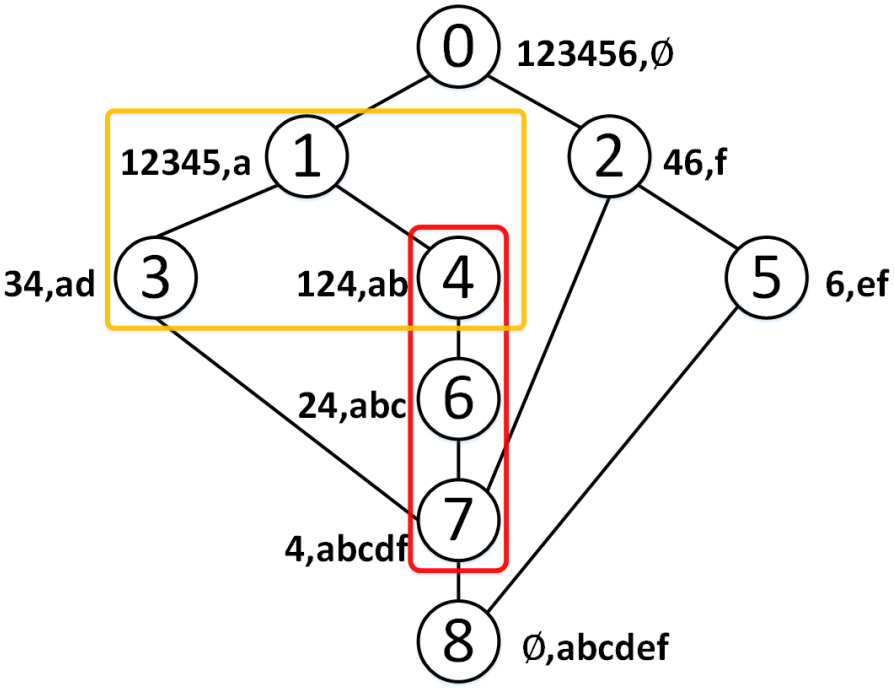


Figure 3.9: Clusters in a Hasse diagram.

If the process of ontological clustering is based on the semantic relations described in Fig. 3.8, to repeatedly merge two nearest clusters into one, there will be two mechanisms to generate clusters. The process is to traverse the whole Hasse diagram to find out all the concepts having corresponding semantic relations.

The first one is to discover inherited relations shown in Fig. 3.8a. The main characteristic is that some patterns in the extent of one concept cannot be found in the extents of its subconcepts. It refers to Line 7 to 9 and 16 to 19 in Algorithm 3.

Example: in Fig. 3.9, the red rectangle including nodes 4, 6, and 7 highlights the inherited relation. Pattern g_1 in node 4 disappears in the extents of the sub nodes 6 and 7. This is because the disappeared patterns are the superclasses having fewer attributes than the subclasses in the sub nodes.

The second one is based on the semantically similar relation in Fig. 3.8b. If one node has more than one branch, it means that the patterns in its extent are the clustroids and current concept is an ontological cluster. Nevertheless, it is necessary to use the threshold defined in Equation 3.5 to control the merging of clusters. It refers to Line 10 to 19 in Algorithm 3.

Example: in Fig. 3.9, the yellow rectangle including nodes 1, 3 and 4 highlights the semantically similar relation. Patterns in nodes 3 and 4 commonly having an attribute *a*. If the cardinality of the intent in node 1 is bigger than the predefined threshold, the following sub nodes should be merged.

With the help of ontological clustering, the prediction accuracies at the early stages will be improved. When observed data are few and limited, the inference engine will predict the ontological superclass instead of directly predicting an activity. For example, *PrepareCoffee* will be no longer directly predicted, the inference trace will be *PrepareDrinks* $\rightarrow$ *Prepare-*

BlackCoffee $\rightarrow$ *PrepareCoffeeWithoutSugar* $\rightarrow$ *PrepareCoffee*.

3.5 CANDIDATE ASSESSMENT

Because of few observed data, a concept usually has more than one element in its extent, which means that there are several candidates (possible ongoing activities) according to the observed data. Redundant candidates are ambiguous and useless to make decisions for real-time assistance. In this case, we desire to evaluate the relevance of each candidate in a concept and choose the most relevant one as the *local optimal prediction*. The relevance is defined as the similarity between existing learned patterns and the pattern to recognize.

As mentioned in the previous sections, an activity can be accomplished by alternative patterns g_i because of different personal preferences. Furthermore, these derived patterns may have flexible execution orders, repetitive or optional data. At the same time, each resident may have a relatively stable preference to execute an activity. Namely, for the same resident executing an activity, there are only a few deviations among each execution. Based on this hypothesis, we take advantage of historical patterns containing the preferences of residents to generate a knowledge database called *accumulated matrix*. For each sensor data, we calculate its expectant position appearing in each pattern to establish a series of naive distributions.

To measure the contextual similarities between historical patterns and the captured one, average deviations are calculated using Root-Mean-Square Deviation (RMSD). The RMSD serves to aggregate the magnitudes of the errors in predictions. It measures the differences between values predicted by a model and the values observed. In our case, it evaluates the differences between the predicted positions of sensor data and the observed ones. Thus, it makes a quantitative comparison to estimate how well the current behavioral pattern fits accumulated historical data. A lower RMSD score indicates that the prediction is more accurate

due to the better adaptation to the historical patterns.

We propose our assessment as follows: for each candidate in the extent, under the condition of executing g_i, we calculate the deviation between actual average positions in α and the accumulated ones in the matrix. Thus, the local optimal prediction should be the one with minimal deviation which has the best adaptation in comparison with historical data. Obviously, our assessment consists of two modules: accumulation and evaluation.

3.5.1 ACCUMULATION

For each sensor data α_j in a training item α, which is a complete sequence of sensor data describing activity g_i (i.e. $\alpha_j \in \alpha$, $\alpha \in g_i$), we update the accumulated value of corresponding element (g_i, α_j) in the accumulated matrix by Equation (3.6):

$$\sigma_{ij} = \sigma'_{ij} + j \tag{3.6}$$

where j is the position of α_j in α. σ'_{ij} is the previous accumulated value and σ_{ij} is the newly updated one. The number of accumulated values σ_{ij} is the sum of positions of sensor data α_j that appears in each pattern describing activity g_i. If a pattern is stored in an array, the position of sensor data can be defined as its index value in the array. We accumulate such a value in order to calculate the average positions and to calculate the standard deviation for the purpose of measuring the confidence of each average position. Equation (3.7) represents the same accumulation in another global view:

$$\sigma_{ij} = \sum_{k=1}^{N_{ij}} \sigma_{(ij,k)} \tag{3.7}$$

where N_{ij} represents the occurrences of sensor data (g_i, α_j) existing in the whole training dataset. $\sigma_{(ij,k)}$ is the position of α_j in the k-th training item describing activity g_i.

3.5.2 EVALUATION

When sensor data α_j was observed, first of all, we calculate its average position $\overline{\varphi_j}$ in current sequence α. It is calculated by Equation (3.8).

$$\overline{\varphi_j} = \frac{1}{\#\alpha_j} \sum_{k=1}^{|\alpha|} k[\alpha_k = \alpha_j] \tag{3.8}$$

where $|\alpha|$ is the size of current sequence α, and $\#\alpha_j$ is the occurrences of α_j in α. The condition $\alpha_k = \alpha_j$ surrounded by the Iverson bracket is to integrate all the discrete positions of α_j.

And then, for each candidate, we calculate the deviation of α given g_i. Equation (3.9) expresses the root-mean-square deviation D_i of current sequence α executing g_i:

$$D_i = \sqrt{\frac{1}{|\alpha|} \sum_{\forall \alpha_j \in \alpha} (\overline{\varphi_j} - \frac{1}{N_{ij}} \sigma_{ij})^2} \tag{3.9}$$

where σ_{ij}/N_{ij} is the expectant position obtained from accumulated matrix.

Thus, RMSD scores $\{D_1, D_2, ..., D_i\}$ of candidates in the current extent $G_1 = \{g_1, g_2, ..., g_i\}$ were calculated. The element g_i having the minimal RMSD value is the local optimal prediction because of the best adaptation to historical patterns.

3.6 PRIMARY RESULTS

In this section, we use the 10-fold cross-validation and the following criteria to evaluate the experimental results: time cost (in both training and inference phases), activity prediction, and recognition accuracies. The experiments are based on the basic dataset named RDATA, the synthetic dataset named DDATA, and the CASAS benchmark dataset introduced in Appendix A. All the evaluations were carried out on a laptop with Intel Core i7 Processor (2.4GHz) and 8GB RAM, under the Ubuntu 14.04 operating system.

3.6.1 TIME COST

The time costs for training lattices with different sizes are shown in Table 3.2. Compared to RDATA, DDATA has the same statistical information in size because the lattice construction only depends on the binary relations (i.e. lattice structure only depends on the set of constituent actions of each activity). That is also the reason why FCA-based models can well handle the patterns with flexible execution orders without additional training costs. Moreover, in the training phase, the time cost of lattice construction is proportional to the number of classes to classify and the number of features. Thus, training data with fewer classes to classify and a smaller feature space can be trained faster. Compared with Table 3.3, the recognition time is greater that the time taken for training, because the time cost of recognition is proportional to the size of test data and the size of constructed lattice. After comparing the impact factors of the two time costs, we can find that there is no correlation between them.

The CASAS benchmark dataset named Kyoto-1 (see more details in Appendix A) is a dataset mapping from lower-level sensor data to higher-level activities as mentioned in Fig. 1.2. A series of motion and analog sensors monitor five activities in the smart environment. However, every ADL class has diverse behavioral patterns (i.e. 120 different behavioral patterns

derived from five activities, see Table 3.2). Once any pattern is identified by our approach, the affiliated ADL class will be predicted and recognized as well.

Table 3.2: Time Cost for Training Concept Lattices

Dataset	Lattice Size			Time Cost (seconds)
	No. Activity Classes	No. Features	No. Concepts	
RDATA	12	69	24	0.0023
DDATA	12	69	24	0.0047
Kyoto-1	5 (120)	25	430	0.7112

3.6.2 RECOGNITION ACCURACY

Table 3.3 shows the recognition performance of the FCA-based model for different datasets. It is worth mentioning that the ontological clustering does not change the structure of constructed lattice. As an optional extension, it only provide additional information about the superclass of the previous prediction which is predicted without using the clustering. Therefore, the accuracies of recognition will not be affected after the clustering.

Table 3.3: Time Cost and Accuracy of Activity Recognition

Dataset	No. Items	Accuracy	Accuracy Without Clustering	Time Cost (s)
RDATA	240	100%	100%	0.0081
DDATA	96972	100%	100%	5.1789
Kyoto-1	120	86.7%	86.7%	0.0261

We evaluate the three datasets using 10-fold cross-validation. The *k*-fold cross-validation can reduce the unreliable estimation of future performance while increasing the bias [172]. As the results shown in Fig. 5 and Fig. 7 of the research work published by Chien and Huang [72], the recognition accuracy of the Kyoto-1 dataset is better than the experimental results (less than 85%) using incremental training by the classical HMM method, but inferior to the ones using off-line training (with 95.39% accuracy). Cook [173] has shown the accuracies of different data mining approaches, such as naive Bayes classifier (78.38%), HMM (78.38%)

and CRF (97.30%).

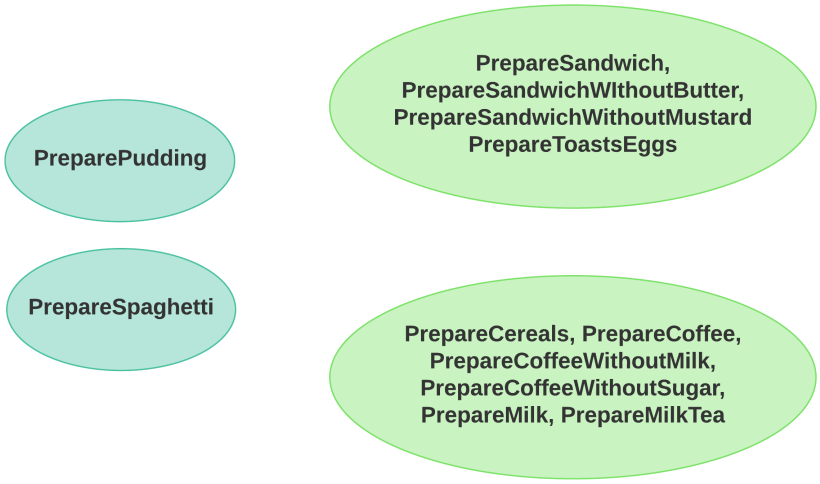


Figure 3.10: Ontological clusters of LIARA dataset

After the ontological clustering, twelve classes of different activities from the LIARA dataset are reclassified into four clusters (see Fig. 3.10). Two clusters that respectively indicate "PrepareSomethingToDrink" and "PrepareSomethingToEat" are generated. Another two small clusters only represent two separate classes, because they are not similar to others. In addition, we automatically classify activities in the Kyoto-1 dataset based on the spatial areas defined by motion sensors. The clustering results are shown in Fig. 3.11.

3.6.3 PREDICTION ACCURACY

Real-time activity prediction and related assessment occur when new data are observed and the corresponding activity is not completed. Successive operations loading new observed data into sequence α are called the *serial stages* and a local optimal prediction will be chosen at each stage. For the LIARA dataset, the total time cost of predictions is 2.1925 seconds, and each prediction takes about 2.03×10^{-5} seconds. For the CASAS dataset, the total time cost of predictions is 0.0204 seconds, about 1.72×10^{-4} seconds per prediction.

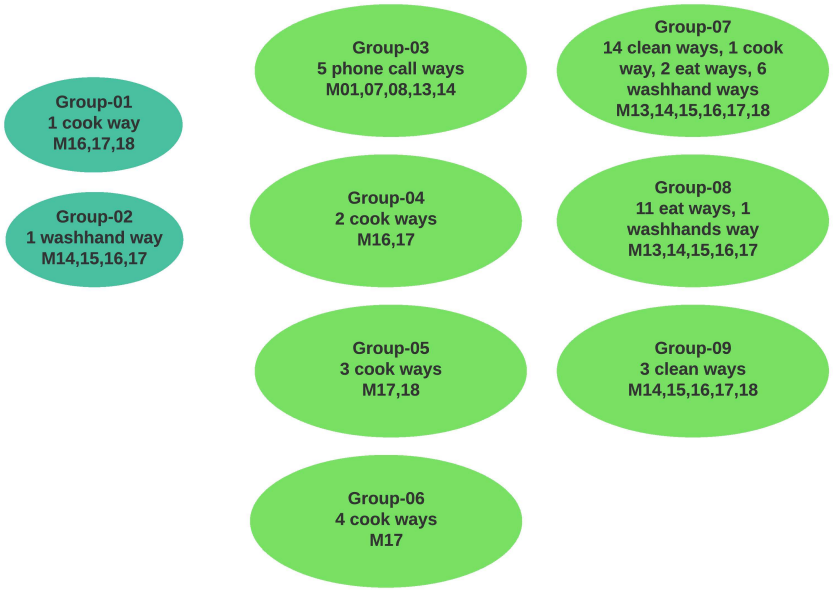


Figure 3.11: Ontological clusters of CASAS dataset

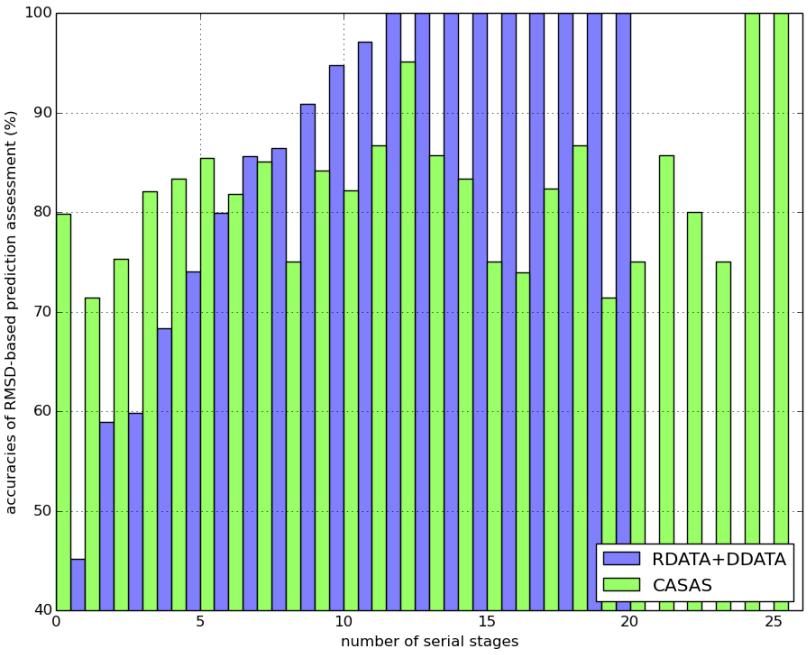


Figure 3.12: Prediction accuracies based on the RMSD at different stages.

Figure 3.12 depicts the average predictive accuracies at different stages and shows the evolution trend. For the RDATA and DDATA datasets, the range of valid stages is from 1 to 20, and for the CASAS dataset, the one is from 1 to 80 (accuracies after Stage 25 are 100%). For the RDATA and DDATA datasets, the accuracies of predictive assessment will be improved gradually when more and more data are being observed and loaded. In the CASAS dataset, a resident must first move to the right place to carry out an ADL. Thus, its predictive accuracies are better than another two datasets at the early stages due to the motion sensors. However, for the CASAS dataset, the accuracies of activity prediction are more susceptible to noise, because the sensor data with weaker semantic correlations are used to describe activities, rather than using the atomic actions with stronger correlations. Therefore, the accuracies will fluctuate.

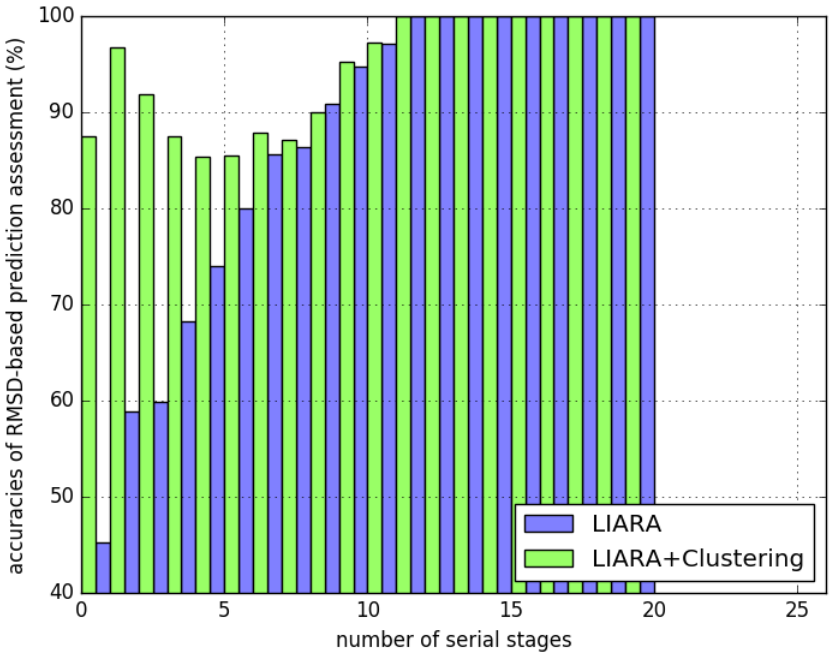


Figure 3.13: Comparison of LIARA recognition results

In Fig. 3.13 and Fig. 3.14, through the clustering method, we can see that the predictive

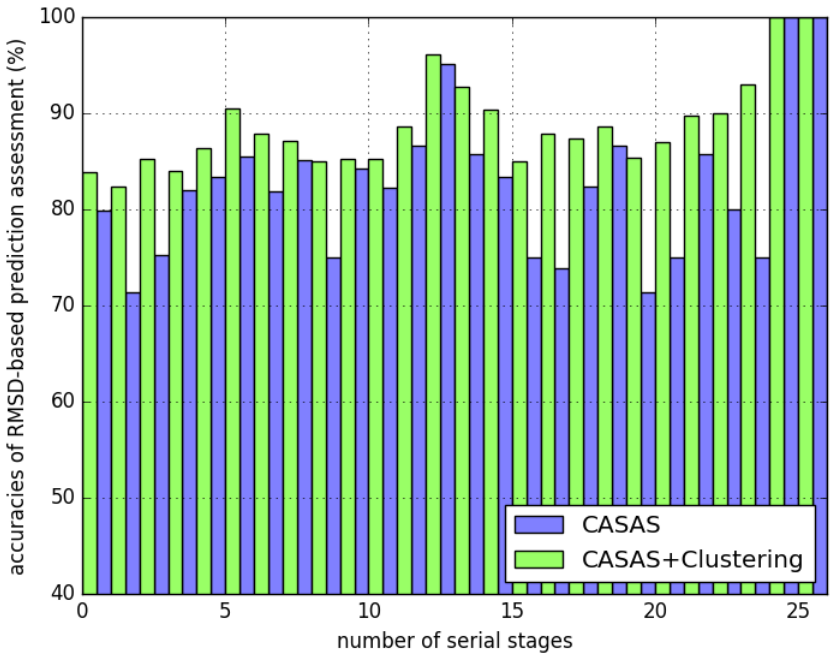


Figure 3.14: Comparison of CASAS recognition results

accuracy has been improved. This is because the inference engine predicts the superclass instead of directly predicting a more precise subclass using few observed data in the early stages. However, for the CASAS dataset, because the behavioral patterns describing the same activity performed by different participants are quite different and motion sensors have limited ability to distinguish different activities, many unseen patterns in the test data may be misclassified as similar patterns existing in the training data. Since new data are continuously observed, the most possible superclass is also gradually corrected and changed among predicted superclasses.

3.7 DISCUSSIONS

The FCA-based model is based on a rigorous mathematical theory. FCA provides a clear framework for better understanding the principle behind inferences. All the things above can

demonstrate that it can work well in practice.

Summing up the results obtained in the previous section, it is possible to conclude that the FCA-based model is suitable and efficient for real-time activity prediction and recognition in ubiquitous computing environments.

3.7.1 ADVANTAGES

First, unlike most expert systems based on scattered deductive reasoning, the hierarchical model based on FCA provides a unified and powerful deductive logic framework. It regards complicated activity prediction and recognition as a graph search problem and spontaneously achieves progressive deductive reasoning. Through representing the relations between activity and sensor data as binary relations, we can obtain enumerable concepts consisting of sensor data (intent) and affiliated activities (extent). With the successive manner loading data in real-time, the scope of probable activities in the extent shrinks gradually and the global optimal concept will be located at the end. All related inferences are automatically deduced by the closure transitions in the Hasse diagram.

Then, as an improved version of BFS, our graph search algorithm has obvious advantages in efficiency and consistency of reasoning. Unlike classical graph traversal algorithms abandoning all the previous reasoning, our incremental way to retrieve inferences needs neither to start over again nor to traverse the whole graph to look for the local optimal concept after observing new data. On the premise of no effect for the final results, our HDS algorithm continues inference retrieval from previous interrupted positions. Moreover, our graph search strategy can also distinguish most activities with multilevel inheritance.

Next, compared with the other statistical or probabilistic methods, our FCA-based model has fewer requirements about the volume of training data due to the data structure based on the

set and graph theories (see RDATA and DDATA statistical information in Table 3.2). In the training phase, because of the same binary relations, new patterns with the same sensor data but different execution orders will not change the structure of the existing FCA model, and only need to update the accumulation binary matrix for the RMSD-based assessment. After that, the real-time predictive assessment will be triggered when new data are observed and evaluated. The relevance of each inference will be evaluated in order to choose the most probable activity that may occur.

Moreover, the FCA-based models have considered the robustness problem about handling noisy sensor data. For each unseen pattern that is not in the training dataset, but in the test dataset, the models will compare its similarity with learned patterns and propose the most possible label as the recognition result. In the worst case, unreliable sensor data will be evaluated and classified into a similar one.

Finally, our approach has great superiority in the knowledge reuse and self-adaptation. The trained Hasse diagram and the accumulation of binary matrix are designed as two independent uncoupled modules. If one module has been modified, there is no influence to another one. As a consequence, accumulation binary matrices can also be reusable for the other scenarios.

3.7.2 DISADVANTAGES

First of all, classical lattice construction methods can only build lattices from Boolean binary relations [169]. This restriction limits that if we try to analyze certain numerical relations, we have to convert them into categorical values by losing precision. For example, in the CASAS dataset, we convert all the positive sensor values into Boolean True when we describe the interactions between ubiquitous sensors data and human activities. Briefly, if a tiny difference between numerical values in binary relations is crucial, we need at least transfer them into

the enumerable nominal values. Even then, it is not achievable in some extreme cases.

Then, activities with multilevel inheritance relations are more readily affected by unreliable data and recognized as one of their similar derivations. Next, for the assessment based on RMSD, the natural lattice structure does not contain temporal information about execution orders, so the bias in the assessment due to incidental factors cannot be completely avoided. At last, as a common problem appearing in the other state-of-the-art prototypes, unseen activities cannot be predicted or recognized if no corresponding training data is available in the dataset [174].

CHAPTER 4

COMPOSITE ACTIVITY RECOGNITION AND ERROR DETECTION

Composite behavioral pattern analysis is always a major challenge for smart home applications [175]. In most activity recognition studies, the processed data streams need to be well segmented with clear boundaries. Moreover, each stream is limited to describe only one activity. However, these assertions are too ideal to be fulfilled in reality. In general, human behaviors are planned and executed in a continuous and composite manner. Compared with the behavioral patterns of basic activities, the composite ones are usually sequential, without clear boundaries. Sometimes, activities are even executed in advanced ways such as interleaved or concurrent manner due to complex personal thinking. Thus, in this chapter, we first address the issue of recognizing composite human activities. The relative research [57] has been published in Journal of Reliable Intelligent Environments.

In addition to revealing suspicious behaviors, error detection is crucial to discover threatening events [176] in order to help people stay supported and safe. In this chapter, we also analyze abnormal behavioral patterns and define them as common errors. The formal definitions of these errors can help us clarify the features of each error and better understand the reason behind those abnormal behaviors. Custom-built error detectors are designed and integrated into our FCA-based inference engine. The inference engine not only recognizes and predicts

human activities, but also detects predefined errors in the sensor data streams. The relative research has been published in the Pervasive Technologies Related to Assistive Environments (PETRA) conference [62].

4.1 RELATED WORK ABOUT COMPOSITE ACTIVITY RECOGNITION

Because of the complexity of analyzing composite human behaviors in non-intrusive smart environments, there are only a few studies in this field. For example, Ruotsalainen et al. [177] introduced a genetic algorithm for detecting interleaved patterns from the sequences of sensor events. It has been used to partition the sequences and only matches them with specific pattern templates. Thus, this method is limited by the low generalization performance.

In other studies, Gu et al. [178] built their activity models based on Emerging Patterns to describe significant changes and differences between two classes to recognize sequential, interleaved and concurrent activities. Rashidi et al. [117] introduced an unsupervised approach in order to discover frequent interesting activity patterns and group similar discovered ones. They created an enhanced HMM to represent and recognize activities and their variants. One of the limitations of these methods is that they only consider specific sequences that occur frequently, but ignore some important problems such as imbalanced distributions in datasets.

As reported by Modayil et al. [179], an interleaved HMM was introduced to recognize multi-tasked activities. After minor modifications to the classical HMM model, the improved model is able to better predict the transition probabilities by recording the last behavioral pattern observed in each activity. Hu and Yang [180] proposed a two-level probabilistic framework for multiple-goal recognition including concurrent and interleaved activity recognition. They used skip-chain conditional random fields (SCCRF) and a correlation graph for modeling interleaved and concurrent activities. The results offered by Singla and Cook in [181] showed

a detailed performance comparison of different techniques involving naive Bayes and the variations of HMM. These methods often have strong noise immunity. Their drawbacks are mainly related to the computational complexity of the training stage. It is usually difficult to train a model with a large number of parameters or large state spaces.

For the other methods, Hallé et al. [182] used the finite-state automaton to decompose the total power load and distinguish the use of each appliance. Consequently, interleaved activities related to energy consumption are indirectly discriminated. However, it cannot handle activities without the use of appliances.

For the knowledge-driven approaches, Riboni et al. [183] proposed an unsupervised method to recognize composite activities by exploiting the semantics from the target activities and contextual data through ontological and probabilistic reasoning. Roy et al. [184] proposed a hybrid recognition model based on the probabilistic description logic. Okeyo et al. [185] combined ontological and temporal knowledge representation to recognize composite activities. Their model established relationships between activities and involved background knowledge. The temporal one defined correlations between constituent activities of a composite activity. Saguna et al. [186] proposed a conceptual framework for spatial-temporal context-aware systems to infer interleaved and concurrent activities. However, these knowledge-based methods require more extra knowledge or predefined inference rules. Their high requirement about domain knowledge makes the maintenance or extension difficult without domain experts.

Another interesting research introduced by Ye and Dobson [187] proposed a knowledge-driven approach for concurrent activity recognition [188]. However, their methods largely depend on domain knowledge, predefined logic expressions, and operations. These factors greatly reduce the efficiency and flexibility. In [52], a semantic-based segmentation approach is proposed to infer whether the incoming sensor event is related to an observed sequence. It

separates and segments the real-time sensor stream into multi-threads by the ontology. The approach consists of terminology and assertion reasoning, generic and user-specific logical rules, dynamic window size analysis and continuous RDF querying language. Its performance is limited by the number of activity threads that request incrementally inferences.

4.2 RECOGNIZING COMPOSITE BEHAVIORAL PATTERNS

Compared with the basic activity recognition, the composite one mainly concentrates on distinguishing composite behavioral patterns belonging to different activities. Recall that there are three types of composite patterns defined in Section 1.5: sequential, interleaving and concurrent ones.

As mentioned, every formal concept (i.e. node) of a Hasse diagram is a cluster regrouping ontological-similar objects that share common features. As a consequence, the behavioral patterns describing the same activity are almost in the same node. Furthermore, a pattern can derive many inherited ones with optional behavioral data that are represented as adjoining nodes. Thus, similar and derived patterns of an activity are represented within a group of clusters having similarly ontological relations. That is, formal concepts provide a powerful way to effectively aggregate long-range correlations among inter-dependent data objects.

If incoming data are excluded by such a cluster, it means that the data have strong ontological differences with other internal activities. As a result, the incoming data are classified as outliers of the current plan which is being executed, and have to be put into another one. The new plan starts a new search from the Supremum.

The principle of deciding whether observed data are necessary to be excluded or not by the current plan is determined by the hierarchy of a Hasse diagram. Suppose that a node (G_1, M_1) is located by the HDS algorithm, the set of relevant data R_e given a target class g is obtained

by Equation (4.1).

$$R_e = \bigcup_{\forall g \in G_1} g' \tag{4.1}$$

where g' is the concept-forming operation shown in Equation (3.1). All the other data, no matter indexed or not by the lattice, will be classified as the outliers of the current plan because the Infimum is immediately located. Once an outlier is detected, a provisional boundary will be marked and a new plan for caching will be created at the same time. The search of the current plan will also rollback from the Infimum to the previous position.

Activities about preparing breakfast		*a*: boil water	*b*: prepare tableware	*c*: add cocoa powder	*d*: pour cereals	*e*: take out breads	*f*: take out teabags	*g*: take out spaghetti	*h*: add sugar	*i*: add milk	*j*: add sauce	*k*: use toaster	*l*: use microwave oven
PrepareHotChocolate	g_1		×	×					×	×			×
PrepareMilkTea	g_2	×	×				×		×	×			
PrepareSpaghetti	g_3	×	×					×			×		
PrepareCaffèMocha	g_4	×	×	×					×	×			
PrepareCereals	g_5		×	×	×					×			×
PrepareToast	g_6		×			×						×	
PrepareSandwich	g_7		×			×					×		

Figure 4.1: Matrix representing the activities g_i carried out in the kitchen and their atomic actions m_j.

Suppose that there are seven activities about preparing breakfast: *PrepareHotChocolate* (g_1), *PrepareMilkTea* (g_2), *PrepareSpaghetti* (g_3), *PrepareCaffèMocha* (g_4), *PrepareCereals* (g_5), *PrepareToast* (g_6) and *PrepareSandwich* (g_7). There are also twelve actions shared among these activities: *boil water* (*a*), *prepare tableware* (*b*), *add cocoa powder* (*c*), *pour cereals* (*d*), *take out breads* (*e*), *take out teabags* (f), *take out spaghetti* (*g*), *add sugar* (*h*), *add milk*

(*i*), *add sauce* (*j*), *use toaster* (*k*) and *use microwave oven* (*l*). The binary matrix is shown in Fig. 4.1.

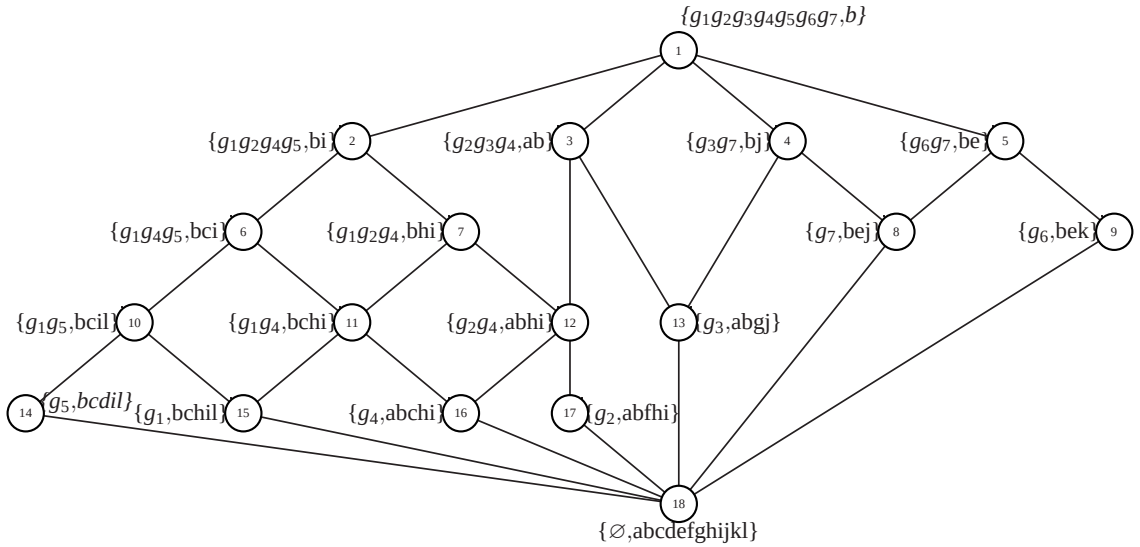


Figure 4.2: Hasse diagram of the binary matrix shown in Fig. 4.1

Considering the lattice shown in Fig. 4.2, suppose $\alpha = \{b \prec e \prec b \prec c \prec i \prec b \prec l \prec g \prec k \prec h\}$ indicating two interleaved activities *PrepareHotChocolate* (g_1) and *PrepareToast* (g_6). There is also an unreliable data g (*take out spaghetti*). Table 4.1 depicts the whole composite activity recognition process. The symbol $\circlearrowleft_{Infimum}$ indicates a rollback operation from the Infimum to the previous search result.

At round 4, when c is observed, $\{bebc\}$ is excluded by the current plan because no subconcept of node 5 contains these observations except the Infimum. Thus, a new plan is created to cache c and launches a concurrent search. At round 8, because g is excluded by all the existing plans. A new concurrent one is created at that moment to cache g. Activities g_6 and g_1 are finally recognized at round 9 and 10, because their sizes of extent are equals to 1 and all the required observations in the intents are observed.

Figure 4.3 illustrates the interweaving situation. There are three plans P_i ($i \in \{0,1,2\}$) in

Table 4.1: Inferring Process of Composite Activity Recognition

Round	Observed Data α	Located Topmost Concept	Predictive Activities
1	$\{b\}$	node 1 $\{g_1g_2g_3g_4g_5g_6g_7$,**b**$\}$	$g_1g_2g_3g_4g_5g_6g_7$
2	$\{be\}$	node 1 $\curvearrowright$ node 5 $\{g_6g_7$,**be**$\}$	g_6g_7
3	$\{beb\}$	node 5 $\{g_6g_7$,**be**$\}$	g_6g_7
4	$\{bebc\}$	node 5 $\circlearrowleft_{Infimum}$ $\{g_6g_7$,**be**$\}$ node 6 $\{g_1g_4g_5$,b**c**i$\}$	g_6g_7 $g_1g_4g_5$
5	$\{bebci\}$	node 5 $\circlearrowleft_{Infimum}$ $\{g_6g_7$,**be**$\}$ node 6 $\{g_1g_4g_5$,b**ci**$\}$	g_6g_7 $g_1g_4g_5$
6	$\{bebcib\}$	node 5 $\{g_6g_7$,**be**$\}$ node 6 $\{g_1g_4g_5$,**bci**$\}$	g_6g_7 $g_1g_4g_5$
7	$\{bebcibl\}$	node 5 $\circlearrowleft_{Infimum}$ $\{g_6g_7$,**be**$\}$ node 6 $\curvearrowright$ node 10 $\{g_1g_5$,**bcil**$\}$	g_6g_7 g_1g_5
8	$\{bebciblg\}$	node 5 $\circlearrowleft_{Infimum}$ $\{g_6g_7$,**be**$\}$ node 10 $\circlearrowleft_{Infimum}$ $\{g_1g_5$,**bcil**$\}$ node 13 $\{g_3$,ab**gj**$\}$	g_6g_7 g_1g_5 g_3
9	$\{bebciblgk\}$	node 5 $\curvearrowright$ node 9 $\{g_6$,**bek**$\}$ node 10 $\circlearrowleft_{Infimum}$ $\{g_1g_5$,**bcil**$\}$ node 13 $\circlearrowleft_{Infimum}$ $\{g_3$,ab**gj**$\}$	g_6 g_1g_5 g_3
10	$\{bebciblgkh\}$	node 9 $\{g_6$,**bek**$\}$ node 10 $\curvearrowright$ node 15 $\{g_1$,**bcilh**$\}$ node 13 $\circlearrowleft_{Infimum}$ $\{g_3$,ab**gj**$\}$	g_6 g_1 ~~g_3~~

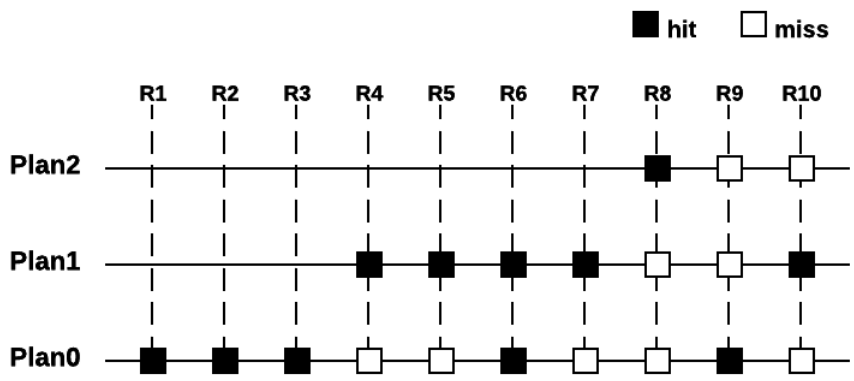


Figure 4.3: Interweaving plans appearing in the process of composite activity recognition

the figure. P_0 is the initial plan. P_1 and P_2 are created when observed data is irrelevant to all the existing plan. Squares indicate two states of observed data: the black ones indicate the observed data is relevant to the patterns in the present P_i (i.e. hit), and the hollow ones indicate the data is irrelevant (i.e. miss). For any incoming data, it can trigger one of the three possible states:

- strictly belongs to one plan: the observed data belongs to a unique plan. For example, $R1$, $R2$, $R3$, $R4$, $R5$, $R7$, $R8$, $R9$ and $R10$ in Fig. 4.3.
- belongs to more than one plan: it always happens to concurrent activities. For example, $R6$ in Fig. 4.3.
- belongs to none of the existing plans: In sequential activities, it is the moment triggering the boundary detection. In interleaved patterns, the resident may start to do another activity or an irrelevant action, or the system may receive an unreliable data. For example, $R4$ and $R8$ in Fig. 4.3.

At the end of the data stream, a completeness check will verify all the existing plans. There

are two objectives: first of all, the amount of predictive activities will be checked. The plan having too many predictive activities will be abandoned due to ambiguity. Otherwise, a further check will verify the completeness of each activity calculated by Equation 4.2.

$$C_i = \frac{|g_i' \cap \alpha|}{|g_i'|} \quad \text{and} \quad g_i \in G \tag{4.2}$$

where $|g_i' \cap \alpha|$ indicates the number of observed data and $|g_i'|$ indicates the required one. An activity having low completeness will be abandoned. In Table 4.1, activity g_3 was finally abandoned due to low completeness, and the cached g was identified as unreliable data.

4.3 SIGNIFICANCE OF ANOMALY DETECTION IN SMART ENVIRONMENTS

In our daily lives, some normal activities such as cooking or adherence of medical instruction may become risky as well [189]. The increasing need for appropriate intervention leads to the emergence of smart homes, which is a typical AAL application [170]. Smart environments desire to avoid some of the potential daily threats. For example: forget to turn off the stove, excessive sodium & sugar consumption, or unintentional overdose of drugs, etc.

4.4 RELATED WORK ABOUT ANOMALY DETECTION IN SMART ENVIRONMENTS

As a common problem, sequential anomaly detection has been discussed in many aspects such as machine learning, data mining and applied mathematics [190, 191, 192]. So far, for AmI problems, we can conclude that errors in sensor data are a kind of the contextual anomaly because a human behavior or sensor data is normal and not inherently unusual. It is only considered abnormal under certain contexts [193]. However, those errors are usually difficult

to detect. Firstly, because of context-sensitive and diverse forms, it is difficult to ensure that all possible anomalies are considered and covered in the training datasets. Moreover, the annotation of abnormal samples is also prohibitively expensive [193]. Fortunately, in the training data, abnormal patterns may be dissimilar under certain criteria in the comparison of normal ones, or they often have rare occurrences [176]. Therefore, most solutions are based on two assertions and are classified as similarity-based and frequency-based methods.

Similarity-based methods are based on the assumption that normal sequential data are dissimilar in several criteria. Thus, these solutions usually focus on the methods such as classification or cluster analysis. Park et al. [194] defined a similarity scoring function using the longest common subsequence (LCS) to determine abnormal human behaviors among low-level sensor data. Zhao et al. [195] clustered activities in the temporal aspect and used Markov chain model to measure whether a sequence of activities is abnormal or not. Duong et al. [196] used a hidden semi-Markov model and durations of activities to detect abnormal deviations from normal patterns. Besides, El-Kechaï and Després [197] proposed a domain-independent formalism to classify possible errors.

For frequency-based methods, most of them are based on the assumption that patterns containing errors occur rarely in the training dataset. They try to identify abnormal patterns with low occurrences which are seemingly biased towards the normal ones. For example, Yin et al. [198] presented a model based on the support vector machine to filter out most of the normal activities, and then handle suspicious ones using kernel nonlinear regression (KNLR) model for further detection.

A key limitation of these previous studies is that they do not address the customization problem and more or less ignore the behavioral features of anomalous patterns. Thus, it is easy to suffer from high missing and false alarm rates. Some abnormal behavioral patterns were

also analyzed in the studies of Roy et al. [170] and Fortin-Simard et al. [58]. On the basis of these previous works, in this chapter, we further analyze significant features existing in the abnormal data streams, and summarize common errors from their abnormal patterns. Relative solutions will be proposed for each predefined error.

4.5 ANOMALY DETECTION PROBLEM SETTINGS

Besides the activity recognition module, we also create an error detection (CED) module to detect particular characteristics in the patterns. In this section, we summarize common errors and discuss how to detect them based on their behavioral characteristics.

Derivative patterns are defined as the various behavioral patterns having changeable data with flexible execution orders, but derived from the same activity. Suppose that there are N_i derivative patterns describing an activity A_i. Thus, a pattern α_j describing A_i is defined as a container (not a set) of:

- Essential Data Set E, where $E = \bigcap_{i=1}^{N_i} \alpha_i$, which contains all essential data existing in all N_i derivative patterns of A_i. That is, the data exists in all the derivative patterns. The arbitrary intersection $\bigcap_{i=1}^{N_i} \alpha_i$ ensures that all the data in the intersection appeared in every pattern describing the activity A_i.

  For example, “boil water” and “pour water into a teacup” are two essential actions for “PrepareTea”, because they exist in any pattern α_i describing the process of making a cup of tea, no matter who does it.

- Optional Data Set O, where $O = \bigcup_{i=1}^{N_i} \alpha_i - \bigcap_{i=1}^{N_i} \alpha_i$, which indicates optional data for the patterns of A_i. The arbitrary union $\bigcup_{i=1}^{N_i} \alpha_i$ aggregates all the data that described A_i. In other words, it indicates all the data that are related to A_i. Thus, the difference of

$\bigcup_{i=1}^{N_i} \alpha_i$ and the essential data $\bigcap_{i=1}^{N_i} \alpha_i$ is the set of optional data, because they described the activity, but not appeared in all the patterns.

For example, ‘add milk’ can be somebody’s personal taste when drinking tea, but not exists in all the patterns describing ‘prepare a cup of tea’. So it is a typical optional action.

- Possible Irrelevant Data Set I, where $I \cap \bigcup_{i=1}^{N_i} \alpha_i = \emptyset$.
  For example, ’take out pasta from cabinet’ is an irrelevant action for ‘prepare a cup of tea’ and it will not exist in any of its normal execution sequences.
- Possible Redundant Data Set R, where $R \subseteq \bigcup_{i=1}^{N_i} \alpha_i$, which contains all the data existing in the entire N_i derivative patterns of A_i. This is because any data can appear twice or more times, and becomes redundant.

All these sets are generated automatically from data without any prior domain knowledge. So we give out our generic symbolic representation of a pattern α_j in the form of a triplet:

$$\alpha_j = (\{E \cup O' \cup I' \cup R'\}, \prec_j, C) \tag{4.3}$$

where $O' \subseteq O$, $I' \subseteq I$, and $R' \subseteq R$. The symbol $\prec_j$ refers to a possible permutation of the union (i.e. a possible execution order). C is a set of order constraints limiting the permutation $\prec_j$. Thus, we assert that α_j is a normal sequence of data without errors if and only if set E is complete, sets I' and R' are empty, and $\prec_j$ satisfies all the constraints in C.

From the definitions above, we can find out that different sets and their permutations play a key role in the constitution of errors. In the next section, we will explain how to detect each error using our inference engine.

4.6 ERROR DEFINITIONS

In this section, by observing and tracking the daily lives of people, first of all, we describe each type of abnormal behavioral pattern appearing in the historical data and define those patterns as errors. And then, through behavioral pattern analysis, we explain how to detect those errors and give out corresponding solutions.

4.6.1 INITIALIZATION

The initialization error is to do nothing at the beginning of an activity. A simple solution is to set a temporal threshold to detect whether a resident does something for accomplishing an activity at the early stage. Because it is not associated with behavioral data analysis, in this section, the initialization error will not be considered.

4.6.2 OMISSION OF ESSENTIAL DATA

The omission of essential data is a failure to do something that ought to be done, but was forgotten, according to the initial planning. It is a very usual scenario in daily life. Sometimes, there is only a limited influence for performing an activity. For example, there is no big deal if a resident forgets to do some behaviors related to the optional data summarized in set O like personal preferences. However, most of the time, the omission of essential behaviors will break the integrity of implementation (e.g. forgetting to add some ingredients while cooking) and the quality of accomplishment will also be affected. In some extreme cases, it will lead to serious or fatal consequences (e.g. forgetting to turn off the oven after use).

As we mentioned above, the optional data in set O are less important than the ones in set E, and bring less trouble while being omitted. Due to the set-based dual structure of concepts,

it is easy to check the final completion of implementation using set theory: if the universal actions of an activity A_i is denoted as U_i, the forgotten actions can be calculated as the relative complement $S^C = U_i - S$, where S is currently observed data. It is worthy to mention that U_i can be quickly obtained by executing the concept-forming operation A_i' or searching the cross table.

	m_1	m_2	m_3	m_4	m_5	m_6
g_1	×	×				
g_2	×	×	×			
g_3	×			×		
g_4	×	×	×	×		×
g_5	×					
g_6					×	×

Figure 4.4: Example of cross table for error detection

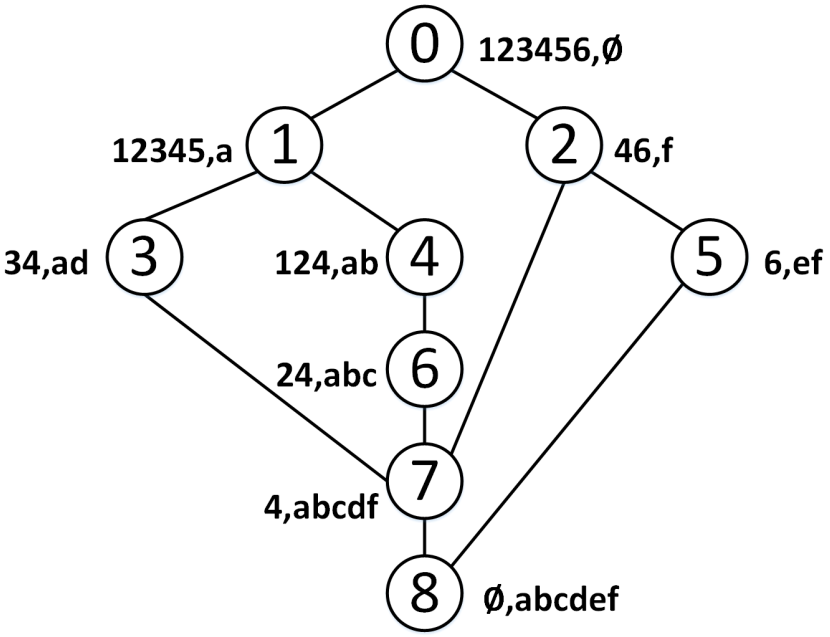


Figure 4.5: Simplified lattice for illustrating how to detect errors

Example: suppose that the actions in sequence $\alpha = \{a \prec c \prec b \prec f\}$ are successively loaded. Considering Fig. 4.5 obtained from the binary matrix shown in Fig. 4.4, node 7 is located

at the end of the extensions. To check the degree of completion of activity g_4 indicating in the extent, we compare current observed sequence $\alpha = \{acbf\}$ with $g_4' = \{abcdf\}$, and the complement $g_4' - \alpha = \{d\}$ is not an empty set, so d is omitted during the execution of g_4.

4.6.3 UNREASONABLE REPETITION

The reason of redundant information existing in the data stream can be various: the periodic sampling of sensors, reasonable intention or anomaly etc. In our case, the redundant information should be the repetitive data existing in the observed sequence of data. All the repetitions, no matter reasonable or not, will be successfully detected, because it is just a simple set operation. In most cases, repetitive behaviors are harmless, even reasonable and necessary to accomplish an activity. For example, we need to regularly check the degree of cooking or intermittently stir the ingredients while preparing a meal. In the other extreme cases, unreasonable repetitive actions will lead to potential threats like excessive consumption (condiments or medications).

The simplest solution is to check if the incoming data exists in the current sequence α. To distinguish the unreasonable repetition and the reasonable ones, we define a weighted array to measure the harm degree of each data being repetitive. For this reason, the detection accuracy of harmful redundancy could be reinforced and the false-positive alert warning harmless redundancy could be reduced. For example, almost all the repetitive data generated by the motion sensors are harmless. If data m is captured periodically in the patterns describing activity A, then its weight is defined as a low value in the array of activity A. In contrast, if m exists only once in each pattern and it is generated by an object sensor, then its weight should be carefully defined as a high value.

4.6.4 MIXTURE OF IRRELEVANT DATA

Sometimes, people may forget current long-term intention or confuse with another one, and then add irrelevant data into the current ongoing activity. From Equation 4.3, we can see that irrelevant dataset I of activity A_i has no intersection with the relevant one $E \cup O$. In other words, an extension caused by incoming data a is acceptable for current planning if and only if $a \in E \cup O$. Thus, full elements in I will be excluded by all the concepts containing A_i.

After a new extension, if updated α is no longer compatible with any concept except the Infimum, there are probably one or more irrelevant data which have mixed into the current sequence, especially the last incoming one should be suspected.

Example: considering Fig. 4.5, suppose sequence α is successively extended by $\{a \prec c \prec e \prec d \prec b \prec f\}$. Node 6 is located after the first two extensions $\alpha \leftarrow ac$. In the third round, $\alpha \leftarrow e$, updated $\alpha = \{ace\}$ is incompatible with current planning because there is no sub-concept (A,B) having $\alpha \subseteq B$ except the Infimum. As a consequence, last incoming e will be treated as irrelevant data which have to be removed from the initial cache and put it aside, into a newly created cache indicating another planning. At the end of the extensions, node 7 is located and the irrelevant data e is identified.

We summarize the logic above and represent it in Algorithm 4. Cache P_0 always denotes the initial planning of a resident. New data a is observed and loaded for an extension at step 3. Step 4 to 7 is to check whether there exist one or more caches in P_i compatible with current observed data. If a is irrelevant to all existing caches (step 9), then create a new cache to save it (step 10 to 11). After extensions, we choose the longest cache, P_0 in most of the time, as the normal sequence performing A_i (step 12), and the data in the other caches will be treated as irrelevant one.

Algorithm 4: detect mixture of irrelevant data

Data: sequence α, lattice $\mathcal{L}$, caches P_i.
Result: set of irrelevant data $\mathcal{I}$.

```
1  begin
2      while α do
3          a ← α.popleft
4          foreach P_i do
5              if ∃(A,B) ∈ L, P_i ∪ a ⊆ B then
6                  P_i ← P_i ∪ a
7          end
8      end
9      if ∄(A,B) ∈ L, P_i ∪ a ⊆ B then
10         P_{i+1} ← a
11         P_i ← P_i + P_{i+1}
12     P_M ← max(size(P_i))
13 end
```

4.6.5 ORDER INVERSION

Suppose two data (actions or sensor data), $\alpha_i \prec \alpha_{i+m}$, appear successively in the sequence $\alpha = \{\alpha_0 \prec ... \prec \alpha_i \prec ... \prec \alpha_{i+m} \prec ... \prec \alpha_n\}$. If the set of order constraints C has limited that α_{i+m} must occur before α_i, represented as $\alpha_{i+m} \preceq \alpha_i$, then there is a order inversion in the sequence [58].

We manually define order constraints and then verify them among data in α. For any data α_i in the sequence, we generate its order pairs by scanning all the data on its right. If one generated pair (α_i, α_j) has the opposite one (α_j, α_i) in C and no α_j appeared before α_i, then the sequential execution $\alpha_i \prec \alpha_j$ is against the predefined constraints. The time complexity of order inversion check is $T(O(n^2))$.

4.6.6 DISTRACTION

The distraction is similar to adding irrelevant data. Compared to original planning, the two errors have the same feature that they are mixed irrelevant data into their sequences, but distraction has created a transformation of quantitative into qualitative changes. Different from the mixture of irrelevant data, this error can be classified as a collective anomaly [193]. The feature of distraction is that at the beginning of the sequence of data, all the performed behaviors belong to a real expected long-term planning. At a specific singular point, the performed behaviors started to differ from the original objective.

Figure 4.6 is an example of distraction. Planning 0 is used to indicate the original planning of a resident and Planning 1 and 2 denote his/her distracted traces. A Black point represents a hit that the loaded data used for extension in this step is accepted by the positioned cache and the Hasse diagram, and a white one indicates a missing. The difference between the distraction error and the concurrent tasks concentrates on their completenesses. The concurrent tasks can always be finished in a period, but the distraction error always has an unfinished original planning.

The distraction really happens in the fourth extension and T_1 indicates this singular position. The loaded data a_4 has not been accepted by the Planning 1 due to its irrelevance. Once data are not acceptable for all existing caches, we need to put them in a new one. There is only one black point at the moment of new cache creation. Moreover, if data are compatible with more than one cache, they must be distributed into each compatible cache. At the end of the extensions, we choose the longest cache having the most compatible data as the normal sequence of data. If the longest cache is not Planning 0, we can assert that the resident has derived from his/her real objective.

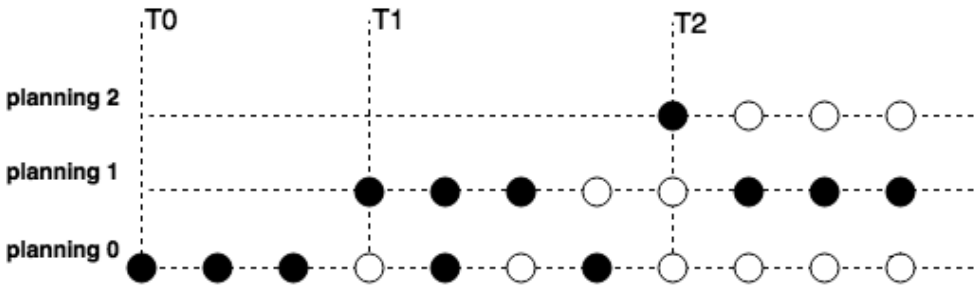


Figure 4.6: Example of distraction

4.7 EXPERIMENTS

In this section, we separately evaluate the performance of our inference engine in recognizing composite activities and detecting errors.

4.7.1 EXPERIMENTS ABOUT COMPOSITE ACTIVITY RECOGNITION

The performances of the inference engine are tested using two datasets created in two smart environments, LIARA, and CASAS testbeds. More information about the two datasets are described in Appendix A. We use the behavioral patterns describing basic activities to train the model and then use it to recognize patterns describing composite activities. The reason is that we hope to establish precise semantic correlations between activities and sensor data (or atomic actions). The common classification metrics, F-measure and accuracy [83, 199] (see Appendix B), are used to evaluate the performance of activity recognition. All the experiments are carried out on a computer with tech specs of Intel Core i7 Processor 2.4GHz and 8GB RAM, under Ubuntu 16.04.

In Table. 4.2, statistical information and F-measure results using FCA-based inference engine are given out. Activities without multilevel inheritance relations have better recognition accuracies in the composite mode. This is because activities with multilevel inheritance relations

Table 4.2: Statistical Information and F-measure Results of LIARA Dataset

Classes	Activities	Amount of Actions	F-measure
ac_1	PrepareSandwichWithoutMustard	11	0.947
ac_2	PrepareCoffeeWithoutSugar	11	0.947
ac_3	PrepareCereals	8	1.000
ac_4	PrepareMilkTea	12	1.000
ac_5	PreparePudding	5	1.000
ac_6	PrepareToastsEggs	20	1.000
ac_7	PrepareMilk	5	0.952
ac_8	PrepareSandwichWithoutButter	9	0.869
ac_9	PrepareSpaghetti	18	1.000
ac_{10}	PrepareCoffee	14	0.976
ac_{11}	PrepareSandwich	15	0.902
ac_{12}	PrepareCoffeeWithoutMilk	11	0.806
	Overall F1 score	-	0.954
	Overall accuracy	-	0.985

Table 4.3: Comparison of Accuracies of CASAS Dataset

Classes	**Naive Bayes** [181]	**HMM** [175]	**FCA-based**
ac_1	50%	58%	100%
ac_2	62%	78%	100%
ac_3	27%	43%	60%
ac_4	39%	46%	95%
ac_5	78%	80%	95%
ac_6	83%	82%	100%
ac_7	89%	81%	100%
ac_8	57%	67%	100%

are easier to be affected by unreliable data and recognized as one of their similar derivations.

In Table. 4.3, we compared the recognition accuracy with different methods [175, 181]. In Fig. 4.7, our method achieves the highest accuracy (93.75%) among naive Bayes (66.08%) and HMM (71%) [181]. In Table. 4.4, we compared the performance of our method with another two methods described in [175, 183] by F-measure. Composite behavioral patterns are classified as eight classes (activities). From these comparisons, we can see that our method outperforms in each recognition case.

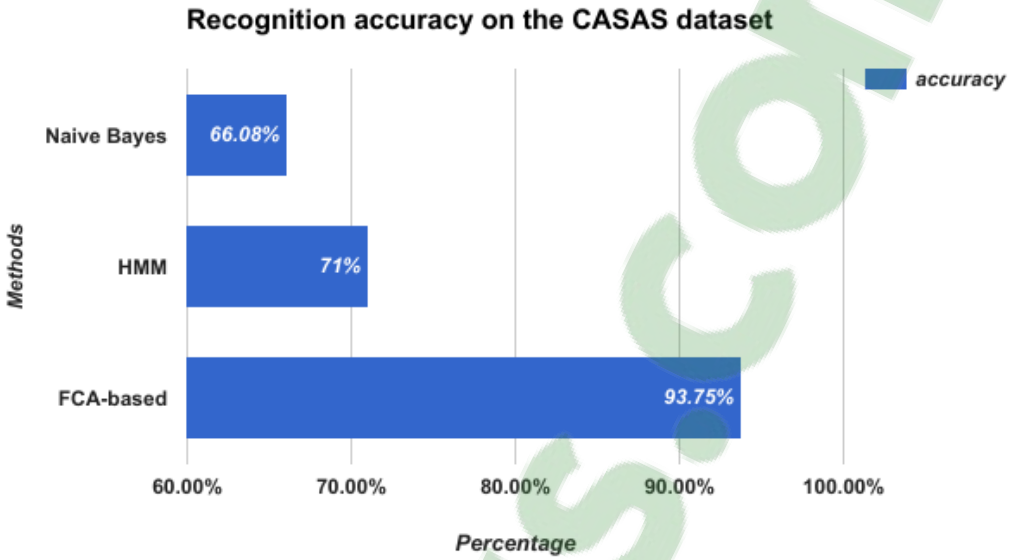


Figure 4.7: Recognition accuracy of different methods on the CASAS Kyoto-3 dataset

Table 4.4: Comparison of F-measure of CASAS Dataset

Classes	**HMM** [175]	**MLN (supervised)** [183]	**FCA-based**
ac_1	0.656	0.803	**1.000**
ac_2	0.862	0.882	**1.000**
ac_3	0.285	0.740	**0.750**
ac_4	0.589	0.688	**0.973**
ac_5	0.828	0.807	**0.974**
ac_6	0.826	0.873	**1.000**
ac_7	0.881	0.781	**1.000**
ac_8	0.673	0.904	**1.000**
avg	0.700	0.810	**0.962**

For the time complexities in both the training and test phases, we give out the statistical information in Table. 4.5. The training phase includes sequential pattern extraction, formal lattice construction, and historical data accumulation. While handling with LIARA dataset, the training and testing times are both very low. Compared with LIARA dataset, CASAS data has much fewer training items, but the training time is much longer than the LIARA one. The reason is that the number of target classes greatly affect the number of clusters. The augmentation of clusters also increases the complexity of searching in the Hasse diagram.

Table 4.5: Statistic Information and Performance of FCA-based Algorithm in Different Datasets

Datasets	Classes	Features	Nodes	Training Items	Training Times	Test Items	Test Times
LIARA	12	70	25	25207	0.0062s	2520	0.8093s
CASAS	160	84	5089	160	40.3625s	20	1.6961s

4.7.2 EXPERIMENT ABOUT DETECTING ANOMALIES

Our experiment is first carried out on two datasets: the LIARA abnormal dataset described in Appendix A that involves predefined errors, as well as the CASAS error dataset described in Appendix A involving the omission and repetition errors.

Table 4.6 sketches the accuracies about errors detection applied on the two test data sets by 3-fold cross-validation. To our best knowledge, very few benchmark publications are available in the literature that use the same dataset to evaluate the performance of error detection.

Table 4.6: Accuracies of Error Detections in Two Datasets

Errors	Datasets / Accuracy	
	LIARA Errors	CASAS Kyoto-2
Omission of Essential Data	100%	88.5%
Mixture of Irrelevant Data	100%	-
Unreasonable Repetition	100%	100%
Order Inversion	100% (M)	-
Distraction	$\geq$ 97.8%	-

From the listed results in Table 4.6, we can see that our model received excellent detection

rates in four errors except for the distraction. One of the reasons is that the detection accuracy of distraction error depends on the singular position when the distraction occurs. Figure 4.8 shows the F-measure at different singular positions. The precision at each position is always equal to 1 (TP=1.0 and FN=0.0). It is worth mentioning that the result of order inversion detection was based on the manually defined order constraints (marked as "M"). The total time cost of the error detection is about 0.4182 seconds.

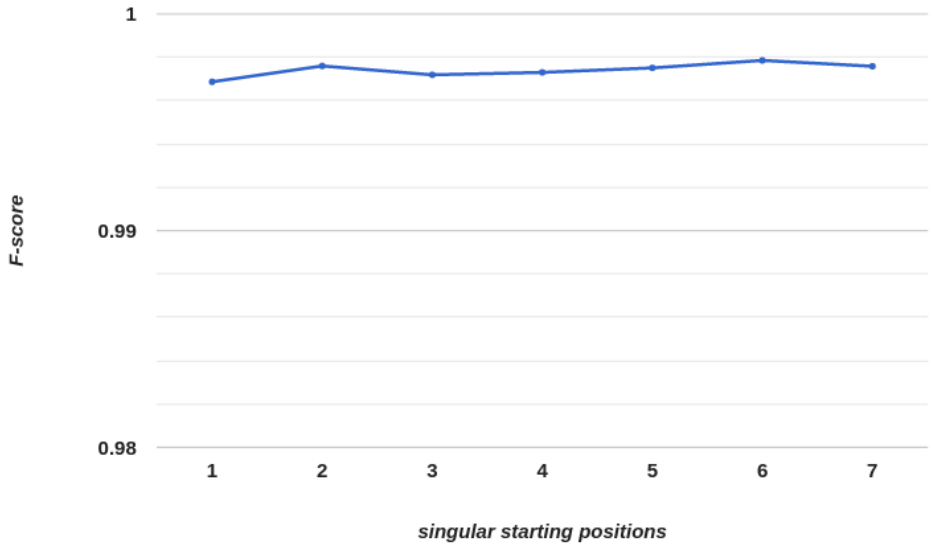


Figure 4.8: Distraction detection of LIARA dataset at different singular positions

For the CASAS dataset, there are only two predefined errors existing in the test samples: omission (did not turn the water off, did not turn the burner off, did not bring the medicine container, did not use water to clean and did not dial a phone number) and repetition (dialed a wrong phone number and redialed, duplicate sampling of motion sensors, etc.). We used "-" to represent the nonexistent results in Table 4.6. Furthermore, we evaluated its results under evaluation metrics, including precision, recall, and F-measure in Table 4.7. The total time cost of the error detection is about 1.01×10^{-3} seconds.

The architecture of CED is sketched in Figure 4.9. After the features analysis of common

Table 4.7: Results of Error Detection in CASAS Kyoto-2 Dataset

Errors	Precision	Recall	F-score
Omission of Essential Data	0.656250	1.0	0.792453
Unreasonable Repetition	1.0	1.0	1.0

abnormal behavioral patterns, we gave out different solutions for detecting predefined errors.

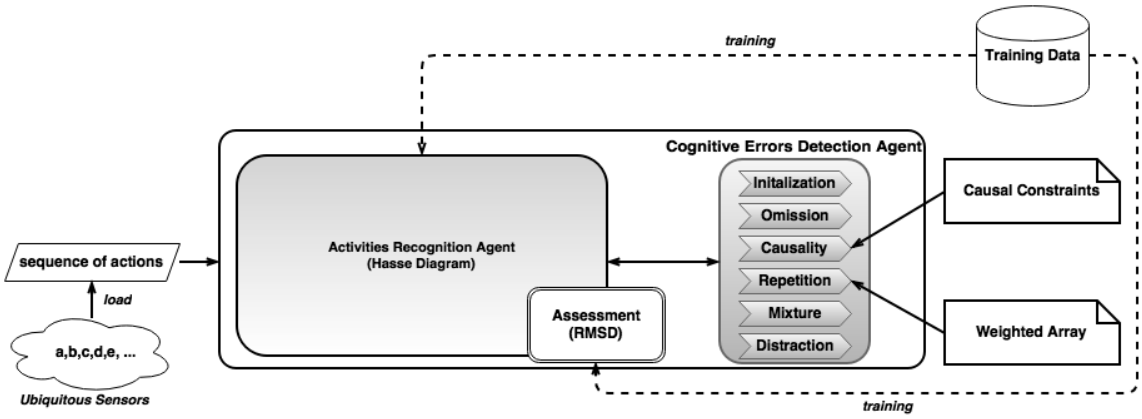


Figure 4.9: Architecture of FCA-based inference engine with error detectors

The omission of essential actions and unreasonable repetition are two errors strongly related to the set theory of discrete mathematics. Through simple algebra of sets and binary operations on sets, they can be easily detected. As shown in Table 4.6, repetitive actions in the sequence were 100% detected, but not all of them are unreasonable. For example, in CASAS, due to the deployment of motion sensors and periodic sampling, sequences are filled with repetitive events. The presence of motion sensors in CASAS also affects the result of the omission error detection. Irregular movements of residents produce massive derivative sets of actions having negligible movements as elements of the optional actions set O. Thus, the repetition and omission existing in the sequence of sensor data will lead to a high false-positive rate (12.3%).

In order to reduce the false-positive rate and to increase the true-positive rate at the same time, it is worthy to note that a weighted array was defined for the unreasonable repetition error

to automatically adjust the detection sensitivity on the basis of the severity of each repetitive data.

To detect order inversion in a sequence, compared to simple binary operations on set, the biggest challenge to overcome is the source of order constraints. As the result shown in Table 4.6, order constraints defined by human experts are accurate and easy to be deployed into conflict detection, but the definition was also prohibitively expensive.

The rest two errors, the mixture of irrelevant data and distraction, are more complex than the others because of the ambiguous singular position between original intention and the abnormal one. Multilevel inheritance and varied singular positions also aggravate the complexity of situations. In the worst case, some samples with distraction errors will be identified as a series of repetition errors in this case. Unlike probabilistic models, our FCA-based model is not easily affected by imbalanced class distributions. Only normal classes corresponding to normal behavior can be used for training a model to identify anomalies in the test data.

4.8 CONCLUSION

In this chapter, we first proposed another search strategy to recognize composite behavioral patterns from complex activities. Unlike most of the data-driven methods depending on large-scale data to discover regularity of probability distribution and drive internal reasoning, FCA-based model emphasizes the internal correlations of activities to recognize. According to the ontological differences, the FCA-based model differentiates sequential, concurrent or interleaved behavioral patterns belonging to different activities in the continuous data flow. The model does not require clear boundaries of the beginning and the end of a sequential pattern describing an activity. Based on the ontological relevance, sensor data can be automatically classified to the most appropriate patterns.

We also formulated the most common errors existing among people. Combined with the FCA-based activity inference engine, we proposed several errors detectors to detect predefined errors in the sequences of data. Moreover, we also defined several dynamic mechanisms to reduce the false-positive rate according to predefined weights. Unlike the other similarity or frequency-based approaches, our approach does not require the fault samples should be available in advance.

However, our approach also has some constraints. The training data are required to cover diverse behavioral patterns describing the same activities as many as possible. Insufficient samples will cause high false alarm rates while detecting omission of essential data and the mixture of irrelevant data. The results of error detection will be more stable in a larger dataset, because the classification of essential and optional data is more precise. All the error detections depending on such a classification will be more accurate.

CHAPTER 5

MULTIPLE RESIDENT ACTIVITY RECOGNITION

In this chapter, we focus on a more complicated issue about multi-resident activity recognition. Section 5.1 outlines why multi-resident activity recognition is an indispensable research subject for smart environment applications. Section 5.2 introduces the recently published related work. Section 5.3 examines how to identify different patterns by using an FCA-based model. Section 5.4 shows excellent recognition results and compares them with other methods using the same benchmark datasets. This chapter has been summarized in the paper "Recognizing Multi-Resident Activities in Non-intrusive Sensor-Based Smart Homes by Formal Concept Analysis" recently accepted in the journal Neurocomputing [79].

5.1 SIGNIFICANCE OF RECOGNIZING MULTIPLE RESIDENT ACTIVITIES

The complexity of activity recognition increases when there are multiple residents in a smart environment [200]. Multiple inference rules must be applied to the same sensors at the same time in the same place. Most living environments have more than one resident. For example, family members get together to prepare dinner, or to do housework at the same time. Multi-resident activities can be carried out in an individual, parallel or cooperative manner. Because of the social characteristics of human beings, activities can be coordinated by multiple resi-

dents. In these cases, each sensor reading may involve more than one resident.

Compared with the single-resident activity recognition, recognizing activities in the multi-resident scenario is equally important. People usually live with other family members like their parents, spouse and children. Based on this assumption, ambient living assistance to monitor the multi-resident activities is still necessary. Moreover, due to obvious differences in behavioral patterns, the inferences of single-resident activity recognition cannot be directly applied to the multi-resident one.

5.2 DEVELOPMENT OF MULTI-RESIDENT ACTIVITY RECOGNITION

In the literature, different solutions are proposed to solve the problem of multi-resident activity recognition based on the sensor-based infrastructure design. They can be categorized as data-driven and knowledge-driven models. However, both of them regard graphical models as the first choice to describe the association among activities and to provide a dynamic description of state transitions. Besides, all the related works in the literature are based on a common hypothesis that we know exactly who has triggered which sensors.

5.2.1 DATA-DRIVEN MODELS

Compared with knowledge-driven models, data-driven ones place more emphasis on using large-scale data to drive internal reasoning [201]. Some mainstream solutions are the models based on the statistical and probabilistic theories, such as HMM, CRFs and their variants. They identify all relevant variables in the smart environment and build dynamic probabilistic models that take into account the regularity of probability distribution and the state transition probabilities.

Probabilistic and Statistical Models for Classification

Using historical behaviors and profiles of residents, Crandall and Cook [202] combine an HMM with a Naive Bayesian Classifier (NBC) to identify residents. The system maps sensor events to the residents who triggered them, and then predicts residents' desires and further interacts with them. In [96], authors present a Bayesian network-based probabilistic generative framework to characterize the structural variabilities of complex activities.

Chiang et al. [100] adopt two graphical models, parallel HMM (PHMM) and coupled HMM (CHMM), to identify activities in a multi-resident environment. Besides, they also propose a new dynamic Bayesian network extending CHMM. To model activity patterns, domain knowledge has been added and sensor data has been categorized in the preprocessing. Benmansour et al. [203] develop an HMM-based combined label (CL-HMM) and a linked HMM (LHMM) to compare their performances against the PHMM and CHMM methods. Besides, Wang et al. [204] study a temporal probabilistic model called Factorial Conditional Random Field (FCRF) to model interacting processes in a sensor-based, multi-user scenario.

In [205], Chiang et al. propose a feature-based knowledge transfer framework to extract and transfer knowledge between two different smart environments. They first use a PCA-like method to reformulate input feature sets, and then measure the divergence among the features by Jensen-Shannon divergence. After that, a graph matching algorithm is used to derive the best feature mapping between training and testing datasets. Liu et al. [206] propose another two-stage approach to firstly cluster the training data by K-means using temporal features like start time, end time and approximate duration, and secondly to recognize the activities in each cluster.

In fact, all these methods suffer from the same drawback, they rely on reliable transition

probabilities and emission matrices which depend on large amounts of training data having stable probability distributions. The probabilities should be calculated from a dataset which probability distributions are quite close to the reality. Generally, data-driven models stress on discovering probabilistic or statistical regular patterns over training data. Thus, reliable probability distributions and statistical stability are the most important factors for the final results. However, small-scale training data could not ensure the distributions of training data are infinitely close to the reality. As a consequence, results of probabilistic models will be sensitive to unbalanced distributions.

Models using Association Rules

Chen and Tong explore a two-stage activity recognition method in [207]. It is an extension of the typical HMM and CRF. It uses association rules to learn combined training sequences at the first stage, and then maps test sequences to multi-resident activities at the second stage.

Prossegger and Bouchachia [109] propose an application of incremental decision trees to classify activities in a multi-resident context. Their model allows leaf nodes to be multi-labeled for representing single or multiple classes and incrementally accommodates new instances as well as new activities.

Deep Learning

Fang and Hu [208] built a deep belief network through restricted Boltzmann machines to recognize human activities. They also compare their results with HMM and NBC. They tested their model in their smart home environment and gave an average accuracy as high as 96.53%. In another work, Zhang et al. [209] combine HMM and DNN models to recognize activities. They tested their model on their created dataset and achieved the best average

precision (93.37%) and the best average recall (93.22%) compared with the Gaussian mixture model (GMM) and random forest.

Moreover, for a part of methods like deep learning algorithm, there is no efficient mechanism to organize discovered knowledge. As black-boxes, if the results are not good in some cases, it is hard to explain the reasons and find out the solutions.

For the data-driven approaches, they try to use mathematical theories to establish probabilistic or statistical models based on the analysis of historical data. However, due to the sensitivity of noisy data, they typically have high requirements for data quality and volume to generate a stable and reusable model. Data scarcity may cause underfitting. Additional operations, such as data cleansing, may be applied before processing. Moreover, most of them have insufficient extensibility. If new training data greatly affects the probability distribution or statistical stability of previous training dataset, the entire model needs to be retrained.

5.2.2 KNOWLEDGE-DRIVEN MODELS

Compared with data-driven approaches, knowledge-driven models are easier to be understood and interpreted by researchers and domain experts in knowledge representation. Their classification results are also easier to explain. When their performance is unsatisfactory, it is easier to find the reason for optimization. Instead of retraining models to find the regular patterns by probability and statistical theories, knowledge-driven models can be easily extended by adding homogeneous new domain knowledge.

Ye and Stevenson [210] presented a knowledge-driven approach combining ontologies with semantic matching techniques to recognize daily human activities. The proposed approach works well for the activities having explicit semantics, but it is limited in distinguishing the ones having ambiguous semantic features. Their successive research [188] continues

to focus on recognizing multi-user concurrent activities from an unsegmented continuous sensor sequence. Combining ontological reasoning with statistical methods, the boundaries of different activities are automatically detected by dividing a continuous sensor sequence into partitions.

Alam et al. [211] investigate the challenges of improving the recognition of complex activities in multi-resident smart homes. They propose a loosely-coupled hierarchical dynamic Bayesian network to identify coarse-grained activities using fine-grained atomic actions and sensor data. Because of the prohibitive computation, they have to discover the key spatio-temporal constraints in the activity contexts across users and learned association rules on the basis of Apriori algorithm to prune the state space of the Bayesian network. However, the context correlations and constraints among activities cannot be generated automatically. These constraints well defined the conflicts for extra and inter-user activities in spatial and temporal correlations.

Explicit semantics are essential for most of the knowledge-driven models. The models usually depend on prior knowledge defined by domain experts or an open ontology to infer results. Thus, their maintenance and extension are difficult for the persons who are not familiar with specific domain knowledge. Moreover, their customization usually requires significant artificial costs. Sometimes, they can distinguish activities with great semantic gaps among sensor events, but cannot well recognize two concurrent activities with similar semantic features [188].

5.3 BEHAVIORAL PATTERNS OF MULTIPLE RESIDENT ACTIVITIES

As shown in Section 1.5.4, multi-resident activities are classified in two categories: parallel and cooperative. Therefore, their behavioral patterns can also be divided into two types.

For multi-resident activities, behavioral data belonging to different residents or activities are often interweaved in their executions. This proposition is based on the analysis of the behavioral patterns of these two categories of activities. For parallel activities, two or more behavioral patterns are independent of each other. Since there is no order constraint between different activities, their behavioral data will be interweaved. In addition, almost all sensor events are triggered by only one resident (see the patterns of reading magazine and hanging up clothes in Fig. 5.1). For cooperative activities, due to the interaction and cooperation of residents, most sensor events are triggered by multiple residents at the same time, it is difficult to determine exactly who triggered which sensor event (see the pattern of play checkers in Fig. 5.1).

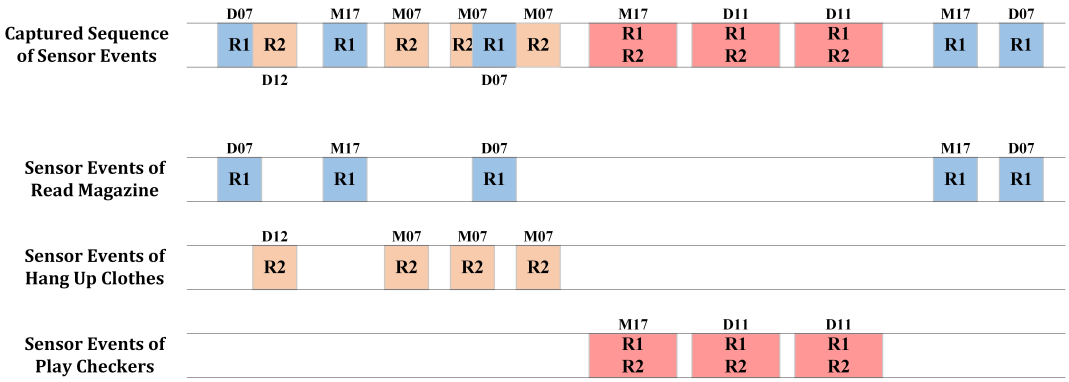


Figure 5.1: Regular behavioral patterns of multi-resident activities in smart homes

In order to simulate the interweaving situation, we create several temporary caches to simulate the long-term intentions of residents (i.e. the activities they are willing to do). As shown in Fig. 5.2, each cache stores the search result of last knowledge retrieval in the Hasse diagram. It indicates the inference about all possible ongoing activities given partially observed sensor events. The system continuously loads subsequently observed sensor events. If a newly captured sensor event makes the new retrieval return the Infimum as the search result, it means that this sensor event is very different from the previously observed data in the ontology. It will be rejected by the current cache (i.e. the current intention) and the cache

itself will rollback. The system will perform a roll polling operation to check if any existing cache can accept it. If all existing caches have triggered the rollback operation, the system will create a new cache to store this sensor event. In other words, a new parallel or cooperative activity may be in progress. In the beginning, there is only one primary null cache for each resident without initial training. As time passes, residents start to interact with the other residents or carry out parallel activities, and more and more caches indicating different inferences are added into the polling.

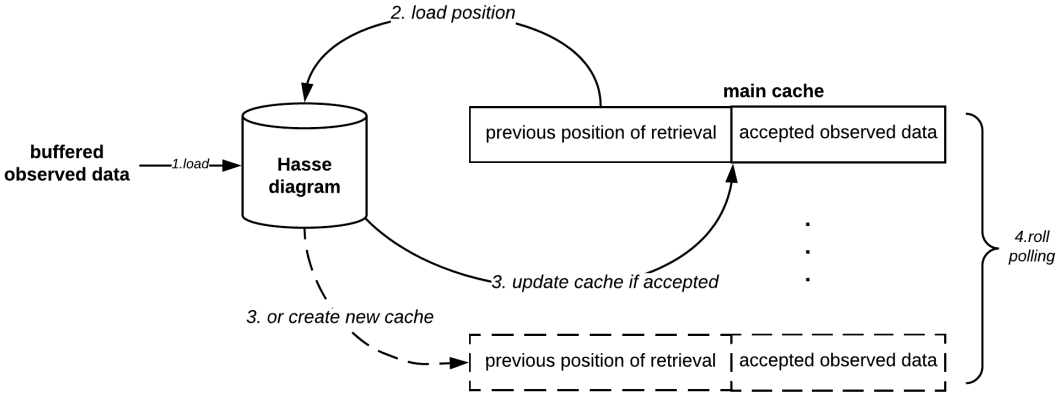


Figure 5.2: Recognition process using Hasse diagram

Once a cache has enough observed sensor events about an activity, the extent of the concept located by the cache determines the final recognition result.

Fig. 5.4 gives a lattice of multi-resident activity recognition obtained from the binary matrix shown in Fig. 5.3. Activities will be considered as recognized when there is only one object in the extent of the final located concept, such as n_{13}, n_{16} and n_{20}, or an object have never shown in its successive concepts, like g_4 in n_{14} could not be found in its subconcept n_{18}.

Suppose $\alpha = \{\text{M09} \prec \text{M06} \prec \text{M17} \prec \text{D13} \prec \text{D07} \prec \text{M13} \prec \text{M07}\}$ is a sequence indicating multi-resident activities g_{13} and g_{14}. Table. 5.1 illustrates the recognition process. The symbol $\curvearrowright$ represents a transition of inference and $\circlearrowleft_{Infimum}$ represents a rollback operation from the Infimum. At round 2, the bottom-up search ensures that node 14 is located, not node 18. At

Simplified CASAS Activities [164]		m_1: D07	m_2: D11	m_3: D12	m_4: D13	m_5: D14	m_6: I04	m_7: M04	m_8: M06	m_9: M07	m_{10}: M09	m_{11}: M13	m_{12}: M17	m_{13}: M23
Fill medication dispenser	g_1	×					×						×	
Hang up clothes	g_2			×						×	×			×
Move furniture	g_3							×						×
Read magazine	g_4								×	×	×			
Water plants	g_5								×	×				
Sweep floor	g_6			×					×	×	×	×	×	×
Play checkers	g_7		×										×	
Prepare dinner	g_8										×	×		
Set table	g_9			×							×			×
Read magazine	g_{10}	×											×	
Pay bills	g_{11}					×							×	
Pack picnic food	g_{12}								×	×				
Pack picnic food	$g_{12'}$								×	×				
Retrieve dishes	g_{13}				×				×	×	×	×		
Retrieve dishes	$g_{13'}$								×	×	×	×		
Retrieve dishes	$g_{13''}$								×	×	×	×		
Pack picnic supplies	g_{14}	×									×		×	
Pack and bring supplies	g_{15}			×		×				×	×		×	×

Figure 5.3: Matrix for illustrating multi-resident activity recognition

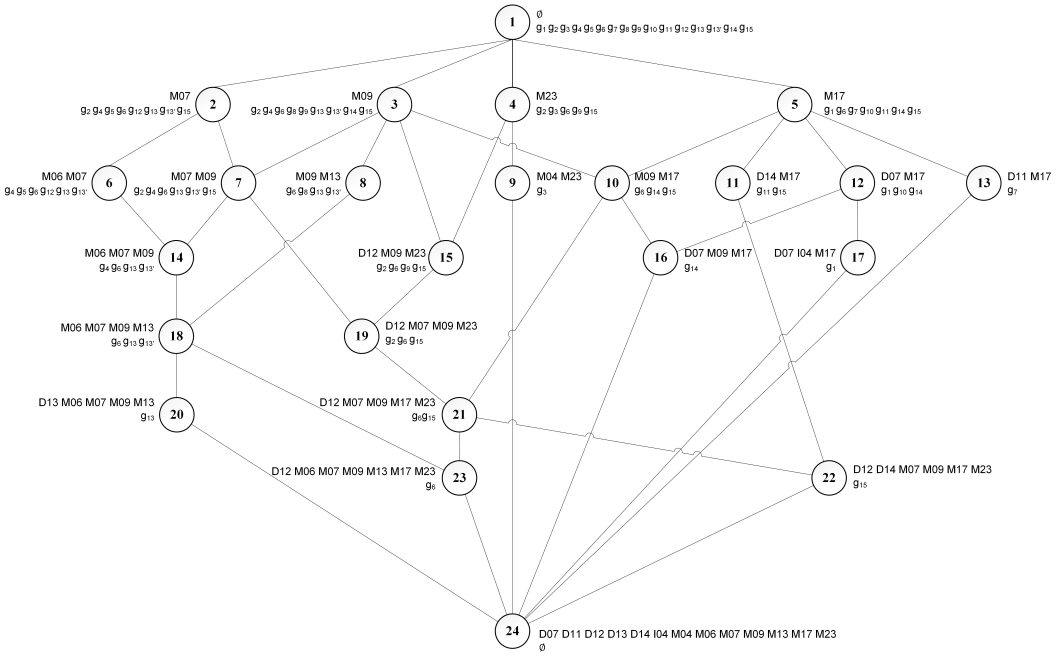


Figure 5.4: Lattice of multi-resident activity recognition

Table 5.1: Example of Inferring for Multi-resident Activity Recognition

Round	Observed Data α	Located Topmost Concept	Predictive Activities
1	{M09}	node 3 {$g_2g_4g_6g_8g_9g_{13}g_{13'}g_{14}g_{15}$,**M09**}	$g_2g_4g_6g_8$ $g_9g_{13}g_{13'}g_{14}g_{15}$
2	{M09M06}	node 3 $\curvearrowright$ node 14 {$g_4g_6g_{13}g_{13'}$,**M06**M07**M09**}	$g_4g_6g_{13}g_{13'}$
3	{M09M06M17}	node 14 $\circlearrowleft_{Infimum}$ {$g_4g_6g_{13}g_{13'}$,**M06**M07**M09**} node 5 {$g_1g_6g_7g_{10}g_{11}g_{14}g_{15}$,**M17**}	$g_4g_6g_{13}g_{13'}$ $g_1g_6g_7g_{10}g_{11}g_{14}g_{15}$
4	{M09M06M17D13}	node 14 $\curvearrowright$ node 20 {g_{13},**D13M06**M07**M09**M13} node 5 $\circlearrowleft_{Infimum}$ {$g_1g_6g_7g_{10}g_{11}g_{14}g_{15}$,**M17**}	g_{13} $g_1g_6g_7g_{10}g_{11}g_{14}g_{15}$
5	{M09M06M17D13 D07}	node 20 $\circlearrowleft_{Infimum}$ {g_{13},**D13M06**M07**M09**M13} node 5 $\curvearrowright$ node 12 {$g_1g_{10}g_{14}$,**D07M17**}	g_{13} $g_1g_{10}g_{14}$
6	{M09M06M17D13 D07M13}	node 20 $\circlearrowleft$ {g_{13},**D13M06**M07**M09M13**} node 12 $\circlearrowleft_{Infimum}$ {$g_1g_{10}g_{14}$,**D07M17**}	g_{13} $g_1g_{10}g_{14}$
7	{M09M06M17D13 D07M13M07}	node 20 $\circlearrowleft$ {g_{13},**D13M06M07M09M13**} node 12 $\circlearrowleft_{Infimum}$ {$g_1g_{10}g_{14}$,**D07M17**}	g_{13} $g_1g_{10}g_{14}$
8	{**M09**M06M17D13 D07M13M07}	node 20 $\circlearrowleft$ {g_{13},**D13M06M07M09M13**} node 12 $\curvearrowright$ node 16 {g_{14},**D07M09M17**}	g_{13} g_{14}

round 3, when M17 is observed, {M09M23M17} is excluded by previously located node 14 because there is no subconcept containing it except the Infimum. Thus, after the roll polling, a new cache is created to store M17. At round 8, when there is no more observable sensor event, the missing data M09 in the second cache will be automatically completed by the previous one observed at round 1.

TRANSITION MATRIX

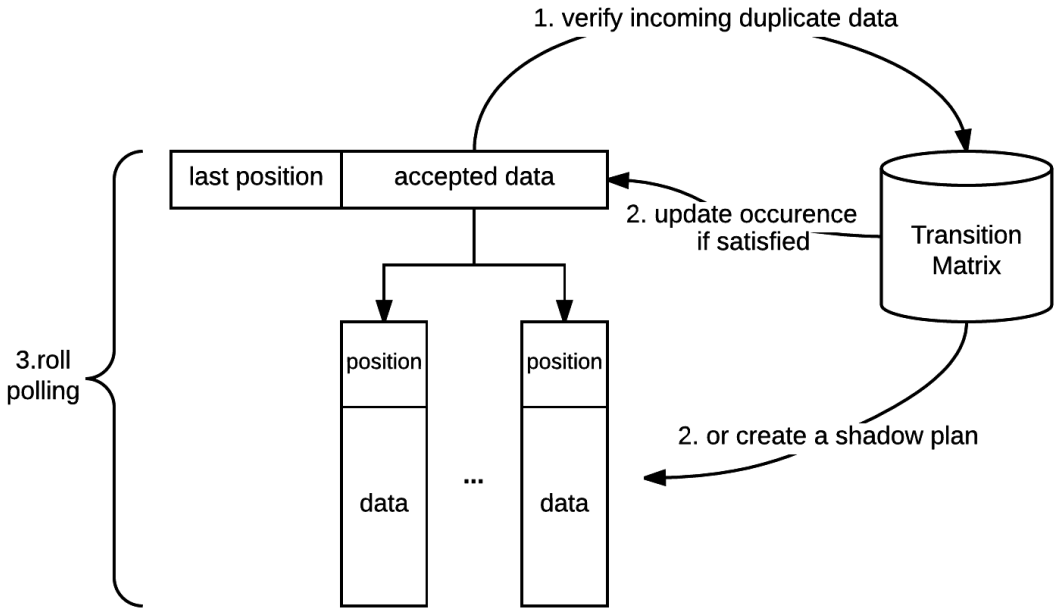


Figure 5.5: Identifying highly similar activities by transition matrix

Besides the FCA-based graphical model, for $|G|$ indexed activities, we define a transition matrix T_i for each of them to record the context information among sensor data (see Fig. 5.6). The objective is to distinguish similar or multi-level inheritance patterns. For instance, g_1 and g_2 are two highly similar activities, and the sensor events of g_1 are the subset of the ones of g_2. If they are performed by two residents at the same time, it is hard to correctly identify the real ongoing activities in the duplicate data without considering context information. Fortunately, transition matrices provide a feasible solution because even two similar patterns

having exactly the same set of sensor data, the transition states among sensor data will be different.

Each T_i is a $N \times N$ square matrix where $N = |M| + 2$ and $|M|$ is the cardinality of indexed sensor events. Its columns or rows indicate an array $\{start, m_1, ..., m_j, ..., m_{|M|}, end\}$ where *start* and *end* are the boundary labels appearing in the training data.

For example, in the training phase, if a sequence describing activity g_5 is $\{start, m_8, m_9, m_9, end\}$, the elements $a_{0,8}, a_{8,9}, a_{9,9}$ and $a_{9,N-1}$ in the matrix T_5 should be updated.

$$T_1 = \begin{pmatrix} a_{00} & 2 & \cdots & 4 \\ 0 & a_{11} & \cdots & 5 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 5 & \cdots & a_{N-1,N-1} \end{pmatrix} \quad ... \quad ... \quad T_{|G|} = \begin{pmatrix} 0 & 0 & \cdots & 20 \\ 7 & 6 & \cdots & 11 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 2 & \cdots & 0 \end{pmatrix}$$

Figure 5.6: Transition matrices of different activities

In fact, duplicate data indicating repeated sensor events comes from frequent sampling or repeated triggering. In the recognition phase, when a new sensor event is repetitive, it will be only checked by the transition matrix. This is because duplicate sensor data will always be accepted by the caches containing it.

For example, because of few sensors deployed in an apartment, g_4, g_5 are two totally different activities, but they have similar sensor data. $g'_5 = \{m_8, m_9\}$ and $g'_4 = \{m_8, m_9, m_{10}\}$, so $g'_5 \subset g'_4$. As shown in Fig. 5.5, suppose the observed data are $\{m_9 \prec m_8 \prec m_{10} \prec m_8 \prec m_9\}$. Duplicated data m_8, m_9 will be detected after being observed (see step 1 in the figure). Because of no clear boundary, we could not simply justify that the duplicated m_8 belong to g_4, so we check the transition matrices to verify the transition $a_{10,8}$ in T_4. A cache will be created to store the duplicated data (see step 2) if and only if a_{ij} is lower than a threshold for any pattern of g_4. A roll polling operation (see step 3) will check each cache when a new duplicated

Table 5.2: Comparison of Recognition Accuracies

Methods	**NBC [76]**	**HMM [76]**	**CRF [103]**	**TSM-HMM [207]**	**TSM-CRF [207]**	**FCA**
Accuracy	63.27	60.90	58.41	75.77	75.38	**94.26**

data is observed.

5.4 EXPERIMENTS ABOUT MULTI-RESIDENT AR

In this section, we use a benchmark dataset to evaluate the performance of our models. To compare the results with other models under the same measures, the following experiments are evaluated by both leave-one-out (LOOCV) and 3-fold cross-validations [212]. The benchmark dataset adopted in the experiments is the CASAS Kyoto-4 multi-resident dataset (see details in Appendix A).

RESULTS AND DISCUSSION

Cooperative activities could also be called joint activities if and only if at the same time, both of resident perform the same cooperative activity. The cooperative could be regarded as well recognized when both of recognitions are correct. We compare our results with other references using the same dataset [76, 100, 103, 133, 164, 203, 207]. The total time cost of recognition is about 4.0756 seconds.

First of all, we compare each activity recognition result with [164] and show the results in Fig. 5.7. Our results also surpass the results shown in Fig. 9 of [207]. The results are based on the same 3-fold cross-validation. As described in [164], HMM-1 is a single HMM model implemented for both residents. For HMM-2, an HMM model is built for each resident. In the results, we could see that most of the recognition are excellent except for two activities: water plants (activity 5) and picnic food (activity 12). The reason has been indicated in [164]

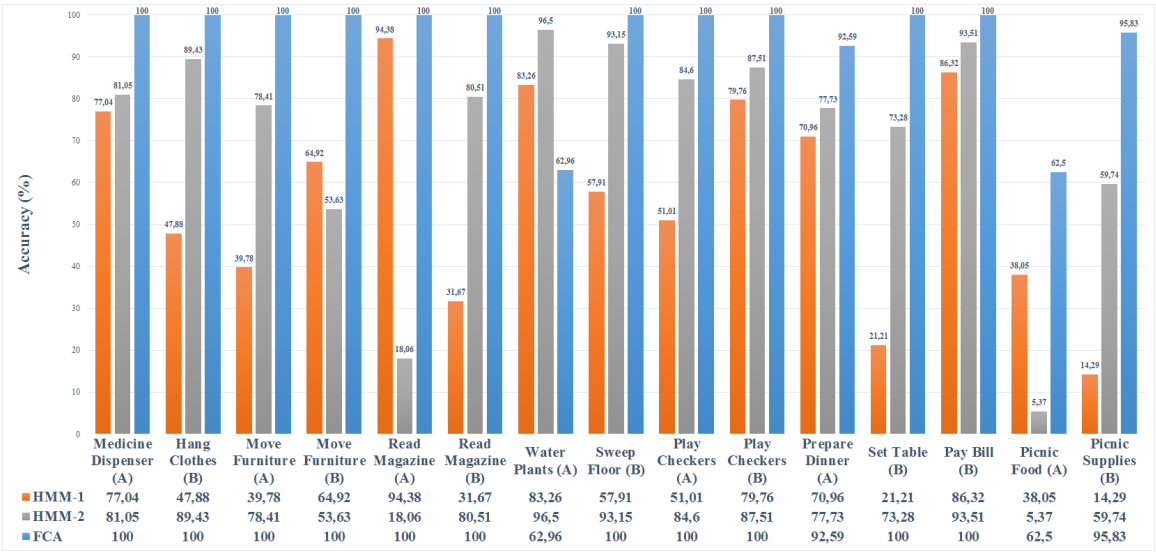


Figure 5.7: Performance of recognizing each multi-resident activity

that the activities with insufficient sensor events will be difficult to differentiate from other activities. In the view of FCA models, the distinguishable ability of a sensor is negatively correlated with the number of shared activities. We also compare our results with other classical algorithms, including naive Bayes classifier (NBC), HMM, CRF and their variants. The results are summarized in Table. 5.2.

After that, we compare our results of independent parallel activity recognition with another reference [203] (see Table. 5.3). In this comparison, we use the leave-one-out method to evaluate the performance. The results are classified by different residents and the types of activities. According to the results under different metrics, we could find that our FCA-based method outperforms the other HMM-based methods. In the part of recognizing joint activities, the FCA-based method also has excellent performance (see Table. 5.4[1]). Although the models based on TSM-HMM and TSM-CRF have better accuracies, our model has more stable performance and obtains better results in terms of F-measure score.

[1]the methods marked by † use the leave-one-out cross-validation, the one marked by * uses the 5-fold cross-validation, and the ones marked by ‡ use the 10-fold cross-validation.

Table 5.3: Comparison of Results Categorized by Different Activity Types and Residents

Approach	Residents	Accuracy	Individual	Cooperative	Average	Precision	Recall	F-measure
CL-HMM [203]	R1	91.33±8.15	91.11±8.41	92.76±21.87	91.78±11.68	92.25±6.99	92.54±6.59	92.38±6.71
	R2	91.61±7.87	92.37±6.64	91.22±11.07	91.8±6.96	91.12±7.43	91.7±7.99	91.35±7.5
	Average	91.47±7.5	91.74±6.07	92.33±11.24	91.91±7.3	91.68±6.1	92.12±6.42	91.89±6.17
LHMM [203]	R1	92.36±8.48	93.86±7.89	65.19±43.57	81.4±21.32	93.25±7.46	91.93±7.56	92.48±6.98
	R2	94.17±5.05	90.8±7.52	96.42±5.48	93.61±5.12	93.9±5.44	93.43±6.4	93.61±5.63
	Average	93.27±6.21	92.33±6.95	82.77±21.3	87.53±11.22	93.58±5.41	92.68±6.18	93.1±5.62
FCA	R1	97.25±7.94	97.25±7.94	96.26±10.17	96.75±0.49	98.90±5.49	98.35±6.04	98.42±4.60
	R2	94.71±8.61	90.38±15.6	99.03±4.81	94.70±4.32	97.05±6.42	97.53±6.15	97.07±4.85
	Average	**95.98±1.27**	**93.81±3.43**	**97.26±2.11**	**95.53±1.73**	**97.97±0.93**	**97.94±0.41**	**97.75±0.68**

Table 5.4: Comparison of Joint Activities Results

Methods	Accuracy	F-measure
FCA†	92.86±12.54	95.10± 9.32
LHMM† [203]	88.23±10.23	80.3±9.84
TSM-HMM∗ [207]	97.40	80.96
TSM-CRF∗ [207]	97.25	79.98
CHMM+Interaction vertices† [100]	78.26	-
Random Forest‡ [133]	88.60	-
SVM‡ [133]	83.70	-
Naive Bayes‡ [133]	81.20	-

The proposed FCA-based model has better capacity than the previous version [57] while identifying similar activities. This is because the newly added transition matrices can be useful when two patterns are highly similar. On the premise of keeping the context information, the FCA-based model with the transition matrices reduces the influence of imbalanced distributions of training data and enforce the impact of internal regulars of patterns. Even two patterns consist of the same sensors events, their sequential contexts would be different. It means that for a sensor event in two highly similar patterns, its previous and successive sensor events will not always be the same ones. Compared with two HMM methods in [164], the overall performance of activity recognition has increased 37.02% and 22.76%. In the LOOCV experiments, our methods improve 4.51% and 2.71% accuracies.

Besides, the FCA-based model simulates the real scenarios that include the interweaving patterns. There is no explicit segmentation to reveal the beginning and end of a sequence

indicating an activity. To determine a sensor data belongs to which patterns, the conventional HMM methods use a series of probabilities such as joint and transition probabilities to judge the affiliations of a sensor data. If a posteriori probability is lower than a threshold, then the systems will judge that it belongs to another pattern. In our method, we do not directly use probability to evaluate the confidential degrees, however, we make the decision from the semantic parts. If a sensor data has great semantic gaps with the others, then it will be judged as one part of another pattern.

Comparing with the HMM methods, the FCA-based models can give a scope of possible ongoing activities and refine the results by the RMSD assessment. However, it works well only for the independent activities performed in parallel. This is because one person's activities will be affected by another one, especially for the cooperative activities. Thus, the RMSD assessment has to wait for enough data to infer the most reliable recognition in the case of cooperative activity recognition.

5.5 CONCLUSION

In this chapter, we address the problem of multi-resident activity recognition in non-intrusive sensor-based smart homes. Using the lattice search strategy, we can automatically and incrementally infer the most possible ongoing activities given a part of observed data. The incremental knowledge retrieval makes the static formal lattice containing ontological knowledge become dynamic. The combination of the graphical knowledge base and the transition information make the FCA-based model reduce the dependency of stable data distribution in the training data. The experimental results show that the recognition accuracy outperforms traditional statistical or probabilistic models. Due to the limited ability of multi-class classification or the complexity to construct a knowledge base, to the best of our knowledge,

there are few available comparative results of the other data mining approaches such as decision trees, association rules or knowledge-driven models solving the multi-resident activity recognition on the same benchmark dataset.

CHAPTER 6

INCREMENTAL LEARNING

In this chapter, we propose a functional improvement of current models associated with incremental learning. The new design for incrementally constructing concept lattice enables our systems to meet the scalability requirement about integrating new training data with new features into constructed models. The chapter is organized as follows. Section 6.1 gives a brief overview of incremental learning in data mining. Section 6.2 emphasizes the significance of incremental learning, especially for the applications in smart environments. Section 6.3 outlines a few studies about incremental learning in activity recognition in smart environments. Section 6.4 details how to use the incremental learning algorithm to enhance the existing FCA-based models. The experimental results are shown in Section 6.5. Brief advantages and disadvantages of our incremental improvement in Section 6.6. The work presented in this chapter will be submitted soon as a journal paper. [80].

6.1 INCREMENTAL LEARNING IN DATA MINING

In fact, many successful machine learning and data mining methods are based on a common assumption that the training and future data must be in the same feature space and have the same distribution [213]. When the distribution or feature space is changed, most statistical

or probabilistic models need to be rebuilt from scratch using newly collected training data. However, this assumption is not suitable for AmI applications.

Incremental learning is usually a higher level requirement with limited memory resources for existing algorithms in the part of adaptation based on a constantly arriving data stream [214]. Non-incremental learning approaches are usually static, which means they first load and store all the available data in memory for training, and then use their unchangeable trained models for prediction, classification or pattern recognition. Most of them can not achieve self-adaption to automatically include new data or features. When non-incremental learning models want to improve their performances with new training data, in most instances, they have to be reconstructed, in order to adapt to new training entities or to bring in new features. However, the time consumption of reconstruction increases with the augmented amount of training data. Without an effective solution, frequent and time-consuming model construction is intolerable for most smart environment applications.

6.2 SIGNIFICANCE OF INCREMENTAL LEARNING FOR AR

Incremental learning is meaningful for the smart environment applications. Although most activity recognition systems can train their models from historical data, the gathered patterns cannot cover all possible patterns. Moreover, different residents may perform the same activities in different ways. To ensure stable recognition accuracy, systems should learn additional information from new training data to improve the accuracy and robustness. Sometimes, the design sensor layout of a smart environment will be expanded by new sensors or new interesting activities. We wish that our system could automatically self-adapt these changes and only update the trained model with these new data.

The scalability of an activity recognition model in terms of integrating new data is one of

the most important requirements for sensor-based smart environments. This is because a smart environment keeps on considering and introducing new situations and the recognition model need to constantly update itself to update these new changes. Moreover, if the current layout of the smart environment is not suitable enough to identify all the activities of interest, new and specific sensors can be deployed to enhance the ability to distinguish misclassified activities.

6.3 STATE OF ARTS ABOUT INCREMENTAL LEARNING APPLIED ON AR

Considering the complexity, flexibility, and variability of the situations when recognizing activities in smart environments, different methods and architectures have been proposed by the scientific communities. Their common practice is to make appropriate changes based on classic algorithms such as decision tree, random forests, naive Bayes and neural networks.

Lu et al. [215] proposed a hybrid user-assisted incremental model adaptation (HUIMA) that reconfigures previously learned activity models within a dynamic environment. HUIMA consists of an automatic mechanism for simplifying the unseen data annotation task, and an enhanced Dynamic Bayesian Network model for incrementally updating the models by new annotated data. They tested their method with their own dataset. However, the correctness of data annotation cannot be always guaranteed. Thus, another data-annotation wizard with human interventions was used in case of ambiguity. However, it will decrease the self-adaptation of the model.

Zhao et al. [216] proposed a class incremental extreme learning machine (CIELM). It was built on the basis of the ELM (Extreme Learning Machine), a neural network algorithm [217] and was tested using their own datasets. In order to implement this non-incremental learning algorithm, CIELM incrementally updates its model using individual samples or data chunks

with new labels. Their performance is slightly worse than the batch learning method because of the trade-off between optimization and restricted resources. Wang et al. [218] combined probabilistic neural networks (PNN) and an adjustable fuzzy clustering algorithm (AFC) to build an incremental learning method for sensor-based human activity recognition. Their process of adding or removing an activity is almost independent of the pattern neurons of other activities. They tested their method with their own dataset. However, the generalization capability of the proposed method was limited by their subject-independent training.

Hu et al. [219] proposed an incremental growing mechanism of the decision tree and a novel splitting strategy to construct Class Incremental Random Forests (CIRF). Their solution can tackle the dynamic changes in activity recognition. However, the CIRF algorithm requires maintaining large-scale training samples all the time.

Because the ID5R incremental decision tree algorithm [220] does not support to handle numeric variables, multi-class classification tasks, or missing values, an extension of ID5R which incrementally augments leaf nodes and allows them to be multi-labeled is proposed in [109]. Because of the neglect of important sequence information, complex activities having complicated relations need a better modeling than the straight and native application of decision tree. Their method was evaluated using ARAS dataset [1]. However, based on the outcome received from the experiments, the efficiency of multi-labeling and the use of counts has to be further analyzed. A loosely-coupled Hierarchical Dynamic Bayesian Network (HDBN) is proposed in [211] to exploit the spatiotemporal relationships across the activities of residents. Their method was evaluated using their own dataset. However, a state space pruning should be performed before employing the model for complex activity recognition.

In brief, the incremental designs of most of the previous studies are limited by their algo-

[1] https://www.cmpe.boun.edu.tr/aras/

rithms, without considering the complicated situations and frequent layout updates in smart environments. Thus, we propose an incremental learning approach which is independent and only focuses on incremental knowledge management to integrate new data and new features.

6.4 NEW INCREMENTAL ALGORITHM FOR CONSTRUCTING CONCEPT LATTICE

As mentioned in Section 3.3, different lattice construction algorithms have quite different performances. Our incremental method is based on an algorithm proposed by Valtchev and Missaoui [165]. This algorithm is an efficient lattice building approach which is more effective than many other classic incremental and batch ones. It investigates the incremental updating of the constructed lattice by a set of previously unseen individuals. Its basic idea is to recognize the lattice parts requiring restructuring and to carry out the reconstructing at a minimal cost. Thus, two categories of formal concepts must be identified: those which changed their extent and those which remain the same. Concepts in the latter category are further validated to see whether they produce new concepts. However, its implementation [221] does not consider about updating new data with new features. In other words, the scenario about adding new sensors in a smart environment has not been considered.

Thus, in Algorithm 5, we illustrate the optimization of incrementally updating a constructed lattice. As defined in Section 1.5.2, the input data is a collection of labeled sequences of sensor events. To achieve the incremental manner, as an extension, the space of features is incrementally updated (lines 2-3). The algorithm initializes a lattice if it does not exist before (Lines 4-8). For each item in the new training dataset, an iteration of the lattice verifies whether the iterated concept should be updated, created or ignored (lines 9-26). We optimize and simplify the logic of an internal function called *minAdjacentParent*, described

Algorithm 5: Optimized Valtchev Algorithm

Data: A constructed lattice $\underline{\mathfrak{B}}$, a training dataset
$\mathcal{D} = (\mathcal{X}, \mathcal{Y}) = \{(x^{(0)}, y^{(0)}), (x^{(1)}, y^{(1)}), ..., (x^{(m)}, y^{(m)})\}$, the space of features $\mathcal{M}$.
Result: Updated lattice $\underline{\mathfrak{B}}^{+}$.

```
1  begin
2      if M ∩ X ≠ X then
3          M = M ∪ X
4      if B = ∅ then
5          supremum = newConcept(y^(0), x^(0))
6          infimum = newConcept(∅, M)
7          createLink(supremum, infimum)
8      modified = ∅
9      foreach (x^(i), y^(i)) ∈ X do
10         foreach c ∈ B do
11             if int(c) ⊆ y^(i) then
12                 ext(c) = ext(c) ∪ y^(i), mark it as modified
13             else
14                 n = newConcept(ext(c) ∪ y^(i), int(c) ∩ x^(i))
15                 m = minAdjacentParent(n,c)
16                 createLink(m, n)
17                 if ext(m) has been modified then
18                     dropLink(m, c)
19                 end
20                 createLink(n, c)
21                 if c==supremum then
22                     supremum=n
23                 end
24             end
25         end
26     end
27 end
```

Algorithm 6: Discover Adjacent Super Concept

Function *minAdjacentParent(m,c)*
    Data: Concept m to compare, current concept c
    Result: Adjacent parent of c having minimal superset of ext(m)
1 *parents = sorted(parents(c))*
2 **foreach** $p \in parents$ **do**
3   **if** $ext(m) == ext(m) \cap ext(p)$ **then**
4   *return p*
5   **end**
6 **end**

in Algorithm 6. The updated lattice $\underline{\mathfrak{B}}^{+}$ normally exists in the memory and can be serialized in a database or in a disk file.

APPLICATIONS OF FCA-BASED MODELS

An overview of the FCA-based activity recognition framework is given in Fig. 6.1. The framework is divided into two individual modules. One module focuses on incremental learning, and the other one focuses on recognizing activities in smart environments. In the recognition module, there are several ad-hoc inference retrieval strategies for different scenarios mentioned in Chapters 3 to 5.

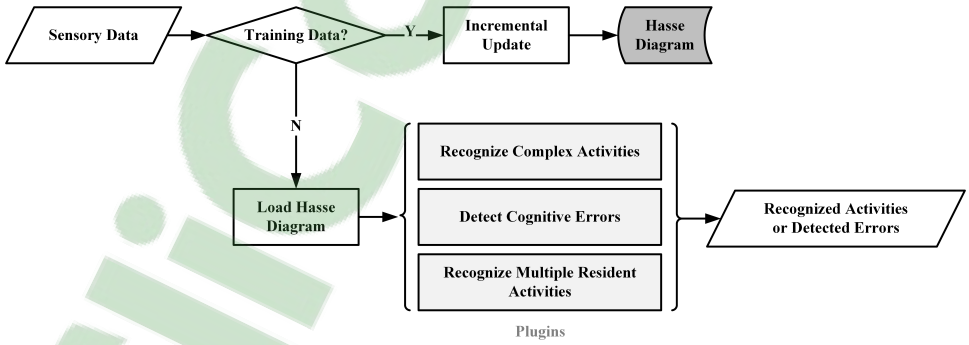


Figure 6.1: Recognizing activities in smart environments

When new sensor data is captured by the system, first of all, it will be judged whether it is a training data. If yes, it will be used for updating current lattice. Otherwise, it will be

processed by the basic, composite or multi-resident activity recognition module as well as error detectors to recognize activities or detect abnormal errors.

6.5 EXPERIMENTS

The experiments are carried out on a desktop with an Intel Core i7-6700 CPU and 8GB of RAM running Windows 10. The benchmark dataset used in the experiment is the Kyoto-4 dataset [2], described in Appendix A.

The final binary matrix consists of 270 rows and 73 columns, and generates a lattice with 29,118 formal concepts. It is worth mentioning that both incremental and non-incremental lattice construction algorithms using the same training dataset will produce the totally same lattice without any difference. Thus, their recognition results are also the same, because lattice construction and recognition depend on two independent modules.

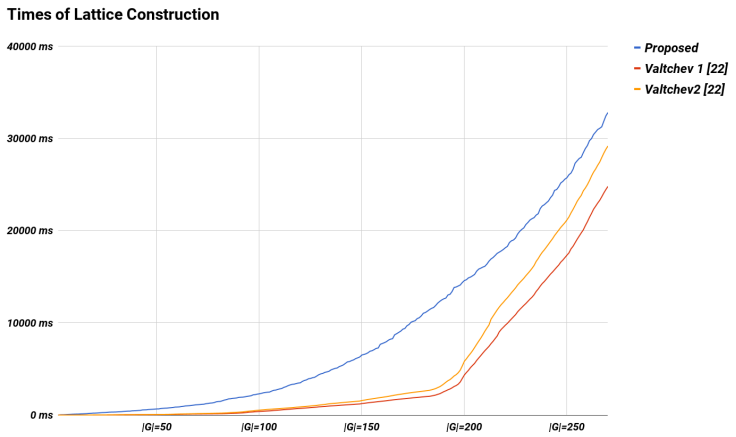


Figure 6.2: Time of lattice construction

[2] http://ailab.wsu.edu/casas/datasets/

6.5.1 COMPARISONS ABOUT LATTICE CONSTRUCTION

We compare our results with both non-incremental and incremental algorithms published in [153, 167, 169, 221], However, Godin and Norris algorithms [221] cannot handle the training data having multi-level inheritance [3] [57]. Thus, Figure. 6.2 presents the time of lattice construction of three incremental algorithms at different stages. The time consumption of lattice construction increases while the amount of target classes ($|G|$) grows. However, almost all the non-incremental algorithms load and generate the lattice by learning on the entire training dataset at once. Once a lattice is constructed, it can not be modified by any new training data. Thus, these algorithms do not update the lattice one by one. Compared with the other two incremental algorithms, ours sacrifices the efficiency in speed in exchange for the functional expansion to incrementally update new data with new features.

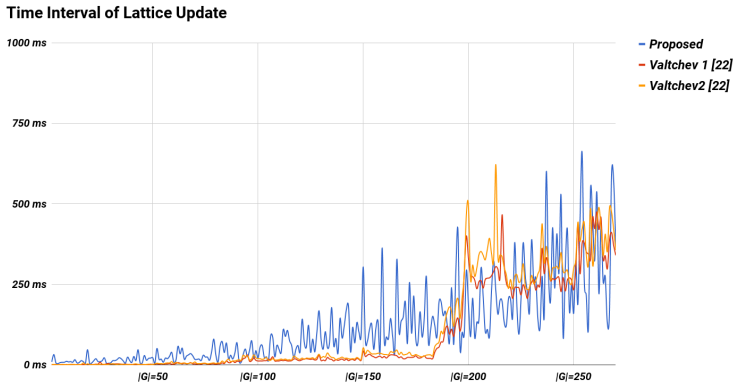


Figure 6.3: Time interval for each incremental update

The time intervals of all the iterations are shown in Fig. 6.3. As shown in this figure, in the beginning, the time of each update tends to be stable, and later, the time intervals begin to fluctuate. This is because when the lattice construction has reached a certain dimension, the complexity of updating becomes uncertain, largely depending on the relationship between

[3] For two activities g_1 and g_2, their features having $g_1' \subseteq g_2'$ or $g_2' \subseteq g_1'$

Table 6.1: Comparison of Results of Lattice Construction by Different Algorithms

Algorithm	Type	Time for Lattice Construction
Bordat [167]	Non-incremental	49.625s
Ganter [169]	Non-incremental	180.331s
Fast [153]	Non-incremental	9216.659s
Valtchev 1 [221]	Incremental	25.449s
Valtchev 2 [221]	Incremental	29.598s
Proposed	Incremental	33.664s

the new data and the old one.

In table 6.1, a comparison of different lattice construction algorithms including incremental and non-incremental ones is given. As shown, incremental algorithms construct faster than the non-incremental ones. This provides us a powerful practical basis for using incremental algorithms.

Our extension has paid an extra cost in speed. However, instead of using all the data to retrain the entire model, new features like sensor events and new activities are allowed to incrementally update constructed lattice.

6.5.2 COMPARISONS ABOUT ACTIVITY RECOGNITION

First of all, we compare our model with another incremental one [211] and show the results in Table. 6.2. Then, we also compare each activity recognition result with the non-incremental method described in [164] and the results are shown in Fig. 6.4. The comparison is based on the same 3-fold cross-validation. In the results, we could see that most of the recognition results are excellent except for two activities: water plants (activity 5) and picnic food (activity 12). The reason has been indicated in [164] that the activities with insufficient sensor events are difficult to be distinguished from other activities and lead to lower recognition results. In the view of FCA models, the distinguishable ability of a sensor is negatively correlated with

Table 6.2: Comparison of F1-score of Two Incremental Models

Activity ID	Activity	CACE [211]	FCA
1	Fill medication dispenser	0.932	1.0
2	Hang up clothes	0.965	1.0
3	Move furniture	0.973	1.0
4	Read magazine	0.607	1.0
5	Water plants	0.593	0.672
6	Sweep floor	0.955	1.0
7	Play checkers	0.945	1.0
8	Prepare dinner	0.976	0.958
9	Set table	0.943	1.0
10	Read magazine	0.923	1.0
11	Pay bills	0.98	1.0
12	Pack picnic food	0.955	0.724
13	Retrieve dishes	0.979	0.978
14	Pack picnic supplies	0.558	0.978
15	Pack and bring supplies	0.615	0.978
	Overall Precision	0.965	0.989
	Overall Recall	0.945	0.948
	Overall F1-score	0.936	0.954

the number of shared activity.

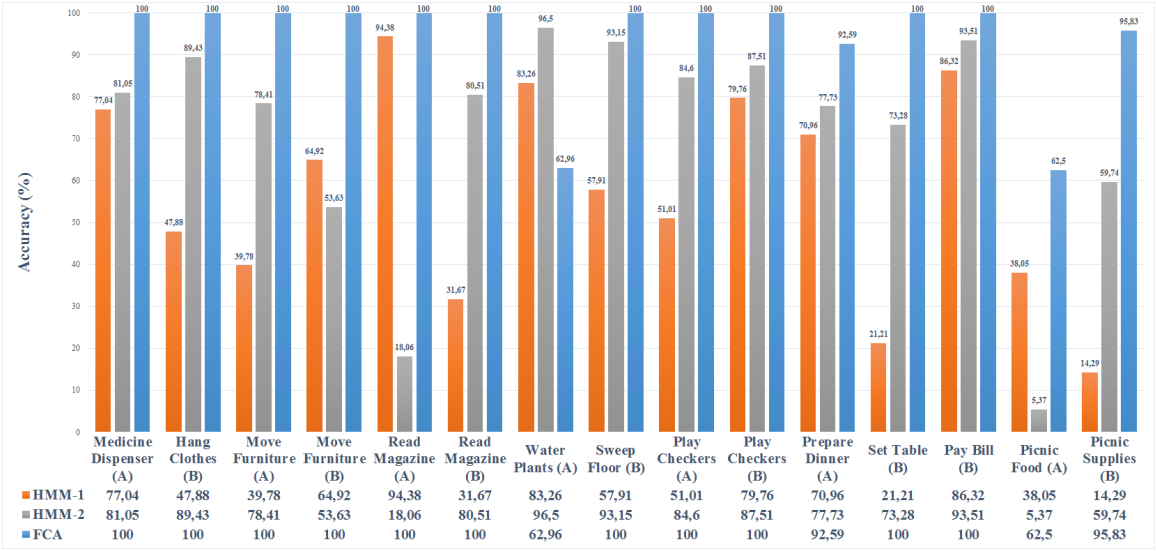


Figure 6.4: Performance of recognizing each multi-resident activity using both non-incremental and incremental methods

To solve this problem, we use new training data with new features to help to distinguish the activities g_5 and g_{12}. We find that activity g_5 must interact with the watering can that is located in the hallway closet, but activity g_{12} does not. Thus, we can add an RFID tag or other sensors to monitor the moving states (e.g. *iCAN_ON* and *iCAN_OFF*) of the watering can. Likewise, for activity g_{12}, food has to be gathered from the kitchen cupboard. Thus, we can monitor the open/close states (e.g. *iCupbord_ON*, *iCupbord_OFF*) of the kitchen cupboard. In the experiment, simulative sequences with four new sensor events, *iCupbord_ON*, *iCupbord_OFF*, *iCAN_ON* and *iCAN_OFF*, are incrementally introduced into the constructed lattice for the enhancement of knowledge base (see Fig. 6.5).

As can be seen from Table. 6.3, the ability distinguishing activities g_5 and g_{12} is greatly improved by new training data with new sensor events. Moreover, the enhancement introducing new features into existing lattice does not reduce the overall recognition rates.

```
M08_ON 5_new
M07_OFF 5_new
M07_ON 5_new
M08_OFF 5_new
M07_OFF 5_new
iCAN_ON 5_new
M07_OFF 5_new
M07_ON 5_new
iCAN_OFF 5_new
```

(a) new training data of activity g_5 with new features iCAN_OFF and iCAN_ON

```
M06_ON 12_new
M07_ON 12_new
M08_ON 12_new
M06_OFF 12_new
M07_OFF 12_new
M08_OFF 12_new
M06_ON 12_new
iCupbord_ON 12_new
iCupbord_OFF 12_new
```

(b) new training data of activity g_{12} with new features iCupbord_OFF and iCupbord_ON

Figure 6.5: Constructed lattice enhanced by new data with new features

Table 6.3: Recognition Results Before and After Incremental Updates with New Features

Activity 5	Before	After	Activity 12	Before	After
Accuracy	0.630	0.889	Accuracy	0.625	0.847
F1-score	0.692	0.933	F1-score	0.724	0.911
Overall Precision	0.989	0.989	-	-	-
Overall Recall	0.948	0.978	-	-	-
Overall F1-score	0.954	0.981	-	-	-

6.6 CONCLUSION

In this chapter, we proposed an activity recognition method based on formal concept analysis in an incremental manner. Its performance is better than most of non-incremental FCA lattice construction algorithms. Moreover, the incremental mechanism for updating the constructed knowledge base is very suitable for sensor-based smart environments. The update does not need to use previous training data and directly modify the constructed lattice by new data. At the same time, the independence of updating and recognition of FCA models could fast updating model without interruption. It will decrease the burden of system maintenance and knowledge base updating.

CHAPTER 7

GENERAL CONCLUSION

As the product of cross-border integration, AI technique plays a more and more important role in the era of big data. Various fields of our society begin to change from digital and interconnected to intelligent. Big data analysis and IoT technology connect all available physical resources to realize the interconnection of information. In this context, they stimulate the exploration, design, and development of AmI applications, especially the future intelligent living environments called smart homes, in order to provide appropriate assistance for their residents and make them live securely.

As one of the most important prerequisites, recognizing human activities is essential for smart homes to understand human behaviors and further predict their objectives. However, it is always a complicated research due to massive data and various categories of behavioral patterns in continuous, composite or multi-resident ways. Thus, we prefer to use the data mining technique to help us recognize activities from sequential and temporal data. The tasks consist of knowledge representation and management, activity recognition and prediction, as well as anomaly detection for preventing potential threats from daily lives.

REALIZATION OF THE OBJECTIVES

Knowledge Representation and Management In this thesis, we proposed a promising sequential pattern mining solution based on the Formal Concept Analysis theory to discover the semantic features from temporal and sequential data. An FCA-based model can extract features from raw data and explore correlations between target classes and features of interest. Behavioral patterns are automatically clustered by different features of interest, such as sensor events or atomic actions. These clusters are sorted by partial orders and form a hierarchy structure called concept lattice. Inferences that are related to activity recognition are encapsulated in such a graphical knowledge base.

Knowledge Base Retrieval Once the hierarchy structure is constructed, the issues of behavioral data analysis, including activity recognition, prediction and error detection, can be transformed to lattice search problems. We have different search strategies to deal with those problems. The observed data can be treated as query conditions and retrieve them within the knowledge base constructed by FCA. However, traditional retrieval method is static and cannot guarantee that suitable inferences are returned each time according to the observed data. Moreover, classical graph traversal algorithms always abandon all previous searches when new data are available. For these reasons, we proposed an HDS algorithm to retrieve suitable inferences quickly and incrementally. Our incremental way to retrieve inferences needs neither to start over again nor to traverse the whole graph to look for the observed data after each extension of observed data. It is a lattice search algorithm that consists of two part: the top-down search quickly locate one of the inferences satisfying the observed data, and the bottom-up one further finds the most optimal inference. It continues the inference retrieval of each new round of reasoning from the previous interrupted position. With the successive manner loading data in real-time, the scope of probable activities shrinks gradually and the

global optimal inference will be located at the end.

Ontological Clustering To distinguish highly similar activities with almost the same behavioral data, we proposed an assessment based on the root-mean-square deviation to measure the fitting between the observed values and the historical ones. For the purpose of reducing the impact of few data at the beginning, we further proposed an ontological clustering method for merging discovered clusters according to their semantic similarities. Thus, the inference engine will predict the ontological superclass instead of directly predicting an activity using few and limited observed data at the early stages.

Activity Recognition The proposed HDS algorithm can well recognize those behavioral patterns describing the basic activities with clear boundaries. However, the captured data from smart homes are always continuous. There are also more complicated ways to perform activities. After analyzing those complicated behavioral patterns, on the basis of the HDS algorithm, we propose several lattice search strategies to recognize composite activities with sequential, interleaved or concurrent patterns, as well as the multi-resident activities with parallel or cooperative patterns. The beginning and the end of a pattern describing an activity is determined by FCA based on the ontological correlations between activities and constituent behavioral data.

Error Detection We defined different abnormal behaviors commonly appearing in the behavioral patterns of residents, and proposed corresponding detectors. To recognize complex and multi-resident activities, we imported similar temporary caches to simulate different long-term intentions of residents. Moreover, for the multi-resident case, we used an additional transition matrix to help us identify two parallel activities performed at the same time.

ADVANTAGES

The FCA-based models have considered as a concise and robust solution to handle sequential and temporal data. For each unseen pattern that is not in the training dataset, but in the test dataset, the models will compare its similarity with learned patterns and propose the most similar activity cluster as the recognition result. In the worst case, unreliable sensor data will be evaluated and classified into a similar activity cluster.

Our approach has great advantages in terms of knowledge reuse and adaptation. The constructed Hasse diagram, accumulated matrices, lattice search strategies, and error detectors are designed as independent uncoupled modules. If one module has been modified, there is no influence to the others. As a consequence, most of them can be reused to the other smart homes with similar infrastructure designs. This is because the correlations between the behavioral patterns of human activities and sensors are established based on their ontological relevances. These relevances are inherent and independent with other factors.

In practice, many datasets are extremely imbalanced. For this reason, most probabilistic methods can not generate robust models by few training items with an unstable probability distribution. The same situation for our methods, inferences are convincing that a particular underrepresented class is not ignored or rejected by the score vote. An FCA-based model allows various behavioral patterns describing the same activity, and it tries to recognize activities by their general correlations.

As mentioned in Chapter 6, stable feature space and distribution are important for many algorithms. Nevertheless, new training data and extensible feature space are essential to maintaining the efficiency of an AmI application. As a result, we improved an incremental algorithm of lattice construction to expand our model incrementally by new data with new

features (i.e. new sensors deployed in a smart home). This is to avoid rebuilding models from scratch.

DRAWBACKS

First of all, most lattice construction methods can only build lattices from Boolean binary relations [169]. Thus, if we try to analyze numerical relations, features with numeric values have to be converted into categorical ones by losing precision. To convert real-valued features to the categorical ones, the simplest way is to split them at their median into two binary features [86]. However, this way will lose their precision [61]. For example, in the CASAS datasets, we convert all the positive sensor values into Boolean True. Briefly, if a tiny difference between numerical values in binary relations is sensitive and crucial, we should at least transfer them into the enumerable nominal values.

Then, activities with multilevel inheritance relations are easier to be affected by unreliable data and recognized as one of their similar derivations. Next, for the assessment based on RMSD, the natural lattice structure does not contain temporal information about execution orders, so the bias in the assessment due to incidental factors cannot be completely avoided.

The training data for the lattice construction are required to cover as many behavioral patterns describing the same activities as possible. Otherwise, insufficient training samples will cause a high false alarm rate while detecting some errors (e.g. omission of essential data and the mixture of irrelevant data).

As a common problem appearing in the other state-of-the-art prototypes, unseen activities cannot be predicted or recognized if no corresponding training data is available in the dataset [174]. However, a behavioral pattern describing an unseen activity will be predicted and recognized as a known activity with similar patterns.

Despite the attractive qualities of FCA-based models, there are still been prohibitively expensive to apply in some extreme cases. As shown in the experimental results, the efficiency of inference retrieval is very high, and the main time consumption focuses on the construction of the concept lattice from raw data. To solve this problem, we have proposed two optional pruning operations to reduce the size of the formal context of our model. Besides, redundant data as duplicate patterns can be refined to improve the efficiency of lattice construction.

PERSONAL ASSESSMENT OF DOCTORAL RESEARCH

The entire doctoral research was a long journey filled with difficulties and challenges. However, it was also the most important and memorable period of my life. Through this fascinating research subject, I became a member of a fabulous research team. I am so glad that I have joined the most promising research community and use the cutting-edge AI technologies to solve the real problems. This experience let me calm down to get into serious research work in my interested fields. It also allowed me to develop my rigorous research ability and communication skills.

My research work has been published in two international conference papers, three journal articles, a book chapter as well as a journal article that will be submitted soon.

[61] Hao, J., Bouchard, B., Bouzouane, A., & Gaboury, S. Real-time activity prediction and recognition in smart homes by formal concept analysis. 12th International Conference on Intelligent Environments, 2016, pp. 103-110.

[62] Hao, J., Gaboury, S., & Bouchard, B. Cognitive errors detection: Mining behavioral data stream of people with cognitive impairment. ACM International Conference on PErvasive Technologies Related to Assistive Environments, 2016, pp.15:1-15:8

[77] Hao, J., Bouzouane, A., Bouchard, B., & Gaboury, S. Activity inference engine for real-time cognitive assistance in smart environments. Journal of Ambient Intelligence and Humanized Computing, 9(3), 2018, 679-698.

[57] Hao, J., Bouzouane, A., & Gaboury, S. Complex behavioral pattern mining in non-intrusive sensor-based smart homes using an intelligent activity inference engine. Journal of Reliable Intelligent Environments, 3(2), 2017, 99-116.

[79] Hao, J., Bouzouane, A., Gaboury, S. Recognizing Multi-Resident Activities in Non-intrusive Sensor-Based Smart Homes by Formal Concept Analysis. Neurocomputing, 2018, pp. 1–21.

[222] Bouchard, K., Hao, J., Bouchard, B., Gaboury, S., Moutacalli, M. T., Gouin-Vallerand, C., ... & Giroux, S. The Cornerstones of Smart Home Research for Healthcare. In Advanced Data Analytics in Health, 2018, pp. 185-200.

[80] Hao, J., Bouzouane, A., Gaboury, S. An Incremental Learning Method Based on Formal Concept Analysis for Human Activity Recognition in Sensor-based Smart Environments

The accomplishment of my doctoral study is a new start of my research career, all my research experience will be my precious treasure that inspires me to continue my AI research.

Foremost, I would like to express my sincere gratitude to my advisors Prof. Sébastien Gaboury and Prof. Abdenour Bouzouane for their continuous and great support for my research. Their guidance with patience, motivation, enthusiasm, and immense knowledge incessantly helped me during the three years, especially in my articles and thesis writing. Besides, I would like to thank Prof. Bruno Bouchard for his guidance and great help at the beginning of my research. I also thank Julien Maître and the other members of the laboratory for their sincere help. At last, I would like to thank my parents for their greatest love and

support for my doctoral study.

FUTURE WORKS

Although the FCA-based model is a promising solution to solve some AmI problems, there are still some areas for improvement. At the moment, FCA-based models can only handle the observed data with categorical values due to the limitation of lattice construction. One possible improvement is to make FCA-based models could deal with numeric attributes like C4.5 or CART algorithms [108].

In addition, the FCA models can integrate themselves with various graphical models, such as probability or statistical models, in order to combine knowledge-driven models with data-driven ones. Such an integration can evaluate the probability of the occurrence of two highly similar activities from the perspective of probability, thus the prediction based on the RMSD assessment can be improved. We may also combine the active learning [56] to enhance the knowledge base.

For our current design, the RMSD assessment cannot well handle with data having a multi-modal distribution. We may use standard deviation to measure the confidence of the average position in our future work. Some factors in the training data such as temporal relations will also be considered.

APPENDIX A: TESTBEDS

INFRASTRUCTURE DESIGNS OF EXPERIMENTAL TESTBEDS

Due to different adopted sensors and flexible home layouts, the infrastructure design of a smart home is often diverse, not unique. However, the core idea of these designs is the same, that is to provide residents with a comfortable and safe living environment, a more convenient interactive experience and the appropriate assistance without disrupting their daily lives. In this appendix, we introduce two typical designs of sensor-based smart environments used in our experiments.

INFRASTRUCTURE OF LIARA SMART HOME

The Laboratoire d'Intelligence Ambiante pour la Reconnaissance d'Activités (LIARA) of Université du Québec à Chicoutimi has designed and built its own smart home. The LIARA smart home is a smart living environment covering an area of approximately 100 square meters. It is designed for elderly people, especially for those patients with Alzheimer's disease, known as an age-related cognitive impairment. It is also an innovative solution about the future living environment that focuses on providing real-time assistance based on ambient

intelligence for its residents. It consists of numerous sensors and actuators, such as passive RFID tags, RFID antennas, pressure mats, electromagnetic contacts, motion sensors, power analyzer, and smart plugs, in order to monitor environmental changes caused by human behaviors inside the smart home by non-intrusive ways.

Figure A1: Sensor layout of the LIARA smart home.

Figure. A1 shows the prototypical design of the LIARA smart home. Most objects in the figure are embedded with low cost controllable and measurable electronic components. For example, infrared, light sensors and RFID antennas have been installed on the walls. The oven in the kitchen zone is monitored and controlled by a built-in microcomputer and temperature sensors. A tablet is also embedded on the refrigerator to control the habitat of experiments, and assist residents with the help of teaching videos. The water consumption is measured by water sensors, and the power consumption is recorded by a power analyzer located at the main electrical panel. The open and closed states of cabinets are detected by binary sensors.

Pressure mats are placed in the bathroom to trace residents' movements. Besides, passive RFID tags are attached to all the other daily commodities to localize and track their spatial positions. The purpose of the LIARA datasets is to recognize human activities by human behaviors. In other words, they achieve the mapping described in Section 1.5.3, which is from intermediate-level atomic actions to high-level activities.

INFRASTRUCTURE OF CASAS TESTBED

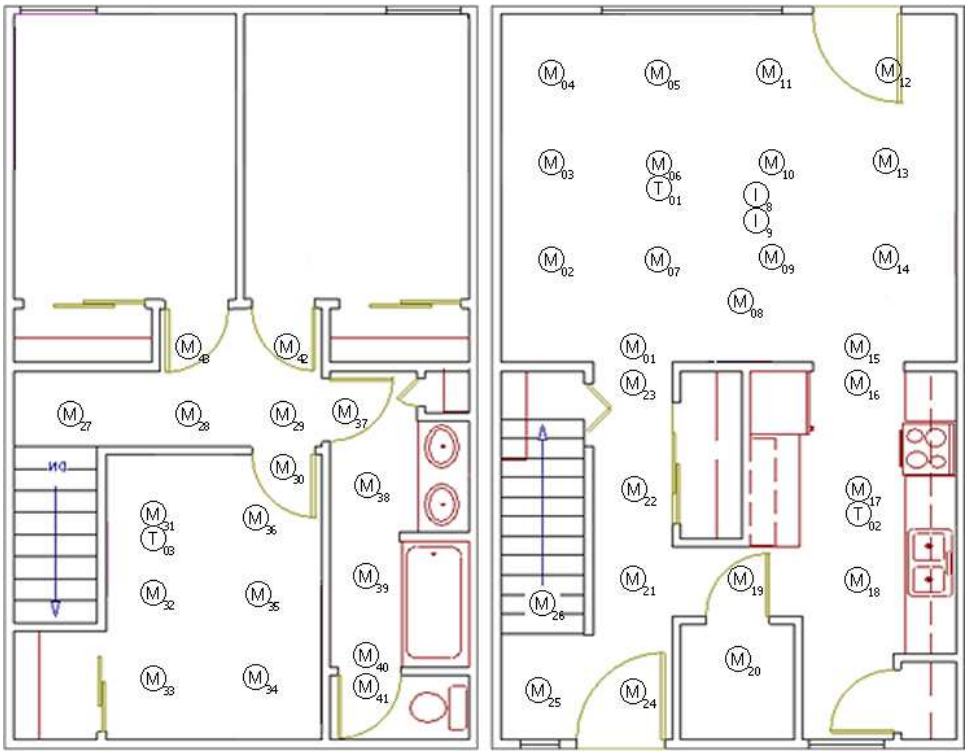


Figure A2: Sensor layout (bedroom) of CASAS intelligent apartment A.

The CASAS smart apartment is designed and constructed by the Center for Advanced Studies in Adaptive Systems of Washington State University. Its benchmark datasets[1] represent sensor data collected in a smart apartment testbed. As shown in Fig. A2, Fig. A3 and Fig. A4, the whole apartment, including bedrooms, a bathroom, a kitchen, and a living room,

[1]available at http://ailab.wsu.edu/casas/datasets/

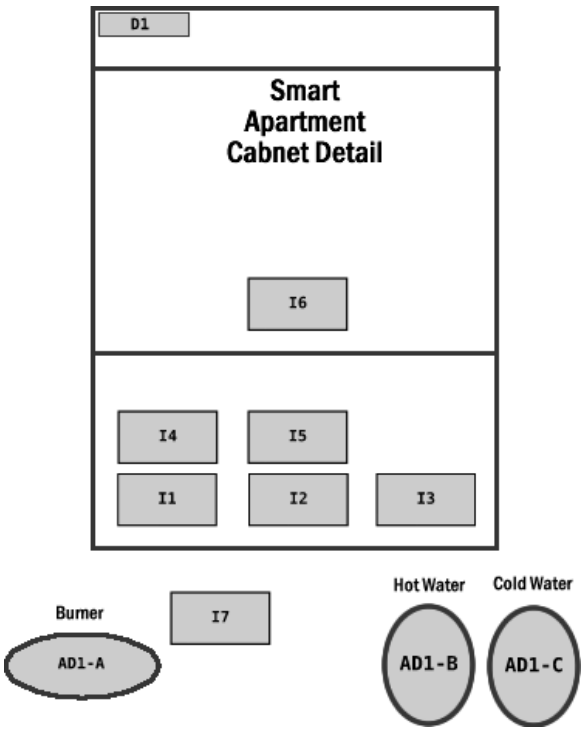


Figure A3: Sensor layout (cabinet) of CASAS intelligent apartment.

is deployed with heterogeneous sensors to capture various environmental states in the same non-intrusive ways.

Instead of using passive RFID tags to track daily objects, the CASAS laboratory directly uses motion sensors to track human movements. Thus, each sensor data in a sequence represents a raw sensor event. Besides, the CASAS smart apartment also includes temperature sensors, light controllers and a variety of item sensors to detect the human-object interactions produced by residents. Moreover, analog sensors monitor the usage of hot water, cold water, and stove burner. The phone usage is captured by Asterisk software and the states of doors and cabinets are captured by contact switch sensors. Pressure sensors monitor the usages of key items such as medicine container, cooking pot, and phone book.

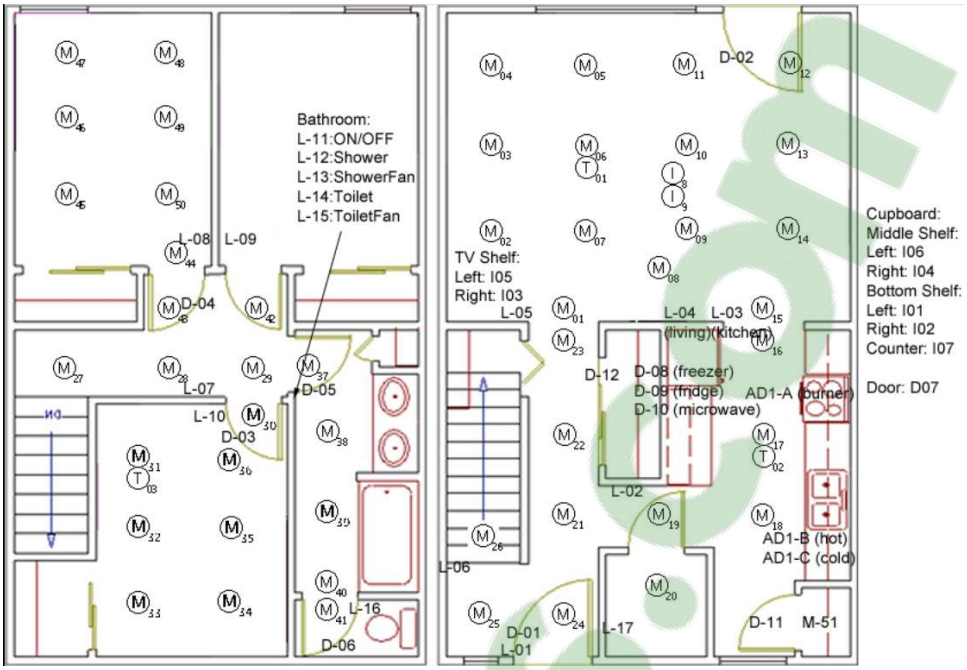


Figure A4: Sensor layout (bedroom) of CASAS intelligent apartment B

DATASET STUDIES

In this section, we describe a series of datasets that are used in various experiments for different AmI problems. Their characteristics including data formats and statistical information are also presented in details.

LIARA DATASETS

Based on the infrastructure design shown in Fig. A1, the researchers of LIARA laboratory created a series of datasets to verify the performance of activity recognition algorithms in different scenarios. Considering more frequent and complex human-object interactions, we chose several kitchen activities as our main research activities. Table. A1 is a training sample of LIARA datasets. It consists of three important data fields: *timestamps*, *atomic actions* and *labels*.

Table A1: Training Sample of LIARA Datasets

Timestamps	**Atomic Actions** $x^{(i)}$	**Label** $y^{(i)}$
2015-08-11 08:22:04	BoilWater	PrepareCoffee
2015-08-11 08:22:26	TakeCupFromCupboard	PrepareCoffee
2015-08-11 08:23:13	TakeOutCoffeePowder	PrepareCoffee
2015-08-11 08:23:23	PutCoffeePowderIntoCup	PrepareCoffee
2015-08-11 08:23:49	StoreCoffeePowder	PrepareCoffee
2015-08-11 08:35:13	PourWaterIntoCup	PrepareCoffee
2015-08-11 08:35:30	TakeOutSugar	PrepareCoffee
2015-08-11 08:35:41	AddSugarIntoCup	PrepareCoffee
2015-08-11 08:35:49	StoreSugar	PrepareCoffee
2015-08-11 08:35:57	TakeOutMilkFromRefrigerator	PrepareCoffee
2015-08-11 08:36:08	PourMilkIntoCup	PrepareCoffee
2015-08-11 08:36:22	StoreMilkInRefrigerator	PrepareCoffee
2015-08-11 08:36:31	BrewCoffee	PrepareCoffee
2015-08-11 08:36:43	PutSpoonIntoSink	PrepareCoffee

The timestamps field indicates the exact moment that an atomic action was performed or captured. Atomic actions are named in camel case, and they were obtained through several sensor data parsings, such as RFID signal analysis and load signatures of appliances. They were ordered by their timestamps and formed a behavioral pattern $x^{(i)}$ as input data of the training model. Their data type can be treated as categorical input values. The ground truth labels $y^{(i)}$ indicate the real activities performed. Thus, LIARA datasets are the data collections that try to recognize high-level activities by intermediate-level atomic actions.

LIARA Basic Dataset The first dataset contains bounded and basic activities, called LIARA basic dataset or RDATA. Its statistical information is shown in Table (A2). There are twelve kitchen activities. Each behavioral pattern $x^{(i)}$ is bounded and describes only one activity. In addition, some of them have a multi-level inheritance relationship, which means that a behavioral pattern of an activity is exactly the subset of a behavioral pattern of another activity. For example, the activity *PrepareSandwich* contains all the component actions of another activity *PrepareSandwichWithoutButter*. Thus, these two activities have the multi-level inheritance

relationship. This relationship is very common in real life and directly affects the accuracy of activity recognition and the high false alarm rate during the error detection.

1. PrepareCoffee: prepare a cup of coffee with sugar and milk. The objects that a resident interacts with are a kettle, instant coffee powder, sugar, milk, a cupboard, water, a cup, a refrigerator, and a spoon.

2. PrepareCoffeeWithoutSugar: prepare a cup of coffee with milk, but without sugar. The objects that a resident interacts with are a kettle, instant coffee powder, milk, a cupboard, water, a cup, a refrigerator, and a spoon.

3. PrepareCoffeeWithoutMilk: prepare a cup of coffee with sugar, but without milk. The objects that a resident interacts with are a kettle, instant coffee powder, sugar, a cupboard, water, a cup, a refrigerator, and a spoon.

4. PrepareMilk: prepare a cup of milk. The objects that a resident interacts with are a bowl, a drawer, milk, and a refrigerator.

5. PrepareSpaghetti: prepare spaghetti. The objects that a resident interacts with are a cauldron, a drawer, water, a stove, pasta, a strainer, a plate, a cupboard, and sauce.

6. PrepareSandwich: prepare a sandwich. The objects that a resident interacts with are bread, a knife, a cupboard, a plate, butter, ham, and mustard.

7. PrepareSandwichWithoutMustard: prepare a sandwich without mustard. The objects that a resident interacts with are bread, a knife, a cupboard, a plate, butter, and ham.

8. PrepareSandwichWithoutButter: prepare a sandwich without butter. The objects that a resident interacts with are bread, a knife, a cupboard, a plate, mustard, and ham.

9. PrepareCereal: prepare a bowl of cereals. The objects that a resident interacts with are a bowl, cereals, a cupboard, a drawer, a refrigerator, milk, and spoon.

10. PrepareToastsAndEggs: prepare toasts and eggs. The objects that a resident interact with are bread, a pan, a refrigerator, a knife, a drawer, butter, stove, a cupboard, a sink, eggs, a spatula, and a plate.

11. PreparePudding: prepare pudding as dessert. The objects that a resident interact with are pudding, a refrigerator, a plate, a spoon, and a drawer.

12. PrepareMilkTea: prepare a cup of milk tea. The objects that a resident interact with are a kettle, water, a teacup, a cupboard, a drawer, tea leaves, milk, a refrigerator, a spoon, and a sink.

Table A2: Statistical Information about LIARA Basic Dataset

Activities y	**Number of Atomic Actions**
PrepareCoffee	14
PrepareCoffeeWithoutSugar	11
PrepareCoffeeWithoutMilk	11
PrepareMilk	5
PrepareSpaghetti	18
PrepareSandwich	15
PrepareSandwichWithoutMustard	11
PrepareSandwichWithoutButter	9
PrepareCereal	8
PreparingToastsAndEggs	20
PreparePudding	5
PrepareMilkTea	12

LIARA Synthetic Dataset Based on the real data, the second LIARA dataset is called the LIARA synthetic dataset, or DDATA. It contains synthetic behavioral patterns that are generated under certain order constraints. Order constraints have limited that some sensor

data must appear before or after other data in order to avoid order inversion. For example, for the activity *PrepareMilkTea*, water should be boiled before pouring into a teacup.

For each indexed activity in the dataset, we kept constituent atomic actions unchanged, but disrupted the internal execution orders under the condition of following the order constraints. In this way, we obtained sufficient derived behavioral patterns to train models or generate test cases with errors.

LIARA Error Dataset Besides, the third dataset, named LIARA error dataset, is also synthetic and contains all the six errors predefined in Section 4.6, including the omission of essential data, the mixture of irrelevant data, unreasonable repetition, order inversion, and distraction. On the basis of derived sequences, we randomly changed their inner structures (e.g. removing, adding, repeating, splicing and swapping data) to create a dataset with those mentioned errors. Table. A3 shows the statistical information about this dataset.

Table A3: Statistical Information of LIARA Error Dataset

Activities	Number of Atomic Actions
PrepareCoffee	14
PrepareCoffeeWithoutSugar	11
PrepareCoffeeWithoutMilk	11
PrepareSpaghetti	18
PrepareSandwich	15
PrepareCereal	8
PreparingToastsAndEggs	20

LIARA Composite Activity Dataset We also created a synthetic dataset in order to recognize composite activities defined in Section 1.5.4. The training data come from the LIARA basic activity dataset without any modification. In other words, each training item only contains the data describing a basic activity. To create test data, first of all, we simulate that each activity was performed twenty times, and then, activities were freely performed in sequen-

tial, interleaved or concurrent ways. Twelve activities as same as the one shown in the basic dataset are described by sequentially observed actions.

CASAS DATASETS

We compared algorithm performance on a collection of datasets [2] from CASAS repository. Their features are either binary or categorical values. Similarly, there are four data fields: triggering data, time, sensor ID and its value.

CASAS Basic Activity Dataset The CASAS *Kyoto-1* basic activity dataset represents sensor events collected in the smart apartment testbed with the infrastructure design illustrated in Fig. A2 and Fig. A3. The data includes all 24 participants performing five activities in the apartment. The five activities are:

1. Make a Phone Call: moves to the phone in the dining room, looks a specific number in the phone book, dials the number, listens to a recorded message and summarizes the listened cooking directions on a notepad.
2. Wash Hands: moves into the kitchen sink and washes his/her hands in the sink, using hand soap and drying their hands with a paper towel.
3. Cook: cooks a pot of oatmeal according to the directions given in the phone message, measures water, pours the water into a pot and boils it, adds oats, then puts the oatmeal into a bowl with raisins and brown sugar.
4. Eat: takes the oatmeal and a medicine container to the dining room and eats the food.
5. Clean: takes all of the dishes to the sink, and cleans them with water and dish soap in the kitchen.

[2] http://ailab.wsu.edu/casas/datasets/

Furthermore, the data is categorized by participants and activities, and saved in different files named according to the participant number and task number. That is, in a separate file, the data contains all the sensor events that describes an activity. Each activity is bounded and indicated by its name with a specific start event and the corresponding end one.

CASAS Error Dataset The CASAS *Kyoto-2* error dataset totally reuses the setting of CASAS basic activity one, except that for each of the five tasks, an error is introduced. The involved errors are:

1. Make a Phone Call: a wrong phone number was initially dialed and has to be redialed.
2. Wash Hands: water is not turned off after washing his/her hands.
3. Cook: the burner is not turned off after cooking the oatmeal.
4. Eat: the medicine container is not brought with the participant to the dining room.
5. Clean: the participant does not use water to clean the dishes.

CASAS Composite Activity Dataset CASAS *Kyoto-3* dataset is a benchmark dataset that evaluates the performance of an algorithm recognizing composite activities. In this dataset, there are twenty participants performing eight basic and instrumental activities in the apartment. First of all, each activity was performed separately, and then these participants are asked to perform the entire set of eight activities again in any order or to perform tasks in concurrent or interleaved way if required. Eight activities were involved: *fill medication dispenser* (ac_1), *watch DVD* (ac_2), *water plants* (ac_3), *answer the phone* (ac_4), *prepare birthday card* (ac_5), *prepare soup* (ac_6), *clean* (ac_7), and *choose outfit* (ac_8). Each sensor reading is tagged with timestamps, a sensor id and its value. The CASAS dataset contains the patterns of sequential and interleaved activities.

CASAS Multi-resident Dataset This benchmark dataset is the CASAS Kyoto-4 multi-resident dataset. It contains sensor events collected from a smart apartment testbed. To generate Kyoto-4 dataset, researchers from CASAS laboratory recruited forty volunteers to perform fifteen activities in their smart apartment. Each time, the multi-resident environment was occupied by two volunteers at the same time to perform assigned tasks concurrently. Collected sensor events were manually labeled with the activity ID to which it belongs, and the ID of the resident who triggered it. However, most of them cannot provide decisive information to distinguish who (or which activity) generated the sensor events.

Table A4: Independent and Cooperative Activities in the CASAS Dataset

Activity ID	Activity	Type	Performers
1	Fill medication dispenser	Individual	R1
2	Hang up clothes	Individual	R2
3	Move furniture	Cooperative	R1, R2
4	Read magazine	Individual	R2
5	Water plants	Individual	R1
6	Sweep floor	Individual	R2
7	Play checkers	Cooperative	R1, R2
8	Prepare dinner	Individual	R1
9	Set table	Individual	R2
10	Read magazine	Individual	R1
11	Pay bills	Cooperative	R1, R2
12	Pack picnic food	Individual	R1
13	Retrieve dishes	Cooperative	R1,R2
14	Pack picnic supplies	Cooperative	R2
15	Pack and bring supplies	Individual	R1

As shown in Table A4, “R1” and “R2” refer to two different residents. Sometimes, two residents performed activities together or in the same space called “joint activities”. For joint activities, residents cooperate to jointly accomplish the task. The remaining independent activities are performed independently and in parallel. The statistical information about average activity times and the number of sensor events generated for each activity are shown in Table. A5.

Table A5: Average Time and Number of Sensor Events Generated for Each Activity

Activity ID	R1 Time (mins)	R1 Events	Activity ID	R2 Time (mins)	R2 Events
1	3.0	47	2	1.5	55
3	0.7	33	3	0.5	23
5	2.5	61	4	1.0	18
7	3.5	38	6	2.0	72
8	1.5	41	7	2.0	25
10	4.5	64	9	1.0	32
12, 15	1.5	37	11	5.0	65
-	N/A	N/A	13, 14	3.0	38

APPENDIX B: MODEL PERFORMANCE AND METRICS

MODEL MEASURES

In the model measures, we use the testing error as the approximation of generalization error. The testing set is mutually exclusive with the training set as far as possible. That is, the testing instances are not used in the training process.

CROSS-VALIDATION

Sometimes, a model can receive excellent results when it evaluates the data existing in the training set. However, once the test data has not been shown before, the recognition result may break down. Cross-validation is an efficient way to indicate the performance of a built model when it is required to predict the data that is not used to create the model.

Stratified 10-fold cross-validation is recommended for estimating accuracy, because of its relatively low bias and variance. However, in our experiments, to compare all the results with existing references under the same measures, we also adopt 3-fold cross-validation and leave-one-out cross-validation.

The objective of 10-fold cross-validation is to evaluate the capacity about generalization, a well-known issue in machine learning. With its help, each pattern in the dataset was removed at least once from the training sets.

LEAVE-ONE-OUT

Leave-one-out cross-validation (LOOCV) is a special case of k-fold cross-validation, where the number of folds k is equal to the number of instances in a dataset. Each instance has a chance to be selected as a single-item test set, at the same time, all other instances are applied as a training set.

Sometimes, LOOCV evaluation can be very costly and hard to be acceptable due to high number of instances [3]. For n instances, we have to create n different training sets and n different test sets, thus, there are totally n iterations for training and testing, each iteration is on $n-1$ instances. Assuming k is not too large and $k < n$, LOOCV is more computationally expensive than k-fold cross-validation [4].

PERFORMANCE MEASURES

In current machine learning research, when performing an empirical validation of new algorithms, it is not enough to simply present accuracy results. Thus, we briefly introduce several measures used for evaluating classification performances in the next experiments.

[3] http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.LeaveOneOut.html
[4] http://scikit-learn.org/stable/modules/cross_validation.html#leave-one-out-loo

CONFUSION MATRIX

Confusion matrix (also called a contingency table), is a two-dimensional matrix that summarizes the classification performance of a classification model with respect to a set of instances for testing (i.e. test data).

In binary classification, each instance can be assigned a label from the set $\{P, N\}$, which indicates a positive or negative class. In order to predict the class membership of instances, a classification model usually assigns discrete class labels or estimated probabilities within different thresholds indicating predicted classes.

Given a model and a labeled instance, there are four possible classification outcomes. If an instance with a positive label is correctly (T) classified as positive (P), it is counted as a *true positive*; if it is wrongly (F) classified as negative (N), it is counted as a *false negative*, also called the Type II error. If an instance with a negative label is correctly (T) classified as negative (N), it is counted as a *true negative*, otherwise, if it is wrongly (F) classified as positive (P), it is counted as a *false positive*, also called the Type I error. Fig. B1 is an example of confusion matrix summarizing statistical outcomes.

		Assigned class	
Actual class		Positive	Negative
	Positive	TP	FN
	Negative	FP	TN

Figure B1: Confusion matrix of binary classification

As shown in Fig. B2, in multi-class classification, the numbers of the major diagonal represent the correct classification, and the rest numbers represent confusions.

Once the confusion matrix is available, we are able to define many common metrics. Equa-

Actual class	Assigned class			
		A	***B***	***C***
	A	10	2	1
	B	0	6	1
	C	0	3	8

Figure B2: Confusion matrix of multi-class classification

tions 1 to 6 are six metrics formed on the basis of the matrix.

Precision (see Equation 1) is the proportion of instances predicted positive that are really positive, while *recall* (see Equation 2) is the proportion of positive instances that have been correctly predicted as positive.

$$Precision\ (P) = \frac{TP}{TP+FP} \tag{1}$$

$$Recall\ (R) = \frac{TP}{TP+FN} \tag{2}$$

True positive rate (see Equation 3) measures the fractions of positive instances that are correctly labeled. In opposite, *false positive rate* (see Equation 4) measures the fraction of negative instances that are misclassified as positive.

$$True\ Positive\ Rate\ (TPR) = \frac{TP}{P} = \frac{TP}{TP+FN} \tag{3}$$

$$False\ Positive\ Rate\ (FPR) = \frac{FP}{N} = \frac{FP}{FP+TN} \tag{4}$$

F1 score (see Equation 5) is the harmonic mean of precision and recall. If an F1 score is high, it means that both its precision and recall are good.

$$F_1 \text{ score} = \frac{2}{1/P + 1/R} = \frac{2PR}{P+R} = \frac{2TP}{2TP+FP+FN} \tag{5}$$

Accuracy (see Equation 6) refers to a measure that can be treated as the proportion of correctly classified instances within the total instances. It is also an important estimation between prediction and reality.

$$Accuracy\ (ACC) = \frac{TP+TN}{TP+TN+FP+FN} \tag{6}$$

BIBLIOGRAPHY

[1] H. H. Goldstine, *The computer from Pascal to von Neumann*. Princeton University Press, 1980.

[2] J. D. Torr, *The Information Age*. Greenhaven Press, 2003.

[3] A. M. Turing, "Computing machinery and intelligence," in *Parsing the Turing Test*, pp. 23–65, Springer, 2009.

[4] M. Chui, J. Manyika, and M. Miremadi, "Four fundamentals of workplace automation," *McKinsey Quarterly*, vol. 29, no. 3, pp. 1–9, 2015.

[5] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[6] DHL and IBM, *Artificial intelligence in logistics - A collaborative report by DHL and IBM on implications and use cases for the logistics industry*. DHL Trend Research, 2018.

[7] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2010.

[8] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[9] T. N. Sainath, A.-r. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for LVCSR," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8614–8618, IEEE, 2013.

[10] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen, *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International Conference on Machine Learning*, pp. 173–182, 2016.

[11] W. Xiong, J. Droppo, X. Huang, F. Seide, M. L. Seltzer, A. Stolcke, D. Yu, and G. Zweig, "Toward human parity in conversational speech recognition," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 25, no. 12, pp. 2410–2423, 2017.

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1026–1034, 2015.

[14] H. Y. Xiong, B. Alipanahi, L. J. Lee, H. Bretschneider, D. Merico, R. K. Yuen, Y. Hua, S. Gueroussov, H. S. Najafabadi, T. R. Hughes, *et al.*, "The human splicing

code reveals new insights into the genetic determinants of disease," *Science*, vol. 347, no. 6218, 2015.

[15] V. Gulshan, L. Peng, M. Coram, M. C. Stumpe, D. Wu, A. Narayanaswamy, S. Venugopalan, K. Widner, T. Madams, J. Cuadros, *et al.*, "Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs," *Journal of the American Medical Association*, vol. 316, no. 22, pp. 2402–2410, 2016.

[16] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun, "Dermatologist-level classification of skin cancer with deep neural networks," *Nature*, vol. 542, no. 7639, pp. 115–118, 2017.

[17] P. Rajpurkar, J. Irvin, K. Zhu, B. Yang, H. Mehta, T. Duan, D. Ding, A. Bagul, C. Langlotz, K. Shpanskaya, *et al.*, "Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning," 2017.

[18] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[19] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[20] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[21] Networking and Information Technology Research and Development Subcommittee, "The federal big data research and development strategic plan," 2016.

[22] S.-C. Fischer, "Artificial intelligence: China's high-tech ambitions," *CSS Analyses in Security Policy*, 2018.

[23] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[24] P. Remagnino and G. L. Foresti, "Ambient intelligence: A new multidisciplinary paradigm," *IEEE Transactions on systems, man, and cybernetics-Part A: Systems and humans*, vol. 35, no. 1, pp. 1–6, 2005.

[25] C. Ramos, J. C. Augusto, and D. Shapiro, "Ambient intelligence—the next step for artificial intelligence," *IEEE Intelligent Systems*, vol. 23, no. 2, pp. 15–18, 2008.

[26] F. Sadri, "Ambient intelligence: A survey," *ACM Computing Surveys (CSUR)*, vol. 43, no. 4, p. 36, 2011.

[27] D. J. Cook, J. C. Augusto, and V. R. Jakkula, "Ambient intelligence: Technologies, applications, and opportunities," *Pervasive and Mobile Computing*, vol. 5, no. 4, pp. 277–298, 2009.

[28] G. Acampora, D. J. Cook, P. Rashidi, and A. V. Vasilakos, "A survey on ambient intelligence in healthcare," *Proceedings of the IEEE*, vol. 101, no. 12, pp. 2470–2494, 2013.

[29] M. Soliman, T. Abiodun, T. Hamouda, J. Zhou, and C.-H. Lung, "Smart home: Integrating internet of things with web services and cloud computing," in *International Conference on Cloud Computing Technology and Science*, pp. 317–320, IEEE, 2013.

[30] M. Baldauf, S. Dustdar, and F. Rosenberg, "A survey on context-aware systems," *International Journal of ad Hoc and ubiquitous Computing*, vol. 2, no. 4, pp. 263–277, 2007.

[31] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (IoT): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[32] S. Amendola, R. Lodato, S. Manzari, C. Occhiuzzi, and G. Marrocco, "RFID technology for IoT-based personal healthcare in smart spaces," *IEEE Internet of Things Journal*, vol. 1, no. 2, pp. 144–152, 2014.

[33] P. Rawat, K. D. Singh, H. Chaouchi, and J. M. Bonnin, "Wireless sensor networks: a survey on recent developments and potential synergies," *The Journal of Supercomputing*, vol. 68, no. 1, pp. 1–48, 2014.

[34] J. Broekens, M. Heerink, and H. Rosendal, "Assistive social robots in elderly care: a review," *Gerontechnology*, vol. 8, no. 2, pp. 94–103, 2009.

[35] L. C. De Silva, C. Morikawa, and I. M. Petra, "State of the art of smart homes," *Engineering Applications of Artificial Intelligence*, vol. 25, no. 7, pp. 1313–1321, 2012.

[36] D. J. Cook and S. K. Das, "How smart are our environments? an updated look at the state of the art," *Pervasive and Mobile Computing*, vol. 3, no. 2, pp. 53–73, 2007.

[37] Q. Lin, D. Zhang, L. Chen, H. Ni, and X. Zhou, "Managing elders' wandering behavior using sensors-based solutions: a survey," *International Journal of Gerontology*, vol. 8, no. 2, pp. 49–55, 2014.

[38] D. Monekosso, F. Florez-Revuelta, and P. Remagnino, "Ambient assisted living," *IEEE Intelligent Systems*, vol. 30, no. 4, pp. 2–6, 2015.

[39] O. D. Lara and M. A. Labrador, "A survey on human activity recognition using wearable sensors," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 3, pp. 1192–1209, 2013.

[40] A. S. Crandall and D. J. Cook, "Smart home in a box: A large scale smart home deployment," in *Workshop Proceedings of the 8th International Conference on Intelligent Environments*, pp. 169–178, 2012.

[41] C. E. Shannon, "A mathematical theory of communication," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 5, no. 1, pp. 3–55, 2001.

[42] S.-R. Ke, H. L. U. Thuc, Y.-J. Lee, J.-N. Hwang, J.-H. Yoo, and K.-H. Choi, "A review on video-based human activity recognition," *Computers*, vol. 2, no. 2, pp. 88–131, 2013.

[43] F. Fusier, V. Valentin, F. Brémond, M. Thonnat, M. Borg, D. Thirde, and J. Ferryman, "Video understanding for complex activity recognition," *Machine Vision and Applications*, vol. 18, no. 3-4, pp. 167–188, 2007.

[44] S. Vishwakarma and A. Agrawal, "A survey on activity recognition and behavior understanding in video surveillance," *The Visual Computer*, vol. 29, no. 10, pp. 983–1009, 2013.

[45] L. Chen, J. Hoey, C. D. Nugent, D. J. Cook, and Z. Yu, "Sensor-based activity recognition," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, pp. 790–808, 2012.

[46] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[47] M. Al-Wattar, R. Khusainov, D. Azzi, and J. Chiverton, “Activity recognition from video data using spatial and temporal features,” in *International Conference on Intelligent Environments*, pp. 250–253, 2016.

[48] S. Hongeng, R. Nevatia, and F. Bremond, “Video-based event recognition: activity representation and probabilistic recognition methods,” *Computer Vision and Image Understanding*, vol. 96, no. 2, pp. 129–162, 2004.

[49] A. Cavallaro, “Privacy in video surveillance,” *IEEE Signal Processing Magazine*, vol. 2, no. 24, pp. 168–166, 2007.

[50] J. Morales and D. Akopian, “Physical activity recognition by smartphones, a survey,” *Biocybernetics and Biomedical Engineering*, vol. 37, no. 3, pp. 388–400, 2017.

[51] Y. Kwon, K. Kang, and C. Bae, “Unsupervised learning for human activity recognition using smartphone sensors,” *Expert Systems with Applications*, vol. 41, no. 14, pp. 6067–6074, 2014.

[52] D. Triboan, L. Chen, F. Chen, and Z. Wang, “Semantic segmentation of real-time sensor data stream for complex activity recognition,” *Personal and Ubiquitous Computing*, vol. 21, no. 3, pp. 411–425, 2017.

[53] M. Pantic, A. Pentland, A. Nijholt, and T. S. Huang, “Human computing and machine understanding of human behavior: a survey,” in *Artifical Intelligence for Human Computing*, pp. 47–71, Springer, 2007.

[54] G. Okeyo, L. Chen, H. Wang, and R. Sterritt, “Dynamic sensor data segmentation for real-time knowledge-driven activity recognition,” *Pervasive and Mobile Computing*, vol. 10, pp. 155–172, 2014.

[55] J. Kramer, “Is abstraction the key to computing?,” *Communications of the ACM*, vol. 50, no. 4, pp. 36–42, 2007.

[56] J. Han, J. Pei, and M. Kamber, *Data mining: concepts and techniques*. Elsevier, 2011.

[57] J. Hao, A. Bouzouane, and S. Gaboury, “Complex behavioral pattern mining in non-intrusive sensor-based smart homes using an intelligent activity inference engine,” *Journal of Reliable Intelligent Environments*, vol. 3, no. 2, pp. 99–116, 2017.

[58] D. Fortin-Simard, J.-S. Bilodeau, S. Gaboury, B. Bouchard, and A. Bouzouane, “Method of recognition and assistance combining passive RFID and electrical load analysis that handles cognitive errors,” *International Journal of Distributed Sensor Networks*, vol. 11, no. 10, 2015.

[59] C. Belley, S. Gaboury, B. Bouchard, and A. Bouzouane, “An efficient and inexpensive method for activity recognition within a smart home based on load signatures of appliances,” *Pervasive and Mobile Computing*, vol. 12, pp. 58–78, 2014.

[60] C. Belley, S. Gaboury, B. Bouchard, and A. Bouzouane, “Nonintrusive system for assistance and guidance in smart homes based on electrical devices identification,” *Expert Systems with Applications*, vol. 42, no. 19, pp. 6552–6577, 2015.

[61] J. Hao, B. Bouchard, A. Bouzouane, and S. Gaboury, “Real-time activity prediction and recognition in smart homes by formal concept analysis,” in *International Conference on Intelligent Environments*, pp. 103–110, IEEE, 2016.

[62] J. Hao, S. Gaboury, and B. Bouchard, “Cognitive errors detection: Mining behavioral data stream of people with cognitive impairment,” in *Proceedings of the 9th ACM International Conference on PErvasive Technologies Related to Assistive Environments*, pp. 15:1–15:8, ACM, 2016.

[63] N. K. Suryadevara and S. C. Mukhopadhyay, “ADLs recognition of an elderly person and wellness determination,” in *Smart Homes*, pp. 111–137, Springer, 2015.

[64] S. E. Bibri, “Ambient intelligence: A new computing paradigm and a vision of a next wave in ICT,” in *The Human Face of Ambient Intelligence*, pp. 23–66, Springer, 2015.

[65] S. Liang, C.-Y. Huang, T. Khalafbeigi, *et al.*, “Ogc sensorthings api-part 1: Sensing,” *OGC® Implementation Standard. http://docs. opengeospatial. org/is/15-078r6/15-078r6. html*, 2016.

[66] M. A. Jazayeri, S. H. Liang, and C.-Y. Huang, “Implementation and evaluation of four interoperable open standards for the internet of things,” *Sensors*, vol. 15, no. 9, pp. 24343–24373, 2015.

[67] D. Miorandi, S. Sicari, F. De Pellegrini, and I. Chlamtac, “Internet of things: Vision, applications and research challenges,” *Ad hoc Networks*, vol. 10, no. 7, pp. 1497–1516, 2012.

[68] L. Chen, C. D. Nugent, and H. Wang, “A knowledge-driven approach to activity recognition in smart homes,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 6, pp. 961–974, 2012.

[69] P. Rashidi and A. Mihailidis, “A survey on ambient-assisted living tools for older adults,” *Journal of Biomedical and Health Informatics*, vol. 17, no. 3, pp. 579–590, 2013.

[70] A. Benmansour, A. Bouchachia, and M. Feham, “Multioccupant activity recognition in pervasive smart home environments,” *ACM Computing Surveys*, vol. 48, no. 3, p. 34, 2016.

[71] S. Zhang, S. McClean, B. Scotney, P. Chaurasia, and C. Nugent, "Using duration to learn activities of daily living in a smart home environment," in *4th International Conference on Pervasive Computing Technologies for Healthcare*, pp. 1–8, IEEE, 2010.

[72] B.-C. Chien and R.-S. Huang, "Activity recognition using discriminant sequence patterns in smart environments," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 329–341, Springer, 2014.

[73] C. C. Aggarwal, *Data classification: algorithms and applications*. CRC Press, 2014.

[74] C. C. Aggarwal and C. K. Reddy, *Data clustering: algorithms and applications*. CRC press, 2013.

[75] J. L. Kolodner, "An introduction to case-based reasoning," *Artificial Intelligence Review*, vol. 6, no. 1, pp. 3–34, 1992.

[76] D. J. Cook, "Learning setting-generalized activity models for smart spaces," *IEEE Intelligent Systems*, vol. 27, no. 1, pp. 32–38, 2012.

[77] J. Hao, A. Bouzouane, and S. Gaboury, "Activity inference engine for real-time cognitive assistance in smart environments," *Journal of Ambient Intelligence and Humanized Computing*, vol. 9, no. 3, pp. 679–698, 2018.

[78] C. Sammut and G. I. Webb, *Encyclopedia of machine learning*. Springer Science & Business Media, 2011.

[79] J. Hao, A. Bouzouane, and S. Gaboury, "Recognizing multi-resident activities in non-intrusive sensor-based smart homes by formal concept analysis," *Neurocomputing*, vol. 1, no. 1, pp. 1–21, 2018.

[80] J. Hao, A. Bouzouane, and S. Gaboury, "An incremental learning method based on formal concept analysis for human activity recognition in sensor-based smart environments," *Submitted*, 2018.

[81] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2016.

[82] Z.-H. Zhou, *Machine Learning*. Tsinghua University Press, 2016.

[83] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

[84] A. Vellido, J. Martín-guerrero, and P. J. Lisboa, "Making machine learning models interpretable," in *Proceedings of European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, pp. 163–172, 2012.

[85] F. Hayes-Roth, D. Waterman, and D. Lenat, "Building expert systems," pp. 405–420, 1984.

[86] B. Letham, C. Rudin, T. H. McCormick, and D. Madigan, "Building interpretable classifiers with rules using bayesian analysis," *The Annals of Applied Statistics*, pp. 1350–1371, 2012.

[87] W. M. Van der Aalst, "Process discovery: An introduction," in *Process Mining*, pp. 125–156, Springer, 2011.

[88] Z.-H. Zhou, *Ensemble methods: foundations and algorithms*. CRC press, 2012.

[89] J. Franklin, "The elements of statistical learning: data mining, inference and prediction," *The Mathematical Intelligencer*, vol. 27, no. 2, pp. 83–85, 2005.

[90] L. E. Sucar, *Probabilistic Graphical Models: Principles and Applications*. Springer, 2015.

[91] V. Mihajlovic and M. Petkovic, "Dynamic bayesian networks: A state of the art," *University of Twente Document Repository*, vol. TR-CTIT-34, 2001.

[92] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[93] T. Van Kasteren and B. Krose, "Bayesian activity recognition in residence for elders," in *International Conference on Intelligent Environments*, pp. 209–212, 2007.

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

[95] E. Nazerfard and D. J. Cook, "CRAFFT: an activity prediction model based on bayesian networks," *Journal of Ambient Intelligence and Humanized Computing*, vol. 6, no. 2, pp. 193–205, 2015.

[96] L. Liu, S. Wang, G. Su, Z.-G. Huang, and M. Liu, "Towards complex activity recognition using a bayesian network-based probabilistic generative framework," *Pattern Recognition*, vol. 68, pp. 295–309, 2017.

[97] E. Kim, S. Helal, and D. Cook, "Human activity recognition and pattern discovery," *IEEE Pervasive Computing*, vol. 9, no. 1, pp. 48–53, 2010.

[98] T. L. Van Kasteren, G. Englebienne, and B. J. Kröse, "Hierarchical activity recognition using automatically clustered actions," in *International Joint Conference on Ambient Intelligence*, pp. 82–91, Springer, 2011.

[99] A. Mihailidis, J. Boger, M. Canido, and J. Hoey, "The use of an intelligent prompting system for people with dementia," *Interactions*, vol. 14, no. 4, pp. 34–37, 2007.

[100] Y.-T. Chiang, K.-C. Hsu, C.-H. Lu, L.-C. Fu, and J. Y.-J. Hsu, "Interaction models for multiple-resident activity recognition in a smart home," in *International Conference on Intelligent Robots and Systems*, pp. 3753–3758, IEEE, 2010.

[101] N. V. Chawla, "Data mining for imbalanced datasets: An overview," in *Data Mining and Knowledge Discovery Handbook*, pp. 875–886, Springer, 2009.

[102] E. Nazerfard, B. Das, L. B. Holder, and D. J. Cook, "Conditional random fields for activity recognition in smart environments," in *Proceedings of the 1st ACM International Health Informatics Symposium*, pp. 282–286, ACM, 2010.

[103] K.-C. Hsu, Y.-T. Chiang, G.-Y. Lin, C.-H. Lu, J. Y.-J. Hsu, and L.-C. Fu, "Strategies for inference mechanism of conditional random fields for multiple-resident activity recognition in a smart home," in *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, pp. 417–426, Springer, 2010.

[104] J. Yin, D. H. Hu, and Q. Yang, "Spatio-temporal event detection using dynamic conditional random fields," in *International Joint Conference on Artificial Intelligence*, vol. 9, pp. 1321–1327, 2009.

[105] E. M. Tapia, S. S. Intille, and K. Larson, "Activity recognition in the home using simple and ubiquitous sensors," in *International Conference on Pervasive Computing*, pp. 158–175, Springer, 2004.

[106] J. R. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986.

[107] J. R. Quinlan, "C4.5: Programming for machine learning," 1993.

[108] J. R. Quinlan, *C4.5: programs for machine learning*. Elsevier, 2014.

[109] M. Prossegger and A. Bouchachia, “Multi-resident activity recognition using incremental decision trees,” in *Adaptive and Intelligent Systems*, pp. 182–191, Springer, 2014.

[110] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman, “Activity recognition from accelerometer data,” in *AAAI*, vol. 5, pp. 1541–1546, 2005.

[111] U. Maurer, A. Smailagic, D. P. Siewiorek, and M. Deisher, “Activity recognition and monitoring using multiple sensors on different body positions,” in *Proceedings of the International Workshop on Wearable and Implantable Body Sensor Networks*, pp. 113–116, IEEE, 2006.

[112] L. Fan, Z. Wang, and H. Wang, “Human activity recognition model based on decision tree,” in *International Conference on Advanced Cloud and Big Data*, pp. 64–68, IEEE, 2013.

[113] L. Bao and S. S. Intille, “Activity recognition from user-annotated acceleration data,” in *International Conference on Pervasive Computing*, pp. 1–17, Springer, 2004.

[114] C. Chen, B. Das, and D. J. Cook, “A data mining framework for activity recognition in smart environments,” in *International Conference on Intelligent Environments*, pp. 80–83, 2010.

[115] C. C. Aggarwal and J. Han, *Frequent pattern mining*. Springer, 2014.

[116] B. Chikhaoui, S. Wang, and H. Pigot, “A frequent pattern mining approach for ADLs recognition in smart environments,” in *IEEE International Conference on Advanced Information Networking and Applications*, pp. 248–255, IEEE, 2011.

[117] P. Rashidi, D. J. Cook, L. B. Holder, and M. Schmitter-Edgecombe, "Discovering activities to recognize and track in a smart environment," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 4, pp. 527–539, 2011.

[118] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions and reversals," in *Soviet Physics Doklady*, vol. 10, 1966.

[119] A. Jurek, C. Nugent, Y. Bi, and S. Wu, "Clustering-based ensemble learning for activity recognition in smart homes," *Sensors*, vol. 14, no. 7, pp. 12285–12304, 2014.

[120] Z. Feng, L. Mo, and M. Li, "A random forest-based ensemble method for activity recognition," in *Engineering in Medicine and Biology Society*, pp. 5074–5077, IEEE, 2015.

[121] B. Krawczyk, "Active and adaptive ensemble learning for online activity recognition from data streams," *Knowledge-Based Systems*, vol. 138, pp. 69–78, 2017.

[122] Y.-J. Kim, Y. Kim, J. Ahn, and D. Kim, "Integrating hidden markov models based on mixture-of-templates and k-NN ensemble for activity recognition," in *International Conference on Pattern Recognition*, pp. 1636–1641, IEEE, 2016.

[123] S. Ryza, U. Laserson, S. Owen, and J. Wills, *Advanced analytics with spark: patterns for learning from data at scale*. O'Reilly Media, Inc., 2017.

[124] M. J. Zaki, W. Meira Jr, and W. Meira, *Data mining and analysis: fundamental concepts and algorithms*. Cambridge University Press, 2014.

[125] T. Huynh and B. Schiele, "Analyzing features for activity recognition," in *Proceedings of the Joint Conference on Smart Objects and Ambient Intelligence: Innovative Context-aware Services: Usages and Technologies*, pp. 159–163, ACM, 2005.

[126] S. Zhao, W. Li, and J. Cao, "A user-adaptive algorithm for activity recognition based on k-means clustering, local outlier factor, and multivariate gaussian distribution," *Sensors*, vol. 18, no. 6, 2018.

[127] L. G. Fahad, A. Ali, and M. Rajarajan, "Long term analysis of daily activities in smart home.," in *European Symposium on Artificial Neural Networks (ESANN)*, 2013.

[128] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning: with Applications in R*, p. 78–129. Springer New York, 2013.

[129] Y. Sun, A. K. Wong, and M. S. Kamel, "Classification of imbalanced data: A review," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 23, no. 4, pp. 687–719, 2009.

[130] L. G. Fahad, A. Ali, and M. Rajarajan, "Learning models for activity recognition in smart homes," *Lecture Notes in Electrical Engineering*, vol. 339, pp. 819–826, 2015.

[131] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, S. Y. Philip, *et al.*, "Top 10 algorithms in data mining," *Knowledge and Information Systems*, vol. 14, no. 1, pp. 1–37, 2008.

[132] D. J. Cook, N. C. Krishnan, and P. Rashidi, "Activity discovery and activity recognition: A new partnership," *IEEE Transactions on Systems Man Cybernetics Part B Cybernetics A Publication of the IEEE Systems Man Cybernetics Society*, vol. 43, no. 3, pp. 820–828, 2012.

[133] R. Mohamed, T. Perumal, M. Sulaiman, N. Mustapha, M. Zainudin, *et al.*, "Multi label classification on multi resident in smart home using classifier chains," *Advanced Science Letters*, vol. 24, no. 2, pp. 1316–1319, 2018.

[134] A. Fleury, M. Vacher, and N. Noury, "SVM-based multimodal classification of activities of daily living in health smart homes: sensors, algorithms, and first experimental results.," *IEEE Transactions on Information Technology in Biomedicine*, vol. 14, no. 2, pp. 274–283, 2010.

[135] S. Althloothi, M. H. Mahoor, X. Zhang, and R. M. Voyles, "Human activity recognition using multi-features and multiple kernel learning," *Pattern Recognition*, vol. 47, no. 5, pp. 1800–1812, 2014.

[136] S. Choi, E. Kim, and S. Oh, "Human behavior prediction for smart homes using deep learning," in *Ro-Man*, pp. 173–179, 2013.

[137] N. C. Krishnan and D. J. Cook, "Activity recognition on streaming sensor data," *Pervasive Mobile Computing*, vol. 10, pp. 138–154, 2014.

[138] B. Ganter and R. Wille, *Formal concept analysis: mathematical foundations*. Springer Science & Business Media, 1999.

[139] C. De Maio, G. Fenza, M. Gallo, V. Loia, and S. Senatore, "Formal and relational concept analysis for fuzzy-based automatic semantic annotation," *Applied Intelligence*, vol. 40, no. 1, pp. 154–177, 2014.

[140] J. Poelmans, D. I. Ignatov, S. O. Kuznetsov, and G. Dedene, "Formal concept analysis in knowledge processing: A survey on applications," *Expert Systems with Applications*, vol. 40, no. 16, pp. 6538–6560, 2013.

[141] C. De Maio, G. Fenza, V. Loia, and M. Parente, "Time aware knowledge extraction for microblog summarization on twitter," *Information Fusion*, vol. 28, pp. 60–74, 2016.

[142] P. Valtchev, R. Missaoui, and R. Godin, “Formal concept analysis for knowledge discovery and data mining: The new challenges,” in *International Conference on Formal Concept Analysis*, pp. 352–371, Springer, 2004.

[143] R. Wille, “Restructuring lattice theory: an approach based on hierarchies of concepts,” in *Ordered Sets*, pp. 445–470, Springer, 1982.

[144] J. Poelmans, P. Elzinga, S. Viaene, and G. Dedene, “Formal concept analysis in knowledge discovery: a survey,” in *International Conference on Conceptual Structures*, pp. 139–153, Springer, 2010.

[145] M. Obitko, V. Snásel, and J. Smid, “Ontology design with formal concept analysis,” in *Proceedings of International Workshop on Concept Lattices and their Applications*, pp. 111–119, 2004.

[146] R. Bendaoud, A. Napoli, and Y. Toussaint, “Formal concept analysis: A unified framework for building and refining ontologies,” in *International Conference on Knowledge Engineering and Knowledge Management*, pp. 156–171, Springer, 2008.

[147] C. De Maio, G. Fenza, V. Loia, and S. Senatore, “Hierarchical web resources retrieval by exploiting fuzzy formal concept analysis,” *Information Processing & Management*, vol. 48, no. 3, pp. 399–418, 2012.

[148] P. du Boucher-Ryan and D. Bridge, “Collaborative recommending using formal concept analysis,” *Knowledge-Based Systems*, vol. 19, no. 5, pp. 309–315, 2006.

[149] Y. Huang and L. Bian, “Using ontologies and formal concept analysis to integrate heterogeneous tourism information,” *IEEE Transactions on Emerging Topics in Computing*, vol. 3, no. 2, pp. 172–184, 2015.

[150] R. J. Cole, *The management and visualisation of document collections using formal concept analysis*. CUMINCAD, 2000.

[151] P. Eklund, J. Ducrou, and P. Brawn, "Concept lattices for information visualization: Can novices read line-diagrams?," in *International Conference on Formal Concept Analysis*, pp. 57–73, Springer, 2004.

[152] W.-C. Lim and C.-S. Lee, "Knowledge discovery through composited visualization, navigation and retrieval," in *International Conference on Discovery Science*, pp. 377–379, Springer, 2005.

[153] C. Lindig, "Fast concept analysis," in *International Conference on Conceptual Structures*, pp. 152–161, 2000.

[154] S. O. Kuznetsov and S. A. Obiedkov, "Comparing performance of algorithms for generating concept lattices," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 14, no. 2-3, pp. 189–216, 2002.

[155] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *Proceedings of International Conference on Very Large Data Bases (VLDB)*, vol. 1215, pp. 487–499, 1994.

[156] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," in *ACM Sigmod Record*, vol. 29, pp. 1–12, ACM, 2000.

[157] L. Lakhal and G. Stumme, "Efficient mining of association rules based on formal concept analysis," in *Formal Concept Analysis*, pp. 180–195, Springer, 2005.

[158] A. Aamodt and E. Plaza, "Case-based reasoning: Foundational issues, methodological variations, and system approaches," *AI Communications*, vol. 7, no. 1, pp. 39–59, 1994.

[159] M. Uschold and M. Gruninger, "Ontologies: Principles, methods and applications," *The Knowledge Engineering Review*, vol. 11, no. 2, pp. 93–136, 1996.

[160] A. Gómez-Pérez, M. Fernández-López, and O. Corcho, *Ontological Engineering: with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web*. Springer Science & Business Media, 2006.

[161] A. Formica, "Ontology-based concept similarity in formal concept analysis," *Information Sciences*, vol. 176, no. 18, pp. 2624–2641, 2006.

[162] C. C. Aggarwal, *Data mining: the textbook*. Springer, 2015.

[163] Oracle, *Oracle Semantic Technologies Overview*. Oracle Doc, 2018.

[164] G. Singla, D. J. Cook, and M. Schmitter-Edgecombe, "Recognizing independent and joint activities among multiple residents in smart environments," *Journal of Ambient Intelligence and Humanized Computing*, vol. 1, no. 1, pp. 57–63, 2010.

[165] P. Valtchev and R. Missaoui, "Building concept (Galois) lattices from parts: generalizing the incremental methods," in *International Conference on Conceptual Structures*, pp. 290–303, Springer, 2001.

[166] H. Fu and E. M. Nguifo, "A lattice algorithm for data mining," *Ingénierie des Systemes d'Informatione*, vol. 9, pp. 109–132, 2004.

[167] J.-P. Bordat, "Calcul pratique du treillis de galois d'une correspondance," *Mathématiques et Sciences Humaines*, vol. 96, pp. 31–47, 1986.

[168] M. Chein, "Algorithme de recherche des sous-matrices premières d'une matrice," *Bulletin mathématique de la Société des Sciences Mathématiques de la République Socialiste de Roumanie*, pp. 21–25, 1969.

[169] B. Ganter, "Two basic algorithms in concept analysis," in *Proceedings of the 8th International Conference on Formal Concept Analysis*, pp. 312–340, Springer-Verlag, 2010.

[170] P. C. Roy, S. Giroux, B. Bouchard, A. Bouzouane, C. Phua, A. Tolstikov, and J. Biswas, "A possibilistic approach for activity recognition in smart homes for cognitive assistance to Alzheimer's patients," in *Activity Recognition in Pervasive Intelligent Environments*, pp. 33–58, Springer, 2011.

[171] J. Leskovec, A. Rajaraman, and J. D. Ullman, *Mining of massive datasets*. Cambridge University Press, 2014.

[172] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, vol. 14, pp. 1137–1145, 1995.

[173] D. J. Cook, "Learning setting-generalized activity models for smart spaces," *IEEE intelligent systems*, vol. 2010, no. 99, p. 1, 2010.

[174] H.-T. Cheng, *Learning and Recognizing the Hierarchical and Sequential Structure of Human Activities*. PhD thesis, Carnegie Mellon University, 2013.

[175] G. Singla, D. J. Cook, and M. Schmitter-Edgecombe, "Tracking activities in complex settings using smart environment technologies," *International Journal of Biosciences, Psychiatry and Technology (IJBSPT)*, vol. 1, no. 1, pp. 25–35, 2009.

[176] L. Akoglu, H. Tong, and D. Koutra, "Graph based anomaly detection and description: a survey," *Data Mining and Knowledge Discovery*, vol. 29, no. 3, pp. 626–688, 2015.

[177] M. Ruotsalainen, T. Ala-Kleemola, and A. Visa, "GAIS: A method for detecting interleaved sequential patterns from imperfect data," in *IEEE Symposium on Computational Intelligence and Data Mining*, pp. 530–534, IEEE, 2007.

[178] T. Gu, Z. Wu, X. Tao, and H. K. Pung, "epSICAR: An emerging patterns based approach to sequential, interleaved and concurrent activity recognition," in *IEEE International Conference on Pervasive Computing and Communications*, pp. 1–9, 2009.

[179] J. Modayil, T. Bai, and H. Kautz, "Improving the recognition of interleaved activities," in *Proceedings of the 10th International Conference on Ubiquitous Computing*, pp. 40–43, 2008.

[180] D. H. Hu and Q. Yang, "CIGAR: concurrent and interleaving goal and activity recognition," in *AAAI Conference on Artificial Intelligence*, pp. 1363–1368, 2008.

[181] G. Singla and D. J. Cook, "Interleaved activity recognition for smart home residents," in *International Conference on Intelligent Environments*, pp. 145–152, 2009.

[182] S. Hallé, S. Gaboury, and B. Bouchard, "Towards user activity recognition through energy usage analysis and complex event processing," in *Proceedings of the 9th ACM International Conference on PErvasive Technologies Related to Assistive Environments*, p. 3, ACM, 2016.

[183] D. Riboni, T. Sztyler, G. Civitarese, and H. Stuckenschmidt, "Unsupervised recognition of interleaved activities of daily living through ontological and probabilistic reasoning," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pp. 1–12, ACM, 2016.

[184] P. Roy, B. Bouchard, A. Bouzouane, and S. Giroux, "A hybrid plan recognition model for Alzheimer's patients: interleaved-erroneous dilemma," *International Conference on Intelligent Agent Technology*, vol. 7, no. 4, pp. 375–397, 2009.

[185] G. Okeyo, L. Chen, and H. Wang, "Combining ontological and temporal formalisms for composite activity modelling and recognition in smart homes," *Future Generation Computer Systems*, vol. 39, pp. 29–43, 2014.

[186] Saguna, A. Zaslavsky, and D. Chakraborty, "Towards a robust concurrent and interleaved activity recognition of mobile users," in *IEEE International Conference on Mobile Data Management*, pp. 297–298, 2011.

[187] J. Ye and S. Dobson, "Exploring semantics in activity recognition using context lattices," *Journal of Ambient Intelligence and Smart Environments*, vol. 2, no. 4, pp. 389–407, 2010.

[188] J. Ye, G. Stevenson, and S. Dobson, "KCAR: A knowledge-driven approach for concurrent activity recognition," *Pervasive and Mobile Computing*, vol. 19, pp. 47–70, 2015.

[189] National Institute on Aging, *Talking with Patients about Cognitive Problems*. NIH Publication, 2011.

[190] M. Gupta, J. Gao, C. Aggarwal, and J. Han, "Outlier detection for temporal data," *Data Mining and Knowledge Discovery*, vol. 5, no. 1, pp. 1–129, 2014.

[191] X. Xu, "Sequential anomaly detection based on temporal-difference learning: Principles, models and case studies," *Applied Soft Computing*, vol. 10, no. 3, pp. 859–867, 2010.

[192] Y. Li, M. Thomason, and L. E. Parker, “Sequential anomaly detection using wireless sensor networks in unknown environment,” in *Human Behavior Understanding in Networked Sensing*, pp. 99–123, Springer, 2014.

[193] V. Chandola, A. Banerjee, and V. Kumar, “Anomaly detection: A survey,” *ACM Computing Surveys*, vol. 41, no. 3, pp. 15:1–15:58, 2009.

[194] K. Park, Y. Lin, V. Metsis, Z. Le, and F. Makedon, “Abnormal human behavioral pattern detection in assisted living environments,” in *Proceedings of the 3rd International Conference on PErvasive Technologies Related to Assistive Environments*, p. 9, ACM, 2010.

[195] T. Zhao, H. Ni, X. Zhou, L. Qiang, D. Zhang, and Z. Yu, “Detecting abnormal patterns of daily activities for the elderly living alone,” in *International Conference on Health Information Science*, pp. 95–108, Springer, 2014.

[196] T. V. Duong, H. H. Bui, D. Q. Phung, and S. Venkatesh, “Activity recognition and abnormality detection with switching hidden semi-markov model,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 838–845, 2005.

[197] N. El-Kechaï and C. Després, “A plan recognition process, based on a task model, for detecting learner’s erroneous actions,” in *Intelligent Tutoring Systems*, pp. 329–338, Springer, 2006.

[198] J. Yin, Q. Yang, and J. Pan, “Sensor-based abnormal human-activity detection,” *Knowledge and Data Engineering*, vol. 20, no. 8, pp. 1082–1090, 2008.

[199] T. Fawcett, “An introduction to ROC analysis,” *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006.

[200] T. K. Hui, R. S. Sherratt, and D. D. Sánchez, "Major requirements for building smart homes in smart cities based on internet of things technologies," *Future Generation Computer Systems*, vol. 76, pp. 358–369, 2016.

[201] D. Barber, *Bayesian reasoning and machine learning*. Cambridge University Press, 2012.

[202] A. S. Crandall and D. J. Cook, "Using a hidden markov model for resident identification," in *Proceedings of the 6th International Conference on Intelligent Environments*, pp. 74–79, IEEE Computer Society, 2010.

[203] A. Benmansour, A. Bouchachia, and M. Feham, "Modeling interaction in multi-resident activities," *Neurocomputing*, vol. 230, pp. 133–142, 2016.

[204] L. Wang, T. Gu, X. Tao, H. Chen, and J. Lu, "Recognizing multi-user activities using wearable sensors in a smart home," *Pervasive and Mobile Computing*, vol. 7, no. 3, pp. 287–298, 2011.

[205] Y.-T. Chiang, C.-H. Lu, and J. Y.-J. Hsu, "A feature-based knowledge transfer framework for cross-environment activity recognition toward smart home applications," *IEEE Transactions on Human-Machine Systems*, vol. 47, no. 3, pp. 310–322, 2017.

[206] Y. Liu, D. Ouyang, Y. Liu, and R. Chen, "A novel approach based on time cluster for activity recognition of daily living in smart homes," *Symmetry*, vol. 9, no. 10, 2017.

[207] R. Chen and Y. Tong, "A two-stage method for solving multi-resident activity recognition in smart environments," *Entropy*, vol. 16, no. 4, pp. 2184–2203, 2014.

[208] H. Fang and C. Hu, "Recognizing human activity in smart home using deep learning algorithm," in *33rd Chinese Control Conference (CCC)*, pp. 4716–4720, IEEE, 2014.

[209] L. Zhang, X. Wu, and D. Luo, "Human activity recognition with HMM-DNN model," in *International Conference on Cognitive Informatics & Cognitive Computing (ICCI*CC)*, pp. 192–197, IEEE, 2015.

[210] J. Ye and G. Stevenson, "Semantics-driven multi-user concurrent activity recognition," in *International Joint Conference on Ambient Intelligence*, pp. 204–219, Springer, 2013.

[211] M. A. U. Alam, N. Roy, A. Misra, and J. Taylor, "CACE: Exploiting behavioral interactions for improved activity recognition in multi-inhabitant smart homes," in *International Conference on Distributed Computing Systems*, pp. 539–548, IEEE, 2016.

[212] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pp. 1137–1143, 1995.

[213] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[214] A. Gepperth and B. Hammer, "Incremental learning algorithms and applications," in *European Symposium on Artificial Neural Networks (ESANN)*, 2016.

[215] C.-H. Lu, Y.-C. Ho, Y.-H. Chen, and L.-C. Fu, "Hybrid user-assisted incremental model adaptation for activity recognition in a dynamic smart-home environment," *IEEE Transactions on Human-Machine Systems*, vol. 43, no. 5, pp. 421–436, 2013.

[216] Z. Zhao, Z. Chen, Y. Chen, S. Wang, and H. Wang, "A class incremental extreme learning machine for activity recognition," *Cognitive Computation*, vol. 6, no. 3, pp. 423–431, 2014.

[217] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1-3, pp. 489–501, 2006.

[218] Z. Wang, M. Jiang, Y. Hu, and H. Li, "An incremental learning method based on probabilistic neural networks and adjustable fuzzy clustering for human activity recognition by using wearable sensors," *IEEE Transactions on Information Technology in Biomedicine*, vol. 16, no. 4, pp. 691–699, 2012.

[219] C. Hu, Y. Chen, L. Hu, and X. Peng, "A novel random forests based class incremental learning method for activity recognition," *Pattern Recognition*, vol. 78, pp. 277–290, 2018.

[220] P. E. Utgoff, "Incremental induction of decision trees," *Machine Learning*, vol. 4, no. 2, pp. 161–186, 1989.

[221] P. Valtchev, D. Grosser, C. Roume, and M. R. Hacene, "Galicia: an open platform for lattices," in *International Conference on Conceptual Structures*, pp. 241–254, Shaker Verlag, 2003.

[222] K. Bouchard, J. Hao, B. Bouchard, S. Gaboury, M. Tarik Moutacalli, C. Gouin-Vallerand, H. Kenfack Ngankam, H. Pigot, and S. Giroux, *The Cornerstones of Smart Home Research for Healthcare*, vol. 93. Springer, 2018.