Process-aware Pattern Recognition and Deviation Detection Under Uncertainty

Jiawei Zheng

Doctor of Philosophy

Artificial Intelligence Applications Institute
School of Informatics
University of Edinburgh

2024

Abstract

Humans naturally follow specific patterns in their daily activities, whether driven by best practices, physical layouts, social norms or established processes or routines. Monitoring and understanding these human activities is a challenging problem, with the ultimate goal of identifying anomalies from desired behaviour, such as those related to healthy living or efficient working. Recognising process patterns and routines, and detecting anomalies can provide crucial insights. However, this presents significant challenges across different domains.

Varying structures of processes in different domains complicate the task of capturing and analysing these patterns, as well as understanding the multifaceted nature of anomalies. Some domains are structured, like manufacturing, therefore it is easier to detect workflow patterns and deviations from them, but others are semi-structured or unstructured, like daily living. In the latter, even defining what an anomaly means is a challenge. Moreover, data are often filled with noise and uncertainties, which may lead to misinterpretation of normal occurrences as anomalies.

To address these challenges, we first discuss the characteristics of different types of domains, i.e., structured, semi-structured, and unstructured domains. We delve into the diverse sources of anomalies across different domains, which present unique challenges in the identification and interpretation of anomalies. Then we propose four approaches to integrating process information into pattern recognition and anomaly detection in different domains.

Firstly, in structured domains, we propose a workflow recognition approach that can automatically correlate the generated data with its corresponding processes, which lays the foundations for leveraging process information for further analysis. Secondly, we propose a process-aware deviation detection approach, specifically designed to operate effectively with uncertain data. Thirdly, we propose a process-driven approach to recognising human activities, aimed towards leveraging process information from semi-structured domains to enhance the accuracy of activity recognition. Fourthly, in unstructured domains, we propose a temporal pattern recognition and anomaly detection approach with a focus on the domain of Activities of Daily Living (ADLs). The proposed approach involves identifying both short-term and long-term deviations in daily activities, as well as detecting changes of complex behaviours over time.

In summary, this thesis underscores the effectiveness of incorporating process information to interpret patterns and detect anomalies in different domains.

Lay Summary

Humans naturally follow specific patterns in their daily lives, which can be influenced by best practices, physical layouts, social norms, or established processes. Understanding and identifying these patterns, and spotting when something deviates from the expected are crucial for improving efficiency, enhancing productivity and promoting health.

A major challenge of this task is that different areas of life and work have different structures, which can make it difficult to capture process patterns and identify anomalies from those. For instance, in highly structured domains, such as manufacturing, it is easier to spot deviations because there are clear, established processes. However, in less structured settings, like someone's daily routine at home, there is no clear baseline of "normal" behaviour, making it difficult to pinpoint what counts as an anomaly. Another challenge is data collected by sensors during daily life are noisy and uncertain. This complicates the task of determining whether a deviation is merely noise caused by the sensors or a genuine anomaly.

This thesis introduces Artificial Intelligence (AI)-based strategies designed to incorporate process information for recognising patterns and detecting anomalies in different types of domains, varying from structured, and semi-structured to unstructured domains. In structured domains, we present a method for correlating data with specific processes, which helps in accurately tracking progress. We also present a method for effectively identifying deviations under uncertain data.

In semi-structured domains, which have some level of organisation but are not fully predictable, we introduce a technique that uses process information to enhance the accuracy of identifying what activities are being performed.

In unstructured domains, such as everyday living activities, the thesis offers a new approach to recognising daily patterns and identifying anomalies and changes in behaviours based on data collected by unobtrusive ambient sensors in smart homes. This approach offers valuable insights into the well-being of individuals, which can be incredibly useful for the individuals themselves or their caregivers.

Acknowledgements

First of all, I would like to give my great thanks to my supervisors, Dr. Petros Papapanagiotou, Prof. Jacques Fleuriot and Prof. Jane Hillston. Thanks for their continuous support, patience, encouragement and hands-on advice. Their guidance not only helped me in all the time of research but also benefited me in personal improvement.

Secondly, I would like to thank my family for their endless support and encouragement. I am grateful to my parents, my sister and my partner Xuerui Yi for their love and support.

I would like to express my gratitude to my examiners Prof. Juliana Bowles and Prof. Dave Robertson for making the viva an enjoyable experience while providing crucial and insightful comments. I would also like to thank a number of people who contributed ideas and input, whether in the form of detailed feedback or with short but constructive comments, including but not limited to Prof. Bob Fisher, Dr. Valerio Restocchi, Dr. Luo Mai and Dr. Nadin Kokciyan.

I would also like to thank a number of people in the AI Modelling Lab (AIML) for stimulating discussion and encouragement, including but not limited to Dr. Ricardo Contreras, Dr. Guillermo Romero Moreno, Dr. Jorge Gaete Villegas, Filip Smola, Ramon Fernández Mir, Zonglin Ji, Lauren DeLong, Konstantin Georgiev, Fiona Smith, Richard Schmoetten, Simon Yu, Leo Kravtchin, Dr. Paola Galdi, Dr. Jake Palmer, Dr. Luna De Ferrari and Dr. Mark Chevallier.

Next, I would like to thank my lab mates and friends for the stimulating discussions and all the fun.

Last but not least, I would like to thank the funding bodies for my research, the School of Informatics, University of Edinburgh. I have also received funding from Artificial Intelligence and its Applications Institute, Advanced Care Research Centre and the Scottish Informatics and Computer Science Alliance.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Jiawei Zheng*)

Table of Contents

Chapter 1

Introduction

In the modern digital world, recognising patterns and detecting anomalies in human activities can ensure efficient operations, enhance productivity and promote human health (Theodoridis and Koutroumbas, 2006; Chandola et al., 2009). For example, in the context of manufacturing, consistent patterns of machine operations indicate that the machine is operating predictably, while any deviations or anomalies could be indicative of malfunctions or operational inefficiencies. Early detection of these anomalies can reduce downtime and prevent waste of resources. Similarly, in the context of daily living, identifying unusual patterns in routine activities, such as abrupt changes in physical activity or sleeping patterns, can provide early warnings of health issues or deteriorating conditions, allowing for timely intervention and promoting better health outcomes.

Humans inherently follow distinct patterns in their activities and behaviours, driven by established best practices or workflows, physical needs, and adherence to social norms. For instance, in manufacturing, production lines are organised according to workflows in order to build a specific part or product from raw materials, while ensuring quality standards. In the context of healthcare, when a patient presents specific symptoms, healthcare professionals rely on established processes to diagnose and treat them (Meier et al., 2015). Even Activities of Daily Living (ADLs) are influenced by processes. For instance, people tend to follow routines in their daily lives, such as a morning routine involving waking up, personal hygiene, eating breakfast, and leaving for work. Also, the activity of preparing a meal might involve a process, following a recipe, such as selecting ingredients, preparing, cooking, and serving (Tabira et al., 2022). These processes, while predictable to an extent, can vary or diverge much more than the rigid processes of manufacturing.

In traditional computational methods of pattern recognition and anomaly detection, such as those involving machine learning and deep learning, patterns and anomalies are identified based solely on the inherent traits and relationships within the data (Liu et al., 2022; Gupta et al., 2022). These methods do not consider the external *process* context in which the data was generated or used, because there is no standard methodology for integrating such contextual information (Dash et al., 2022). The absence of process context consideration can impact the efficacy of these methods, particularly in situations where data are noisy and uncertain, or where behaviour patterns exhibit high variability.

For example, in the advanced application of Industry 4.0, factories want to measure efficiency, track asset movements, measure operation timings, and detect errors and incorrect operations. This can be achieved by integrating Internet of Things (IoT) systems. One example technique is WiFi localisation, which uses the strength and origin of WiFi signals to determine the physical location of devices within a specific area (Hayward et al., 2022). Incorrect localisation may lead to several issues, such as false positives, inaccurate efficiency measurements, and operational disruptions. Incorrect localisation is often due to uncertainty and noise in the data caused by environmental factors such as signal disturbances, as well as the inherent lack of precision in WiFi systems for detecting small distances. For instance, if an asset supposed to move from assembly to testing area is mistakenly detected at the nearby welding station, it might be flagged erroneously as a deviation from the norm. The computational results from localisation algorithms are inherently uncertain, typically providing probabilities of different locations (e.g., a higher probability of the asset being at the welding station versus a lower probability at the testing area). By integrating process context into the algorithm, such as anticipating that the asset should move to the testing area next, the system can adjust its inference and prevent this mislocalisation and false positive alerts.

Moreover, incorporating process context can better recognise human activities. For example, the activity of *drinking from a cup* and *answering a phone* may exhibit similar motion patterns and partially share the same movements (Gupta and Davis, 2007), making it challenging to distinguish between these two activities based solely on motion data. If process information is integrated, such as an individual's typical pattern of drinking from a cup after picking it up, or the subsequent action of putting down the cup, this process can be used to make more accurate inferences about the current activity. For example, if the previous activity is *picking up a cup*, it could be inferred that the individual is more likely drinking rather than answering the phone.

However, harnessing the power of process information comes with its challenges. The vast complexity and variability of real-life processes across different domains, influenced by different objectives and various levels of constraints (Pesic, 2008), makes it challenging to extract and model process information. In some domains, the underlying processes are structured with strict constraints governing how the execution should be performed. For example, in the context of manufacturing, each step in a production process is clearly defined, and workers follow a specific sequence to assemble products efficiently (Papapanagiotou et al., 2021). In unstructured domains, there are very few constraints on behaviours. These domains are characterised by a wide range of potential behaviours and paths to reach the end result. For example, ADLs can be considered an unstructured domain, since individuals may perform daily activities and spend their days in various ways based on their needs and preferences, rather than a pre-defined structured workflow (Banovic, 2017; Aminikhanghahi et al., 2019). Therefore, each type of domain presents a unique set of challenges when it comes to leveraging process information to interpret patterns and detect anomalies.

This thesis sets out to tackle those challenges in order to interpret patterns and detect anomalies across the spectrum of different domains. We acknowledge the immense complexity of problems in different domains and it is a massive undertaking fraught with numerous variables, settings, and goals. Our research focuses on addressing specific key challenges, informed by real-world examples and problems encountered in practical situations. Our goal is to make significant contributions and pave the way forward in this critical area of study.

This chapter discusses the characterisation of different types of domains in Section 1.1. Then we discuss diverse sources of anomalies in different domains in Section 1.2. With the related concepts being addressed, we present the research problems and challenges in Section 1.3. We introduce the objectives and contributions of this thesis in Section 1.4 and Section 1.5, respectively. Finally, the structure of this thesis is presented in Section 1.6.

1.1 Characterisation of domains

As hinted from our discussion for far, domains can be categorised into three types based on the structure of their underlying processes: structured, semi-structured and unstructured. This classification helps us understand the structure of processes within each domain and aids in developing appropriate strategies for incorporating process information to interpret patterns and detect anomalies in data. In the following subsections, we describe the key characteristics of each type.

1.1.1 Structured domains

In structured domains, underlying processes are characterised by well-defined, orderly, and organised procedures designed to accomplish specific objectives. These processes are governed by explicit rules, constraints, and standards that outline each step and dictate how execution should progress to complete tasks consistently and efficiently. In these domains, all activities are specified to follow pre-determined paths for execution. For example, in the domain of manufacturing, each step in the production process is clearly defined, and workers follow a specific sequence to assemble products efficiently. The key characteristics of structured domains are as follows.

- Step-by-step approach: A structured process breaks down a complex task into smaller, more manageable steps, with each step designated to accomplish specific tasks.
- Constrained: A set of rules and constraints govern each step of the process to complete tasks.
- Well-defined inputs and outputs: The process specifies the inputs required at each stage and the expected outputs or deliverables.
- Repeatability and consistency: Structured processes are designed to be repeatable and produce consistent results.

1.1.2 Semi-structured domains

Processes in semi-structured domains offer a level of flexibility that lies between the strictness of structured domains and the absence of constraints in unstructured domains. They involve a mix of pre-defined rules and constraints, as well as elements that do not strictly conform to a fixed sequence or rigid structure. Semi-structured processes exist in domains where data or information has some inherent structure, but not all aspects can be precisely defined in advance. For example, the act of eating can be considered a semi-structured process, which we will discuss in Chapter 5, because there are certain basic steps involved in eating, such as selecting a utensil, using it to pick up food, then putting food in the mouth, etc. However, the overall process allows for flexibility and variations depending on individual preferences, the environment or personal capabilities. For instance, rather than adhering strictly to the use of utensils, an individual might choose to drink directly after picking up utensils or simply opt to eat with their hands.

1.1.3 Unstructured domains

In unstructured domains, there is a lack of constraints governing how the execution should be performed. The processes of performing a task are driven by experience, intuition, or rules-of-thumb. The steps and sequences of actions may vary based on changing circumstances or the discretion of individuals involved. Such processes are characterised by their flexibility, with multiple potential paths leading to desired outcomes. For example, ADLs, which we will encounter later in Chapter 6, can be considered an unstructured domain. Different individuals' daily routines and activities can significantly differ, shaped by their individual needs and preferences, rather than following a predetermined, structured workflow.

The key characteristics of unstructured domains include:

- Flexibility and variability: Processes can adjust depending on conditions, discretion and judgement of individuals involved.
- Lack of constraints: Execution of processes relies on discretion and judgement rather than pre-defined constraints.
- Non-linear progression: Activities in a process do not follow a fixed sequence, and different paths can be taken to reach the same outcome.
- Unpredictability: It is difficult to determine in advance the way the process will be executed.

1.2 Different sources of anomalies

Anomalies, also known as *outliers* or *deviations*, are data points or patterns that significantly differ from the normal or the expected behaviour within a given domain. Anomalies have been categorised into *point anomaly*, *collective anomaly* and *contextual anomaly* in (Chandola et al., 2009; Ibidunmoye et al., 2015). Point anomalies refer to a data instance that is much different from others. Collective anomalies refer to a group of data whose patterns or properties deviate significantly from groups of other data. When contextual information is considered for anomaly detection, the detected point anomaly is called a contextual anomaly. Therefore, according to whether considering contextual information, anomaly detection can be categorised into *data point anomaly* and *contextual anomaly*. In this thesis, our focus is predominantly on the detection of contextual anomalies.

Contextual anomalies, unlike data point anomalies that stand out based solely on the characteristics of values, are identified by considering the context in which the data are generated and used. Incorporating contextual information can improve the accuracy of anomaly detection. For example, in the domain of ADLs, an instance of late sleep for an individual might be mistakenly flagged as an anomaly without acknowledging the context, such as whether the individual typically sleeps late on weekends. This behaviour would not be considered anomalous but rather a normal part of their weekly routine.

In different domains with varying structures of underlying processes, anomalies can arise from different perspectives and sources.

In domains with constraints governing the process execution, i.e. structured and semi-structured domains, anomalies typically refer to instances where the actual execution of a process deviates from the pre-defined or expected sequence of steps and activities. When the execution of a process deviates from its predefined sequence of activities, this results in a *control-flow deviation*. For example, in the context of manufacturing, an assembly line follows a specified standard protocol that outlines the exact sequence of steps required to assemble a product. A control-flow deviation might occur if a critical component is installed at a different stage than prescribed by the protocol, potentially leading to assembly errors or quality issues in the final product.

In unstructured domains, on the other hand, the lack of constraints and a high degree of flexibility of execution mean that there is no clear baseline representing what is considered normal behaviour. For example, in the context of ADLs, people can perform their daily lives in various ways. Therefore, anomalies can originate from a variety of sources in unstructured domains, making it particularly challenging to define what constitutes an anomaly.

For instance, the emergence of novel behaviour or patterns within the data itself can be a source of anomalies. Consider an elderly person who has a consistent pattern of going to bed by 9 PM and waking up at 6 AM every day. If there is a change where the individual starts staying up until midnight or later and begins sleeping until late in the morning, this shift in sleep pattern represents a novel behaviour that deviates from their established routine.

Moreover, what constitutes an anomaly can vary significantly from one individual to another. A change in sleep patterns that is concerning for an elderly person might be perfectly normal for a younger individual, whose lifestyle and obligations might naturally accommodate later nights and mornings. This variability of anomalies of ADLs underscores the importance of understanding the individual's baseline and the context of their behaviour.

Anomalies in ADLs can arise not only from individual activities but also from complex behaviour that involves multiple interconnected activities. For example, consider the behaviour pattern of medication intake. Typically, an individual might have a routine where they take medication before a meal. If there is a deviation in this pattern, such as taking medication after meals or inconsistently adhering to the prescribed pattern over a period, it could indicate an anomaly. This change might be due to forgetfulness, a change in medical instructions, or other health-related issues. Such anomalies, which involve a sequence of activities, can often provide more significant insights into a person's well-being than deviations in single activities alone. We elaborate on the detailed challenges of anomaly detection in ADLs in Chapter 6.

1.3 Research problems and challenges

Given the characteristics of different domains and various sources of anomalies, we identify several challenges regarding incorporating process information for pattern recognition and anomaly detection in different domains.

- **Diverse process structures**: Firstly, different structures of underlying processes within domains complicate the task of capturing and modelling process information. For example, in semi-structured domains, there is a mix of flexibility in operations and structured processes governing activities. This flexibility within the structure requires an approach that can adapt to both expected process flows and flexible elements. Moreover, in unstructured domains, specifically ADLs, modelling complex behaviours that involve multiple interdependent activities poses challenges due to the inherent flexibility of these domains. Effective modelling of these behaviours requires more than simply understanding the sequence of activities; it must also account for the flexibility with which these activities are carried out. For instance, consider the routine of taking medication before a meal. While it is crucial that the medication is taken prior to eating, there is no fixed rule that it must be taken immediately before the meal. Someone might take their medication, engage in various other activities such as watching TV or relaxing, and then eat their meal. This means that a rigid model that expects medication to be taken immediately before a meal might incorrectly flag normal behaviour as anomalous.

- **Noisy and uncertain data**: Secondly, generated data is noisy and uncertain, complicating the distinction between genuine anomalies and normal occurrences. Historically, collecting data and understanding the patterns of what people were doing was primarily based on self-report. Individuals would manually log their activities, often in written formats, which was time-consuming and prone to human errors or basis (Cleland et al., 2014; Yaya et al., 2020). With the advent of the Internet of Things (IoT) and sensor technologies, data can be collected automatically by sensors during the execution of the processes (Schönig et al., 2020). For example, wearable devices can generate real-time data reflecting heart rate, steps taken, etc. RFID sensors can be used to track the movement of assets, enabling location tracking. However, the data collected by sensors is also typically noisy with errors caused by various factors, such as inaccuracy or malfunctions of sensors, etc. (Liu et al., 2020). For instance, a sensor may generate data with incorrect timestamps, leading to an apparent lack of synchronisation in the data. This misalignment makes it difficult to correlate the generated data with process execution accurately, leading to erroneous conclusions. For instance, an incorrect timestamp might delay a critical alert, leading to missed interventions.

  Moreover, due to the inherent uncertainty of machine learning and deep learning methods, the predictions or classifications they produce are often accompanied by a level of confidence, rather than absolute certainties, which results in uncertain data (Guo et al., 2017; Ghahramani, 2015). For example, in the context of Human Activity Recognition (HAR), it is challenging to distinguish activities with similar motion characteristics due to inter-class similarity (Akila and Chitrakala, 2019), such as activities of *answering a phone* and *drinking from a cup* (Gupta and Davis, 2007). Detecting anomalies without considering the uncertainty of data may lead to erroneous flags or missed detection of anomalies.

A misclassification of *drinking from a cup* as *answering a phone* may lead to a false alert of anomaly, for instance, related to the level of hydration. In high-stakes domains, like healthcare, and security, such precision becomes even more crucial, as errors in these can have significant influences.

Process mining techniques enable the leverage of the data created during the execution of specific tasks to understand how processes behave and unfold in reality (van der Aalst, 2016). However, there has been little effort to handle the uncertain data in process mining approaches (Felli et al., 2022).

- **Complex data-process mapping**: When incorporating process information, mapping the generated data with its corresponding process is a problem in scenarios with complex tasks that require the coordination of multiple workflows. The completion of each task in these scenarios generates multiple event sequences. Identifying which specific workflow an event sequence belongs to can be particularly problematic for organisations. This issue complicates efforts to integrate process information for deviation detection and pattern recognition. For example, in a hospital setting, cancer treatment encompasses multiple stages, such as consultation, diagnosis, and follow-up. Patients undergo numerous clinical tests according to a distinct workflow at each stage. However, with just a record of the tests a patient has undergone, physicians often struggle to swiftly pinpoint the specific phase of therapy the patient is currently navigating. Therefore, it may result in delays in treatment and monitoring of the patient's progress.

- **Anomaly detection amid varying process constraints**: The level of constraints governing process execution can impact the identification of anomalies in different domains. In structured domains, behaviour and activities are typically governed by underlying processes, that delineate what is considered normal or expected behaviour. When the observed behaviour diverges from the process, it is flagged as an anomaly. However, unstructured domains lack pre-defined rules or clear baseline models that represent normal behaviour. This makes it challenging to identify anomalies because there is no standard baseline for comparison. Moreover, unstructured domains often experience a lack of labelled data, primarily because it is hard to label anomalies. The scarcity of labelled data complicates the use of supervised learning techniques for anomaly detection.

  For example, in the context of ADLs, there is a considerable degree of variability and flexibility in how an individual conducts their daily activities. This diversity in daily routines means that what constitutes an anomaly for one individual might be a regular occurrence for another. For example, for one person, going to bed late and waking up late might be their norm, while for another, such a pattern could signify a deviation from their usual early bedtime and morning rise. Or sleeping late may just be part of the individual's weekly routine on the weekends. Therefore, identifying anomalies of ADLs is particularly difficult.

In summary, each type of domain presents unique challenges for pattern recognition and anomaly detection, stemming from various factors. These include the absence of defined constraints that delineate expected behaviour, inherent data noise and uncertainty, and the varying structure of underlying processes. This thesis aims to address these challenges, developing methods that can effectively incorporate process information for interpreting patterns and detecting anomalies in structured, semi-structured, and unstructured domains.

1.4 Objectives

The aim of this thesis is to propose practical methods for pattern recognition and anomaly detection by incorporating process information in different domains. To address the problems and challenges mentioned in the previous section, we will introduce methods to

(i) establish the groundwork for further analysis that involves the incorporation of process information, which entails mapping generated data with its corresponding processes;

(ii) augment the capabilities of existing process mining techniques to handle uncertain data;

(iii) enhance the accuracy of existing human activity recognition approaches that rely solely on data-driven models by incorporating process information in semi-structured domains;

(iv) tackle the difficulties associated with identifying anomalies within unstructured domains.

Further details about the contributions are outlined in Section 1.5.

1.5 Contributions

We set out to explore the possibilities of how to leverage process information to address the pattern recognition and anomaly detection problems over noisy and uncertain data in various domains. We give a brief overview of the key contributions of this thesis below. For structured domains, our contributions are the following, which correspond to Chapter 3 and Chapter 4, respectively.

- **Workflow recognition approach.** We propose a novel framework for automatically recognising the specific workflow being executed based on noisy data generated by sensors, which aims to achieve Objective (i). The proposed framework provides us with passive monitoring of the operating workflows, which enables us to accurately recognise the specific workflow in operation. This work lays the fundamental work for further studies aimed to incorporate process information for detecting deviation in Chapter 4 and for enhancing activity recognition in Chapter 5.

- **Deviation detection approach over probabilistic events.** We extend an alignment conformance checking algorithm to function under probabilistic event data, aimed at Objective (ii). The proposed algorithm can leverage the process knowledge captured in the process model to address the levels of uncertainty in the event data. It can effectively enhance the accuracy of deviation detection, achieving a lower rate of both false positives and false negatives in comparison to similar methods. This work is based on the published conference paper (Zheng et al., 2024).

  J. Zheng, P. Papapanagiotou, and J. D. Fleuriot, "Alignment-based conformance checking over probabilistic events", In Proceedings of the 57th Hawaii International Conference on System Sciences, Jan. 2024.

For semi-structured domains, we present the following contribution, which corresponds to Chapter 5:

- **Process-driven human activity recognition approach.** We present an innovative framework that merges structured process information within the semi-structured domains and the probabilistic output of machine learning-based classification models. By constructing an alignment between processes and the probabilistic output of predicted classes, the approach can enhance the performance of classification by considering the classes with lower probability but can better align with the processes. This work aims towards Objective (iii).

For unstructured domains, our contributions are as follows, which aim to achieve Objective (iv) and are detailed in Chapter 6:

- **Temporal pattern recognition and deviation detection approach.** We develop a predictive approach that can identify temporal patterns based on historical data, as well as an anomaly detection approach based on identified patterns. We focus on identifying both short-term deviations and long-term patterns. This work is based on the published conference paper (Zheng and Papapanagiotou, 2022).

  J. Zheng and P. Papapanagiotou, "Predictive Behavioural Monitoring and Deviation Detection in Activities of Daily Living of Older Adults", in 15th International Joint Conference on Biomedical Engineering Systems and Technologies, Volume 5 - HEALTHINF, Feb. 2022, pp. 899–910.

- **Behavioural changes detection approach.** We propose a probabilistic model checking-based approach that quantitatively assesses the probability of certain behaviours over different periods. The proposed approach can model the daily routine and assess the probabilities of complex behaviours in a formal way. Additionally, a statistical approach is proposed, which allows us to detect abrupt behaviour changes and identify periods in which behaviours deviate from the others.

1.6 Thesis structure

The structure of the rest of the thesis is organised as follows:

- In Chapter 2, we describe basic concepts and methods involved in this thesis, including pattern recognition, anomaly detection, and process mining.

- In Chapter 3, we present the workflow recognition approach in structured domains. We also conduct an experimental study on a real-life manufacturing dataset.

- In Chapter 4, we present our deviation detection approach over probabilistic events in structured domains. We compare the accuracy of deviation detection with two other similar approaches based on two commonly used process mining datasets.

- In Chapter 5, we present the approach of process-driven human activity recognition in semi-structured domains. We evaluate the effectiveness of our approach based on a real-life dataset that is used for recognising eating behaviours.

- In Chapter 6, we present our approach for detecting temporal anomalies and behavioural change in unstructured domains. We first introduce the datasets on which we conduct experimental studies. Next, we present the temporal pattern recognition and deviation detection approach. We then present the approach of behavioural change detection and its evaluations.

- In Chapter 7, we conclude the thesis by summarising our work and discussing potential future work.

Chapter 2

Background

In this chapter, we give a brief overview of the foundational concepts underpinning this thesis, namely pattern recognition and anomaly detection. In addition, we introduce the background of process mining, which is the primary technique investigated in this thesis.

2.1 Pattern recognition

Pattern recognition is concerned with how machines can observe the environment, learn to identify and discover regularities, and make decisions of categorising patterns (Theodoridis and Koutroumbas, 2006). Pattern recognition is an important problem in a variety of domains, such as tumour detection in healthcare (Saba et al., 2020), recognising and locating objects within images (Amit et al., 2020), analysing heart rhythm patterns of a patient (Teplitzky et al., 2020), etc. Given a pattern recognition problem, it can be categorised as supervised learning or unsupervised learning according to the type of learning procedure used to generate the output value (Sathya et al., 2013).

2.1.1 Supervised learning

Supervised learning assumes that a set of labelled data is available, consisting of a set of instances as input (termed *features*) with properly labelled correct output (referred to as *labels*). A learning procedure creates a model that attempts to generate the expected output for each instance in the training dataset. This procedure is called model training. The model learns the relationship between the features and their corresponding labels.

Once a model is trained, it can be used to predict outcomes for new, previously unseen instances of data.

Two major tasks of supervised learning are *classification* and *forecasting*. Classification aims to determine a category label for a given input instance, based on past observations. Examples include email spam detection (spam or not spam) (Crawford et al., 2015), image recognition (identifying objects, people or activities in images) (Amit et al., 2020), etc. In Chapter 5, we investigate the application of classification models for recognising human activities. The goal of forecasting is to predict continuous numeric values for a given input instance. Examples include predicting house or stock prices based on various features that may influence these (Lu et al., 2021). In Chapter 6, we investigate the forecasting methods aimed at predicting daily behaviours, such as sleep duration.

The output of classification tasks is a class label, indicating which category a data instance belongs to. The most commonly used models for classification tasks include Decision Trees, Support Vector Machines, Random Forests, Gradient Boosting Machines (such as eXtreme Gradient Boosting (XGBoost) and Light Gradient Boosting Machine (LightGBM)), Neural Networks, etc (James et al., 2023).

In evaluating the performance of classification models, we often use specific terms to describe the outcomes of model predictions relative to actual labels, including *true positives* (TP), *false positives* (FP), *true negatives* (TN), and *false negatives* (FN). True positives are the cases where the model correctly predicts positive classes. False positives are the cases where the model incorrectly predicts positive classes. True negatives are the cases where the model correctly predicts negative classes. False negatives are the cases where the model incorrectly predicts the negative classes.

Based on these terms, we can derive several key metrics used to measure the performance of a classification model, such as *Accuracy*, *Precision*, *Recall*, *F1-score*, etc. (Lever, 2016).

Accuracy measures the proportion of total corrected predictions. It is calculated using the equation (2.1). Precision assesses the accuracy of positive predictions, which is calculated as equation (2.2). Recall measures the model's ability to correctly identify all relevant instances within a dataset, calculated by equation (2.3). F1-score is the harmonic mean of precision and recall, providing a single score that balances both precision and recall. It is calculated by equation (2.4).

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \tag{2.1}$$

$$Precision = \frac{TP}{TP+FP} \tag{2.2}$$

$$Recall = \frac{TP}{TP+FN} \tag{2.3}$$

$$F1\text{-}score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{2.4}$$

Forecasting is the process of making predictions about future values based on historical data. The methods for forecasting tasks include time series methods, such as autoregressive integrated moving average (ARIMA) (Ho and Xie, 1998) and Prophet (Taylor and Letham, 2017), and regression models, such as Linear regression, Logistic regression, etc. (James et al., 2023). Error metrics are often used to measure how well the forecasting model's prediction aligns with the actual outcomes, such as Mean Absolute Error (MAE), and Root Mean Squared Error (RMSE) (Botchkarev, 2018).

2.1.2 Unsupervised learning

Unsupervised learning, on the other hand, assumes that there is no labelled output data to guide the learning procedure (Barlow, 1989). It aims to identify inherent patterns, structures or relationships within the data itself.

Compared to supervised learning, unsupervised learning is often much more challenging. Unsupervised learning is inherently more subjective, lacking a straightforward objective like the prediction of stock prices. Furthermore, it can be harder to assess the results obtained from unsupervised learning methods, since there is no clear benchmark for performing validation (James et al., 2023).

In unstructured domains, presented in Chapter 6, as there are no definite baselines to represent what is considered normal behaviour, therefore, our proposed anomaly detection methods operate in an unsupervised way.

2.2 Anomaly detection

Anomaly detection refers to the problem of identifying data points that do not conform to expected behaviour (Chandola et al., 2009). Anomaly detection has been widely applied in various domains, such as fraud detection for credit cards, intrusion detection for cyber security, and unusual health condition detection in healthcare. Detecting anomalies is crucial because they represent significant, critical, actionable information in a wide variety of application domains. For example, anomalies in credit card transactions could indicate credit card fraud or identity theft (Yang et al., 2019). Anomalies of the heart rate of a patient might indicate arrhythmia, myocardial infarctions, or other cardiac conditions (Šabić et al., 2021).

A straightforward anomaly detection approach is to define a pattern representing normal behaviour and identify any observation in the data that does not fit this pattern as an anomaly. However, it is usually challenging to define a normal pattern. Some datasets contain labels associated with each instance denoting whether that instance is normal or abnormal. In such cases, anomaly detection can be transformed into a supervised classification problem. With labelled data, it is possible to train a classification model that learns the patterns characteristic of both normal and abnormal behaviours. It should be noted that obtaining labelled data that is accurate and represents all types of normal and abnormal behaviours is often expensive because labelling work is often done manually by domain experts and needs substantial effort and time. To overcome the challenges posed by the scarcity of labelled data, one strategy is to establish a pattern of what represents normal behaviour by incorporating contextual information, such as process knowledge from domains, as we discussed in Section 1.2.

On the other hand, anomaly detection can operate in an unsupervised way, which means we do not require labelled train data with normal or abnormal classes (Goldstein and Uchida, 2016). Unsupervised approaches have to analyse the datasets to infer the real sense of abnormality or make an assumption based on specific problems. An example approach in this case is clustering-based anomaly detection, which makes an implicit assumption that normal instances are far more frequent than anomalies in the datasets (Yahaya et al., 2021a). Therefore, these techniques presume data that groups inside small clusters are prone to be anomalous.

Isolation Forest is one of the most commonly used algorithms for unsupervised anomaly detection (Liu et al., 2012). The Isolation Forest is achieved by randomly selecting a feature and then randomly selecting a split value between its maximum and minimum values. This partitioning process can be represented by a tree structure. Because anomalies are more susceptible to being isolated in this random partitioning process, it will produce noticeably shorter paths for anomalies than normal values by recursive partitioning the values. By aggregating the path lengths over a forest of such trees, the algorithm assigns an anomaly score to each data point, i.e., the shorter the path length, the higher the likelihood of the point being an anomaly. We adopt the

Isolation Forest algorithm in Chapter 6 to identify abnormal behaviour.

2.3 Process mining

This thesis primarily investigates process mining techniques for understanding and discovering process information. Process mining aims to extract process knowledge from event logs generated during the execution of the processes in the physical world. It provides insights into the actual operation of processes and allows for a detailed comparison between expected and actual behaviours. By identifying where and when deviations occur, process mining facilitates the identification of inefficiencies, bottlenecks, and non-compliance issues, enabling organisations to undertake targeted improvements. Beyond gaining insights into process execution, process mining has been applied to a variety of domains, such as financial, manufacturing, healthcare, etc., to enhance operational efficiency, process optimisation and compliance monitoring (van der Aalst, 2016).

There are many perspectives in process mining, such as control-flow, resource, data and time (Mannhardt, 2018). Each perspective sheds light on different facets of processes. Control-flow perspective focuses mainly on the ordering of the activities. Essentially, it maps out the route of a process, determining which steps follow others and under what conditions between activities. For example, in a manufacturing process, the control-flow perspective would detail how raw materials are first inspected, then moved to assembly, followed by quality control, and finally shipped to the customer, outlining any conditional branches such as rework or disposal in case of quality issues. The resource perspective aims to understand resource involvement, allocation and interaction in the execution of tasks. It examines how human resources or equipment are utilised and coordinated to achieve the tasks. For example, in a hospital setting, this perspective would analyse how doctors, nurses, medical equipment, and rooms are assigned to treat patients. The data perspective focuses on the data objects involved and produced during the process execution, e.g., compensation amount in financial domains. This can help organisations ensure data integrity and consistency along the process execution. The time perspective is centred on the temporal properties of a process. This involves looking at duration, frequencies, and time intervals between activities. This can help us understand process bottlenecks and idle times, and provide insights on process efficiency. In this thesis, we mainly focus on the control-flow perspective of processes.

There are three main parts of process mining (van Der Aalst and Carmona, 2022): i) *process discovery* learns a process model from an event log through different discovery algorithms; ii) *conformance checking* refers to comparing the observed behaviour in the event logs with process models to identity discrepancies; and iii) *enhancement* enriches and improves the process models based on additional information in the event log, e.g., improving the model to make it better adapted to the changing market conditions. We present the tasks of process discovery and conformance checking in detail in Section 2.3.3 and Section 2.3.4, respectively, as these two tasks are closely related to our work.

2.3.1 Process mining requirements

The basis of process mining is an event log. This stores information about activities that were recorded during the execution of a process. Each execution of a process, referred to as a process instance, generates a sequence of events, referred to as an event trace (van der Aalst, 2016). Therefore, an event log contains multiple event traces.

Events captured in logs can exhibit varying degrees of alignment with the activities they represent in a process. Some events may directly relate to activities in the process. For instance, an event labelled "Invoice Approved" directly mirrors an approval activity within a financial process. However, not all recorded events are at the same granularity as the activities in the process. In some cases, the execution of a single activity may trigger the generation of multiple events in the event log, which do not individually correspond to distinct activities in the process. These are referred to as low-level events (Mannhardt et al., 2016). For example, in a healthcare process, the activity "Patient Treated" might generate several low-level events such as "Medication Administered", and "Vital Signs Checked".

Table 2.1 shows an example of an event log in the domain of handling requests for compensation (van der Aalst, 2016). Each row in the table refers to an instance of an event. Each event in the log has an associated *activity*, such as *register request*, *check ticket*, etc. Each event belongs to a particular *case*, indicating a *process instance*. Moreover, events within a case need to be ordered. In addition to these essential elements of an event, i.e. *activity* and *case*, events may also have additional information. For example, in Table 2.1, the *time* column shows the time the activity was performed, which is usually used to measure performance-related properties, e.g., the average duration of a case. *Resources* refer to the machine or person executing the activity. For example, row 1 in Table 2.1 means that Pete performed an activity of *register request*. Moreover, the table also shows the *cost* associated with each event. This is an example of a data attribute. There may be many other data attributes. For example, in this case, it would be interesting to record the outcome or time spent on the examinations and checks. Another data attribute that could be useful for analysis is the amount of compensation requested. This could be an attribute of the whole case or stored as an attribute of the *register request* activity.

Case ID	Event ID	Time	Activity	Resource	Cost
1	3323	2010/12/11 10:12:11	register request	Pete	50
1	3324	2010/12/11 12:15:13	examine	Mike	400
1	3325	2010/12/12 10:17:23	check ticket	Sara	100
1	3326	2010/12/12 14:45:24	decide	Tom	200
1	3327	2010/12/13 10:18:54	reject compensation	Bob	200
2	3328	2011/01/05 09:23:52	register request	Bob	50
3	3329	2011/01/05 10:45:23	register request	Mike	50
2	3330	2011/01/05 11:25:56	check ticket	Ellen	100
3	3331	2011/01/05 11:45:56	examine	Ellen	400
2	3332	2011/01/05 13:25:26	examine	Pete	400
2	3333	2011/01/05 15:15:36	decide	Sara	200
2	3334	2011/01/06 08:43:52	pay compensation	Mike	150
3	3335	2011/01/07 08:52:12	check ticket	Mike	100
3	3336	2011/01/07 09:32:22	decide	Sean	200
3	3337	2011/01/08 15:29:48	reject compensation	Tom	200
4	3338	2011/01/09 08:53:24	register request	Mike	50
...	...	...	...	...	...

Table 2.1: Example of an event log related to the process of handling compensation requests (van der Aalst, 2016).

Overall, the relationship between these different concepts in event logs is listed as follows:

- A *process* includes many *activities* to be executed according to a particular order.
- A *process* consists of multiple *instances*, referred to as *cases*.
- A *case* consists of many *events*, and each *event* belongs to one case.

- *Events* grouped in one *case* are ordered.
- *Events* are related to *activities*, associating to the *activity* within the *process*.
- *Events* can have many attributes, such as *time*, *costs*, *resources*, etc.

2.3.2 Quality criteria of process mining

Determining the quality of a process mining result is difficult and is characterised by many dimensions. We introduce the four most used quality dimensions: *fitness*, *simplicity*, *precision*, and *generalisation* (van der Aalst, 2016).

The *fitness* dimension measures how well the model can reproduce or "replay" event traces in the event log. A model having good fitness is able to replay most of the traces in the event log. A model has perfect fitness if all traces in the event log can be replayed by the process model from beginning to end.

The *simplicity* dimension assesses whether a process model is as simple as possible. This means that the simplest model that can explain the behaviour seen in the event log, is the best model. One method for measuring simplicity is by analysing the complexity of the process model. For the detailed assessment of model complexity, one can refer to the relevant literature (Mendling et al., 2007).

The *precision* measures the level at which the allowed traces of a process model belong to the original event logs. A process model is *precise* if it does not allow behaviour that is unrelated to what was seen in the event log. In other words, a process model with high precision indicates the allowed traces account for a large proportion of traces in the event log. A process model having poor precision is underfitting, which means that the model over-generalises the example behaviour in the event log. An example of underfitting is that a process model allows for traces very different from what was seen in the event log.

The *generalisation* dimension indicates that the process model should not restrict traces to the examples seen in the event log. A model that does not generalise is overfitting. Overfitting indicates the process model is very specific, only representing a particular sample event log, but another sample event log that belongs to the same process model may not be allowed by this model.

It turns out to be challenging to balance the four quality dimensions. For example, an oversimplified model is likely to have a low fitness or a low precision. Moreover, there is an obvious trade-off between precision and generalisation, which are indicative of overfitting and underfitting, respectively.

After presenting the requirements and quality criteria of process mining, we will next detail two main tasks in process mining: process discovery and conformance checking, which are directly related to this thesis.

2.3.3 Process discovery

A process discovery algorithm takes an event log as input and maps the event log onto a process model such that the model is "representative" of the behaviour seen in the event log (van der Aalst, 2016). The process models can be represented by Petri nets (Peterson, 1977) or Business Process Models and Notations (BPMN) (White, 2004).

To illustrate the principle of process discovery, we show an example of discovering a process model from the event log shown in Table 2.1 (van der Aalst, 2016). This event log contains 4 cases, and 6 activities are executed. Each case starts with the activity of *register request*, and is followed by either *examine* or *check ticket*. The execution of *decide* activity should be done after the completion of *examine* and *check ticket*. These cases end with either *pay compensation* or *reject compensation*. The goal of process discovery is to discover a process model that can represent the behaviour observed in this event log. Based on this information, the process model can be deduced, as shown in Figure 2.1. The discovered process model is represented by a Petri net, which is defined in Definition 1.

Definition 1 (Petri net) *Given a set of activities A, a Petri net is defined as a tuple* $N = (P, T, F, \alpha)$*, where P and T are sets of places and transitions respectively. F is a set of arcs representing flow relations between transitions and places, so that* $F \subseteq (P \times T) \cup (T \times P)$*. A labelling function* $\alpha : T \rightarrow A \cup \{\tau\}$ *assigns either an activity from A or* τ *(immediate transition not associated with any activity) to each transition in* T.

As Figure 2.1 shows, the process starts with *register request*, which is modelled by a *transition* (represented by a square) labelled *register request*. Transitions are connected through *places* (represented by a circle) that model the states of the process. If all the input *places* have a *token* (represented by a dot), the *transition* of the Petri net is enabled to be activated. This means that the corresponding activity is enabled and ready to execute. This is also termed as *firing*. For example, the transition of *register* *request* has only one input place (start) and this place initially has a token. Therefore, the activity of *register request* is ready to execute as the start of the process. When *firing*, the transition consumes a token from input places to generate a token for output places. Hence, the firing of transition *register request* results in the removal of the token from *start* place and the production of two tokens: one in place *c1* and another in place *c2*. These two tokens enable further activities, i.e., *examine* and *check ticket*.

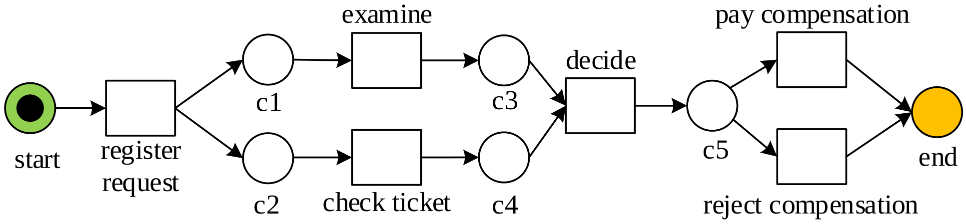


Figure 2.1: Discovered process model.

Examples of algorithms for process discovery on the control-flow perspective include Alpha miner (van der Aalst et al., 2004), Inductive miner (Leemans et al., 2014), Heuristic miner (Weijters et al., 2006), etc. Van der Aalst provides a comprehensive overview of these algorithms, which helps select the appropriate process discovery algorithm for different use cases (van der Aalst, 2016).

The Alpha miner is one of the first process discovery algorithms. However, its applicability is limited due to its sensitivity to noise, its struggle with infrequent paths, and its inability to handle incomplete event traces effectively, making it less practical for real-world applications.

The Heuristic miner represents a significant advancement by taking frequencies of events and sequences into account when constructing a process model. This algorithm uses a dependency threshold to distinguish between frequent and infrequent paths, thereby focusing on the more common behaviours observed in the event logs. This approach allows for a more robust model that better reflects the dominant processes.

The Inductive miner is currently one of the leading process discovery approaches as it was designed to handle a wide range of process behaviours, including complex structures, noise and infrequent paths. This algorithm applies a recursive approach, where the step of splitting the log, constructing process trees for segments, and then integrating these trees into a model is repeated until the entire process is modelled (Leemans et al., 2014).

2.3.4 Conformance checking

Conformance checking techniques need both an existing process model which may have been discovered by algorithms or handmade, and an event log capturing the real execution as the input (Carmona et al., 2018). The goal of conformance checking is to find commonalities and discrepancies between the modelled behaviour in the process model and the observed behaviour in the event log. It can compare how well the event log conforms to the process model. Therefore, conformance checking is used to detect deviations by showing the alignment between real execution and the known process model.

Two well-known conformance checking algorithms are token replay (De Medeiros et al., 2008) and alignment (van der Aalst et al., 2012).

Token replay algorithms replay each trace of the event log on the model, i.e., one event at a time. It records all situations where a transition is forced to fire without being enabled. For example, typically, the replay of a trace against a process model is halted when an issue is encountered, resulting in the trace being deemed non-conforming. However, rather than stopping, the algorithm makes and records a local correction, allowing the replay process to continue. The local correction may be for example to skip/ignore a transition in the process model or to skip/ignore an activity in the log. We can gain insights into how well the event log conforms to the process model by counting the number of recorded corrections.

A general limitation of token replay algorithms is that the error correction is performed locally when an error is encountered. Because the corrections are made locally, the algorithm may not efficiently identify the root cause or the smallest set of changes needed to align the event log with the process model. Essentially, while local corrections can allow the replay to continue, they do not guarantee the identification of the most efficient or minimal set of corrections. Practically, this means that it cannot explain and locate where the deviations between the event log and the process model are.

The alignment algorithm can solve the limitation of the token replay algorithms by finding an optimal alignment between the observed trace and the process model. Therefore, it is guaranteed to find the most consistent path in the model in comparison to the trace of the event log. It can provide better diagnostics and pinpoint non-conforming cases with respect to the process model. Therefore, an alignment conformance checking algorithm is applied in this thesis.

To illustrate the intuition of the alignment algorithm, we consider an example trace $\sigma = \langle register\ request, examine, decide, pay\ compensation \rangle$ and the process model shown in Figure 2.1. We can achieve an alignment, which is shown in Table 2.2. The top row corresponds to the trace σ in the event log and the bottom row corresponds to a path from the start to the end of the process model. If the event in the log and the activity in the model correspond to each other, it means a *synchronous move* between the event log and the process model. For example, the event *register request* in the event log corresponds to the activity *register request* in the process model, which forms a synchronous move.

Event log	register request	examine	$\gg$	decide	pay compensation
Process model	register request	examine	check ticket	decide	pay compensation

Table 2.2: An example of optimal alignment between the given trace σ and the process model shown in Figure 2.1.

The "$\gg$" symbol denotes a misalignment, representing a *move* either in the event log or in the process model. For example, in Table 2.2, the process model makes a *check ticket* move, while the event log is not possible to make this move because of the lack of the *check ticket* event. This example denotes a *model move*, indicating a misalignment that an activity should have been executed according to the model, but there is no corresponding event in the log. This alignment indicates that the event log skips the *check ticket* before the *decide* activity happens. Conversely, a *log move* means that an event in the log indicates that an activity has been executed, even though it should not have been executed according to the model.

The goal of alignment conformance checking is to select an optimal alignment. To achieve this, we associate *cost* to each type of move. Therefore, the problem of finding an optimal alignment is transformed into selecting an alignment with the lowest total costs (van der Aalst, 2016; Carmona et al., 2018). Typically, the cost of the synchronous move is 0. The model move and log move usually have a higher cost than the synchronous move, i.e., greater than 0. The standard cost function assigns cost 1 to model moves and log moves (Adriansyah et al., 2011; van der Aalst, 2016). One exception is that there may be hidden activities, labelled as τ, in the process model which represent the activity that can be skipped, the cost of model move on these hidden activities is 0. As an example, Table 2.2 shows an optimal alignment between the given trace σ and the process model shown in Figure 2.1.

This example shows that the alignment algorithm can provide detailed diagnostics per case of the event log. These diagnostics can be aggregated to provide further insights, such as a specific activity is often skipped or occurs at times when it is not supposed to happen.

Conformance checking algorithms provide a methodology for evaluating the quality of discovered process models by computing *fitness* measures, based on the given process model and event logs. The fitness metric offers insights into how well the model can replay traces in an event log, as discussed in Section 2.3.2.

The fitness metric can be calculated based on the defined cost for different types of moves. The fitness value is between 0 and 1, where 0 means poorest fitness and 1 means perfect fitness. The fitness for a given log L is calculated by equation (2.5).

$$f = 1 - \frac{\sum_{\sigma \in L} \text{cost of the optimal alignment for } \sigma}{\sum_{\sigma \in L} \text{cost of worst-case alignment for } \sigma} \tag{2.5}$$

where *worst-case alignment* is that there are no synchronous moves and only model and log moves. The *cost of the worst-case alignment* is defined as the cost of aligning an empty trace to the model plus the cost of treating all events as log moves. Therefore, it will result in an alignment where all events in trace σ are converted to log moves and the shortest path from the start to the end of the model is converted to a sequence of model moves. For example, consider the same trace $\sigma = \langle \textit{register request}, \textit{examine}, \textit{decide}, \textit{pay compensation} \rangle$ and the process model shown in Figure 2.1, the worst-case alignment is shown in Table 2.3.

Event log	register request	examine	decide	pay compensation	≫	≫	≫	≫	≫
Process model	≫	≫	≫	≫	register request	examine	check ticket	decide	pay compensation

Table 2.3: An example of worst-case alignment between the given trace σ and the process model shown in Figure 2.1.

Therefore, according to the standard cost function, the worst-case alignment has a cost of 9, i.e., the cost of 4 moves on the event log at the beginning and the following 5 moves on the process model. The cost of the optimal alignment, shown in Table 2.2, is 1, i.e., only one model move. Therefore, the fitness of this trace against this model is $f = 1 - \frac{1}{9} = 0.89$.

We will explore the application of the fitness metric in conformance checking for workflow recognition in Chapter 3.

2.4 Conclusion

This chapter introduces the background concepts relevant to the core themes of this research. It begins by giving a brief background on pattern recognition and anomaly detection. It then goes on to the detailed background of process mining, which is the main technique we focused on in this thesis. Process mining is able to discover process information from events generated during the execution of processes in the physical world. The major aim of this work is to leverage the process information to solve the problems of pattern recognition and anomaly detection in different domains.

In the next chapter, we investigate the challenge of accurately identifying workflows that correspond to a given sequence of events in structured domains. Moreover, we propose a workflow recognition approach based on conformance checking techniques to tackle this issue.

Chapter 3

Workflow Recognition in Structured Domains

3.1 Introduction

In structured domains, operations are directed by workflows, delineating specific sequences of activities to complete tasks. For example, in a manufacturing factory, workflows might include assembly lines for product manufacturing where products are built step by step, quality control processes to ensure that products meet standards, and packaging workflows that prepare items for shipment. Monitoring the progress of these diverse workflows is crucial for maintaining the efficiency and productivity of the production process. It allows for the timely identification of bottlenecks, inefficiencies, or deviations from expected performance, ensuring that the factory operates smoothly and effectively to meet its production targets. The failure to effectively monitor operational processes can result in delayed response to issues, reduced productivity, and increased costs caused by unnoticed inefficiencies and errors.

With the advancements of the Internet of Things (IoT) technologies, numerous IoT devices can be deployed to monitor the execution of processes (Schönig et al., 2020). These devices can generate a huge amount of event data during the execution of processes. The generated data offers a goldmine for process monitoring and analytics. Moreover, process mining techniques can be incorporated into analysing the generated data to gain insights into the efficiency, discrepancies, and potential bottlenecks of the processes (van der Aalst, 2016).

However, a significant challenge in leveraging process mining for these analyses lies in the fact that the generated event data is often at a granular level that does not directly correlate with the activities modelled in the workflows. For example, looking back at the example of Industry 4.0, such as integrating IoT systems for operational monitoring in factories described in Section 1, WiFi-based localisation is an example of a technique for tracking asset movement and monitoring the production process. In this application, the generated raw data primarily consists of coordinates that pinpoint locations within a factory. This low-level data must be interpreted and aggregated to match high-level activities such as "using a specific machine". Therefore, it is important to bridge the gap between granular event data and the activities modelled in workflows.

Moreover, the generated data is typically noisy, with errors caused by many factors such as inherent limitations in sensor accuracy, or sensor malfunctions. For example, in the context of using Radio Frequency Identification (RFID) tags for tracking component movements within a factory, errors can arise due to interference from nearby electronic devices, obstructed signals, or simply tags not being scanned properly. These inaccuracies can lead to incorrect or missed readings, complicating the task of accurately monitoring the processes.

Another significant challenge arises in scenarios involving complex tasks requiring the coordination of multiple workflows. In these cases, each task completion generates multiple sequences of events. Identifying the specific workflow an event sequence belongs to can become particularly challenging. For instance, consider the same example of using RFID for component movement tracking, if a single component undergoes various machining processes such as drilling, milling, and polishing. Each stage generates its own sequence of RFID tag readings. It becomes challenging for the factory to distinguish whether a sequence of RFID events corresponds to drilling, milling or polishing. Without a clear mapping between the generated sequences of events and the corresponding workflow, we are not able to conduct further analyses, such as those aimed at incorporating process information for identifying discrepancies, which we will discuss in Chapter 4, and enhancing the understanding of patterns, which we will discuss in Chapter 5.

This challenge also exists in other contexts, such as identifying clinical treatment process steps in healthcare. For example, cancer therapy includes several phases, e.g., consultation, diagnosis, follow-up, etc. A patient undergoes various clinical examinations following a specific workflow in each phase. Given a record of examinations that a patient has undertaken, unfortunately, physicians cannot simply and quickly identify which therapy phase the patient is in (Meier et al., 2015). This leads to inefficient processes in daily clinical routine and considerable time spent on retrieving patient clinical data for recognising the current treatment phase.

The current practice to address this problem involves manual labelling. For example, within the manufacturing sector, an individual is tasked with documenting the workflow that a component follows (Papapanagiotou et al., 2021). Similarly, if a patient is undergoing a series of treatments, such as pre-operative procedures, post-operative care, or a diagnostic testing workflow, nurses must manually enter each step of these processes into the hospital's information system (Meier et al., 2015; Munoz-Gama et al., 2022). Although the manual entry serves its purpose, i.e., ensuring that there is a record of which specific workflow or process each patient or asset is following at any given time, it is subject to some limitations. Manual interventions are not only time-intensive but also prone to errors, potentially becoming a bottleneck in terms of improving efficiency. Boosting efficiency and reducing the workload associated with labour-intensive tasks demand the adoption of passive monitoring and the automatic recognition of the specific workflow in operation.

In this chapter, we propose a novel framework for automatically recognising the workflow of the current operation based on generated low-level event sequences during the execution of the process. Our approach leverages complex event processing (CEP) techniques to abstract events from a low level to a high level, aligning them with the activities in the process. We employ Petri nets for modelling the various workflows, offering a structured and visual representation of processes. We utilise conformance checking algorithm to determine which workflow the generated event sequences most closely match, thereby identifying which workflow is being executed. This framework provides us with a way of passive monitoring of the operation and enables us to accurately and automatically recognise the specific workflow in operation, enhancing our understanding and management of the underlying processes. Moreover, this work lays the groundwork for further analysis aimed at leveraging the process information for deviation detection in Chapter 4 and enhancing human activity recognition in Chapter 5.

We further motivate our work with an example in the domain of smart manufacturing in Section 3.2. We present the background of our work in Section 3.3. The proposed method for workflow recognition is presented in Section 3.4. An experimental study based on a real-life manufacturing dataset is presented in Section 3.5. Finally, we discuss and conclude our work in Section 3.6 and Section 3.7.

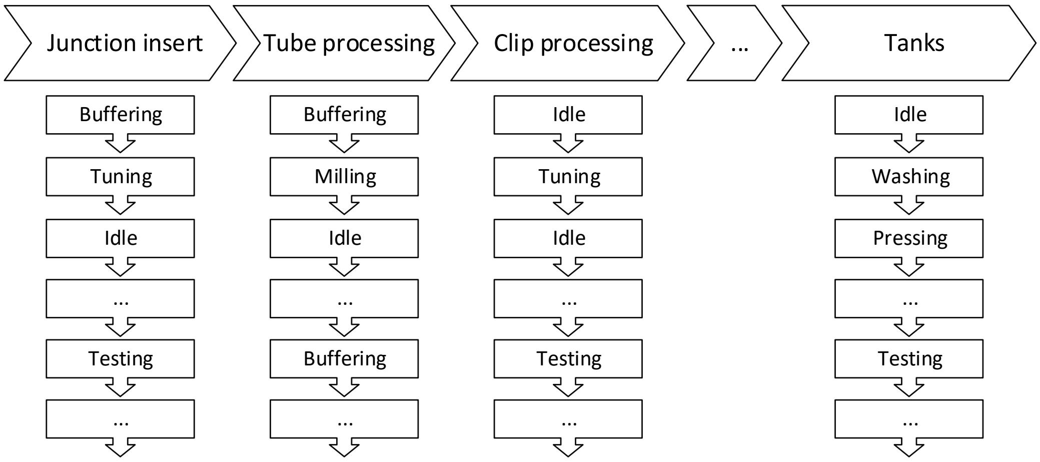


Figure 3.1: An example of a production line for manufacturing a pen.

3.2 Motivating example

In the context of smart manufacturing, IoT techniques, such as indoor localisation, have been widely used to monitor the location of assets and track the progress of production processes (Kirikkayis et al., 2023). By providing real-time insights into the movement and status of materials and equipment, these technologies play a pivotal role in enhancing operational efficiency. For instance, Geofencing and WiFi devices are deployed in different production areas of a factory to track the location of assets. This location information can be further mapped to the activities of production processes to monitor production progress and ensure the smooth operation of manufacturing workflows.

A production line usually consists of multiple small steps, which are organised and managed to achieve the final production targets. For example, a production line for manufacturing a pen includes steps like *junction insert*, *tube processing*, *clip processing*, etc (Papapanagiotou et al., 2021), as shown in Figure 3.1. Each step is typically responsible for completing a specific stage of the production, and has to be performed according to a pre-defined production workflow, which defines a sequence of activities such as *buffering*, *tuning* and *milling*. These activities are conducted in different production areas or by different machines in a factory. Therefore, indoor localisation techniques can be deployed to track the location of assets. For instance, when an asset moves into the *tuning* area, it triggers a raw sensor event signalling it enters into this specific zone. If such locations are associated with particular production activities, this enables the tracking of production progress through the analysis of location data.

In the factory, it is common for multiple assets to be produced simultaneously, leading to the generation of numerous sequences of events, each associated with a specific asset. For example, as assets enter different production zones like the *tuning* area, each movement is captured as a unique sensor event linked to that particular asset. This results in a complex stream of data representing various assets moving through different stages of production simultaneously.

Given the multiple generated event sequences, operators face a significant challenge in pinpointing exactly which workflow each asset is following. Without a clear understanding of which workflow an asset's specific sequence of events undergoes, determining the current step within the production process for any given asset becomes a challenging task. This may cause problems in resource allocations, leading to delays and interruptions that affect production efficiency and overall progress. Currently, the event sequence has to be labelled manually, which introduces delays in process monitoring and further analyses, such as detecting anomalies.

3.3 Background

In this section, we first introduce the concept of complex event processing (CEP), online conformance checking and their related work. Then we introduce the related work in the field of workflow recognition. Finally, we present a summary highlighting how our work differs from existing research.

3.3.1 Complex event processing

CEP is designed to identify meaningful patterns, relationships, and abnormalities within streams of data (Cugola and Margara, 2012). The core of CEP involves the detection of complex events, which are significant insights discerned from the raw stream of event data. These complex events offer actionable insights for organisations into ongoing operations. For example, a single failed login request of a system is not interesting, but several consequent failed login attempts may require attention, as they may indicate malicious activity, such as an attempt to breach the system's security.

These complex events are identified through the application of rules or patterns that can group specific sequences or aggregate some simple events. Filtering, correlation, and aggregation techniques are used to develop complex events. For example, in the context of detecting fraud in the financial domain, a complex event rule could be set to detect when multiple withdrawals are made from a single account at geographically distant ATMs within a short timeframe, such as several withdrawals across different states or countries within a few hours. This pattern of activity, unlikely under normal circumstances, could trigger an alert for potential fraud.

CEP can be adeptly applied to various domains, ranging from finance, and healthcare to manufacturing. In each domain, CEP can be tailored to meet the specific data processing needs of the domain. For instance, in a financial system, CEP can be applied to identify fraudulent transactions by analysing patterns and anomalies in real-time transaction data. Similarly, in the healthcare domains, CEP can be utilised to monitor patient health by interpreting a range of health signals, such as vital signs and laboratory results, to detect early signs of deterioration.

CEP platforms are designed to perform analysis to interpret data carried by events. They are able to group the data streams together and define patterns according to the time sequence relationship and aggregation relationship between events. One of the most commonly used CEP platforms is Siddhi (Suhothayan et al., 2011). Siddhi is open-sourced and provides comprehensive logic for event data processing, such as filtering, aggregation, enriching, and merging of event streams. The event processing logic can be written using SQL-like queries via graphical and source editors.

Many studies have demonstrated the suitability of using CEP techniques for cleaning and pre-processing IoT data (Chen et al., 2014). The complex events can be derived from low-level events generated by IoT sensors through operations like filtering, aggregation and correlation. For instance, Chen et al. present a distributed complex event processing (CEP) engine for event aggregation (Chen et al., 2014). Moreover, Rahmani et al. apply CEP techniques into healthcare domains for analysing patient data collected by IoT sensors (Rahmani et al., 2021). Although these studies support the technical process of deriving complex events from low-level events, they do not map the complex events to activities in a process model for further analysis.

Event abstraction aims to bridge the gap between raw low-level events generated by IoT sensors (e.g., coordinates information) and events at the process level (e.g., using a specific machine). The focus of the event abstraction problem is how to generate abstraction logic.

Several studies leverage machine learning techniques to devise the abstraction logic. For instance, Margara et al. introduce the problem of automated CEP rules generation and present a framework to address the problem (Margara et al., 2014). Tax et al. propose a supervised learning method for automatic event abstraction (Tax et al., 2018).

They extract features from a window of events, which are used to train a model for abstracting events. Works of this type assume that annotations with high-level interpretations of the low-level events are available.

Compared to using machine learning techniques to obtain abstraction logic, some existing approaches leverage domain knowledge from experts to construct event abstraction logic. Baier et al. propose a semi-automatic approach that maps events to activities by transforming the mapping problem into a constraint satisfaction problem (Baier et al., 2014). They define constraints according to the pre-defined process model. Event mapping is achieved by mapping the similar behaviour relationship between the events in the event log and the activities in the process model. They further extend this approach by using a declarative process modelling language, named Declare, to model the constraints (Baier et al., 2015). They derive Declare constraints from the process model and the event log, respectively, which are used to find possible matches between these two sources of constraints. This work is further extended by introducing natural language processing for analysing the labels of activities and events (Baier et al., 2018).

Similarly, Mannhardt et al. propose an alignment method to map the low-level events to activities (Mannhardt et al., 2016). They rely on a behaviour activity pattern that captures domain knowledge on the relation between activities and events. Low-level events are aligned to the behaviour activity pattern to achieve the mapping of events and activities.

3.3.2 Online conformance checking

Conventional conformance checking techniques are designed to function in offline settings, where we have an entire event log describing past process execution behaviour. They do not apply to online scenarios where it is important to raise notifications as soon as the deviations are detected. However, raising early process-oriented deviation detection is critical for many organisations in different domains. For example, in the healthcare domain, deviating process executions often lead to higher costs, and/or delays in examination time in the hospital. If we can detect the deviation just after the execution of an activity, we can take proactive actions to reduce the costs, such as assigning other nurses for assistance. In recent years, some algorithms have been proposed for these online scenarios where the conformance checking is performed upon a new event occurring.

Van Zelst et al. present an alignment-based online conformance checking method, which entails an incremental algorithm allowing the calculation of optimal and approximate prefix-alignment (van Zelst et al., 2019). Upon the occurrence of a new event, this method recycles the prefix-alignment of preceding events, incorporating the new event into the existing alignment framework. This technique enhances the memory efficiency involved in computing alignments by building upon previously established alignments rather than starting from scratch for each new event. However, this approach tends to underestimate the overall cost in comparison to traditional alignment methods. The reason is that it operates based on a locally optimal strategy, focusing on the immediate inclusion of each new event without necessarily considering the global optimum across the entire sequence of events.

Burattin et al. propose an online conformance checking approach using behaviour patterns (Burattin et al., 2018). It abstracts the underlying process into behavioural patterns and detects the conformance of the observed behaviour patterns. This abstraction balances the performance cost and does not need an assumption on the starting point of cases, but it can lose some critical information due to its abstraction.

Lee et al. present a Hidden Markov Model-based approach for online conformance checking that can handle warm start scenarios, where the event occurs in the middle of the process model (Lee et al., 2020). This approach estimates the position of the process model upon a new event occurrence, and computes conformance from such estimated position. They also propose new metrics for evaluating conformance and evaluate the correlation between the new metrics and the *cost* in the standard alignments method. However, they do not propose a comprehensive metric, such as *fitness*, to measure how well the event log aligns with the process model.

3.3.3 Workflow recognition

Workflow recognition is identified as a key technique for monitoring the step being performed in a complex task and detecting the missing step. The existing research on workflow recognition mainly focuses on the recognition of activities within a specific workflow. This focus includes determining the start time and end time of each task to facilitate workflow analysis. Such efforts in workflow recognition are closely linked to the broader fields of activity and action recognition (Gupta et al., 2022). For example, in the surgical context, automatically recognising surgical activities enables a comprehensive understanding of the ongoing situation in the operation room at any time (Czempiel et al., 2023). In contrast, our work diverges by aiming to identify which specific workflow is being executed, rather than focusing solely on the recognition of individual activities or tasks within a workflow. This distinction underscores our contribution to expanding the capabilities and applications of workflow recognition technologies.

The existing workflow recognition work relies on the data collected by wearable sensors, ambient sensors and cameras. Methods based on wearable sensors utilise on-body sensors, e.g., accelerometer, gyroscope, etc., to sense the movements of body parts and infer steps of the workflows. For instance, Scholl et al. present a wearable system which combines a Google Glass and a wrist-worn accelerometer to capture and recognise steps in a wet laboratory environment (Scholl et al., 2015).

Methods based on ambient sensors focus on deploying dedicated sensors to record the environmental variables, such as temperature, movement, etc. and provide context-aware information for activity recognition. Zhang et al. present an unobtrusive workflow recognition system (Zhang et al., 2017). They perceive the use and movement of associated objects in the workflow by extracting information from low-level RFID signals. The usage and movement of these objects are further utilised to infer the most likely activities in the workflow via the Hidden Markov Model (HMM).

Existing methods using video data employ cameras to record video sequences, which are subsequently analysed to recognise workflows through the application of computer vision algorithms (Czempiel et al., 2023; Hu et al., 2020). For example, Hu et al. propose a workflow recognition framework based on video data (Hu et al., 2020). The proposed framework can extract spatial and temporal features from videos to recognise the actions of workers and machines.

Apart from these methods, Meier et al. present an approach for automatically classifying patient specific information, i.e., the record of clinical examinations, into the corresponding workflow step in a clinical treatment process (Meier et al., 2015). They develop a HMM model to represent clinical workflows. Given a set of unknown clinical patient specific information, this model can predict the current treatment phase.

3.3.4 Summary

Complex event processing techniques have been developed for event abstraction, which bridges the gap between granular, low-level events and the activities in process models. Various methods, including machine learning approaches and domain knowledge-based mappings, have been developed to establish the logic for such abstraction, tailored to address specific challenges in each domain. In our work, we utilise the Siddhi complex event processing platform to define event abstraction patterns based on domain knowledge.

Online conformance checking provides a way to conduct conformance checking under a non completed event trace. It can raise an alert as soon as a deviating behaviour from the desired process is observed. In our work, we also consider workflow recognition in online settings. However, we still use the traditional alignment conformance checking, because it can provide us with a global optimal *fitness* metric to measure how well the process model can replay the event log, rather than a local fitness proposed in (van Zelst et al., 2019).

The related work in the field of workflow recognition focuses on the problem of identifying specific activities within a workflow, linking to the broader field of activity recognition. However, the focus of our research shifts from the conventional goal of activity recognition to the novel objectives of identifying which particular workflow is being executed.

3.4 Method

The proposed workflow recognition method consists of the following stages: Firstly, we incorporate CEP techniques to abstract low-level events to high-level events that align with the activities modelled in workflows. Next, we create process models utilising standard operating procedures documented from each domain. With these two elements, i.e., event sequences and process models, we conduct conformance checking to examine the relationship between the sequences of events and predefined workflows. We determine that the identified workflow for a sequence of events is the one to which the sequence aligns most closely. We describe each stage in more detail next.

3.4.1 Assumptions

Our approach to workflow recognition is based on two key assumptions. First, we posit the existence of multiple distinct workflows, each of which is well-defined and structured. We also assume that the knowledge or logic developed for event abstraction is available, which can effectively translate low-level events into a format that is meaningful at the process activities level.

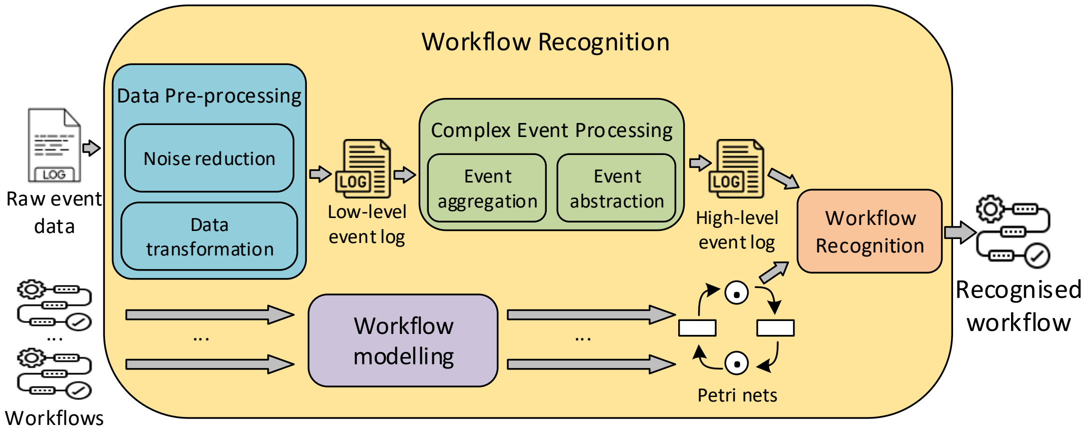


Figure 3.2: The architecture of workflow recognition framework.

These assumptions are reasonable in practice. Firstly, multiple distinct workflows being well defined and structured are realistic reflections of how operations are organised in many structured domains, such as manufacturing, healthcare, and logistics (Papapanagiotou et al., 2021; Meier et al., 2015). Secondly, the knowledge for event abstraction patterns can be obtained through leveraging machine learning techniques and domain-specific knowledge from experts, which are discussed in Section 3.3.1. These two elements, i.e., workflows and event abstraction patterns, which match the low-level events to the activities modelled in workflows, are essential to identify which workflow the generated events belong.

3.4.2 Overview

Figure 3.2 shows the framework of workflow recognition. The workflow recognition approach consists of four main modules: *event data pre-processing*, *complex event processing*, *workflow modelling* and *workflow recognition*. The complex event processing module is responsible for transforming low-level event data into complex events, which can map to activities in the process model. The aim of process modelling is to construct process models leveraging domain knowledge, such as documented workflows for assembling a product. Finally, in the workflow recognition module, we conduct alignment-based conformance checking between the abstracted events and the process models to identify and match similar behaviours between them.

3.4.3 Event data pre-processing

The module introduced in this section aims to pre-process raw event data into an event log. This phase includes two units: noise reduction and data transformation. IoT devices can sometimes produce erroneous readings due to sensor malfunction, environmental factors, or communication errors. The noise reduction module aims to identify such noises in the data and apply existing techniques to eliminate or reduce such noise. For example, the noise reduction module involves removing records having missing values or records with unknown correlation with process executions.

Raw IoT data often comes in various formats and granularities, depending on the sensor type and its purpose. Data transformation aims to transform the IoT sensor data into a standardised event data format. As presented in Section 2.3 regarding the requirements of process mining, for effective application of process mining techniques, the data at least contains the *case id*, the name of the performed activity, and the timestamp of the event that occurred. Typically, a record from a sensor is characterised by the sensor ID, the timestamp, and the value returned by the sensor. To identify which workflow the given execution is following, we identify the sensor id of the event as the *case id* to classify different executions of workflows.

3.4.4 Complex event processing

This module aims to abstract the low-level sensor events into high-level events, which can correlate to the activities in the process model. For some straightforward scenarios, an IoT event might directly correspond to a specific activity in the workflow. However, the events generated by IoT sensors are usually at a lower level. A sequence of IoT events might correspond to a single high-level event, which can be associated with the activity in the workflow. In order to establish this correlation between IoT events and activity in the provided workflows, complex event processing is employed to achieve this.

Complex event processing raises the abstraction of IoT events to the process level through various techniques, such as event querying, aggregation and abstraction. For example, events can be aggregated based on time windows, shared attributes, or defined patterns. Then, the aggregated events are transformed into standardised formats for further downstream analysis. The output of this phase is multiple sequences of events. Each event within the sequence corresponds to a specific activity in the workflow.

In our framework, we integrate the Siddhi CEP platform (see Section 3.3.1) as the engine of the complex event processing module. We assume that the domain knowledge about the correlation between generated events by IoT devices and activities in the process exists. The domain knowledge allows us to develop the CEP patterns that link the IoT events to the activities modelled in the process, raising the abstraction of IoT events to the process level. The abstraction logic can also be partially identified by using machine learning, as introduced in Section 3.3.1.

3.4.5 Workflow modelling

The aim of this module is to transform the workflows into process models. Typically, these workflows are encapsulated within the standard operating procedures, enterprise resource planning systems or workflow management systems of each domain, serving as step-by-step instructions compiled by organisations to help people carry out routine operations consistently.

We utilise Petri nets (presented in Section 2.3.3) as our process modelling tool because Petri nets are particularly well-suited for process mining techniques. Petri nets serve both as a graphical and mathematical tool for representing process models (Aalst, 1998). They enable state space analysis, which encompasses exploring all possible paths within the process model and assessing their alignment with the sequences of events observed in reality. Moreover, Petri nets allow for quantitative analysis in conformance checking techniques, as we introduced in Section 2.3.4. Conformance checking can provide quantitative measurements regarding the fitness of the process model, which measures how well the model can reproduce the observed events, indicating the extent to which all recorded events are represented in the model.

Each of the workflows is modelled as a distinct Petri net. In the Petri net model, *places* are used to represent the various stages of workflow execution. These places are interconnected by *transitions*, which represent the activities within the workflow. The firing of transitions signifies the occurrence or completion of an activity. The routing of activities can be represented by routing constructs of Petri net, such as sequential, parallel, conditional and iterative routing (Aalst, 1998).

3.4.6 Workflow recognition

Once we have the abstracted event traces and Petri nets modelling distinct workflows, in this step, we use conformance checking to identify which workflow the event trace follows. As illustrated in section 2.3.4, conformance checking is able to quantitatively measure how well an event trace conforms to a Petri net. Specifically, a key metric in conformance checking is *fitness* (see Section 2.3.2), which evaluates the extent to which the process model can replay the event log.

We conduct alignment-based conformance checking (see Section 2.3.4) between the event trace and each of the Petri net models. This involves evaluating the fit of the event trace against each Petri net to ascertain which model most closely represents the observed behaviour. The Petri net that yields the highest fitness score is identified as the workflow corresponding to the event trace. This ensures that we can accurately match event traces to the most appropriate workflow, based on the evidence of conformance. The detailed steps are shown by the pseudocode in Algorithm 1.

Algorithm 1 Workflow recognition

Require: $workflows$, an event sequence σ

1: $AlignmentResult \leftarrow \{\}$ ▷ Define an empty dictionary to store results
2: **for** $net \in workflows$ **do** ▷ Iterate every Petri net of production workflows
3: $Result \leftarrow$ **Alignment**(net, σ) ▷ Conduct alignment conformance checking
4: $AlignmentResult[net.ID] \leftarrow Result.fitness$
▷ Assign fitness score (value) to corresponding workflow (key)
5: **end for**
6: $Rec\_workflow \leftarrow$ **Max_Key**$(AlignmentResult)$
▷ Get the key with maximum value
7: **return** $Rec\_workflow$

3.4.7 Online workflow recognition

Moreover, we also consider the workflow recognition problem in online settings, as described in Section 3.3.2. Compared to having a complete sequence of events earlier, we conduct alignment-based conformance checking in an iterated manner to measure fitness scores as new events are produced. The detailed steps of recognising workflow in online settings are shown by the pseudocode in Algorithm 2.

This method is designed to mimic real-time, online scenarios where workflow recognition is conducted immediately upon the occurrence of a new event. The aim is to continuously update our understanding of the workflow being executed as events unfold, providing a more dynamic and responsive system for workflow recognition.

Algorithm 2 Online workflow recognition

Require: $workflows$, an event sequence σ

$Rec\_workflows \leftarrow []$ ▷ Define an empty list for storing results
$\sigma^* \leftarrow []$ ▷ Initialise an empty list σ^* to store previously observed event
for e **in** σ **do** ▷ Iterate each event in σ to simulate newly observed event
    $Append(\sigma^*, e)$ ▷ Append newly observed event into σ^*
    $workflow \leftarrow$ ALGORITHM 1(σ^*)
        ▷ Call the Algorithm 1 to get recognised $workflow$
    $Append(Rec\_workflows, workflow)$
        ▷ Append the recognised $workflow$ of each iteration in the final result
end for
return $Rec\_workflows$

3.5 Experimental study

In this section, we conduct an experimental study on a manufacturing dataset to evaluate our proposed method.

3.5.1 Dataset description

We have a dataset collected from a previous project, Digiflow, which involved a collaboration with a pen manufacturer (Papapanagiotou et al., 2021). Digiflow focuses on digitising, monitoring, and planning industrial workflows using IoT sensors. Bluetooth geofencing sensors were deployed around groups of machines of the same type in the pen factory. When assets enter or exit a particular production area, Bluetooth geofencing generates corresponding events showing the actions of the assets entering and leaving the respective zones. Therefore, it is possible to track the production process based on these events. The floor plan of geofencing areas in the pen factory is shown in Figure 3.3.

The production processes in this factory include 11 workflows, each representing a series of production activities that transition between various machines located in different areas. These activities include washing, polishing, and milling. In Digiflow project, these workflows are systematically documented as part of the standard operating procedures of the factory. An example of the workflow is shown in Table 3.1. The workflow encompasses the activities along with their associated operators, duration, and the specific locations within the factory where they are conducted.

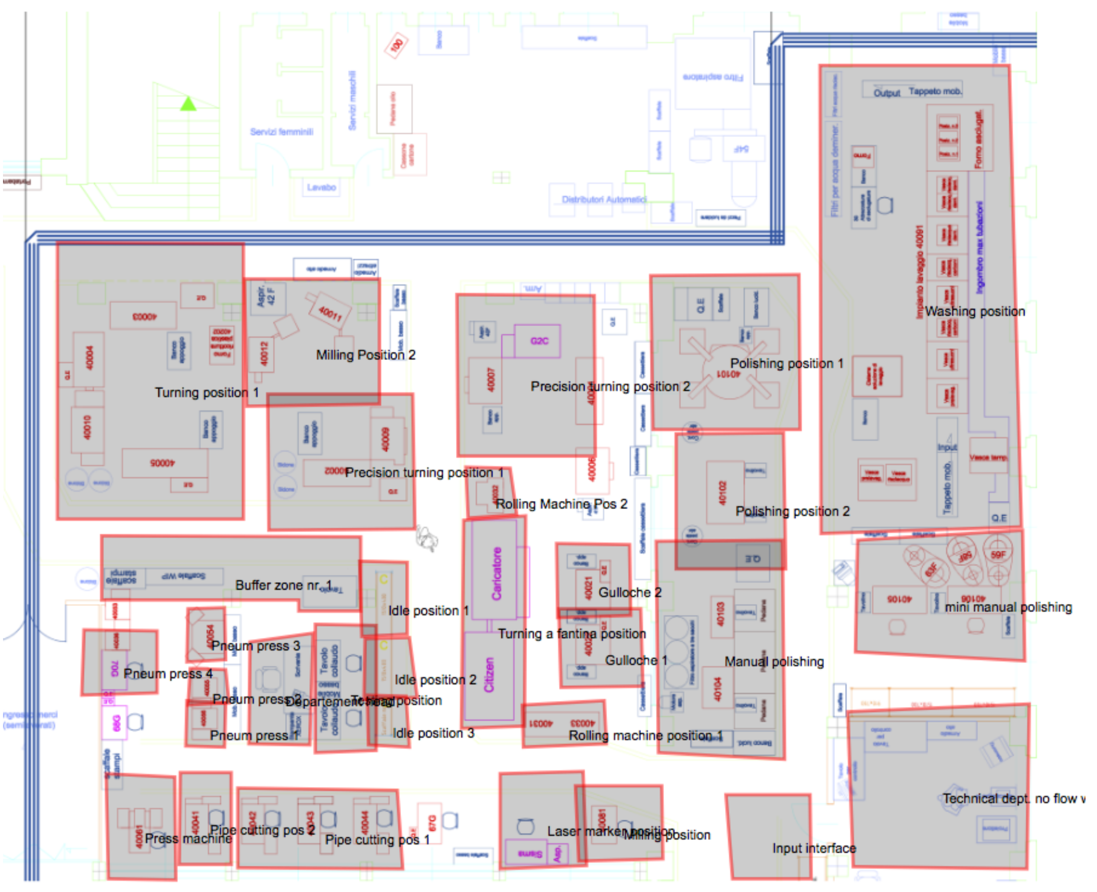


Figure 3.3: The geofencing area in the pen factory.

Steps	Position Name	Operator Name	Activity Name	Activity Duration
1	Raw-Material	Department Head	Input Interface	5 minutes
2	Buffering before entering in process	Department Head	Buffering	N/A
3	Turning - first working location	Turner	Turning 1	1 minute turning
4	Idle station	Employee	Idle station	N/A
5	Polishing	Polishing worker	Polish	2 hours (for production lot)
6	Washing	Washing worker	Washing	1 hour (for production lot)
7	Idle station	employee	Idle Station	N/A
8	Turning - second working location	Turner	Turning 2	1 minute turning
9	Turning - third working location	Turner	Turning 3	1 minute turning
10	Idle station	employee	Idle Station	N/A
11	pieces' testing	Tester	Testing	1 hour per lot
12	Output interface	Employee	Output interface	N/A

Table 3.1: An example of a workflow in the pen factory. This documents the position, operator, and duration of each activity in the workflow.

	timestamp	id_zone	id_settings	entrance_timestamp	metadata/customKey	id_asset	action_type
1	1562140945154	941	15398183895bc7c395d4c294.07482959	1562140945154	turning_position_1	553	accessing
2	1562141246086	941	15398183895bc7c395d4c294.07482959	1562140945154	turning_position_1	553	abandoning
3	1562141269533	941	15398183895bc7c395d4c294.07482959	1562141269533	turning_position_1	553	accessing
4	1562141393759	941	15398183895bc7c395d4c294.07482959	1562141269533	turning_position_1	553	abandoning
5	1562141498852	941	15398183895bc7c395d4c294.07482959	1562141498852	turning_position_1	553	accessing
6	1562142200466	941	15398183895bc7c395d4c294.07482959	1562141498852	turning_position_1	553	abandoning
7	1562142213202	955	15398183895bc7c395d4c294.07482959	1562142213202	idle_position_1	553	accessing
8	1562142399496	955	15398183895bc7c395d4c294.07482959	1562142213202	idle_position_1	553	abandoning
9	1562142421022	944	15398183895bc7c395d4c294.07482959	1562142421022	polishing_1	553	accessing
10	1562155850393	974	15398183895bc7c395d4c294.07482959	1562155850393	washing_position	553	accessing
11	1562158676452	944	15398183895bc7c395d4c294.07482959	1562142421022	polishing_1	553	abandoning
12	1562159424673	974	15398183895bc7c395d4c294.07482959	1562155850393	washing_position	553	abandoning
13	1562159424673	955	15398183895bc7c395d4c294.07482959	1562159424673	idle_position_1	553	accessing
14	1562159449784	956	15398183895bc7c395d4c294.07482959	1562159449784	idle_position_2	553	accessing
15	1562159792039	955	15398183895bc7c395d4c294.07482959	1562159424673	idle_position_1	553	abandoning
16	1562142200466	975	15398183895bc7c395d4c294.07482959	1562142200466	buffer_1	553	accessing
17	1562159806475	956	15398183895bc7c395d4c294.07482959	1562159449784	idle_position_2	553	abandoning
18	1562159806475	942	15398183895bc7c395d4c294.07482959	1562159806475	precision_turning_position_1	553	accessing
19	1562159817169	975	15398183895bc7c395d4c294.07482959	1562142200466	buffer_1	553	abandoning

Table 3.2: Example of raw data in the manufacturing dataset.

As part of the testing and validation of Digiflow, labelled data was collected by having an individual imitate each workflow by navigating the production area in accordance with the specified workflows. This ensures that the collected data accurately represents the unique sequences inherent to each specific workflow, offering a reliable ground truth for our workflow recognition analysis. The dataset is organised into two distinct groups, each comprising a collection of sequences that correspond to the 11 workflows. Therefore, we treat one group as the development set for refining our approach, and the other as the testing set to evaluate the accuracy of our approach.

The example of the raw data set is shown in Table 3.2, where the *timestamp* field represents the time when the event occurs, *id_zone* represents a corresponding production area in the manufacturing. There is no substantive meaning of *id_settings*, representing an identifier of the specific context. The *entrance_timestamp* represents the time of the asset entering one area. The *customKey* means the corresponding operation related to this event, while many events are missing this value. Moreover, the *id_asset* means the specific asset being processed. The *action_type* indicates if an asset enters or exits an area. Specifically, *accessing* represents the asset entering the area, whereas *abandoning* means the asset exits the area.

There are also several factors that contribute to the overlap of geofencing areas, resulting in some noise in the collected data. In the pen factory, the production areas occupy limited physical spaces and are situated near each other. The limited separation between zones naturally leads to overlapping geofencing areas. Moreover, signal inference and device sensitivity issues also exacerbate this overlap. For example, as illustrated in Figure 3.3, the areas designated as *idle position 2* and *idle position 3* overlap with the *testing position*. This overlap leads to the generation of noisy data, such as the occurrence of consecutive entry and exit events for a single area, which can be observed in rows 1-6 of Table 3.2. We present more detailed noisy data analysis in Section 3.5.2.1.

3.5.2 Implementation and evaluation

In this section, we evaluate our proposed workflow recognition approach based on the datasets described above. Firstly, we conduct data pre-processing to identify noises and errors in data and employ complex event processing to clean and abstract events. Then we model the workflows as Petri nets and conduct alignment conformance checking for workflow recognition. More detail of each step is described below.

3.5.2.1 Complex event processing

Due to the overlap of geofencing areas, the data we gather tends to be riddled with noise. In such situations, for example, the sensor may erroneously produce events that suggest an asset is simultaneously present in two distinct locations. Through data pre-processing, we identified some issues of the data as below:

- **Delayed exit events:** One of the assets enters a particular area and then enters the next area without exiting from the first one. It was after several days that we received the exit event. The example is shown in rows 9-11 in Table 3.2, where the asset enters the *polishing* zone in row 9, while it then enters the *washing position* without exiting from the polishing zone. This results in assets sometimes being in two places at once. In this example, the data indicates that the asset is in *polishing* and *washing position* at the same time.

- **Consecutive entry and exit events for one area:** Assets can produce consecutive entry and exit events for one area in a short time, as shown in rows 1-6 in Table 3.2. These consecutive events regarding one area do not correspond to any actual movement in reality.

- **Missed exit events:** This means that the assets remain in one area when the workflow ends, without exiting the production area.

- **Events are not chronologically ordered:** The sequence of events we receive is not chronologically ordered according to their timestamps. This means that certain events that occurred earlier are recorded with timestamps suggesting they happened later. We can see rows 15-17 in Table 3.2, where the data shows that the asset should enter *buffer_1* position earlier than the point in the table since the timestamp of row 16 is less than row 15.

We adapt complex event processing techniques to clean these noisy and transform these events into high-level events that are mapped to the process activities.

As the activities within the workflows are correlated to specific production areas, we link the generated raw data to activities using the field of *id_zone*, which corresponds to process areas in the factory. For example, an asset entering the *id_zone* of *941* indicates that the asset is engaged in the *Turning 1* activity at the *turning_position_1* area.

Within the generated data, a pair of an accessing event followed by an abandoning event, both linked to the same area, indicates the asset has completed a specific activity. For example, the rows 7-8 in Table 3.2 indicates a completion of the *Idle* activity. Therefore, we define a pair consisting of an accessing event and an abandoning event for the same zone as a single activity. The timestamp of the assessing event is set as the starting time of this activity, and the timestamp of the abandoning event is set as the ending time of the corresponding activity.

The issues of missing and delayed exit events indicate that the asset sometime is in two positions at the same time, which is impossible in practice because the asset should exit one area to be able to enter another area. To address this, we define an event abstraction pattern in Siddhi to denote that if the event enters a different area, it means that the asset exits the current area. This pattern defines that the timestamp of entering the next zone is also treated as the exiting timestamp of the current zone. Consequently, this pattern allows us to group an accessing event with an abandoning event in the same zone as one complex event, indicating a completed production activity. The principles of this pattern are summarised by the pseudocode in Algorithm 3.

Next, we introduce a sliding time window mechanism to address the issue of consecutive entry and exit events for one area. This mechanism is detailed in line 23 of Algorithm 3, which employs a sliding window time interval to filter out redundant events occurring in quick succession. The duration of this time interval is tailored to the specific conditions of the scenario being modelled. Given that our dataset was generated through manual simulation of the production process by an individual moving

	Accessing time	id_zone	Identifier	id_asset	Leaving time
0	21/11/2018 09:49:09	975	buffer_1	526	21/11/2018 09:49:50
1	21/11/2018 09:49:43	941	turning_position_1	526	21/11/2018 09:50:24
2	21/11/2018 09:50:24	955	idle_position_1	526	21/11/2018 09:51:06
3	20/11/2018 10:08:50	969	laser_marker_position	526	21/11/2018 09:51:20
4	20/11/2018 10:08:37	946	polishing_manual	526	21/11/2018 09:51:22
5	21/11/2018 09:51:20	944	polishing_1	526	21/11/2018 09:51:59
6	21/11/2018 09:51:59	974	washing_position	526	21/11/2018 09:52:43
7	21/11/2018 09:52:53	955	idle_position_1	526	21/11/2018 09:53:23
8	21/11/2018 09:53:20	942	precision_turning_position_1	526	21/11/2018 09:54:26
9	21/11/2018 09:54:15	943	precision_turning_position_2	526	21/11/2018 09:55:19
10	21/11/2018 09:55:25	957	idle_position_3	526	21/11/2018 09:56:21
11	21/11/2018 09:56:21	958	department_head	526	21/11/2018 11:00:34

Table 3.3: Example data after complex event processing.

through production areas, rather than from actual production activities, we set the time interval at 10 seconds for this context.

Regarding the observation that raw events are not sequentially organised according to their timestamps, one plausible explanation could be synchronisation delays among different Bluetooth gateways. These discrepancies might stem from slight differences in internal clocks or network latency. Hence, we assume the correct order of events to be the order in which they are arranged in the log, even if it is not ordered by timestamp.

After event processing, we get a sequence of events with *id_zone* and *identifiers* which are associated with production areas. Each event sequence also includes the details on the times when assets access or leave these zones, labelled as *accessing time* and *abandoning time*. An example of final data related to asset 526 is shown in Table 3.3.

3.5.2.2 Workflow modelling

In this dataset, there are 11 distinct workflows, each encapsulating the production process between various machines, as shown in Table 3.1. Each activity within the workflow is associated with a specific production area in the factory, as illustrated in the field of *Position name* of Table 3.1. Leveraging Bluetooth geofencing technology enables the tracking of asset locations across different production areas as they proceed following the workflows. This correlation between locations and activities facilitates the modelling of workflows based on the production areas tied to different activities.

Algorithm 3 Complext event processing

```
Require: RawData, TimeInterval                         ▷ Input: senor data
                                   ▷ Input: the interval of the time window
AbstractData ← []                  ▷ Define an empty list for abstracted events
for e_1 in RawData do              ▷ Iterate every event in raw sensor data
    if e_1.action_type = "accessing" then
        e_2 ← subsequent event
                  ▷ Get subsequent event, not necessarily immediate next
        e ← empty event                           ▷ Define an empty event
        if e_2.id_zone = e_1.id_zone and e_2.action_type = "abandoning" then
            e.id_zone ← e1.id_zone
            e.entry_time ← e1.timestamp
            e.exit_time ← e2.timestamp
              ▷ Assign the timestamp of abandoning event as the exiting time
            e.id_asset ← e1.id_asset
            e.entry_time ← e1.timestamp
            Append(AbstractData, e)   ▷ Insert the abstracted event into the list
        else if e_2.id_zone ≠ e_1.id_zone and e_2.action_type = "accessing" then
            e.id_zone ← e1.id_zone
            e.entry_time ← e1.timestamp
            e.exit_time ← e2.timestamp
              ▷ Assign the timestamp of accessing different zone as the exiting time
            e.id_asset ← e1.id_asset
            e.entry_time ← e1.timestamp
            Append(AbstractData, e)   ▷ Insert the abstracted event into the list
        end if
    end if
end for
DeduplicateData ← deduplicate(AbstractData, TimeInterval)
                          ▷ Remove the consecutive events by time window
return DeduplicateData
```

Following this modelling, we then convert the structured workflows into Petri nets. An example Petri net of the modelled workflow is shown in Figure 3.4.

It is important to note that the workflows in this dataset are linear, characterised by a straightforward sequence without loops or branching paths. However, it is crucial to emphasise that our approach for workflow recognition is not limited to linear models alone. It is designed to be adaptable, and capable of handling a wide range of workflow structures, including those with complex loops and branching paths.

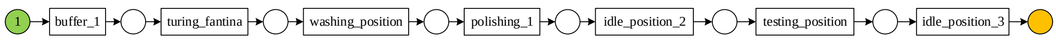


Figure 3.4: Example Petri net of a manufacturing workflow.

3.5.2.3 Workflow recognition

With the processed event sequences and modelled workflows, alignment-based conformance checking is conducted to recognise the workflow. We fit the processed event sequences into the Petri nets to perform alignment.

The initial phase of our analysis focuses on the development set (see Section 3.5.1). We calculate the fitness scores for each event sequence across the different modelled workflows. The results of fitness scores corresponding to different workflows for each sequence are shown in Table 3.4.

	Flow1	Flow2	Flow3	Flow4	Flow5	Flow6	Flow7	Flow8	Flow9	Flow10	Flow11
event_log_1	**0.909**	0.364	0.455	0.333	0.353	0.400	0.444	0.316	0.444	0.333	0.240
event_log_3	0.385	0.385	**0.692**	0.364	0.286	0.316	0.364	0.435	0.364	0.357	0.276
event_log_4	0.500	0.300	0.500	**0.625**	0.400	0.308	0.500	0.353	0.500	0.364	0.261
event_log_5	0.471	0.353	0.471	0.462	**0.833**	0.200	0.462	0.286	0.462	0.316	0.300
event_log_6	0.400	0.133	0.400	0.182	0.200	**0.750**	0.182	0.167	0.182	0.118	0.111
event_log_7	0.316	0.316	0.316	0.533	0.429	0.333	**0.800**	0.375	0.667	0.381	0.364
event_log_8	0.333	0.444	0.444	0.429	0.308	0.364	0.429	**0.667**	0.429	0.300	0.286
event_log_9	0.556	0.444	0.444	0.571	0.615	0.182	0.571	0.400	**0.714**	0.400	0.476
event_log_10	0.370	0.296	0.444	0.261	0.273	0.200	0.261	0.333	0.261	**0.690**	0.200
event_log_11	0.333	0.333	0.400	0.231	0.240	0.174	0.385	0.296	0.385	0.313	**0.788**

Table 3.4: The result of fitness scores between each sequence with specific workflows in the development dataset. The maximum fitness score of each sequence is highlighted in bold.

We determine the workflow that the event sequence corresponds to by selecting the one with the highest fitness score, which is highlighted in bold within Table 3.4.

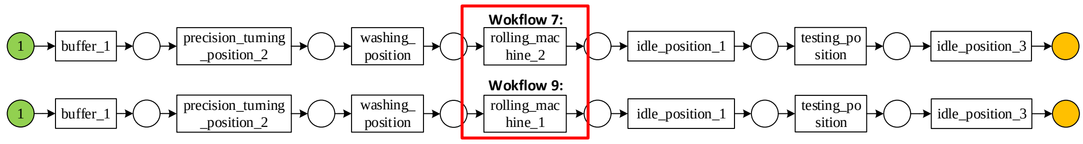


Figure 3.5: The difference between workflow 7 and workflow 9.

The result shows the accuracy of workflow recognition is 100%. This means that each event sequence was accurately matched to its corresponding workflow, indicating that our algorithm effectively identifies workflows within the development dataset.

Following the development set evaluation, we proceed to assess our proposed algorithm using the testing set, which comprises 14 event sequences each associated with different workflows. The workflow recognition results indicate that all the event sequences were accurately matched to their corresponding workflows, except for a single case, i.e., *event_log*_583. In this specific case, the event log that was intended to correspond to workflow 7 was inaccurately identified as workflow 9.

Following this, we investigate the reason for our algorithm's misrecognition of the event sequence *event_log*_583 as workflow 9 instead of workflow 7. We identified that the distinguishing factor between workflows 7 and 9 lies in the specific rolling machine utilised: workflow 9 involves *rolling machine 1*, while workflow 7 employs *rolling machine 2*, as depicted in Figure 3.5. It is important to note that *rolling machine 1* and *rolling machine 2* represent distinct activities and are located in different locations. This small variation in equipment choice signifies different processing paths within the production environment, underscoring the importance of precise event sequence mapping to the correct workflow.

However, the event sequence *event_log*_583 associated with workflow 7 mistakenly includes data for rolling machine 1 instead of rolling machine 2. This discrepancy led to a higher fitness score for workflow 9 in comparison to workflow 7, explaining the misrecognition.

This misrecognition prompts us to consider the accuracy of the workflow simulation conducted by the individual responsible for generating the event data. It appears that there was a deviation from the intended path, resulting in the pen visiting the incorrect rolling machine.

3.5.2.4 Online workflow recognition

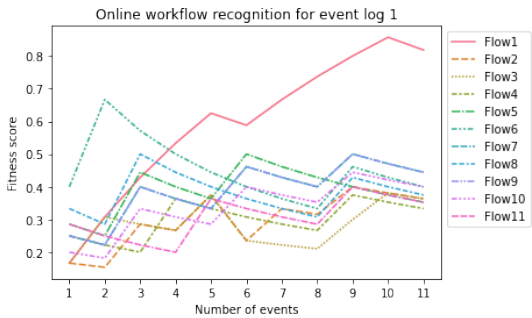
We present the illustrative results of online workflow recognition in Figure 3.6, which demonstrates how fitness scores evolve with the occurrence of subsequent events. As each new event is introduced and aligned with the possible workflows, the corresponding fitness scores adjust, providing insights into how closely each sequence of events continues to align with the modelled workflows over time.

Taking Figure 3.6a as an example, we can see it represents *event log 1*, which is associated with workflow 1. The analysis indicates that precise workflow identification becomes possible once more than 4 events are observed. In the case of *event log 3*, illustrated in Figure 3.6b, correctly identifying the associated workflow necessitates observing more than 8 events. Conversely, for *event log 4*, accurate association with workflow 4 is achievable after merely observing 2 events, as shown in Figure 3.6c. Interestingly, Figure 3.6d shows that the results fluctuate between workflow 7 and 9 for the initial three events. It is only with the introduction of the fourth event that a clear identification of workflow 9 emerges, underscoring the significance of this particular event in distinguishing between the two similar workflows. This pattern highlights the variable nature of event significance across different workflows, where certain events play pivotal roles in clarifying the workflow identification process.

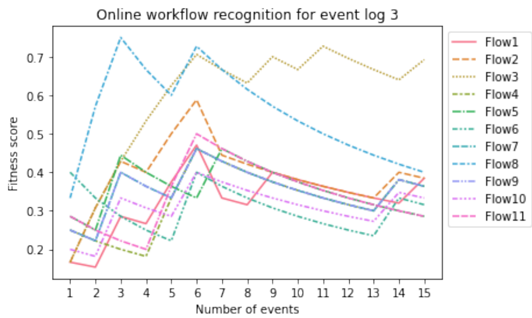
3.6 Discussion

This chapter introduces a novel framework that integrates an alignment-based conformance checking algorithm for workflow recognition. The experimental study on a real-world dataset demonstrates the complex interplay between generated low-level events by IoT sensors and workflow models. This initiative in developing a workflow recognition framework stands out as a substantial contribution, tackling the previously overlooked challenge of determining the specific workflow to which an event sequence belongs. Additionally, this approach lays the groundwork for deviation detection in Chapter 4 and process-driven activity recognition discussed in Chapter 5. This is achieved by establishing the mapping of the generated event data with corresponding process models.

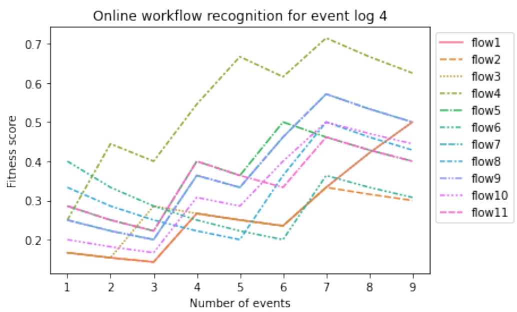
The experimental study presented, from the straightforward identification for most of the workflows to the nuanced differentiation between workflow 9 and workflow 11, illustrates the practical challenges and considerations inherent in applying these tech-



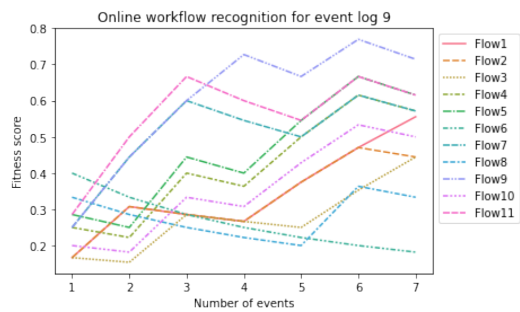
(a) The result of event log 1 corresponding to workflow 1.



(b) The result of event log 3 corresponding to workflow 3.



(c) The result of event log 4 corresponding to workflow 4.



(d) The result of event log 9 corresponding to workflow 9.

Figure 3.6: Variation in fitness score with each successive event for different event sequences.

niques to real-world scenarios. The misidentification of a workflow due to a single deviant event in the data collection emphasises the importance of precision in data collection and the potential for human error to impact the workflow recognition process.

Furthermore, the examination of workflow recognition in online settings demonstrates how fitness scores change over the subsequent events. The demonstrated oscillation between workflows in the early events of a sequence before settling on a definitive identification underscores a critical aspect of online workflow recognition, i.e., the evolution of certainty over time. This can provide early insights into which workflow a sequence of events is likely to correspond to in real-time scenarios. This exploration presents a promising avenue for further explorations of recognising workflows in online settings in terms of computational efficiency.

While the proposed workflow recognition framework offers significant advancements in addressing the gap of correlating event sequences with specific workflows, several challenges and constraints remain.

- **Dependence on event quality**: The accuracy of workflow recognition heavily relies on the quality and granularity of the event data. Misrecorded, missed, or not ordered events can lead to misidentification of the workflow. As evidenced by the confusion between workflow 9 and 11, a misrecorded step in data collection leads to the misidentification of an entirely different workflow. This underscores the significance of event data quality, especially in the context where there are only subtle differences between workflows.

- **Generalisation across different domains**: Due to the constraints on data availability, the approach has been tested and validated within a specific dataset and set of workflows. Although our approach is designed to be flexible, capable of handling various structures of workflows, its applicability and effectiveness across different domains with varying workflow complexities and event logging practices are not evaluated. This limitation underscores the necessity for further exploration and validation across a broader spectrum of operational contexts.

- **Requirement for domain expertise**: The effectiveness of workflow recognition hinges on accurately modelling workflows and the ability to abstract low-level events, generated by IoT devices, to a level that matches the activities within these workflows. Both workflow modelling and event abstraction require domain knowledge tailored to specific context of its application. This requirement for specialised knowledge presents a challenge to swift development.

3.7 Conclusion

In structured domains, specific workflows govern each step of processes. To effectively monitor the execution of processes, especially in scenarios involving complex tasks that integrate multiple workflows, it is crucial to identify the specific workflow which the generated events follow.

In this chapter, we introduce an approach for workflow recognition based on generated low-level events during the execution of processes in structured domains. Our proposed approach comprises four modules: event data processing, complex event processing, workflow modelling and workflow recognition through the alignment-based conformance checking algorithm. An experimental study, utilising a manufacturing dataset, has been undertaken to validate our methodology. The results demonstrate that our approach can discern the underlying workflow from a given sequence of events. Besides, we also investigate the setting of online workflow recognition. The result underscores that our approach can accurately identify the correct workflow with the observation of only a few events.

The next chapter considers the uncertain event data in structured domains. We extend the alignment-based conformance checking algorithm for deviation detection, which can function under probabilistic event data.

Chapter 4

Deviation Detection over Probabilistic Events in Structured Domains

4.1 Introduction

As discussed in Chapter 1, structured domains are governed by pre-defined constraints that describe what behaviours are considered normal or expected. However, data generated by sensors and IoT technologies, which are used for monitoring the execution of processes in these domains, inherently possess a degree of uncertainty. Detecting deviations without considering such uncertainties can result in erroneous identification of normal activities as anomalies or, conversely, the failure to detect genuine anomalies.

Conformance checking (see Section 2.3.4) is the task of comparing the behaviour captured by event logs recorded during the execution of a process to the intended behaviour of a corresponding process model (Carmona et al., 2018). It evaluates how well the execution matches the modelled process and allows us to detect when things are diverging from a desired or expected workflow. A widely used conformance checking approach is based on the *alignment* of the events in the log with the activities in the process model, where any misalignments are potential deviations (Adriansyah et al., 2011).

The standard *alignment* conformance checking approach is only able to fit deterministic events that are assumed to reflect reality with certainty (Cohen and Gal, 2021). This means that probabilistic event data has to be reduced to a deterministic event log in order to be used. This removes information from the event log, thus lowering the confidence on any detected deviations.

In this chapter, we present an extension of alignment-based conformance checking under the assumption of a probabilistic event log with a categorical distribution over a set of activities. We introduce a cost function that takes activity probabilities into consideration, and a custom parameter that allows the algorithm to align activities of lower but sufficiently high probability that better agree with the process model, as opposed to always assuming the most probable activities occurred. In effect, this leverages the knowledge captured in the process model to address levels of uncertainty in the event data, with the aim of reducing the number of false positives and false negatives in deviation detection.

We further motivate our work with the same example from Section 1, discussing the use WiFi localisation for asset movement tracking, in Section 4.2. We present the related work and preliminaries in Section 4.3 and Section 4.4, respectively. We give a formal presentation of our algorithm in Section 4.5, explain the operation and intuition behind its parameter in Section 4.6. Comparative experiments with other approaches are conducted on two real-life datasets in Section 4.7, which demonstrates that our approach can better tolerate the uncertainty of data and effectively decrease both false positive and false negative rates in deviation detection. We implement our algorithm as a Python package based on the PM4Py framework (Berti et al., 2019) and open-sourced on GitHub[1].

4.2 Motivating example

As we discussed in the example of WiFi localisation in Section 1, WiFi-based indoor location tracking systems have been widely used in the context of manufacturing to monitor production progress (Hayward et al., 2022). For instance, tracking the indoor location of specific assets can provide real-time information about current operations and also allow for the assessment of conformance to predefined production workflows.

With different access points installed at various locations in a factory, an indoor location tracking system can identify locations according to the signal strength, a method known as Received Signal Strength Indicator (RSSI) based indoor location tracking (Chen et al., 2016). RSSI works by measuring the strength of the WiFi signal from multiple access points and using algorithms to triangulate the position of an asset based on these measurements. For example, an indoor location tracking system can identify an asset's location in various factory zones (see Section 3.2), such as *milling area*, *polishing area*, and *buffering area*. These locations enable subsequent

[1] https://github.com/jia-wei-zheng/ProbCost

analysis, including conformance checking, to determine whether the asset is following pre-defined production workflows.

However, while such systems typically assume that the asset is located in the most probable location identified by RSSI-based tracking, the accuracy of this method can be compromised, especially when access points are located close to each other or when signals are obstructed by walls or other obstacles (Carrasco et al., 2018). This can lead to uncertainties of the asset's location, which can impact the reliability of subsequent analyses like conformance checking for identifying potential deviations. Detecting deviations without accounting for the uncertainties in location tracking could lead to false alerts. For example, if an asset is inaccurately identified as being in an unexpected production zone due to signal interference or other factors affecting the accuracy of the RSSI-based tracking, it may incorrectly trigger an alert for a workflow deviation.

For example, an indoor location tracking system may indicate that an asset is located in *milling area* with 0.33 probability, *polishing area* with 0.33 probability and *buffering area* with 0.34 probability. All three of these areas are closely situated. When conformance checking is conducted to detect deviations, a misidentified location of *milling area* as *buffering area*, which is only 0.01 more likely, may lead to an alert of a deviation. This would be an unnecessary and false positive caused by the fact that the conformance analysis only considers the most probable locations without considering other contextual information. Instead, we want an algorithm that operates under an assumption of uncertainty, such that considers the production workflows as an additional source of information of what may have occurred in reality, beyond the uncertain location tracking. For instance, if the production workflow describes that the asset currently should move to the *milling area* after the polishing process, this information should be factored into the analysis. For example, even if location tracking data suggests a slight probability favouring the asset being in the *buffering area* rather than the *milling area*, understanding the required workflow sequence allows the system to rationalise that this is likely not a deviation but rather a normal progression of tasks.

In contrast, if the probability of a location is too low (e.g. a 3% probability of *milling area* compared to a 64% probability of *buffering area*), we may want to consider it as non-conforming, even if it is expected by the production workflows. Therefore, incorporating the level of *confidence* in the contextual information of processes compared to the uncertainty of the location tracking is a key requirement.

4.3 Related work

We have presented the background of conformance checking Conformance checking in Section 2.3.4. Conformance checking is a technique to compare process execution logs to a corresponding process model and evaluate how well the observed events followed the model. Alignment algorithm is one of the most commonly used conformance checking algorithms (see Section 2.3.4), which transforms the problem of computing alignments between event logs and process models into an optimal search problem using the A* algorithm, i.e., minimising the alignment cost function (Adriansyah et al., 2011; van Zelst et al., 2017).

Existing literature extends traditional conformance checking methods to accommodate the complexities of process executions. For example, Koorneef et al. extend the alignment algorithm to use a probability-based alignment cost function (Koorneef et al., 2018). The aim of this approach is to get the most probable alignment given historical traces. The idea is to calculate frequencies of events using historical event logs and transform them to probabilities of each model, log, and synchronous move.

However, most conformance checking tasks operate under the assumption that the input event data accurately and comprehensively reflects reality. However, in many settings, this assumption proves to be overly optimistic. Events can be missing or recorded incorrectly, either partially or entirely, due to a range of issues such as human errors, malfunctioning logging devices, or inaccuracies in event data acquisition, for instance, through sensor errors. To mitigate these issues, there has also been more recent work on conformance checking under uncertainty in event logs with 3 different focal points: (i) *uncertain mappings between events and activities* (van der Aa et al., 2020), (ii) *uncertain process models* (Bergami et al., 2021), and (iii) *data quality and imperfections* (such as invalid or missing timestamps in the log) (Felli et al., 2022; Pegoraro et al., 2021),

In the line of *uncertain mappings between events and activities*, Van der Aa et al. introduce a probabilistic conformance checking algorithm that can handle uncertain mappings (van der Aa et al., 2020). Here, uncertain mappings mean that multiple events may map to a single activity. In this chapter, we do not consider this kind of mapping uncertainty. We assume the event corresponds to an activity in a one-to-one relationship.

Conversely, Bergami et al. address the uncertainty of conformance checking by considering *uncertain process models* (Bergami et al., 2021). They propose the use of stochastic Petri nets to obtain probabilistic trace alignments. Compared to this approach which considers probabilistic process models and deterministic event logs, we are considering uncertain event logs and deterministic process models.

In the line of *data quality and imperfections*, Pegoraro et al. propose a conformance checking algorithm over uncertain event data (Pegoraro et al., 2021). This study mainly focuses on control-flow and time perspectives of the process (see Section 2.3). By estimating and repairing the missing and incorrect values in the data preprocessing process, they build a dependency graph representing the uncertain trace by enumerating all the possible traces of such uncertainty in event logs. The algorithm can get upper and lower bounds for fitness scores as opposed to getting only one optimal alignment. This approach though takes the uncertainty of events into account in conformance checking, it does not consider how the probability of events influences the fitness score explicitly.

Likewise, Felli et al. introduce a framework for data-aware conformance checking over uncertain logs (Felli et al., 2022). The framework employs Data Petri nets for modelling multi-perspective processes (see Section 2.3). Data Petri nets enhance the traditional Petri net by incorporating data variables, allowing them to model not only the control-flow but also the data perspective of a process. They address the uncertainty of data perspectives in event logs following the approach in (Pegoraro et al., 2021). They achieve optimal alignment by finding an alignment with minimum cost among all possible realisations of the uncertain traces, compared to establishing upper and lower bounds in (Pegoraro et al., 2021).

Cohen et al. presented a review of challenges related to probabilistic event data in conformance checking techniques (Cohen and Gal, 2021). They highlighted the demand for new conformance checking techniques that can adapt multiple types of uncertain event data, which further motivates our work.

Based on the proposed challenges identified by Cohen et al., Bogdanov et al. present a conformance checking algorithm over stochastic logs, which takes occurrence probabilities of events into account explicitly, similarly to our algorithm (Bogdanov et al., 2022). They apply their algorithm on *trace recovery*, i.e., recovering the original trace from a probabilistic one based on a maximal alignment to a reference process model (Bogdanov et al., 2023). However, in their approach, an event is aligned to a corresponding activity in the process model regardless of the probability that the activity actually occurred, as long as it is non-zero. This assumes the occurrence of activities with very low or even close to zero probability, as long as they align with the

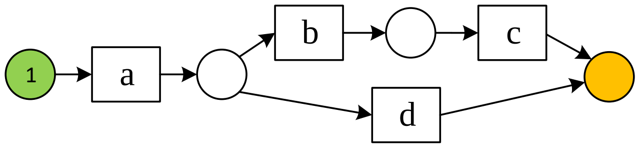


Figure 4.1: Example of a process model represented by Petri net.

process model, therefore discarding potential true positive deviations.

Overall, although the need for conformance checking under uncertainty has been recognised by researchers, the related research is still in its infancy. Our work focuses on a control-flow perspective and takes the probability distribution of event occurrences into account explicitly when achieving the alignment.

4.4 Preliminaries

Our approach is built upon well-established literature on alignment-based conformance checking (see Section 2.3.4). We provide formal definitions of relevant concepts in this section.

Petri nets offer a standard representation for *process models*, such as the example shown in Figure 4.1. The formal definition of Petri nets is introduced in Section 2.3.3 in Definition 1.

As discussed previously in Section 2.3.1, an *event log* captures information about the execution of multiple cases (or instances) of a process model. Each case includes a trace of events $\sigma = \langle e_0, e_1, ..., e_n \rangle$ that correspond to *observations* of the activities that occurred during the process. We assume each event corresponds to a single activity in the process. An event is *deterministic* if it is associated with an activity in the process model in a deterministic way.

When aligning a trace of events to a model, the trace is transformed into a *Petri net* called a *trace model*.

Definition 2 (Trace model) *Given a sequence of events $\sigma = \langle e_0, e_1, ..., e_n \rangle$ over a set of activities A can be converted into a Petri net $N_t = (P^t, T^t, F^t, \alpha^t)$, called a* trace model. *In this, each event e_i is mapped to a transition in T^t. The transitions are then interleaved with places in P^t to form a linear sequence.*

We combine the trace model with a given process model in a single Petri Net called a *synchronous product net*. In this, the set of transitions T^s consists of 3 subsets, namely *synchronous moves* T^{SM}, *model moves* T^{MM}, and *log moves* T^{LM}. Combining a transition for the same activity a from the 2 models results in a *synchronous move* $(a,a) \in T^{SM}$, meaning that an event corresponding to a is *aligned* to activity a in the process model. Transitions in the original process model that do not have a corresponding transition in the trace model are represented as *model moves* $(\gg, a) \in T^{MM}$ or *model moves on* τ $(\gg, \tau)$ and transitions in the original trace model that do not correspond to the process model as *log moves* $(a, \gg) \in T^{LM}$ (Carmona et al., 2018).

An alignment is a *sequence of transition firings* in the synchronous product net. There exists at least one alignment that contains only model moves, followed by log moves. However, the goal is to find the *optimal* alignment between the process and trace given some cost function $c(t)$ for each transition $t \in T^s$.

Definition 3 (Cost function, Standard cost function) *A cost function* $c : T^s \rightarrow \mathbb{R}^+$ *associates a non-negative cost to the firing of each transition of the synchronous product net (Bloemen et al., 2018), such that:*

$$c(t) = \begin{cases} 0, & t = (\gg, \tau) \ (model\ move\ on\ \tau) \\ 0, & t \in T^{SM} \ (synchronous\ move,\ e.g.\ (a,a)) \\ 1, & t \in T^{LM} \ (log\ move,\ e.g.\ (a,\gg)) \\ 1, & t \in T^{MM} \ (model\ move,\ e.g.\ (\gg,a)) \end{cases}$$

Given this, the problem of finding an *optimal alignment* is reduced to searching an execution sequence of the synchronous product with the minimum cost. The standard cost function ensures an optimal number of synchronous moves, i.e. events and process transitions that align together at the same time. A *fitness* score will be derived from the alignment (see Section 2.3.4).

4.5 Conformance checking over probabilistic events

This section presents `ProbCost`, our alignment algorithm. Traditional alignment-based conformance checking only considers traces of *deterministic* events. In our work, we instead consider a trace of m events $\langle e_0, ..., e_{m-1} \rangle$ each of which can correspond to one of n possible activities $\{a_0, ..., a_{n-1}\}$ with the *discrete probability* $p_{i,j} = p(a_j | e_i)$ that event e_i is an observation of activity a_j. This can be modelled by a probability matrix as follows:

$$\begin{array}{c} \\ a_0 \\ a_1 \\ \cdot \\ a_{n-1} \end{array} \begin{array}{c} \begin{array}{ccccc} e_0 & e_1 & \cdots & e_{m-2} & e_{m-1} \end{array} \\ \left\| \begin{array}{ccccc} p_{0,0} & p_{1,0} & \cdots & p_{m-2,0} & p_{m-1,0} \\ p_{0,1} & p_{1,1} & \cdots & p_{m-2,1} & p_{m-1,1} \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ p_{0,n-1} & p_{1,n-1} & \cdots & p_{m-2,n-1} & p_{m-1,n-1} \end{array} \right\| \end{array} \tag{4.1}$$

where entries $p_{i,j} \in [0,1]$ and $0 \leq i \leq m-1, 0 \leq j \leq n-1$ represent the probability that event e_i corresponds to activity a_j. The sum of the probabilities for each event should be 1, i.e. $\sum_{j=0}^{n-1} p_{i,j} = 1, \forall 0 \leq i \leq m-1$.

An example trace of 3 events $\langle e_0, e_1, e_2 \rangle$ each of which may correspond to 3 activity categories a, b, c can be modelled by the following probability matrix:

$$\mathbf{P} = \begin{array}{c} \\ a \\ b \\ c \end{array} \begin{array}{c} \begin{array}{ccc} e_0 & e_1 & e_2 \end{array} \\ \left\| \begin{array}{ccc} 0.3 & 0 & 0 \\ 0.7 & 0.7 & 0.3 \\ 0 & 0.3 & 0.7 \end{array} \right\| \end{array} \tag{4.2}$$

where the first event e_0 has a 0.3 probability to be associated with activity a and 0.7 probability to be associated with b.

This probability matrix can be modelled as a *weighted trace model* as follows:

Definition 4 (Weighted trace model) *Let A be a set of n activities and σ a trace of m events, where each $e_i \in \sigma$ corresponds to each $a_j \in A$ with probability $p_{i,j}$. A weighted trace model is a Petri net with weights, i.e., $(P^{wt}, T^{wt}, F^{wt}, \alpha^{wt}, w^{wt})$, where:*

- $P^{wt} = \{P_0, ..., P_{m-1}\}$ *a set of m places,*
- $T^{wt} = \bigcup_{i,j}\{t_{i,j} \mid p_{i,j} > 0\}$*, where $t_{i,j}$ is a transition for event e_i if activity a_j corresponds to it with probability $p_{i,j} > 0$,*
- $F^{wt} = \bigcup_{i,j}\{(P_{j-1}, t_{i,j}),\ (t_{i,j}, P_j) \mid t_{i,j} \in T^{wt}\}$
- $\alpha^{wt}(t_{i,j}) = a_j$*, for each transition $t_{i,j} \in T^{wt}$,*
- $w^{wt} : T^{wt} \to [0,1]$*, where $w^{wt}(t_{i,j}) = p_{i,j}$ is a function assigning a weight to each transition based on the probabilistic matrix of the event log,*

Note that, in contrast to deterministic trace models, a weighted trace model has multiple transitions for every event, linked to possible activities with different weights. The weighted trace model corresponding to probabilistic matrix (4.2) is shown in Figure 4.2.

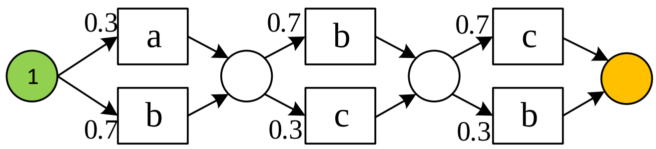


Figure 4.2: Weighted trace model of matrix (4.2).

Definition 5 (Weighted synchronous product net) *Let A be a set of n activities, $N_p = (P^p, T^p, F^p, \alpha^p)$ a process model and σ a captured probabilistic trace of m events with categorical probability $p_{i,j}$ over the activities, with the corresponding weighted trace model $N_{wt} = (P^{wt}, T^{wt}, F^{wt}, \alpha^{wt}, w^{wt})$. A synchronous product net is a Petri net with weights, i.e., $N_{ws} = (P^{ws}, T^{ws}, F^{ws}, \alpha^{ws}, w^{ws})$, such that:*

- $P^{ws} = P^p \cup P^{wt}$,
- $T^{ws} = (T^{MM} \cup T^{LM} \cup T^{SM})$, *where* $T^{MM} = \{\gg\} \times T^p$ *denotes moves on model,* $T^{LM} = T^{wt} \times \{\gg\}$ *denotes moves on log,* $T^{SM} = \{(t_1, t_2) \in T^p \times T^{wt} \mid \alpha^p(t_1) = \alpha^{wt}(t_2)\}$ *denotes synchronous moves.*
- $F^{ws} = \{(p, (t_1, t_2)) \in P^{ws} \times T^{ws} \mid (p, t_1) \in F^p \vee (p, t_2) \in F^{wt}\} \cup \{((t_1, t_2), p) \in T^{ws} \times P^{ws} \mid (t_1, p) \in F^p \vee (t_2, p) \in F^{wt}\}$,
- $\alpha^{ws}((t_1, t_2)) = (l_1, l_2)$ *for all transitions* $(t_1, t_2) \in T^{ws}$, *where* $l_1 = \alpha^p$ *if* $t_1 \in T^p$, *otherwise* $l_1 = \gg$; *and* $l_2 = \alpha^{wt}$ *if* $t_2 \in T^{wt}$, *otherwise* $l_2 = \gg$,
- $w^{ws} : T^{ws} \to [0, 1]$ *a weight function for each transition, where* $w^{ws}((t_1, t_2)) = w^{wt}(t_1)$ *if* $(t_1, t_2) \in (T^{LM} \cup T^{SM})$, *otherwise* $w^{ws}((t_1, t_2)) = 1$,

Note that the worst-case of the number of transitions T^{ws} is exponential in the number of events, which occurs when (i) the number of events in the trace is equal to the number of activities in the process model, (ii) each event has multiple transitions, and (iii) all transitions match to a corresponding labelled activity in the process model. This means that all events can be matched to every activity in the model. However, such high level of uncertainty and generality is unrealistic in the context of the applications that have motivated this work.

Figure 4.3 shows the weighted synchronous product net of our example process model (Figure 4.1) and weighted trace model (Figure 4.2).

Knowing the probabilities of each activity given an observed event allows us to calculate the cost of each transition based on the likelihood that the corresponding

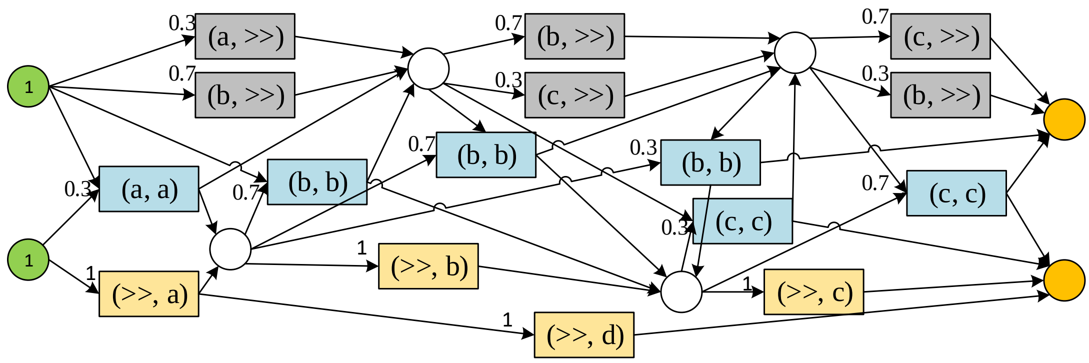


Figure 4.3: Weighted synchronous product net of our example models from Figure 4.1 and 4.2, annotated with the weight function w^{ws} and with different colours for model moves (yellow), log moves (grey) and synchronous moves (blue).

activity actually occurred. This likelihood is reflected in the weight w^{ws} of each transition. Activities with higher probability should have lower cost so that the algorithm is more likely to select them in the optimal alignment. We use a *log* transformation to transform probability products to sums of log probabilities. Based on this, we define a new, weighted cost function $c(t)$ in Definition 6:

Definition 6 (Weighted cost function) *Given a weighted synchronous product net denoted as $N_{ws} = (P^{ws}, T^{ws}, F^{ws}, \alpha^{ws}, w^{ws})$, we define a weighted cost function for each transition $t \in T^{ws}$ of the synchronous product net as follows:*

$$c(t) = \begin{cases} 0, & t = (\gg, \tau) \\ -\log(w^{ws}(t)), & t \in T^{SM} \\ -\log(w^{ws}(t)) - \log(\varepsilon), & t \in T^{LM} \\ -\log(\varepsilon), & t \in T^{MM} \end{cases}$$

where $\varepsilon \in (0,1)$ is a parameter representing the level of confidence in the event log (see Section 4.6).

We apply the negative logarithm function, denoted as $-\log$, to transform the probabilities, which range from 0 to 1, into non-negative values. This transformation is designed so that a lower probability corresponds to a higher cost, and conversely, a higher probability results in a lower cost.

Using this cost similarly to the standard alignment algorithm (see Section 2.3.4), we find an optimal alignment by using the A* algorithm (Russell and Norvig, 2016) and derive a *fitness* score based on the alignment using the same way.

Event log	$\gg$	b	b	c
Process model	a	$\gg$	b	c

(a) Standard alignment.

Event log	a	b	c
Process model	a	b	c

(b) Probabilistic alignment.

Table 4.1: Alignment results of 3 events with probability matrix (4.2) and the process model from Figure 4.1.

Overall, `ProbCost` includes the following steps: (i) build the process and weighted trace models, (ii) construct the weighted synchronous product net, (iii) use A* to find an optimal alignment based on the weighted cost function, (iv) tune the parameter ε based on the context, (v) obtain final alignment using the tuned ε.

Example 1 *Consider the process model (Figure 4.1) and the example probabilistic trace in* (4.2)*, and their composed weighted synchronous product net in Figure 4.3, we could get the cost of the synchronous move, e.g.,* (a,a) *is* $-log(0.3)$*, the cost of the model move, e.g.* $(\gg,a)$ *is* $-log(\varepsilon)$*, and the cost of the log move, e.g.* $(a,\gg)$ *is* $-log(0.3)-log(\varepsilon)$*. The optimal alignment computed by* `ProbCost` *(*$\varepsilon = 0.4$*) is shown in Table 4.1b, with 100% fitness. Compared to the results of standard alignment algorithm (shown in Table 4.1a, with 67% fitness) which assumes the most likely activities for each event, i.e. the trace* $\langle b,b,c\rangle$*,* `ProbCost` *chooses activity a for the first event, even though it has a lower probability (0.3) than b (0.7) and forms a perfect alignment with the process model. This is because the cost of choosing the same trace* $\langle b,b,c\rangle$ *as the standard alignment algorithm is higher than the cost of trace* $\langle a,b,c\rangle$*.*

This fits our intuition that even though a is less likely to have occurred given our first event observation, the process model captures the knowledge that a is expected *to occur (e.g. based on prior observations or physical restrictions), thus increasing our perceived likelihood for a. We thus use our confidence in the modelled process to mitigate the uncertainty of the event log.*

Example 2 *Considering the same process model and event log with Example 1, we change* ε *to be* 0.8*.* `ProbCost` *gets the same optimal alignment as the standard alignment algorithm, as shown in Table 4.1a.*

Compared to Example 1, this shows that the ε *parameter is able to control the acceptable probability of events. Instead of blindly accepting the activity* expected *by the process model, changing* ε *allows us to determine whether we trust the event log's*

Event log	a
Process model	a

(a) Assuming a occurred.

Event log	b	$\gg$
Process model	$\gg$	a

(b) Assuming b occurred.

Table 4.2: Possible optimal alignments of a single event with 2 possible activities a, b and process model with single step a.

indication of the most probable activity or the process model's expectation. We analyse ε *more formally in Section 4.6.*

Complexity analysis Compared to the standard A* alignment approach, whose worst-case space complexity is linear to the number of reachable states of the synchronous product net (Carmona et al., 2018), we have the same number of reachable states and, therefore, the same space complexity. Furthermore, time complexity depends on the number of transitions between initial and final states of the synchronous product net (Carmona et al., 2018). Therefore, the worst-case time complexity of `ProbCost` is exponential compared to the standard approach, when the worst-case number of transitions occurs in the synchronous product net. An empirical evaluation of computation time is discussed in Section 4.7.2.3.

4.6 Threshold parameter ε

Our weighted cost function $c(t)$ (see Definition 6) includes a "threshold" parameter ε. As hinted in the previous section, ε allows the user to control the level of confidence, or *trust*, in the uncertain event log. An intuitive way to express this is by posing the following question:

> *How probable does an event need to be for it to be judged to move synchronously with a matching activity in the process model?*

Let us explore a minimal example whereby ε allows us to control the answer to that question.

Consider a trace with a single event e_0 that corresponds to either of two possible activities a and b with probabilities $p_{0,a} = x$ and $p_{0,b} = 1 - x$, for some $x \in [0, 1]$. We want to check the conformance of this trace with a model containing a as a single step in the process. In this case, we have 2 possible optimal alignments shown in Table 4.2.

The first (Table 4.2a) contains a synchronous move which assumes the event corresponds to activity a. The second (Table 4.2b) considers that the event corresponds to activity b, leading to a model move and a log move. The choice between the two should be based on the probabilities of a and b. The question is *what value does x need to be for the first alignment to be optimal?*

In the extreme cases, if $x = 1$ then a occurred with probability 1, so the first alignment should be optimal, and symmetrically if $x = 0$ then the second alignment should be optimal. Furthermore, if we only consider the most likely activity as the one that actually occurred, which is in fact standard in activity recognition (Sztyler et al., 2016), then the second alignment would be optimal only when $x < 0.5$.

However, we posit that this decision may be influenced by the process model. For instance, consider the extreme case where the process model reflects domain knowledge that *only activity a is actually possible*, whereas activity b is not possible *at all*[2]. A high probability $p_{0,b}$ might then be attributed, for example, to noise in the data, and the second alignment should *always* be sub-optimal.

More generally, we may choose to *trust* the reality (or our expectation of it) as reflected by the process model more than a noisy and uncertain event log, i.e. perform a synchronous move on a even when $x < 0.5$. Notably, (Bogdanov et al., 2022) follows an inflexible approach compared to ours, that they always consider the first alignment to be optimal even when x approaches zero.

In our approach, the parameter ε allows us to control how low x can be for a to be chosen by the algorithm. Given Definition 6, the total cost of the alignment in Table 4.2a is $-log(x)$ (synchronous move on a), whereas the total cost of the alignment in Table 4.2b is $-log(1-x)-2log(\varepsilon)$ (move on log b and move on model). The former is optimal under the condition $\frac{x}{1-x} \geq \varepsilon^2$.

For the synchronous move on the model's expected activity a to be optimal (i.e. to assume that a occurred) the ratio of $p_{0,a}$ to $p_{0,b}$ should be more than ε^2.

The relationship between $p_{0,a}$, ε and the assumed activity is shown in Figure 4.4. For each combination of values for $p_{0,a}$ and ε above the blue line, the algorithm assumes a occurred and picks the first alignment as optimal. Otherwise below the blue line, it assumes b occurred and picks the second alignment as optimal. The lower the threshold, the lower the probability of a needs to be for a synchronous move, and therefore the more we trust the process model instead of the probability distribution of the event log. With $\varepsilon \in (0,1)$, the threshold for the probability of a can be set to be anywhere in $(0,0.5)$.

[2] Of course, such an extreme model would defeat the purpose of conformance checking to begin with, since we already know what is possible, but it helps illustrate our point.

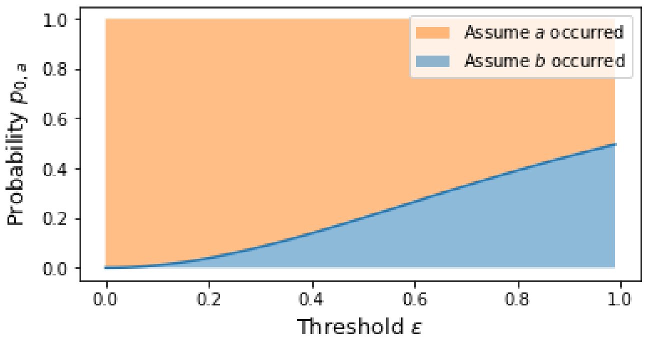


Figure 4.4: Relation between ε, $p_{0,a}$, and assumed activity.

When considering more complex process models and larger traces, with multiple possible activities per event, the interaction between ε, the activity probabilities and the alignment costs become harder to analyse. This makes the selection of the appropriate value of ε more difficult. However, a general principle applies: *when* ε *is closer to* 1*, the algorithm trusts the probabilities of the event log and considers the most probable activities. As the value of* ε *decreases, the algorithm puts more trust in the process model and accepts activities with lower probability in the event log if they align with those in the model.*

This selection of the appropriate value for ε depends on the context of each case study and our confidence in the process model. In many of our examples when choosing between optimal alignments, we have observed ε to be the threshold for the *ratio of probabilities* of the activities corresponding to each alignment. This pattern is also observed in our experimental study. Pending further research on the methods to determine the appropriate value of ε, we select its value based on (i) the developed intuition (higher values mean higher trust in the log), (ii) the ratio of the involved probabilities and (iii) tuning ε on training datasets in a supervised way (see Section 4.7.2.3).

4.7 Experimental study

In this section, we evaluate the performance of `ProbCost` for deviation detection. We first evaluate `ProbCost`'s ability to replicate the trace recovery results as presented in (Bogdanov et al., 2022, 2023) to confirm that our method is able to reflect their approach as a baseline. Next, we evaluate `ProbCost`'s performance in deviation detection compared to the standard alignment and Bogdanov et al.'s approach.

Our experiment employs two publicly available real-world datasets in the domains with structured processes information: (i) BPI Challenge (BPIC) 2012[3] related to personal loan processes, which contains 13087 cases over 36 activities (assessing the application, calling after sent offers, etc.), (ii) BPIC 2019[4] related to purchase order handling processes including 251734 cases over 42 activities, (submit order, payment, etc.). Both BPIC datasets have been used to evaluate algorithms in relevant literature on uncertain data settings (Bogdanov et al., 2022; Pegoraro et al., 2020).

4.7.1 Evaluating trace recovery

In our first experiment, we investigate the ability of `ProbCost` to recover the real trace from a probabilistic event log (trace recovery) under the same assumptions as (Bogdanov et al., 2023). Replicating their strategy on the same BPIC 2012 dataset, we first discover a process model (see Section 2.3.3) from a set of randomly selected traces, then generate probabilistic events by adding noise to the original activities of a different set of 100 traces.

For each event in each trace, we attach a second alternative activity to its original label, selected randomly from the set of activities in the dataset. The original activity is assigned a randomly chosen probability p, whereas the added activity is assigned $1-p$. A parameter $P_h \in [0,1]$ specifies the proportion of original activities with higher probabilities than their added alternatives ($p > 1-p$). When $P_h = 1$ all the original activities are assigned higher probability than the added alternatives, and vice versa when $P_h = 0$.

We then perform conformance checking with our algorithm and extract the optimally aligned sequences of events as the recovered traces. We calculate the recovery accuracy by dividing the number of correctly recovered events (compared to the original ones) by the total number of events, and compare the results against the Argmax sequence (choosing the activity with the highest probability for each event).

The threshold parameter ε is set to a low value (0.01) to simulate the intention of Bogdanov et al. to maximally align activities to the process model regardless of probability. For the model discovery, we use the Inductive Miner (Adriansyah et al., 2011) as implemented in PM4Py (Berti et al., 2019), on separate sets of 15 and 100 traces. Our results compared to Argmax are shown in Figure 4.5.

Our results match those of (Bogdanov et al., 2023, Figure 5). Specifically, when P_h is 0, recovery accuracy based on the model discovered by 15 traces (0.81) is better than

[3]https://data.4tu.nl/articles/_/12689204/1

[4]https://icpmconference.org/2019/icpm-2019/contests-challenges/bpi-challenge-2019/

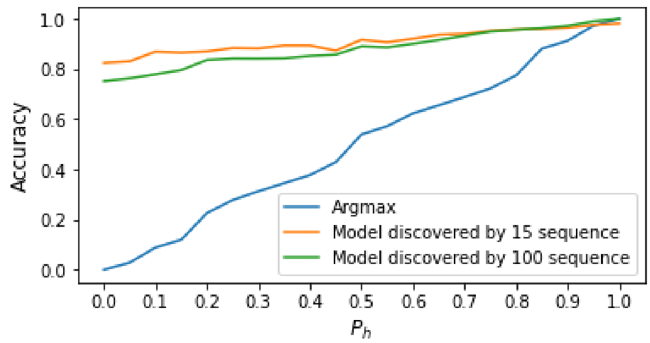


Figure 4.5: Trace recovery accuracy for different number of traces in model discovery.

the result using 100 traces (0.75), which is the same as their result. As P_h increases, the accuracy increases, until P_h reaches 1, where the same result as Argmax is obtained. This validates that their approach can be replicated by our algorithm by setting a low value of ε, close to 0 (e.g. 0.01) as a baseline.

4.7.2 Evaluating deviation detection

4.7.2.1 Experiment design

In this experiment, we evaluate the performance of our algorithm when detecting deviations on the two BPIC datasets, i.e. identifying events that do not fit the process model. In the context of alignment-based conformance checking, a log move indicates a deviation (the event is inconsistent with the process model), while a synchronous move means a normal occurrence (the event fits with the activity in the process model). We compare the performance of our approach with the standard alignment algorithm over the Argmax sequences and the approach of Bogdanov et al. Note that in contrast to the assumption we made in Section 4.7.1, if events fit the process model but occur with low probability, we regard them as deviations.

We aim to synthesise a set of probabilistic event traces in such a way that we can control the ratio of deviations, and so that there exist low probability activities that are not deviations. We accomplish this as follows.

We first perform model discovery using 20 randomly chosen traces and choose another 100 traces from the rest to generate probabilistic events using the same setting as previously. We use $P_h = 0$ so that all the added activities have higher probabilities than the original ones ($p < 1 - p$).

Standard conformance checking over the traces against the discovered model resulted in 100%, and 96% fitness for the two datasets respectively. However, for the sequences of *added* activities, the average fitness and standard deviation (SD) are 23% (0.08 SD) and 8% (0.11 SD), respectively. This means that the original traces conform to the process model (even though they were not used in discovery), whereas the added activities do not have a good fit. Based on this, we define deviations in a probabilistic trace by considering the original activities as *normal occurrences* and the added ones as *deviations*.

Furthermore, under our assumption of noisy data, we introduce a *deviation confidence* parameter T_d. Specifically, for each event, if the odds of the original activity happening, i.e. the ratio of the probability of the original activity p to that of the alternative, is higher than T_d, i.e. $\frac{p}{1-p} \geq T_d$, then we classify this event as a normal occurrence. Otherwise, the event is a deviation.

For example, for $T_d = 0.50$, if $p < 0.32$ then the alternative occurred (true deviation). However, if $p = 0.40$, then we determine that the event is in fact an actual occurrence. Note that $p < 0.50$ always, since $P_h = 0$.

If T_d is close to 0, the original activity is a normal occurrence even when its probability is very low, as long as it is non-zero, resulting in very few deviations. As T_d increases from 0 to 1 the number of normal events decreases while the number of deviations increases.

Therefore, in this setup, we control the ratio of deviations using T_d and normal occurrences have low probabilities ($P_h = 0$).

We use *accuracy* (see equation (2.1) in Section 2.1.1), *F1-score* (see equation (2.4) in Section 2.1.1), *sensitivity* ($\frac{TP}{TP+FN}$) and *specificity* ($\frac{TN}{TN+FP}$) to measure the performance of deviation detection based on the counts of True Positives (TP), False Positives (FP), True Negatives (TN) and False Negatives (FN) (Arifoglu and Bouchachia, 2019). We also use the Geometric mean (G-mean) metric to balance both sensitivity and specificity on imbalanced datasets (Kubat et al., 1997) by equation 4.3.

$$G\text{-}mean = \sqrt{sensitivity \times specificity} \tag{4.3}$$

4.7.2.2 Tuning threshold ε

Next, we set the *deviation confidence* T_d to 0.25, which means the probability of the original activity should be higher than 0.20 to be considered as a normal occurrence. We evaluate the performance of `ProbCost` over the development set (70% of the whole data) under different values of ε, ranging from 0.05 to 1 with a 0.05 step. The accuracy, F1-score and G-mean are calculated for each value of ε to determine the optimal value that yields the best deviation detection performance, as these metrics are suitable for evaluating imbalanced datasets. The results are presented in Figure 4.6. The best performance was achieved for ε between 0.25 to 0.30 for two datasets (variation due to class proportions), which is basically equal to the deviation confidence T_d. This follows the ε selection strategy discussed in Section 4.6. Based on this, we set $\varepsilon = T_d$ for further experiments.

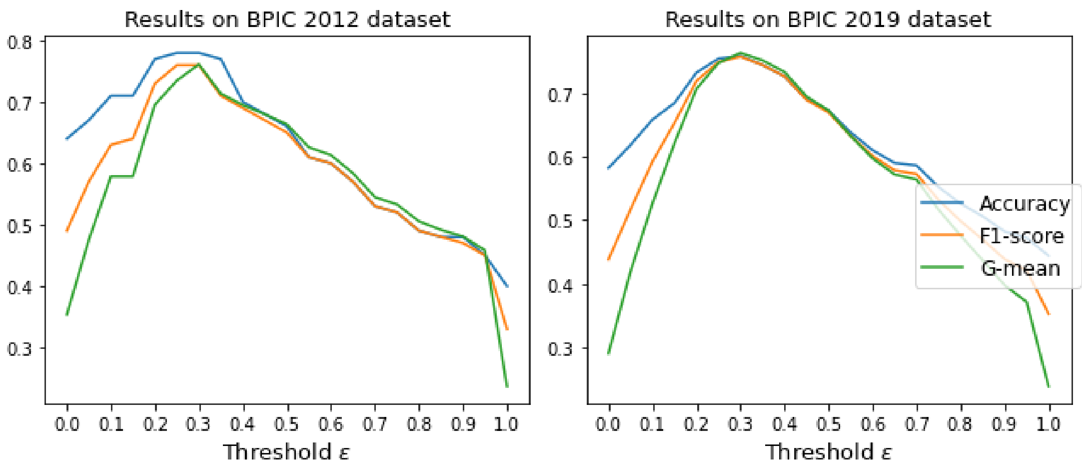


Figure 4.6: Deviation detection performance for different values of ε ($T_d = 0.25$).

4.7.2.3 Results

We assess the deviation detection results using our algorithm ($\varepsilon = 0.25$) compared to the standard alignment conformance checking algorithm as implemented in PM4Py and the approach by Bogdanov et al. (replicated by our algorithm with $\varepsilon = 0.01$). The comparative results are shown in Figure 4.7.

Based on these results, `ProbCost` performs better than both the standard alignment algorithm and Bogdanov et al. in accuracy, F1-score and G-mean metrics. The approach of Bogdanov et al. has the lowest *sensitivity* and mostly highest *specificity* because it always selects the activities that fit in the process model even when their probability is very low, leading to more false negatives. In contrast, the standard alignment algorithm has the highest *sensitivity* and lowest *specificity*, because it may ignore the normally occurring activity even if its probability is close to the probability of the alternative, leading to more false positives. `ProbCost` achieves a better *G-mean* score than the other two approaches, which suggests that it is better at correctly identifying both positive and negative cases, resulting in a better overall performance. In brief,

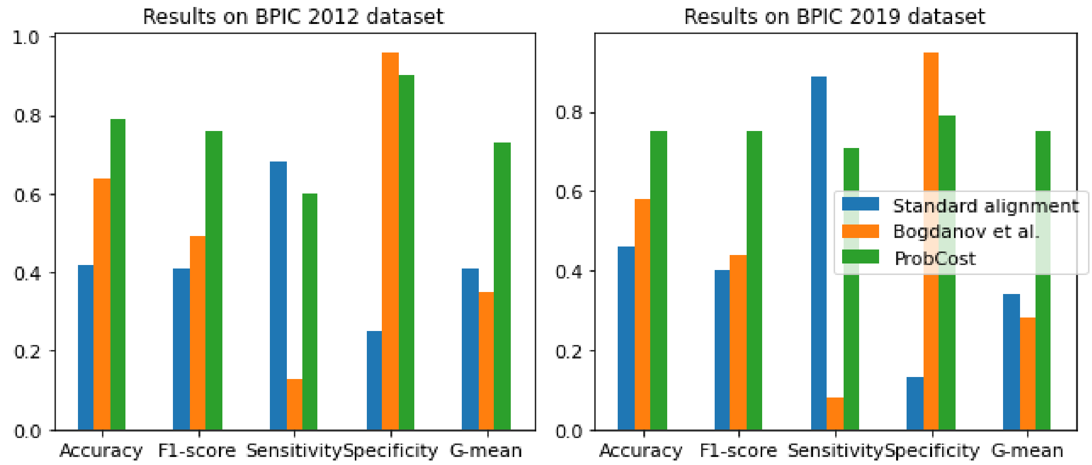


Figure 4.7: Results for deviation detection on different datasets ($T_d = 0.25$).

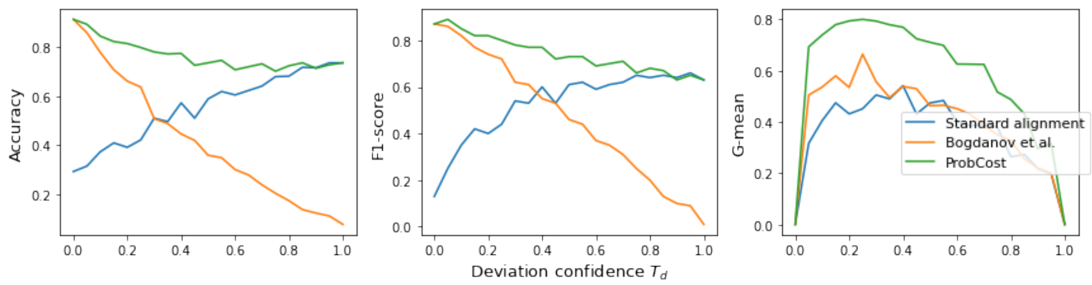


Figure 4.8: BPIC 2012 results for different values of T_d.

our algorithm can better balance the confidence between the process model and the uncertain event log to obtain better results than both the others.

We further investigate the three algorithms under different deviation confidence values T_d. For this, we iterate T_d from 0 to 1 with a 0.05 step to generate deviation data before running the 3 algorithms. As mentioned previously, we set $\varepsilon = T_d$. The approach of Bogdanov et al. is replicated by setting $\varepsilon = 0.01$.

`ProbCost` outperforms the others for all the values of T_d in the two datasets. Results for the different datasets show similar tendencies — the results of BPIC 2012 and BPIC 2019 datasets are shown in Figure 4.8 and Figure 4.9 respectively. For lower values, as expected, our performance matches that of Bogdanov et al. as no matter the probability of the original activity, it can always be aligned with the expected activity in the process model to identify a normal occurrence. However, the standard alignment algorithm gives the lowest accuracy and F1-score because it chooses the most probable activity for each event, resulting in more false positives. All three algorithms produce the lowest G-mean when $T_d = 0$, because there are no deviations, causing the

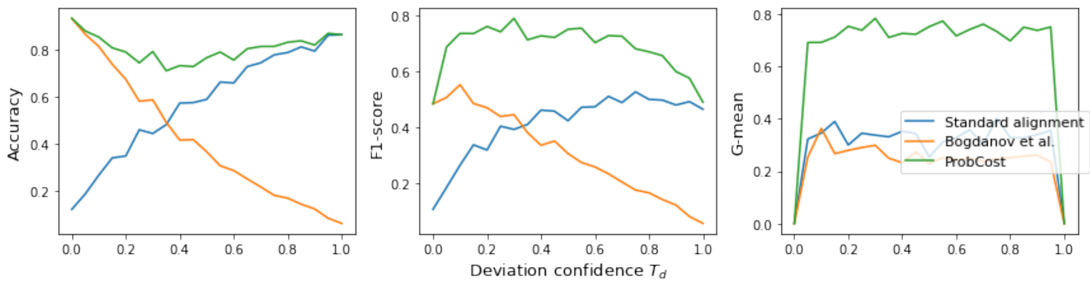


Figure 4.9: BPIC 2019 results for different values of T_d.

sensitivity to be 0. Conversely, there are no normal occurrences when $T_d = 1$, causing specificity to be 0.

As the deviation confidence T_d increases, `ProbCost` achieves the best performances among the approaches. The accuracy and F1-score of the approach by Bogdanov et al. decrease, while they increase for the standard algorithm. This happens because as T_d increases, the original activities need to have a higher probability to be considered as normal occurrences, but Bogdanov et al. accept activities expected by the process model even with low probability, resulting in more false negatives. `ProbCost` can better balance the confidence on uncertain events and select the activity with sufficiently high probability to achieve the best deviation detection accuracy. When T_d is increased to 1, our approach yields the same accuracy and F1-score as the standard alignment algorithm. As we put more trust in the event log (higher ε), we end up considering the most probable activities, similarly to the standard alignment algorithm.

Finally, we assess the execution time of `ProbCost` on two datasets with different sizes of events and process models. The results (shown in Table 4.3) indicate that the execution time of `ProbCost` is longer than the standard approach as expected by the complexity analysis (Section 4.5) because the number of transitions increases. The execution time is approximately double that of the standard approach, but it is still considered acceptable given the enhanced capabilities and complexity of `ProbCost`. This indicates that our approach is applicable in practice.

Overall, our approach demonstrates more resilience to noise and uncertainty in the events, and flexible control of how much the algorithm should trust the event log over the structure of the process model via ε.

Table 4.3: Comparison of computation time between standard alignment and `ProbCost` for different datasets.

Datasets	#Cases	#Events per case				Process model		Computation time per case (in seconds)	
		min	max	avg	median	#places	#transitions	Standard	`ProbCost`
BPIC 2012	100	3	56	24.7	27.5	44	59	356.150	865.201
BPIC 2019	100	1	10	5	5	12	18	0.174	0.629

4.8 Conclusion

In this chapter, an extended alignment conformance checking algorithm is proposed for deviation detection with probabilistic event logs in structured domains. Specifically, we propose a weighted trace model for events with a categorical probability distribution, a weighted alignment cost function, and a custom threshold parameter that controls the level of confidence on the event log vs. the process model. Furthermore, we compared the complexity results with the deterministic alignment approach. Two sets of experimental studies comparing our approach to the deterministic alignment algorithm and the recent relevant approach by Bogdanov et al. show that our algorithm takes into consideration occurrence probabilities explicitly and is able to accommodate activities with lower, but sufficiently high probabilities if they fit the model. Based on this, we argue that our algorithm can better tolerate noise in the event log by leveraging the knowledge captured in the process model with the aim of reducing the false positive and false negative rates in deviation detection. As such, it is better suited to perform deviation detection under an uncertain event log, such as one produced by sensors or AI-based algorithms (Cohen and Gal, 2021).

We recognise the necessity to delve deeper into the impact of the parameter ε in our algorithm. Currently, our approach relies on empirical evaluation on the training set to guide the selection of an appropriate ε value. Further formal investigation into this area is essential to refine our understanding of the parameter's influence, which will help guide the selection of parameters based not just on empirical training but also on domain-specific knowledge. We have currently considered the occurrence probabilities of events and it would be interesting to explore other types of uncertainty of event data, e.g., repeating or missing events.

This chapter investigated how structured process information can help reduce the uncertainty in event logs for detecting deviations. The next one will explore how semi-structured process information can be integrated into recognising activities. We will present a process-driven activity recognition framework that integrates the `ProbCost` algorithm to utilise process information from semi-structured domains.

Chapter 5

Process-driven Human Activity Recognition in Semi-structured Domains

5.1 Introduction

Semi-structured domains occupy the space between structured and unstructured ones, characterised by a degree of flexibility yet governed by underlying processes. This inherent flexibility introduces unique challenges, particularly in defining processes that accurately represent the varied patterns of behaviour observed within these settings. This chapter focuses on investigating the potential of utilising process information within semi-structured domains to improve pattern recognition, with a particular focus on human activity recognition (HAR).

Analysing patterns of human activities offers a multitude of benefits across various domains, from enhancing personal fitness and healthcare monitoring to optimising work environments for safety and efficiency. For example, in healthcare, monitoring stroke patients' daily activities and movements can track their recovery progress and provide intervention when necessary. In elder care, recognising human activities can alert caregivers in case of falls or an unusually long period of inactivity in one location. These alerts can be sent to caregivers for quick response and attention, improving the quality of life for the elderly.

With the rapid development of sensor technologies and smart devices, large amounts of data related to human movement and behaviour are generated. Human activity recognition (HAR) refers to using Artificial Intelligence (AI) techniques to identify and recognise human activities based on the gathered data from various sensors. These sources include wearable sensors (Cook and Schmitter-Edgecombe, 2021), sensors equipped on smart phones, e.g., inertial sensor (Pratap et al., 2022), camera devices (Raza et al., 2023), ambient sensors (Cook et al., 2013a), etc. The HAR techniques have been adopted in various domains, such as healthcare (Liu et al., 2022), surveillance (Jindal et al., 2022), remote care to elderly people living alone (Thakur and Han, 2021), and performance monitoring applications for sports and exercise (Shao et al., 2020).

However, traditional methods of HAR mainly focus on leveraging the power of the machine and deep learning techniques to extract information from data for activity recognition (Gupta et al., 2022). HAR based on these methods relies solely on the inherent traits and relationships within the data. Such an approach can only learn from whatever data are fed to it, but neglect the context in which data are generated. This may result in low accuracy and lack of interpretability, particularly in situations with limited data, or poor quality data (Gupta and Sheng, 2020). For example, as we discussed in Section 1, the activities of *drinking from a cup* and *answering a phone* involve similar arm movements (Gupta and Davis, 2007). Thus, it is difficult to distinguish these kinds of similar activities based on motion data alone.

To address this limitation, the idea of incorporating domain knowledge or contextual information into machine and deep learning methods has been investigated to improve the accuracy and interpretability (Guo et al., 2023). Semi-structured domains encapsulate structured contextual information of established best practices or standardised processes of conducting activities. Incorporating this type of information into the data-driven HAR models mentioned above can help enhance their performance. Taking the example of distinguishing between the activities of *drinking from a cup* and *answering a phone*, it is helpful to consider the typical sequence of actions associated with each activity. For instance, the process of drinking often starts with approaching the location of the cup, followed by reaching for and picking up the cup, then drinking from it and finally putting down the cup. By understanding and recognising this process, we can more accurately infer that if a series of actions match this pattern, the current activity is likely *drinking from a cup* rather than *answering a phone*.

However, capturing this process information poses challenges in semi-structured domains where activities may not follow a strictly defined sequence. In these domains, the variability of how tasks are performed makes it difficult to define a structured process. For instance, the same activity of drinking from a cup could involve different sequences, e.g., an individual could be interrupted when they receive a text message or doorbell ringing, altering the usual sequence.

Therefore, the aim of this work is to enhance the performance of traditional HAR models by incorporating the process information from context. In this chapter, a novel framework is proposed that learns from both the outputs from traditional HAR models and process information from semi-structured domains, and adaptively weights these two sources during the training phase. An experimental study is investigated based on a dataset collected by cameras whilst observing people eat (Raza et al., 2023). The result shows our approach can achieve better accuracy and Macro F1-score compared to two well-known Graph Convolutional Networks (GCNs) algorithms (Shi et al., 2019; Chen et al., 2021) for activity recognition.

We further motivate our work with an example of eating behaviour in Section 5.2. We present the related work of human activity recognition in Section 5.3. Then we present our proposed process-driven HAR method in Section 5.4. Section 5.5 presents an experimental study based on a real-life dataset. Finally, we conclude this work in Section 5.6.

5.2 Motivating example

Eating behaviour monitoring has seen increased attention in recent research, particularly in the context of social care and healthcare (Wu et al., 2022). Recognising an individual's eating actions such as picking up food and placing food in their mouth, can help analyse their eating behaviour and provide valuable insights into their health condition. For example, consistently monitoring how quickly a person eats, the size of the bites they take, or their ability to steadily hold utensils can reveal indicators of potential health issues, such as neurological disorders or physical impairments (Tufano et al., 2022). Moreover, the importance of recognising eating actions also extends to the development of assistive technologies, such as robotic arms for assisting eating, particularly for individuals with disabilities or the elderly.

Existing work has investigated how machine learning and deep learning-based classification models can be used for recognising activities (Wu et al., 2022; Raza et al., 2023). These models are proficient in extracting features from various types of input data, such as images and sensor readings. They generate an output matrix that represents the prediction probabilities for the input across different classes of eating actions, like *chewing*, *picking up cutlery*, and *putting down cutlery*. This output matrix is a probability distribution across possible activity classes for any given input. The recognised activity is determined to be the one associated with the highest probability in the matrix.

However, due to the similarities between different classes of actions, selecting the class with the highest probability does not always guarantee accuracy. This is because similar actions may result in similar probabilities. For example, the algorithm may classify some sensor input associated with the activity of *putting down cutlery* as the *picking up cutlery* activity with 51% probability and *putting down cutlery* with 49% probability, resulting in a misclassification.

Eating behaviour possesses characteristics of a semi-structured domain, where there is inherent structured process information embedded within the execution of eating activities. For example, there is a typical order of eating activities, such as *picking up a fork*, *picking up food using the fork*, *putting food into mouth*, *chewing*, etc. Understanding these structured aspects, such as the processes involved in eating, allows us to capture the context in which the activity is performed, thus improving our ability to make accurate activity recognition. For example, *picking up cutlery* is likely to be followed by an activity related to food consumption (like *picking up food using a fork*), whereas *putting down cutlery* might be followed by activities like wiping one's mouth or reaching for a drink. Therefore, by incorporating the process information, we can reduce the uncertainty of activity recognition algorithms caused by inter-class similarities and enhance the accuracy of activity recognition.

However, eating behaviour also includes a degree of flexibility in the sequence and manner of performing eating actions, which complicates the task of establishing a structured process that could be incorporated to enhance activity recognition. For example, while a typical sequence at mealtime might involve picking up cutlery, using the cutlery to pick up food, and then eating the food, this sequence can be altered based on situational factors. For instance, a person could be interrupted by a phone call or need to attend to a child, causing them to put down their cutlery unexpectedly and alter their usual sequence.

Moreover, individuals often have personal eating habits that vary widely, such as the order in which they eat their food. Some might prefer starting with lighter items like soup before progressing to heavier main dishes, while others might dive straight into the main course. Additionally, the type of food being consumed often determines the utensils required. For example, soup is eaten with a spoon, steak requires a knife and fork, and sushi is typically eaten with chopsticks. These variations mean that eating different types of food can involve distinct sequences of actions. Sometimes, people might choose to eat with their hands instead of using cutlery, bypassing some of the steps in the typical sequence. These variations make it challenging to capture structured process information of eating behaviour.

5.3 Related work

In this section, we explore the related research in HAR, which includes three main categories: data-driven HAR, knowledge-driven HAR and hybrid HAR that integrates both data and knowledge. Lastly, we provide a summary and explain how our work links to the existing work.

5.3.1 Human activity recognition frameworks

HAR aims to identify activities using AI methods. Similarly to any standard machine learning approach, HAR techniques include four main stages: data acquisition, data pre-processing, model training and performance evaluation.

In the stage of data acquisition, the devices for collecting data are selected depending on the target application. For example, in the applications of surveillance or surgical procedures, cameras are predominantly employed to gather visual data (Jindal et al., 2022). Conversely, in applications where we want to monitor an individual's routine activities, sensors, like wearable or unobtrusive ambient sensors, are favoured due to their non-intrusive nature and reduced computational demands (Cook et al., 2013a). The collected data sometimes suffer from issues of noise due to malfunctions of sensors or network latency, which poses challenges in pre-processing data.

In the data pre-processing stage, initial efforts are focused on data cleaning. This involves utilising various filters to suppress noise, reduce extraneous information, or enhance the quality of images (Jindal et al., 2022; Liu et al., 2022). Subsequent to data cleaning, data segmentation is conducted, which involves dividing the continuous stream of data into meaningful and analysable segments. This segmentation is a critical step as it enables the effective isolation of relevant activity patterns necessary for accurate recognition and analysis. For instance, data is segmented into discrete instances corresponding to different activities. Then the segmented data can be used for building models.

The approaches of building models for HAR can be divided into two broad categories: data-driven approaches and knowledge-driven approaches. For data-driven approaches, it is very important to have a robust feature extraction scheme to ensure better prediction. There are approaches where machine learning (ML) models are trained using hand-crafted features, which are meticulously designed based on domain expertise (Cook et al., 2013b). These features often encapsulate temporal and spatial characteristics pertinent to the activities of interest. Alternatively, deep learning (DL) techniques streamline this process by providing automated feature extraction capabilities (Liciotti et al., 2020). Techniques such as Graph Convolutional Networks (GCNs) (Shi et al., 2019), Convolutional Neural Networks (CNNs) (James et al., 2023), and Recurrent Neural Networks (RNNs) (James et al., 2023) have the ability to autonomously identify relevant features from raw data, thus potentially uncovering complex patterns that might elude human designers. In knowledge-driven approaches, an activity model is built through the incorporation of rich prior knowledge and heuristics gleaned from the application domain. Example knowledge-driven approaches include employing ontology or leveraging logical reasoning techniques (Bouchabou et al., 2021).

After building models, the evaluation of the HAR model's performance is conducted. This stage involves using a set of metrics that may include, accuracy, precision, recall, F1-score, etc., to evaluate the model's effectiveness in accurately classifying and recognising activities (see Section 2.1.1).

5.3.2 Data-driven HAR models

ML and DL techniques have been extensively applied in exploiting generated data for activity recognition. There are two types of data-driven approaches for HAR: supervised and unsupervised (see Section 2.1). In supervised approaches, a mathematical model is created based on the relationship between input activity data and output labels of activities. The idea behind the unsupervised approach is to detect patterns in input activity data without prior knowledge of output labels.

ML-based approaches rely on hand-crafted features for activity recognition. Hand-crafted features intend to retrieve the temporal and spatial features from datasets based on human perception and historical contexts. For example, Aminikhanghahi et al. propose an activity recognition algorithm based on data collected by ambient sensors in smart homes (Aminikhanghahi and Cook, 2019). They extract features into three categories: temporal features, such as time of the day, sensor features, such as the number of occurrences of each sensor, and window features, such as the most frequent sensors within a fixed time window. A Random Forest classifier is utilised to recognise activities based on the extracted features.

There also has been a lot of interest in using unsupervised learning techniques to recognise activities. Unsupervised approaches are suitable in scenarios where the majority of the acquired activity data has no labels. Clustering techniques are mostly used in these scenarios for the extraction and the selection of similarities and common characteristics. Clustering techniques can leverage both hand-crafted features and automatically extracted features by neural networks. For example, Sheng et al. propose an unsupervised learning approach for HAR using wearable sensors (Sheng and Huber, 2020). This approach leverages neural networks to extract temporal coherence features of time series data. The k-means clustering algorithm is used to cluster the data based on extracted features.

DL-based activity recognition is gaining much attention because of its remarkable performance and power in learning high-level abstractions from data. The basic idea behind DL models is data representation, which is able to automatically extract optimal features for activity recognition tasks. Deep neural networks, such as CNNs, RNNs, and RNN's variant long short-term memory (LSTM), etc., has been researched in the application of HAR. For example, Deep et al. leverage CNNs for vision-based activity recognition (Deep and Zheng, 2019).

Alshammari et al. present a comparative evaluation of state-of-the-art machine learning techniques to classify activities, including Gradient Boosting Machines, Decision Trees, Support Vector Machines (SVM), and deep neural networks (Alshammari et al., 2018). Their results indicate that DL-based techniques have shown superiority over the other tested techniques.

ML and DL techniques require a large amount of data for training. In contrast, classic statistical methods are more effective when only a smaller dataset is available. One of the well-known statistical methods, i.e., Hidden Markov Models (HMM), has been widely studied for HAR (Crandall and Cook, 2009; Kim et al., 2015). For example, Crandall et al. leverage HMM for recognising activities in multiple residents environments (Crandall and Cook, 2009).

5.3.3 Knowledge-driven HAR models

Besides the traditional ML approaches leveraging large data sets for training models, there is another strand of research focusing on modelling human activity patterns based on prior knowledge. The commonsense knowledge of how people perform daily activities, provides rich links between the environment and activities (Chen et al., 2012). For example, the activity of brushing teeth contains actions involving using a toothbrush and toothpaste, operating a water tap, rinsing with a cup and drying with a towel. In addition, an inhabitant's preferences of performing daily activities provide prior personal level details about their daily activities. Such domain knowledge and prior knowledge of personal preferences are valuable in creating activity recognition models.

Knowledge-driven HAR approaches are established upon the observations that most human activities, specifically, Activities of Daily Living (ADLs), take place within specific temporal and spatial context (Bouchabou et al., 2021). For example, brushing teeth typically takes place in the bathroom, conventionally twice daily—once in the morning and once before bedtime. This activity is characterised by the interaction with particular objects, such as toothpaste, toothbrushes, cups, etc. These inherent relationships between activities, their associated time and space, and the objects involved provide a diversity of hints and heuristics for activity recognition.

Perkowitz et al. propose a method of building activity models through mining the human descriptions of the ways of performing activities (Perkowitz et al., 2004). The human descriptions include directions for performing a particular activity and the objects involved in this activity, etc. This information is transformed into an activity model utilising Dynamic Bayesian Networks (Murphy et al., 2002), which is subsequently leveraged for inferring activities.

Chen et al. propose an event calculus-based logical framework for behaviour reasoning (Chen et al., 2008). They encode the commonsense knowledge of performing activities and personal preferences into event calculus formulae, which then can be applied to infer the activity.

Yamada et al. use ontology to represent the relationship between objects and activities (Yamada et al., 2007). For example, a teapot is used in an activity of tea preparation. This approach can automatically detect possible activities related to an object.

Chen et al. propose an ontological activity modelling and representation approach for activity recognition (Chen et al., 2012). They manually construct ontology-based activity models, modelling activities as a hierarchy of classes with each class described by a number of properties. Their properties include the interrelations between objects, e.g., door, cabinet, etc., and activities, as well as how the activity is performed, e.g., objects used and the sequential order they are used in. With this ontology-based model, the task of activity recognition is transformed into ontological reasoning, i.e., to discover the most closely matched activity model.

Civitarese et al. propose an unsupervised activity recognition framework based on ontology (Civitarese et al., 2021). They derive semantic correlations between activities and sensors from the ontology that models activities and sensor deployment infrastructure. A Markov Logic Network (MLN) (Richardson and Domingos, 2006) is transformed based on the derived semantic correlations. Knowledge-based constraints are derived to express the conditions of sensor events during the occurrence of a particular activity. Based on the MLN and knowledge-based constraints, they perform probabilistic reasoning to calculate the most probable occurring activities, which is identified as the recognised activity.

Compared to the data-driven ML approaches, knowledge-driven approaches are semantically clear in modelling and representation, which offers a high capability of explanation and interpretation. However, it requires extensive domain expertise to develop the rules and logic used for activity recognition. These knowledge-driven models might struggle with scalability and flexibility compared to data-driven models, which can automatically adjust to new data through learning algorithms.

5.3.4 Hybrid HAR models

There are also research initiatives that incorporate domain knowledge into data-driven models for recognising human activities. By integrating domain knowledge, these models can guide the learning process, ensuring that the patterns recognised by the machine learning algorithms align with established theoretical understandings of human behaviour.

For example, domain knowledge can inform feature selection, helping to identify which aspects of the data are most critical for recognising specific activities. Niemann et al. propose a context-aware activity recognition approach for recognising human activities relevant to production and logistics (Niemann et al., 2022). The context information, i.e., objects involved in human activities like picking carts and various racks, are also incorporated as features for building machine learning models.

The domain knowledge can also be used to structure data-driven models' output, ensuring that the results are interpretable and meaningful within the context of the application. Asim et al. propose an approach for recognising fine-grained activities based on smartphone accelerometer data by incorporating location information (Asim et al., 2020). Their methodology begins with the identification of basic activities like running, sitting, walking, etc., utilising a data-driven approach that relies primarily on patterns detected in accelerometer data. Then they incorporate the location information to refine the basic activities into fine-grained activities. For example, if the accelerometer data indicates “sitting”, the addition of location data can further classify this activity based on the person’s location, such as distinguishing between “sitting in a car” if the GPS data indicates movement along roads, or “sitting in a meeting” if the location corresponds to an office setting.

Our work also incorporates domain knowledge to refine the data-driven models’ output. Specifically, we leverage process information from context to enhance the performance of data-driven HAR models.

5.3.5 Summary

In summary, the approaches for HAR can be divided into three categories: data-driven approaches, knowledge-driven approaches, and hybrid approaches. Knowledge-driven approaches propose to model activities following domain-specific knowledge. They prioritise the logical structuring of activities according to predefined rules and concepts from the relevant domain. Data-driven approaches emphasise the identification and extraction of features from data, seeking to establish a connection between the input (data) and the output (recognised activities). These approaches leverage algorithms to learn patterns and correlations within the data, enabling them to make predictions or classifications without relying on explicit domain knowledge. Hybrid approaches utilise both domain knowledge and data when building HAR models.

Our work adopts hybrid approaches, focusing on integrating process information into data-driven HAR approaches. This integration aims to leverage the strengths of both approaches, using the learning capacity of data-driven HAR models while enhancing their accuracy with the process information in the domain.

5.4 Method

Our proposed approach for process-driven human activity recognition consists of 4 stages: i) extracting a probability distribution matrix from ML or DL models, ii) discovering process models, iii) conducting an alignment by using our proposed `ProbCost` approach in Chapter 4, and iv) retrieving activities from the alignment. We describe each stage in more detail next.

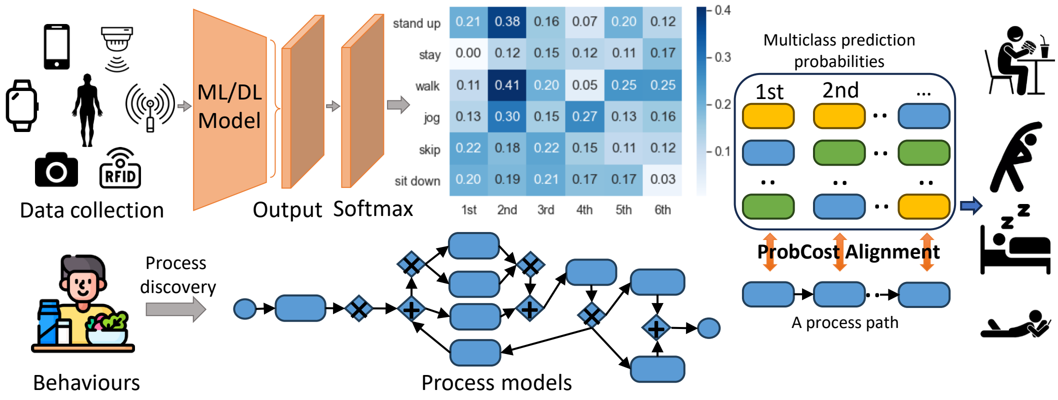


Figure 5.1: The framework of process-driven human activity recognition.

5.4.1 Overview of the framework

Figure 5.1 shows the framework of our proposed approach of process-driven human activity recognition. The framework takes the output of multiclass prediction probabilities from ML or DL algorithms as uncertain event logs. Process models are discovered from activity-labelled data. Taking both uncertain event logs and process models as input, we apply our proposed method of alignment-based conformance checking over probabilistic events, `ProbCost`, as presented in Chapter 4, to achieve an alignment. `ProbCost` is used to calibrate the results of ML/DL-based activity recognition algorithms by incorporating process information. An optimal alignment is obtained between the process model and the uncertain event log. Then we retrieve the events from the achieved alignment as the final recognised activities.

We present the details of different parts in the framework next.

5.4.2 Process discovery

We use Process mining techniques (see Section 2.3) to discover process models in human behaviours based on data collected from various sensors. This data labelled with specific activities are used to build an event log, which contains the relevant information for process mining (as we introduced in Section 2.3.1), including timestamps, performed activities, and cases. Each case in the log can be defined based on a specific period, such as every day or week, or by specific instances of an activity, such as each mealtime. Then the event log can be used in the available process mining tools for discovering processes, such as PM4Py.

Underlying processes in semi-structured domains are less structured with a certain degree of flexibility compared to the processes in structured domains discussed in Chapter 3 and Chapter 4. As we presented in Section 2.3.3, there are various process discovery algorithms, each suited to different scenarios. Inductive miner is known for its robustness in creating process models that guarantee replayability of the logs. The Inductive miner is particularly effective in structured domains, ensuring that every detail in the event log is accounted for and accurately represented in the model. Heuristic miner is particularly effective for discovering meaningful process models in scenarios where data exhibits considerable variation yet contains structured patterns. The Heuristic Miner is well-suited for such settings because it focuses on identifying the most frequent paths in the event log, capturing the main activity flows. Compared to the Inductive miner which might struggle with the flexibility and irregularities in semi-structured domains, the Heuristic miner emphasis on frequency helps to identify the main behaviour in the event log.

Therefore, in the context of semi-structured domains, we apply Heuristic miner to discover meaningful process models. The discovered process models are represented as Petri nets for further alignment analysis.

5.4.3 Extracting probabilistic traces

Traditional ML and DL-based classification models can provide a probability distribution over a set of potential classes for a given input, termed as *multiclass prediction probabilities*. For example, neural networks often conclude their prediction output with a softmax layer, while decision trees and random forest algorithms can estimate probabilities for each class (James et al., 2023). These probabilistic outputs serve as a measure of the model's confidence level in its prediction. Each class's probability reflects the likelihood that the given input belongs to that class.

The probabilities associated with each potential activity or class can be extracted as probabilistic events. We construct a *probabilistic event trace* by following the chronological order in which activities occur, which is modelled as a matrix defined in equation (4.1) in Section 4.5.

Each row in the matrix corresponds to a potential class of activity for which the model provides a probability. Each column in the matrix represents a specific activity or time step at which predictions are made. Each cell in the matrix contains the probability (ranging from 0 to 1) that the model assigns to the corresponding class for the given activity. The sum of the probabilities for each activity should be 1.

5.4.4 Alignment between probabilistic traces and process models

With the discovered process models and extracted probabilistic event traces, we can perform alignment between these two components to calibrate the classification results. The aim of the alignment is to find a possible activity class in each probabilistic event by incorporating inherent structured process information in the domain to enhance activity recognition. Our proposed method `ProbCost` acts as a bridge between the probabilistic events from activity recognition algorithms and the discovered process model.

We apply `ProbCost` to achieve an alignment between the probabilistic event traces output from ML predictions with the process model. Activity classes selected in the achieved alignment serve as the final recognised activities.

As we introduced in Section 4.5, there is a confidence threshold ε in the cost function of `ProbCost`. This threshold ε adjusts the cost associated with each type of move in the alignment based on the likelihood that a particular activity occurred as predicted by ML algorithms. This allows the alignment to take into account probabilistic events that may have a lower probability but align better with the process model. The threshold ε serves as a control mechanism to balance the trust between these two aspects. As presented in Section 4.6, a lower value of ε means that we allow an activity with a lower probability to achieve an alignment with the process model. This lower value indicates that we can select an activity class with low probability in ML predictions but can better mirror actual occurrences represented by the process model. Conversely, a higher value of ε means that an activity to be aligned with the process model needs a higher probability. In this case, activities with higher probabilities are selected to achieve the alignment. Therefore, the alignment results with higher ε are similar to the ML predictions, i.e., selecting activities with higher probabilities.

We treat the ε as a hyperparameter to tune the cost function with the aim of achieving the highest activity recognition accuracy. To determine the optimal setting of ε, we utilise a portion of our datasets as a validation set for tuning and selecting the ε (see Section 5.5.2). By adjusting the value of ε and observing the resulting changes in accuracy on the validation set, we select the ε that can achieve the highest accuracy in the validation set as the optimal value of ε for final evaluation.

5.5 Experimental study

5.5.1 Dataset description

Our experimental study is based on the EatSense dataset, which uses RGBD cameras to capture the eating behaviours of participants (Raza et al., 2023). This dataset records the upper-body movements of individuals while they were eating. It encompasses data from participants who use a range of utensils to consume different types of food. Within this dataset, 16 distinct eating actions such as *chewing*, *drinking*, *food in hand at table* have been densely labelled. The full list of actions is shown in Table 5.1.

No.	Activities
1	chewing
2	move hand towards mouth
3	pick food from utensil with tools in both hands
4	pick food from utensil with one hand
5	food in hand at table
6	pick food from utensil with tool in one hand
7	pick up tools with both hands
8	move hand away from mouth
9	no action
10	other
11	pick up a cup/glass
12	put the cup/glass back
13	put one tool back
14	pick food from utensil with both hands
15	eat it
16	drink

Table 5.1: The list of eating actions included in the datasets.

The dataset includes a trimmed dataset, where the videos have already been manually segmented into separate clips for each action. These trimmed clips are arranged in chronological sequence, mirroring the order in which the actions were performed by the individuals. Each video includes a complete process of eating behaviour for one subject.

The dataset includes videos capturing the eating behaviours of 6 subjects eating

9 different types of food, such as rice, toast, roti (also known as chapati, which is a common staple in India), etc. The distribution of videos across these subjects and food types is detailed in Table 5.2a for subject categories and Table 5.2b for food categories.

For each video, the skeleton of upper-body poses is extracted. The datasets consider 8 human body keypoints, including the nose, chest, right and left shoulder, etc. The extracted skeleton data can be used by GCNs for activity recognition (Shi et al., 2019).

Subject	Numbers of Videos
0	5
1	19
2	2
3	2
4	2
5	2

(a) The number of videos for each subject.

Food Type	Numbers of Videos
Rice	4
Toast	4
Only-Drinks	3
Roti	13
Egg	1
Soup	3
Wafers	1
Chicken-steaks	1
Noodles	2

(b) The number of videos for each food type.

Table 5.2: The distribution of videos across subjects and food types.

5.5.2 Experimental design

Recognising that different individuals consuming different types of food entails a distinct set of processes, we undertake a classification of the videos into distinct categories through two strategic approaches to accurately capture the process information in eating behaviour. Firstly, we focus on discovering the process model associated with each individual, recognising the personal patterns and habits in their eating behaviours. The second strategy is to discover the process models linked to consuming various types of food, taking into account that each food category demands specific eating actions, sequences, and possibly different utensils.

Within the EatSense dataset, the number of videos associated with *subject 1* is notable for comprising more than 10 videos, providing a significant sample size for analysing processes. However, it is important to recognise that while this number allows for meaningful analysis, it still represents a relatively small dataset. Similarly, the videos related to *eating roti* are distinguished by having 13 video cases. Even though these are the highest number of videos within their respective sets they are still relatively low numbers for machine learning purposes.

Despite this, the available quantity of videos allows for training machine learning models and a detailed exploration of how various actions are coordinated in the process of eating. Given these consideration, we concentrate our analysis on the videos related to *subject 1* and the videos capturing *eating roti* as our analysis target.

To get probabilistic events, we apply GCNs for activity recognition based on the skeleton-based dataset, because GCNs are widely used for skeleton-based action recognition and achieved remarkable performance (Shi et al., 2019). GCNs are adept at representing human body skeletons as spatiotemporal graphs, enabling the modelling of complex movements and interactions among different body parts over time (Yan et al., 2018). We choose to use the top two performing GCNs models as identified in the study by Raza et al. as our baselines. These two models are two-stream Adaptive Graph Convolutional Network (2s-AGCN) (Shi et al., 2019) and Channel-wise Topology Refinement Graph Convolutional Network (CTR-GCN) (Chen et al., 2021). Our objective is to utilise these two GCNs models to recognise eating activities as baselines and to extract probabilistic event traces from their outputs.

In each category, we choose 60% of the videos as our training set, 20% of the videos as the validation set and the rest of the videos as the testing set. The training set is used to train the GCN models and discover the process model. We use the validation set to tune the parameter of process discovery and the threshold ε in `ProbCost` algorithm.

We use the 3D keypoints skeleton of 8 upper-body joints from EatSense datasets as the input for the GCNs models. The GCN models are trained using the training set, and their performances are evaluated on the testing set. Subsequently, we convert the output predictions of the GCN models into a probabilistic matrix by applying a softmax function to the results of their output layer.

With the discovered process model and extracted probabilistic event traces, we apply `ProbCost` to achieve an alignment between these two components. The confidence threshold ε is tuned on the validation set with the aim of maximising the activity recognition accuracy. Then the tuned confidence threshold ε is applied to the testing set for the performance evaluation.

Following that, we conduct a comparative analysis of the activity recognition performance, contrasting the results obtained from the original GCNs models against those achieved after applying the `ProbCost` alignment. We use accuracy and macro F1-score to evaluate the performance (see Section 2.1.1). Accuracy provides a straightforward measure of the model's overall correct predictions, while the macro F1-score offers a balanced view of the precision and recall across all classes.

5.5.3 Process discovery

We employed the Heuristic miner implemented in the PM4Py library to discover the processes associated with *subject 1* and *eating roti*. The Heuristic miner requires setting a dependency threshold to eliminate less frequent paths between activities. We varied the dependency threshold from 0.8 to 0.95 with a 0.05 step for discovering different process models based on the training set. We compute the fitness and precision metrics of the discovered process models using the PM4Py process evaluation package. As we discussed in Section 2.3.2, the fitness score measures how well the discovered process model can replay the observed behaviours. Precision, essentially measures the model's ability to generate only those traces that are actually present in the event logs, thereby evaluating how closely the process model aligns with the real-world processes it aims to represent. We also measure the F-score of the precision and fitness by equation 5.1.

$$F\text{-}score = 2 \times \frac{Precision \times Fitness}{Precision + Fitness} \tag{5.1}$$

The result of the discovered process model of *subject 1* is shown in Table 5.3. It indicates that the highest F-score is achieved at a dependency threshold of 0.85.

Dependency threshold	Fitness	Precision	F-score
0.80	0.43	0.64	0.52
0.85	0.72	0.54	0.62
0.90	0.76	0.31	0.44
0.95	0.41	0.41	0.41

Table 5.3: The fitness, precision and F-score of *subject 1* process model in different dependency threshold.

The corresponding process model of *subject 1* is shown in Figure 5.2, where the numbers indicate the frequency of relevant activities and paths. It shows that the eating process of *subject 1* starts from *other* activity, which represents some preparatory

Dependency threshold	Fitness	Precision	F-score
0.80	0.61	0.30	0.40
0.85	0.64	0.34	0.44
0.90	0.84	0.42	0.56
0.95	0.81	0.42	0.55

Table 5.4: The fitness, precision and F-score of *eating roti* process model in different dependency threshold.

activities, such as adjusting seating and setting the table, involved in the eating but not specifically catalogued within our dataset. Then *subject 1* tends to perform the action of *pick food from utensil with tools in both hands* or *pick up tools with both hands*, demonstrating the use of cutlery or other eating instruments. This action is closely followed by the action of *move hand towards month*. Once the food is brought to the mouth, the primary activity is *eating it*, where the food is consumed. Occasionally, this step alternates with *drink*, indicating the consumption of a beverage as part of the meal. After each instance of eating or drinking, *subject 1* performs the action of *move hand away from mouth*. Then the subject performs *chewing*, where the food is processed for swallowing, *put the cup/glass back* if *drink* is performed before, or picking food again for eating either with tools or without tools depending on the consumed food.

We can see the most frequent path of the eating process is starting from *other*, then *pick food from utensil with tools in both hands*, *move hand towards month*, *eat it*, *move hand away from month*, *chewing*, etc. It shows that this process model effectively captures the structured sequences of actions in the eating process. Therefore, we choose the process model of *subject 1* discovered at the dependency threshold of 0.85 as the process model for alignment.

The evaluation of the discovered process model associated with *eating roti* is shown in Table 5.4. It shows that the highest F-score is achieved at a dependency threshold of 0.9. We visualise the discovered process at 0.9 dependency threshold in Figure 5.3. It depicts the process of eating roti, i.e., starting from *pick food from utensil with both hands*, *pick food from utensil with one hand*, *move hand away from month*, *chewing*, etc. Compared to the discovered process model of *subject 1* (Figure 5.2), there is no *pick food from utensil with tools in both hands*, because eating roti does not need tools. We adopt this process model of eating roti for subsequent alignment.

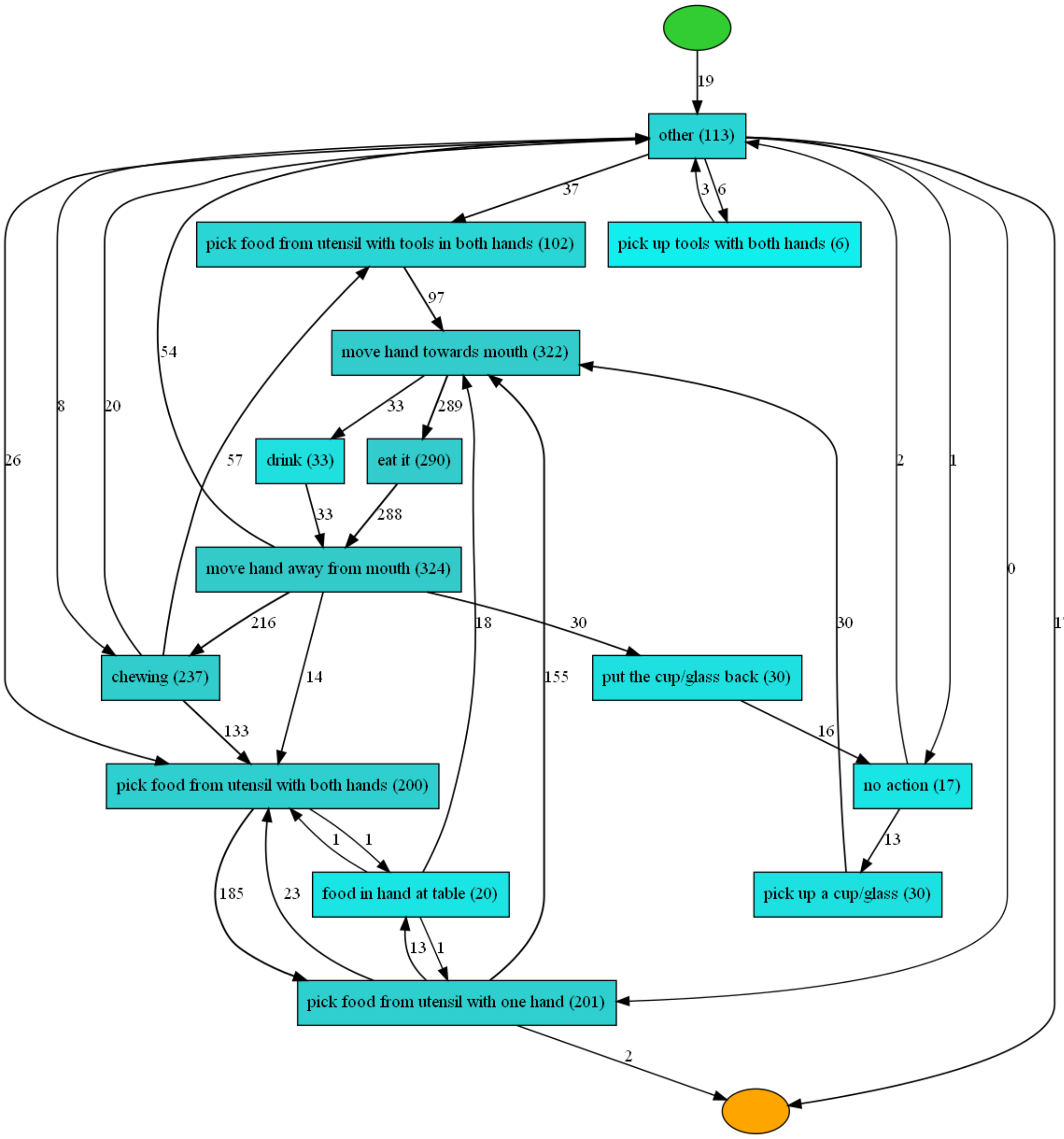


Figure 5.2: The discovered process model of *subject 1* by the Heuristic miner with dependency threshold 0.85. The number in the node label indicates the frequency of the corresponding activity. The number in the arc indicates the frequency of the corresponding path.

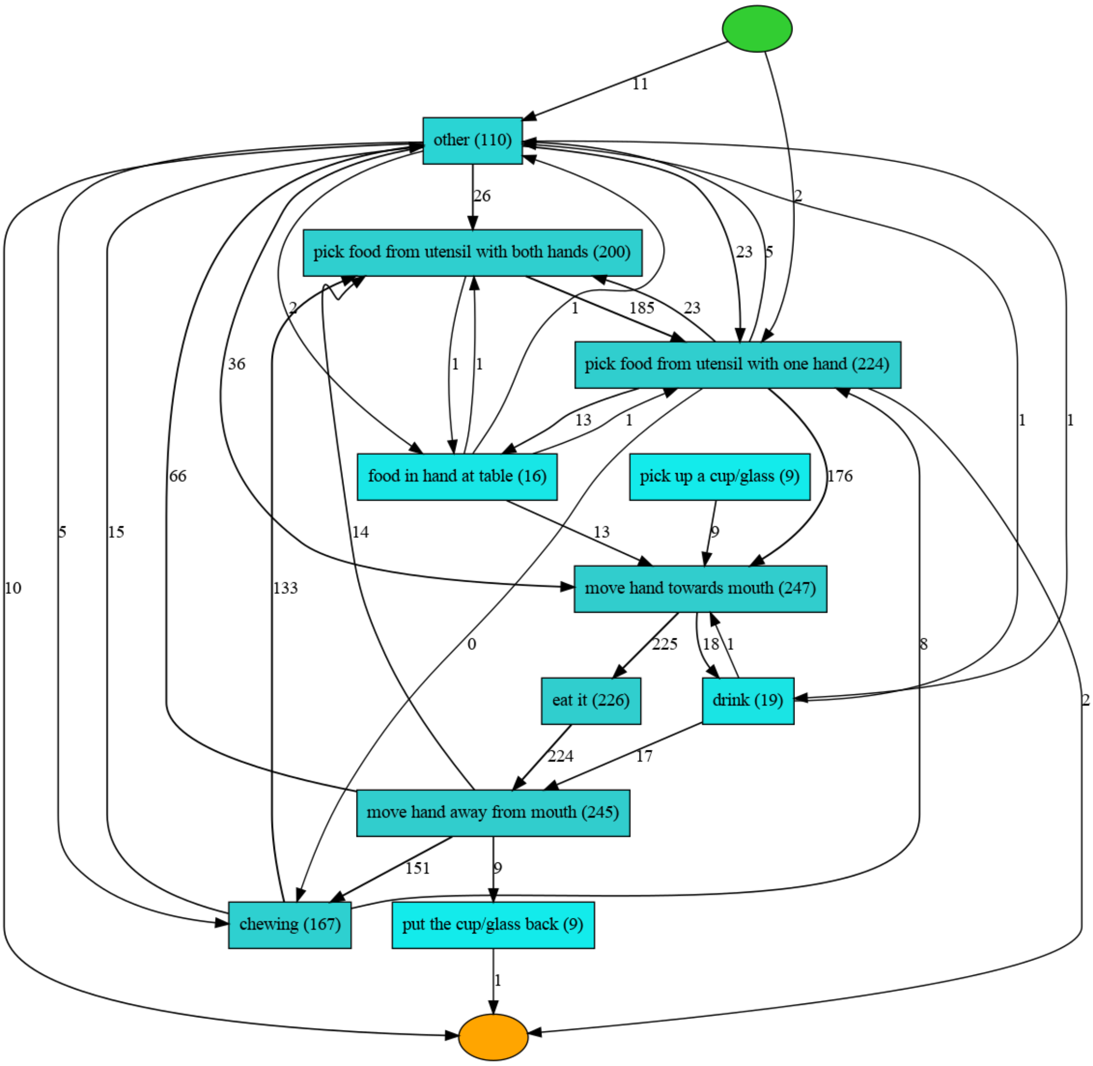


Figure 5.3: The discovered process model of *eating roti* by the heuristic miner with dependency threshold 0.9. The number in the node label indicates the frequency of the corresponding activity. The number in the arc indicates the frequency of the corresponding path.

5.5.4 Results and discussion

Firstly, we focus on investigating activity recognition results for the data related to *subject 1*. The confidence threshold ε in `ProbCost` is tuned in the validation set. We set the ε from 0.05 to 1 with a 0.05 step to test the activity recognition accuracy. The average accuracy in validation set with different ε is shown in Figure 5.4. The result shows it achieves the highest accuracy when the confidence threshold ε equals 0.1. Therefore, we set the parameter of the ProbCost as 0.1 for the evaluation on the testing set.

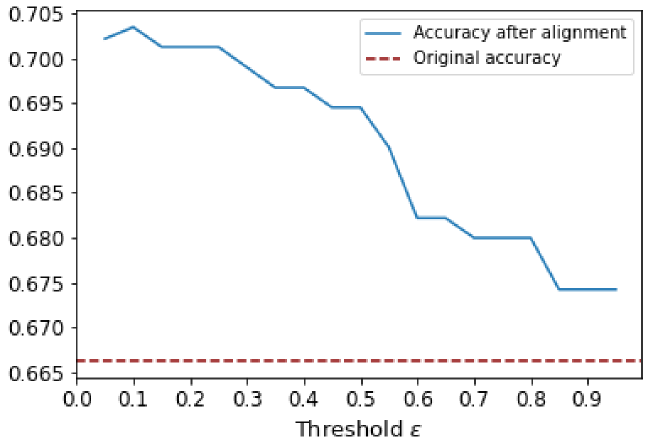


Figure 5.4: Accuracy of activity recognition across varying confidence thresholds ε for subject 1 in the validation set.

Table 5.5 displays the activity recognition performance, comparing the original results from the 2s-AGCN and CTR-GCN models with their performance post-alignment with the process model via ProbCost. The result shows an improvement in both metrics for the 2s-AGCN and CTR-GCN methods. This improvement indicates the benefit of incorporating structured process information into the activity recognition.

Methods	Accuracy	Macro F1	Accuracy-Alignment	Macro F1-Alignment
2s-AGCN	0.67	0.44	0.70	0.46
CTR-GCN	0.57	0.32	0.71	0.48

Table 5.5: Performance comparison for the data related to subject 1.

Next, we investigate the result in the category of eating roti. Similarly, we tune the confidence threshold ε of `ProbCost` in the validation set. The ε is set from 0.05

to 1 with a 0.05 step to determine the optimal setting that maximises activity recognition accuracy. The result is shown in Figure 5.5. It indicates the highest accuracy is achieved at the threshold of 0.1. Therefore, we set the confidence threshold ε of the ProbCost as 0.1 for the evaluation on the testing set.

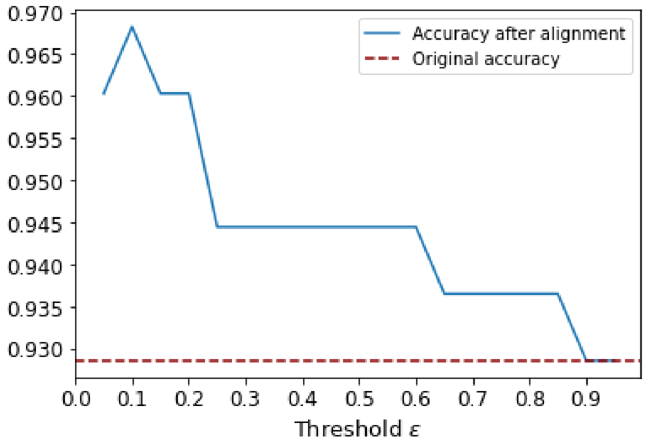


Figure 5.5: Accuracy of activity recognition across varying confidence thresholds ε for eating roti in the validation set.

We compare the performance after alignment via `ProbCost` with the original output performance from 2s-AGCN and CTR-GCN models. The results are shown in Table 5.6. After alignment with the process model, both models show improved performance. For example, initially, the 2s-AGCN model achieved an accuracy of 0.90 and a Macro F1 score of 0.75, indicating a strong performance in identifying activities based on the model's predictions. However, after applying the alignment process via `ProbCost`, there was a notable increase in both metrics. The improvement in the Macro F1 score suggests a better balance between precision and recall across different classes. This enhancement underscores the effectiveness of incorporating structured process information into the activity recognition. The alignment process essentially calibrates the model's predictions to better reflect the actual sequence of activities as defined by the process model.

Methods	Accuracy	Macro F1	Accuracy-Alignment	Macro F1-Alignment
2s-AGCN	0.90	0.75	0.99	0.83
CTR-GCN	0.81	0.73	0.99	0.87

Table 5.6: Performance comparison in the category of food.

Next, we delve into an example where the activity classes are accurately rectified through their alignment with process models.

We observe an instance where the 2s-AGCN model predicted the sequence of activities as i) *pick food from utensil with both hands*, ii) *pick food from utensil with one hand*, iii) *chewing*, and iv) *move hand towards mouth*. However, the actual sequence, or ground truth, is i) *pick food from utensil with both hands*, ii) *pick food from utensil with one hand*, iii) *food in hand at table*, and iv) *move hand towards mouth*. This discrepancy indicated that the third activity, *food in hand at table*, was incorrectly identified as *chewing* by the 2s-AGCN model. The decision is based on the probability distribution (as shown in Table 5.7), which shows *chewing* having the highest probability, leading the 2s-AGCN model to classify the third activity erroneously.

Upon incorporating process information through `ProbCost`, we are able to rectify this misrecognition. The process model of eating roti (as shown in Figure 5.3) suggests that if the third activity were indeed *chewing*, the subsequent action of *move hand towards mouth* would be illogical. Given the high confidence (0.8 probability of GCNs predictions) in recognising the fourth activity as *move hand towards mouth*, it was clear that if the third activity is *food in hand at table*, the sequence adheres more coherently to the discovered process model. Specifically, the activity of *food in hand at table* indicates that the individual is holding food in their hand while at the table, prepared to bring the food to their mouth for eating. Thus, despite *chewing* initially appearing as the most probable activity for the third step, we identified *food in hand at table* as the correct activity, aligning with the process model. This adjustment showcases the concrete example of how the `ProbCost` integrates process information with data-driven models to enhance activity recognition accuracy.

5.6 Conclusion

In this chapter, we present a novel framework aimed at enhancing human activity recognition by integrating process information within the semi-structured domains into the analysis of machine learning model outputs. We apply the proposed method of `ProbCost` to achieve an alignment between probabilistic event traces generated by ML models and the process model. The alignment essentially calibrates the model's predictions by incorporating the actual sequence and nature of activities as defined by the process model. Through experimental comparison based on the Eatsense dataset, with the original performances of Graph Convolutional Networks, including 2s-AGCN and CTR-GCN, our approach demonstrates significant improvement in activity recognition

Activities	Probabilities
chewing	0.352730
move hand towards mouth	0.325867
pick food from utensil with tools in both hands	0.082761
pick food from utensil with one hand	0.082365
food in hand at table	0.034219
pick food from utensil with tool in one hand	0.029754
pick up tools with both hands	0.026519
move hand away from mouth	0.024972
no action	0.009012
other	0.008487
pick up a cup/glass	0.007892
put the cup/glass back	0.006343
put one tool back	0.005486
pick food from utensil with both hands	0.002836
eat it	0
drink	0

Table 5.7: The probability distribution of an activity, which is corrected after alignment.

accuracy and Macro F1 scores.

The results underscore the effectiveness of incorporating process information for activity recognition within semi-structured domains. We consider the inherent flexibility of performing activities when discovering process information in semi-structured domains. Moreover, this work highlights the importance of combining machine learning models with domain-specific process knowledge. Future research can build on these findings, exploring the application of process model alignment framework across a broader range of activities and domains.

The next chapter shifts focus to recognising patterns and detecting deviations in unstructured domains, such as Activities of Daily Living (ADLs). The inherent variability and absence of predefined models representing usual behaviour in these domains pose significant challenges for detecting abnormal behaviour because it is hard to establish a baseline for what is considered “normal”. In the next chapter, we delve into the detailed problems associated with anomaly detection in unstructured domains. We explore the use of predictive models for identifying temporal patterns of behaviour based on historical data and detecting both short and long-term deviations. We also investigate the application of probabilistic model checking for detecting changes over time in complex behaviours.

Chapter 6

Temporal Pattern Recognition and Anomaly Detection in Unstructured Domains

6.1 Introduction

Unstructured domains are characterised by few constraints or pre-defined models to restrict behaviours and represent normal behaviours. These domains, rich in variability and complexity, significantly hinder the identification of regular patterns and anomalies, and pose unique challenges for anomaly detection. This chapter focuses on how we can recognise patterns, establish constraints to define expected behaviours and identify anomalies within unstructured domains, with a special emphasis on the context of Activities of Daily Living (ADLs).

Deviating from expected behaviours in ADLs can be critical indicators of emerging health issues, changes in cognitive or physical abilities, or alterations in lifestyle that could have significant implications for an individual's well-being (Albanese et al., 2020). For example, a sudden decrease in activity level might signal the onset of a physical condition, while changes in routine might indicate cognitive decline, such as that associated with dementia or Alzheimer's disease. Detecting these behavioural changes is crucial for timely interventions, personalised care, and maintaining or improving quality of life (Sepesy Maučec and Donaj, 2021). Moreover, proactive monitoring and analysis of short and long term anomalies from a regular routine in ADLs can provide vital insights on a person's health and a continuous evaluation of their physical and cognitive ability (Sepesy Maučec and Donaj, 2021). Develop a dashboard to visualise detected anomalies and deterioration in behaviour can provide a clear and actionable overview of daily patterns that can facilitate timely interventions. The dashboard can also enhance the observability and accessibility of data for caregivers, medical professionals and even the individuals themselves.

In order to identify abnormal behaviours of daily activities, one possible approach is to impose constraints that define what constitutes expected behaviours. This task is particularly challenging given the inherent unstructured characteristics of daily activities and the variability in individual routine (Brady et al., 2023), as we discussed in Section 1.3. For example, sleeping late might be a part of an individual's regular routine on a Friday night and not necessarily an anomaly in their behaviour. This underscores the importance of understanding the temporal patterns of daily activities for each individual.

Moreover, to effectively identify behavioural changes, it is essential to analyse not just behaviours involving individual activities, such as sleeping patterns, but also complex behaviour patterns that involve multiple activities. This is because people's daily behaviours are intricate, often encompassing a series of activities that are aware of the process in which these activities occur. For instance, a morning routine might progress from waking up to personal hygiene, and then dressing. These behaviours are characterised by having dependencies between multiple activities, such as the necessity of cleaning after cooking or taking medication with meals. Modelling these complex behaviour patterns can reveal changes that single activities alone might not indicate. Therefore, it is crucial for identifying meaningful behavioural changes but introduces additional complexity.

Modern and emerging smart home and wearable sensor technologies allow us to collect continuous data on daily living in an unobtrusive and affordable manner. This involves time series data with status information and timestamped event data when a status change occurs (Cook et al., 2013a). Activity Recognition and Machine Learning (ML) techniques can then produce fine-grained daily activity data. Further AI modelling can allow us to develop rich temporal patterns of daily routines, including sleep duration, number of meals, and levels of active movement. Visualisation techniques can help translate and interpret data into an intuitive format that can be easily understood.

Our work aims to utilise these techniques to detect anomalies and behavioural changes based on ADLs data collected by sensors. We focus on extracting temporal patterns of daily activities and detecting anomalies on individual days and deterioration in the longer term. These anomalies include, for example, staying in the toilet too long, too frequent toilet visits during the night, activity delays, and sleep disruption and deterioration. We develop a dashboard to visualise the detected deviations and deterioration, providing an intuitive overview of individuals' daily routine, trends and anomaly behaviour. Moreover, we aim to detect changes of complex behaviour over time, such as reduced morning efficiency, increased probability of forgetting to take medication before meals, etc.

6.1.1 Sources of ADL data

Different sources of ADL data come with inherent limitations, particularly in terms of labelling activities in the data collection process, which pose challenges for accurately measuring temporal patterns of daily activities.

The methods for collecting data on ADLs have evolved significantly. Traditionally, studies in investigating the relationship between ADLs and health outcomes relied on questionnaires and self-reporting methods (Kanti Majumdar, 2014; Cook and Schmitter-Edgecombe, 2021). For example, participants are required to recall and report the activities they performed and their duration at the end of each day. The result of this may be affected by retrospective memory limitations (Palmer, 2018).

In contrast, recent advancements in sensor technology have provided a way of passive and continuous monitoring of ADLs, providing a more objective and concrete analysis of behavioural patterns and routines. Sensor-based systems capture real-time data without requiring active participation from users. Sensors are typically installed in an individual's living environment or worn on the body to collect data continuously. Despite this advancement, there is still a need for participants to occasionally record their activities manually. This helps in providing labels for the sensor data, which are crucial for training and validating machine learning models for recognising human activities automatically (Bouchabou et al., 2021), as we explored in Chapter 5.

However, while sensor-based systems offer numerous advantages, they also come with inherent limitations tied to the modalities of the sensors used. Most ambient sensor setups primarily capture movement and lighting data, limiting the scope of activities that can be monitored. For example, accurately determining the exact moment an individual falls asleep is challenging with only ambient sensors. Unlike magnetic resonance imaging (MRI) or specialised brain activity recording devices, ambient sensors in ADL monitoring setups lack the capability to directly measure physiological measurements, such as heart rate or brain activity, which are crucial for sleep studies (de Zambotti et al., 2019). This means that certain aspects of ADLs might not be captured accurately.

Integrating additional modalities, such as wearable sensors, can enhance activity detection. Wearable sensors can track a range of physiological and environmental variables, such as heart rates, and geographic location, providing a broader view of an individual's activity and physiological states (Cook and Schmitter-Edgecombe, 2021). For instance, a smart watch equipped with an accelerometer and a heart rate sensor can more precisely detect periods of sleep (Roberts et al., 2020), but their accuracy is still uncertain (Lee et al., 2019).

Moreover, the granularity of data collected from both ambient and wearable sensors may not always capture the full complexity of human behaviour. Activities that involve little or no physical movement, such as reading or watching television, may be harder to distinguish based solely on sensor data. Although cameras can help to identify these activities, this raises significant privacy concerns (Yang et al., 2024). Despite these enhancements, uncertainties in automatically recognising activities remain an ongoing challenge, as we discussed in Chapter 5.

Overall, these limitations highlight the complexity of collecting robust data on daily activities, specifically with regard to obtaining accurately labelled activity data. These challenges are particularly significant when it comes to establishing accurate temporal activity patterns essential for effective monitoring. Beyond these issues, as discussed in Chapter 3 and 5, there are additional challenges such as noise in collected data caused by sensor inaccuracies, malfunctions, or inherent uncertainty of machine learning algorithms. In this chapter, we do not directly address these issues related to data uncertainty and noise, focusing instead on analysing temporal patterns of daily activities and detecting anomalies. This discussion highlights areas for potential future research, where addressing the challenge of noisy and uncertain data could further enhance the accuracy and reliability of our approach for detecting anomalies and behavioural changes.

6.1.2 Challenges of anomaly detection in ADLs

The absence of constraints or pre-defined models to represent normal behaviours, makes it particularly challenging to identify anomalies in ADLs, because there is no baseline for comparison. Additionally, there is often a lack of labelled data to distinguish between what constitutes abnormal behaviour and normal behaviour in ADLs, making it difficult to train and validate anomaly detection algorithms. This scarcity of labelled data requires anomaly detection to be performed in an unsupervised manner. We breakdown the challenges associated with identifying anomalies in ADLs as follows:

- **Lack of ground truth**: A key challenge in the analysis of real-life datasets is the absence of ground truth of what are considered deviations. This limits our ability to evaluate the precision of our deviation detection and to validate the detected behaviour changes against an objective standard or expected behaviour. Without ground truth, assessing the precision of detected deviations or anomalies is complicated. It is difficult to determine whether identified deviations truly represent abnormal behaviour or if they are merely variations within the range of normal human activity. Considering an example scenario of reduced sleep duration following a Friday night party, if this pattern occurs consistently every week, it could be interpreted as part of the individual's regular weekly routine rather than an anomaly.

- **Data availability**: The collected data usually represents a snapshot of behaviour within a specific time frame. This temporal limitation presents a challenge in understanding and modelling daily behaviour, as it may not fully encapsulate the evolution of routines over long periods. As human behaviours are inherently dynamic, and subject to change due to various factors, such as seasonal variations, life events and shifts in personal situations. Therefore, data confined to a narrow time frame might offer a snapshot that, while valuable, lacks the depth and context needed to derive comprehensive insights. For instance, observing reduced sleep duration on a Friday night without considering the broader context may miss the pattern that this reduction consistently occurs every week due to social activities or personal relaxation routines.

- **Varying sensitivity of behaviours**: Different behaviours have different sensitivity to deviations from their normal patterns. For instance, variations in sleep duration typically have a more significant impact on an individual's health compared to fluctuations in reading time. This differentiation underscores the need for an adaptable deviation detection approach that accommodates behaviour differences without overwhelming individuals with false positive alerts.

- **Variability of individual routines**: People can perform their daily routines in various ways, depending on factors such as demographics, health conditions, personal preferences, and lifestyle. For example, sleeping late might be part of one individual's regular routine, while for another, it might signify a deviation from their normal behaviour. Since normal behaviour varies greatly among individuals, it requires an anomaly detection approach to consider personalised behavioural patterns.

- **Modelling complexity**: People's daily behaviours are intricate, often encompassing multiple interdependent activities, such as certain medications must be taken before a meal. Therefore, identifying anomalies should not only consider behaviour involving single activities but also more complex behaviour involving multiple activities. For instance, the order of medication intake and meal consumption can impact the effectiveness of certain medications and overall health. Many medications need to be taken with food to enhance absorption or mitigate side effects, while others must be taken on an empty stomach. If an individual usually takes medication before meals but suddenly begins taking it after eating, or vice versa, this could signal a deviation from their established routine. Such a change might not only affect the efficacy of their medication but could also indicate potential issues like changes in appetite, forgetfulness, or cognitive decline. Effectively modelling complex behaviour and assessing their changes over time also poses a significant challenge.

- **Sensitivity to changes**: The model's sensitivity to minor behavioural changes could either be an advantage or a limitation. Models that are highly sensitive can detect minor changes in behaviour. However, this heightened sensitivity also risks identifying normal variability in human behaviour as deviations, leading to false positives. Conversely, models with low sensitivity might overlook subtle yet meaningful changes, resulting in false negatives. For example, a model might interpret a single night of poor sleep as a concerning change, disregarding the possibility that it could be a one-time event unrelated to broader behavioural trends.

6.1.3 Contributions

Given the challenges and limitations of ADL data discussed above, extracting daily patterns and detecting anomalies is a highly complex problem. In this work, we specifically focus on addressing several key challenges: the varying sensitivity of behaviours, the variability of individual routines, and the complexity of modelling behaviours. By concentrating on these aspects, we have two objectives. One is to develop an anomaly detection method that can effectively adapt to the diverse, varying nature of daily activities and detect both short-term deviation and long-term drifts of individual activities. Another objective is to develop a method that can detect changes of complex behaviour over time. We aim to ensure that our methods are both sensitive and specific to the unique patterns of individuals, and anomalies detected are meaningful and can provide actionable insights for caregivers and individuals.

To achieve the **first** objective above, we present a predictive method for building personalised temporal profiles of daily behaviour based on historical data to represent typical patterns and an anomaly detection method based on these established profiles.

To build a meaningful temporal profile, we focus on extracting behaviour markers that describe an activity at a higher level, such as sleep duration and time, etc. which are more significant for understanding a person's routine and may also be more relevant to health outcomes. For instance, focusing on sleep duration provides a direct indicator of sleep quality and overall health, while potentially ignoring less critical details such as the number of times a light was switched on and off overnight. These behaviour markers like total sleep duration, can help us effectively analyse daily patterns and abstract away much of the noise that does not impact the essence of outcomes.

In the anomaly detection method, we introduce a customisable threshold for detecting anomalies of different activities, which accounts for the varying sensitivity of behaviours. This includes a deviation score that evaluates anomalies for a given day, as well as a long-term pattern recognition approach that can detect long-term drift from the normal routine. We also develop an interactive dashboard, which can visualise personal temporal profiles of daily behaviours including daily routines, trends of each behaviour marker and potential deviations. The dashboard provides intuitive behavioural statistics of the individual that may be useful both for self-monitoring and management and providing useful information to care providers. The efficacy of this method is validated through an empirical study utilising CASAS datasets (Cook et al., 2013a), including data over a 2 months period.

To address the **second** objective of this chapter, i.e., to effectively model complex behaviours and detect their changes over time, we introduce an approach based on formal methods, specifically probabilistic model checking (Kwiatkowska et al., 2018). Probabilistic model checking formulates a system as a probabilistic model, such as Discrete-Time Markov Chains (DTMCs) (Lawler, 2018), known as a stochastic process (Parzen, 1999). It can quantitatively verify whether the probabilistic model satisfies certain properties modelled by logical formulas. In the context of daily routines, sequences of activities performed each day can be transformed as a stochastic process, represented as a DTMC. Each state in the Markov chain represents a specific activity or a stage in the daily routine, and the transitions between states are governed by probabilities that reflect the observed behaviours or patterns. Probabilistic Computation Tree Logic (PCTL) (Hansson and Jonsson, 1994) allows for the precise specification of properties involving probabilities, enabling the formal modelling of behaviours that are expected to occur. By applying probabilistic model checking, we can verify properties such as the likelihood of transitioning from one activity to another, the probability of engaging in certain behaviours over a period, or the probability of certain sequences occurring under given conditions.

Based on probabilistic model checking, we model daily routines as user activity models based on DTMCs, describing the stochastic processes of activities performed every day. To model complex behavioural patterns, we define a set of behaviour properties using PCTL formulas. We use probabilistic model checking techniques to assess the probability of these properties against the user activity models in different periods. We further develop a statistical method for detecting abnormal probabilities of the individual's behaviour that might require attention. This focuses on spotting significant behavioural changes that could signify issues that might impact the individual's well-being. The robustness and effectiveness of our method are validated through a sensitivity analysis based on a synthetic dataset, which evaluates the responsiveness of our method to varying extents of behaviour changes. To showcase the practicality and applicability of our proposed method, we undertake a series of empirical studies utilising both synthetic datasets and real-life CASAS datasets.

The rest of this chapter is organised as follows. Section 6.2 reviews existing literature related to detecting anomalies in the domain of ADLs. The datasets used in our analysis and the methods applied for data preprocessing are detailed in Section 6.3 and Section 6.4, respectively. We then describe our approach to identifying temporal behaviour patterns and detecting anomalies, and its experimental study in Section 6.5.

Following that, the method of detecting behaviour changes over time and its experimental study are introduced in Section 6.6. Finally, the chapter concludes with a discussion and summary of our findings in Section 6.7.

6.2 Related work

Research in the field of ADLs has recently been receiving increasing attention, especially in the context of supporting independent living for older people and providing effective care. Yahaya et al. provide a comprehensive list of recent efforts on anomaly detection in ADLs (Yahaya et al., 2019), which are categorised in the following four categories: *classification-based*, *threshold-based*, *clinical score-based*, and *formal methods-based* approaches.

6.2.1 Classification-based Deviation Detection

Classification methods treat deviation detection as a binary classification problem. They require ADL data labelled as *normal* or *abnormal*, where the latter reflects a pre-specified pattern of behaviour, such as the behavioural difficulties of people with dementia (Arifoglu and Bouchachia, 2019). Due to the scarcity of abnormal data in real datasets, it is common to train and generate synthetic data for the abnormal class. More recent approaches consider any deviations from a normal routine that is learned from historical data as an anomaly (Yahaya et al., 2019; Pazhoumand-Dar et al., 2020; Yahaya et al., 2021c).

Classification-based approaches are limited by the availability of prior activity annotation of abnormalities. Even though generating synthetic data of abnormalities is a solution, it is infeasible to generate every manifestation of abnormalities. This limitation stems from the inherent diversity and variability of behaviours across different individuals.

In this work, we explore the field of anomaly detection without relying on prior annotations to distinguish between normal and abnormal behaviour. In other words, our approach to anomaly detection operates in an unsupervised manner.

6.2.2 Threshold-based deviation detection

Some deviation detection approaches rely on detecting whether the sensor information exceeds a fixed threshold of values. Collected data is used as training data representing the normal behaviours and subsequent activity data are used as testing data for the learned model. Data that have significant variations past certain thresholds are defined as outliers (Arifoglu and Bouchachia, 2019).

Another common approach to detecting deviations from regular routines of the daily activities of a monitored person involves the use of wearable sensors and detecting whether the sensor information exceeds a fixed threshold of values. For example, Pierleoni et al. propose a fall detection method based on the fusion data collected from a triaxial accelerometer, gyroscope, and magnetometer on wearable devices (Pierleoni et al., 2015). If the body orientation falls below a pre-defined threshold for a certain period of time, the system will issue an alarm.

However, approaches that rely on wearable sensors are not always applicable in practice. For instance, some people may not feel comfortable constantly wearing a device or may forget to put it on or charge the battery. Moreover, missing data and false positives can make these approaches less reliable (Pazhoumand-Dar et al., 2020). For example, lying down on a bed suddenly may be mistaken for the movement pattern of an accidental fall.

Some approaches have overcome these limitations by using ambient sensors in a smart home (Cook et al., 2013a). The data are collected from environmental sensors when the subjects have interactions with their environment. For example, Pazhoumand-Dar et al. use Kinect sensors composed with power consumption data to monitor daily behaviours (Pazhoumand-Dar et al., 2020). Their training data is aimed to model the regularity and frequency of important activities and does not need to be labelled in advance.

Howedi et al. use ADL data collected from ambient sensors to detect deviations based on entropy measures (Howedi et al., 2020). Activities with entropy values exceeding a certain range are detected as abnormalities. Similarly, Yahaya et al. propose an ensemble of abnormal detection approach by detecting if the test data differs significantly from the training data based on a threshold for a defined Normality Score (Yahaya et al., 2019).

In our work, we also adopt threshold-based approaches using data without prior annotation of abnormalities. We focus on identifying temporal patterns of daily behaviours, e.g., wake up time, sleep duration, etc., based on historical data and detect deviations based on identified patterns.

6.2.3 Clinical score-based deviation detection

Clinical score-based approaches involve an assessment of human by clinical experts through various factors of their daily activities, such as cognitive health, functional mobility, etc (Dawadi et al., 2016; Alberdi Aramendi et al., 2018; Cook and Schmitter-Edgecombe, 2021).

For example, Dawadi et al. propose a Clinical Assessment using Activity Behaviour, called CAAB, to model a resident's daily behaviours in a smart home and predict the clinical score in the future (Dawadi et al., 2016). The score is provided by a clinician analysing cognitive status. They establish a statistical correlation between the clinician-provided cognitive scores and CAAB-predicted scores. The results illustrate ambient sensors are able to predict clinical scores.

Similarly, Alberdi Aramendi et al. enrich the above method by introducing more assessment fields than CAAB (Alberdi Aramendi et al., 2018). They evaluate functional health scores through the IADL-C questionnaire, which classifies different functional abilities into four factors: (1) money and self-management, (2) home daily living, (3) travel and event memory and (4) social skills. Using the data collected from ambient sensors in smart homes and corresponding clinical scores, they create a regression model to predict functional health scores in the future. Besides this, a classifier is trained to predict changes in positive and negative fluctuations in recurrent health scores. The results show it is hard to predict negative and positive fluctuations using home behaviour data, but changes in social skills are predictable.

In the same line, Cook et al. propose a method to investigate the predictability of clinical scores from a person's behaviours (Cook and Schmitter-Edgecombe, 2021). They propose a joint inference technique to improve the predictive performance of high-dimensional data, including, participant demographics, self-report, and external observation-based health scores. The result shows this approach can get an improvement in predictive performance when joint inference is employed.

The assessments are usually conducted at regular time intervals, for example, every 6 months, and a total score representing the health status of the participant is calculated. After data collection, a computation model is trained to map the clinical score to the data collected by ambient sensors and predict future scores based on that. The assessments are carried out by self-reporting, using an appropriate questionnaire. However, the result from this type of assessment may be affected by the respondent's subjective view and state of mind when filling in the questionnaire (Yahaya et al., 2021b). As a consequence, the score may not accurately reflect the actual health condition of the respondent.

In our work, we employ the idea of modelling deviation as a score, which assesses the degree to which behaviour deviates from the norm. Similarly, we propose the use of a deviation score to evaluate anomalies for a given day, which quantifies the extent of anomalies.

6.2.4 Formal methods-based deviation detection

Formal methods have been widely employed in ensuring the safety design of the software and hardware systems. Such methods including automata, Markov Chains, etc., can also be employed in modelling the expected behaviour patterns.

For example, Saives et al. build a finite automaton to model the recommended behaviours according to the medical staff (Saives et al., 2015). They compare the real behaviour performed by inhabitants with the modelled recommended behaviours to detect behaviour deviations.

Moreover, the nondeterministic behaviour of inhabitants living in smart homes requires the introduction of quantitative properties when analysing behaviour and verifying the extent to which the behaviour satisfies a specified property, such as, what is the probability of completing an activity with a set sequence of behaviours. Probabilistic model checking is a well-established technique for quantitatively and qualitatively analysing state-based models, such as Markov Chains, Markov Decision Processes, etc (Kwiatkowska et al., 2018).

For example, L'Yvonnet et al. translated dynamic human activities into formal models based on Discrete Time Markov Chains (DTMCs) to simulate the real patients' behaviours (L'Yvonnet et al., 2021). A probabilistic model checker, named PRISM (Kwiatkowska et al., 2011), is used to diagnose particular diseases by modelling and checking the properties of medical interest. It is intended that this modelling approach enables a new way of interpreting medical patient performance.

Gao et al. proposed a probabilistic model checking-based approach to predict the next activity, aimed to assist patients with Alzheimer's diseases (Gao et al., 2023). Moreover, they also demonstrated the process of detecting abnormal temporal behaviours by using probabilistic model checking to check the pre-defined expected behaviour properties.

In the same line, Wang et al. proposed a method of human behaviour analysis and sensor fault detection (Wang et al., 2023). They build DTMCs models based on the transitions of sensors generated from a smart home dataset. The DTMCs models were verified to check whether the models satisfy the required properties. They also build Hidden Markov Models (HMMs) to analysis the probability of occurrence of arbitrary sequences of complex behaviours.

Compared to the work discussed above, we employ DTMCs to model the daily routines of an individual, with a specific focus on daily activities, rather than on sensors in (Wang et al., 2023). We use probabilistic model checking techniques to detect how behaviour shifts over time, not only identifying anomalies.

6.2.5 Summary

Different from the classification-based anomaly detection approaches discussed above, which rely on pre-labelled data with abnormal and normal annotations, our work operates in an unsupervised manner and does not rely on such annotations. We focus on creating a personalised temporal profile for each individual. We employ a threshold-based method for anomaly detection, where we compare observed behaviour against the established temporal profiles. Moreover, we adopt a formal method-based approach to model complex behaviour patterns and detect behaviour changes over time.

6.3 Data description

As we discussed in Section 6.1.1, ADL data is difficult to acquire and different sensing modalities have inherent limitations. Our work utilises CASAS datasets, which is collected by ambient sensors from smart homes (Cook et al., 2013a). Our work specifically focuses on the *hh* datasets within the CASAS collections. These datasets involve data collected from senior residents living independently and are the most recently updated datasets (2021). We choose these datasets because they are real-world ones and encompass a wide range of daily activities, providing a comprehensive view of typical behaviours within a home environment. By employing data collected from actual living environments, we ensure that our methods are tested against the complexities and variabilities inherent in real-life scenarios.

While the CASAS data provide a foundational platform for demonstrating our methods, the methods themselves are designed to be generalisable to other datasets involving data with labelled daily activities and their temporal information.

The datasets contain continuous data from unobtrusive ambient sensors including motion sensors, door sensors, light switches and light sensors, deployed in single-family residences. Our work is based on data from 11 individuals, which are denoted by hh101-hh111. Each set of individual data is labelled with activities over a period of approximately two months. Each sensor event is manually labelled with a corresponding activity, including sleeping, cooking, eating, napping, going to the toilet, and working, in a total of 33 different activities. Events that do not correspond to any specific daily activities are categorised as "other activity". A sample of data is shown in Table 6.1a.

The full list of the activities is (in alphabetical order): *Bathe, Bed_Toilet_Transition, Cook, Cook_Breakfast, Cook_Dinner, Cook_Lunch, Dress, Drink, Eat, Eat_Breakfast, Eat_Dinner, Eat_Lunch, Enter_Home, Entertain_Guests, Groom, Leave_Home, Morning_Meds, Other_Activity, Personal_Hygiene, Phone, Read, Relax, Sleep_Out_Of_Bed, Sleep, Step_Out, Take_Medicine, Toilet, Wash_Breakfast_Dishes, Wash_Dinner_Dishes, Wash_Dishes, Wash_Lunch_Dishes, Watch_TV, Work, Work_At_Table*

The raw sensor data is recorded in a time-series format with the following fields:

- **Timestamp**: The date and time of the event.
- **Sensor**: The name of the sensor. For example, `M007` and `D005` in Table 6.1a.
- **Room**: The room-level sensor location, such as `kitchen`, `dinning room`, etc.
- **Location**: The fine-grained location of the sensor, such as `refrigerator`, etc.
- **Message**: The value generated by the sensor, such as `on`, `off`, etc.
- **Sensor type**: The type of sensor generating the event, such as motion sensors, door sensors, and light sensors. In the datasets, these are specified as `Control4-Motion`, `Control4-LightSensor` (see Table 6.1a), etc. This labelling provides context to the generated message.
- **Activity**: A manual label of the corresponding activity of this event, such as sleeping, eating, etc.

Due to inherent uncertainty in the environment and human behaviours, the dataset is noisy. We specifically identified 4 categories of potential noise below:

	Timestamp	Sensor	Room	Location	Message	Sensor type	Activity
1	2011-06-15 09:58:27	M007	Kitchen	Kitchen	ON	Control4-Motion	Cook_Breakfast
2	2011-06-15 09:58:42	M007	Kitchen	Kitchen	OFF	Control4-Motion	Cook_Breakfast
3	2011-06-15 09:58:45	D005	Kitchen	Refrigerator	Close	Control4-Door	Cook_Breakfast
4	2011-06-15 09:58:46	M005	DiningRoom	DiningRoom	ON	Control4-Motion	Eat_Breakfast
5	2011-06-15 09:58:47	M008	Kitchen	Kitchen	ON	Control4-Motion	Cook_Breakfast
6	2011-06-15 09:59:05	LS001	Kitchen	Kitchen	4	Control4-LightSensor	Cook_Breakfast
7	2011-06-15 09:59:06	LS007	Kitchen	Kitchen	6	Control4-LightSensor	Cook_Breakfast
8	2011-06-15 09:59:10	M005	DiningRoom	DiningRoom	ON	Control4-Motion	Eat_Breakfast
9	2011-06-15 10:00:29	LS005	DiningRoom	DiningRoom	13	Control4-LightSensor	Eat_Breakfast
10	2011-06-15 10:00:46	LS015	DiningRoom	DiningRoom	14	Control4-LightSensor	Eat_Breakfast
11	2011-06-15 10:00:47	M008	Kitchen	Kitchen	OFF	Control4-Motion	Cook_breakfast

(a) An excerpt of raw ADL sensor data labelled with activities.

Activity	Start time	End time	Duration	Daycase	Interval
Cook_Breakfast	2011-06-15 09:58:27	2011-06-15 09:59:06	00:00:39	2011-06-15	00:00:04
Eat_Breakfast	2011-06-15 09:59:10	2011-06-15 10:00:46	00:01:36	2011-06-15	00:00:01

(b) ADL data resulting after pre-processing. The duration of each activity does not mean the total duration of the corresponding activity, because the activity may resume after interrupted by other activity. For example, *cooking breakfast* will resume after the *eating breakfast*, as they are interleaved.

Table 6.1: Sample of ADL sensor data.

- **Accuracy of labelling**: Since the particular dataset we are examining was manually labelled post-hoc, there is no measure of the accuracy of the labelling. In fact, newer versions of the dataset seem to have some of the labels updated. More generally, even with an automated activity recognition algorithm, such as the one by Cook et al. (Cook et al., 2013b), there remains an inherent level of uncertainty associated with machine learning algorithms that can impact the results, as we discussed in Chapter 5.

- **Lack of end event**: The observed activity events indicate when an activity is detected. However, there is no clear indication of the exact end time of the activity or the transition time to the next one. This may cause some inaccuracy in our calculated activity intervals (see Section 6.4).

- **Distinguishing activities**: In some cases, the same small set of 1-2 sensors may be used to detect multiple types of activities with similar action patterns or taking place in the same location, such as washing the dishes and preparing a meal. This can lead to data that inaccurately represents the actual activity performed, leading to errors in the activity recognition process.

These noises are primarily due to the accuracy of labelling, which is beyond the scope of this chapter. Whilst we do not explicitly address these noise factors in our modelling, they must be taken into into consideration when making potential decisions or interventions based on our produced insights.

6.4 Data preprocessing

Given that our work focuses on analysing daily activities data, this section details how we transform the sensor event data into a format that intuitively represents discrete daily activity intervals. We extract the essential temporal features of activities, such as starting time and duration.

The first processing step involves the removal of noise, as described in the section 6.3, to the extent possible, as well as the removal of unknown activities labelled as *"other activity"*.

In the dataset, we have observed some patterns of activities that pose a challenge when converting the data from timestamped events to activity intervals, as follows:

- **False occurrences**: During the execution of a particular activity, an orphan event corresponding to a different activity may occur, in an unrealistic way. An example is shown in rows 1-7 in Table 6.1a, where a single event representing the *eat breakfast* activity (row 4) occurs while people are *cooking breakfast*, but subsequent events indicate that the person continues to cook breakfast. A single orphan event prevents us from measuring the (assumed to be short) duration of the corresponding activity. We treat such events as false occurrences.

- **Interleaved activities**: A sequence of events indicates that two different activities occur simultaneously or overlap during the same period. As shown in Table 6.1a, a person is *cooking breakfast* when a continuous sequence of *eating breakfast* events occurs. Subsequent events show that the person continues to cook breakfast. This means that the person is cooking while eating and the two activities are interleaved during this time.

In cases of the false occurrences of events, such as row 4 of Table 6.1a, where only a single event is recorded, it becomes impossible to accurately measure the duration of the corresponding activity. Therefore, we have opted to remove these singular event instances from the dataset.

For the interleaved activities, we operate under the assumption that the next activity begins only after the preceding activity has been completed. For example, in rows 8-10 of Table 6.1a, we observe a continuous sequence of *eat breakfast* events. We treat them as the initial activity of *cook breakfast* ended in row 7, then *eat breakfast* started in row 8 and a new *cook breakfast* activity started in row 11.

Based on the above, we detect consecutive events of the same activity and choose the timestamp of the first event as the start of the activity interval and the timestamp of the last event as the end of the activity interval.

This results in a processed dataset (shown in Table 6.1b) that includes the following temporal features of each activity in order:

- **Activity**: the name of the ADL.
- **Start time**: the start time of the activity.
- **End time**: the end time of the activity.
- **Duration**: the duration of the activity in seconds.
- **Daycase**: the date of the activity.
- **Interval**: the interval in seconds until the next activity starts.

As Table 6.1b shows, it is apparent that a duration of only 39 seconds for cooking breakfast seems unusual, as it is merely an excerpt from the data. In this scenario, the individual was managing both cooking and eating concurrently. Therefore, the short recorded duration for cooking is not reflective of the total time spent on this activity. It indicates that the cooking activity was interrupted by eating and is likely to resume after the eating is completed.

6.5 Temporal pattern recognition and deviation detection

In this section, we present the approach for identifying temporal patterns and detecting anomalies and deterioration of daily activities. Finally, we present the results of our approach.

The proposed approach consists of three stages, as shown in Figure 6.1: (i) modelling behaviour markers, (ii) building predictive models, and (iii) deviation detection. We describe each stage in more detail next.

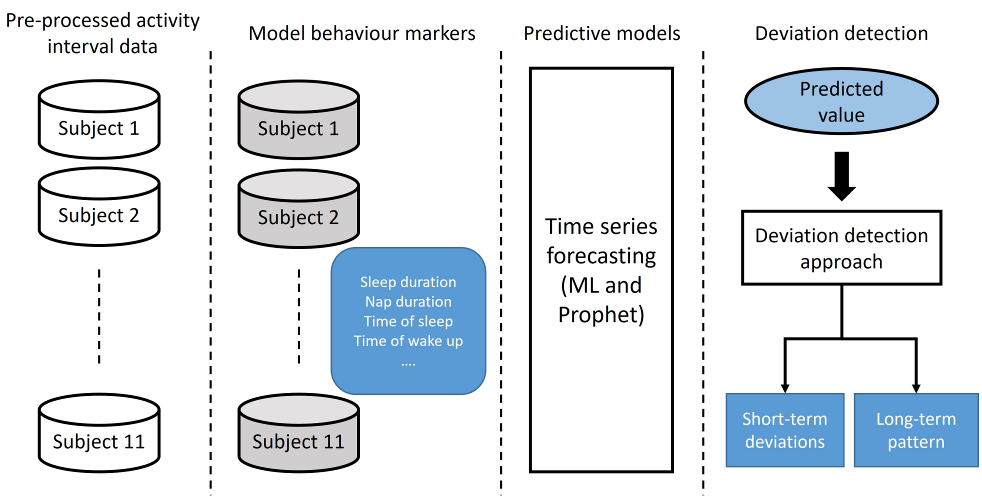


Figure 6.1: An overview of the proposed approach.

6.5.1 Modelling behaviour markers

In order to extract temporal patterns of daily activities, we focus on modelling behaviour markers for each day to build temporal profiles of individuals. Behaviour markers describe activities at a higher level and encapsulate rich temporal information. This allows us to abstract key aspects of daily routines into measurable indicators that represent significant elements of an individual's lifestyle, such as sleep duration, the duration of staying in the toilet, etc.

We model behaviour markers based on the preprocessed data in Section 6.4. Due to the data being continuous and we want to model the behaviour markers for each day, it requires us to split events occurring at midnight into two events, each belonging to the previous and new day respectively. For example, a *Watch TV* event from 22:10:00 to 01:12:23 is split into two events, one from 22:10:00 to 23:59:59 for the previous day and one from 00:00:00 to 01:12:23 belonging to the new day. The only exception is the sleep activity, which we measure from noon to noon of the next day, because people usually sleep overnight. If we choose to measure select the period from midnight to midnight to measure the sleep duration, it will be calculated from the midnight of this day to the midnight of the next day. According to the domain knowledge, the sleep duration calculated in this way does not represent the bedtime of this day but is divided into two days.

The modelled behaviour markers are extracted according to the temporal features related to each daily activity such that provide key insights of the individual's lifestyle and potential links to health indicators.

To focus our analysis on daily behaviour that typically indicates routine activities, we select key activities such as having three meals a day, cooking, sleeping, watching TV, reading, etc. We then extract behaviour markers focusing on temporal information of these activities, such as the start time, end time and duration of the activity. The total number of behaviour markers we defined is 26, with some examples shown in Table 6.2. We subsequently focus our efforts on detecting deviations on these particular behaviour markers.

Behaviour markers	Description
`nap_duration`	total *nap* duration
`sleep_time`	time of *sleep*
`wake_up_time`	morning *wake up* time
`last_personal_hygiene`	time of the last *personal hygiene*
`last_nap_endtime`	end time of the last *nap*
`sleep_duration`	duration of *sleep*
`bed_toilet_duration`	duration of *toilet* visits during *sleep*

Table 6.2: Examples of daily behaviour markers.

6.5.2 Building predictive models

To detect short and long term deviations of ADLs from a routine, which may indicate health deterioration, a key input here is the *ground truth* compared to which certain behaviour is considered a *deviation*. As we discussed in the introduction of the thesis (Section 1.3), each individual may have a significantly different routine and lifestyle. This variance can be exacerbated by the consideration of the large variety of different health conditions that may apply to each person. For instance, the connections between frailty and sleep disturbances have been well studied (Piovezan et al., 2013). We therefore adopted a personalised approach and observed daily living deviations from an individual routine, regardless of whether these routines align with commonly accepted health standards. This means we are specifically monitoring and identifying behaviour when an individual diverges from their regular patterns, even if those patterns might be considered unhealthy by general consensus.

More specifically, given a set of daily behaviour markers for each individual, we create a predictive model to compute their expected values each day. Taking the behaviour marker of `nap_duration` as an example, we develop a model that can predict the total duration of a person's naps during the day given their historical data. We also account for seasonality by including the day of the week (e.g. some people may nap more during the weekend) and the month (e.g. people tend to sleep longer during the winter, particularly if affected by a seasonal affective disorder) as features (Anderson et al., 1994). This model is then used as the ground truth of expected values, against which any new observed behaviour is compared.

To build predictive models that represent the normal daily behaviour markers of an individual, such a model needs to be trained with historical data of routine daily living. Our predictive models are built based on the extracted behaviour markers in the Section 6.5.1. For each individual, we build separative predictive models for each behaviour marker.

In our analysis, we consider special cases of daily behaviour that may need to be filtered out to avoid skewing the results. These include atypical or one-off events that do not reflect the individual's normal routine but could influence the analysis if included. It is crucial to differentiate between true special cases and recurring patterns that may initially appear as anomalies. For example, if an individual regularly sleeps away from home every Friday, this behaviour should not be treated as a special case but rather as a part of their normal routine. In such cases, our approach involves analysing the frequency and regularity of these events to determine whether they constitute a consistent pattern. For example, we identify a single day where an individual is away from home overnight and sleeps elsewhere as special case and remove it.

Then we remove outliers in the data by filtering out values that deviate from the mean by 2 standard deviations, which is a common cut-off for outliers in practice (Ilyas and Chu, 2019).

We use the remaining data to train a regression model (see Section 2.1.1) to predict the future value of behaviour markers. The features incorporated into the model are seasonality-based features to capture the temporal dynamics of daily activities. These include the full date, such as the month and day, which provides specific temporal context; the day of the week, which helps to account for weekly patterns; and the order of data instances, which aids in understanding progression or changes over time.

No.	Model
1	Random Forest Regressor
2	K Neighbors Regressor
3	Extra Trees Regressor
4	AdaBoost Regressor
5	Light Gradient Boosting Machine
6	Gradient Boosting Regressor
7	Decision Tree Regressor
8	Linear Regression
9	Ridge Regression
10	Lasso Regression

Table 6.3: The list of our evaluated machine learning regression models.

Our next goal is to maximise the accuracy of the model. When selecting the best-performing model, we compare the predictive accuracy of different machine learning (ML) regression models and time series forecasting models (see Section 2.1.1). The machine learning models we used include Random Forest Regressor, Linear Regressor, Logistic Regressor, Gradient Boosting Regressor, etc., which are most commonly used models for prediction tasks (James et al., 2023). The full list of machine learning regression models we used is shown in Table 6.3. Then the top 3 best performing among them are integrated as an ensemble using Stacking and Blending techniques (Maclin and Opitz, 1999) to further improve the performance. Stacking is a method where we build a meta-model on top of the predictions of a set of models to make the final predictions. Blending is basically building a voting classifier on top of original models.

We also compare the accuracy of ML regression models with one statistical time series approach named *Prophet* (Taylor and Letham, 2017), because the later is specifically tailored to deal with seasonal effects in time-series data. Prophet is a generalised additive model (James et al., 2023), which allows non-linear functions applied to the regressors and the final prediction is the sum of different components modelled separately, e.g., trend and seasonality. The components of Prophet are combined in the following equation:

$$y(t) = g(t) + s(t) + h(t) + \varepsilon(t) \tag{6.1}$$

where $g(t)$ is the trend function which models non-periodic changes in the value of the time series; $s(t)$ represents periodic changes (e.g., weekly and yearly seasonality); and

$h(t)$ represents the effects of holidays which occur on potentially irregular schedules over one or more days. The error term $\varepsilon(t)$ represents any idiosyncratic changes, which is normally distributed.

We build a separate model for each behaviour marker of an individual. For each model, we use k-fold time series cross validation techniques to evaluate its prediction performance (Bergmeir and Benítez, 2012). Compared to standard cross validation, i.e., randomly shuffling and partitioning data into training and testing sets, time series cross validation maintains the temporal order of data instances. This ensures that the testing data comes strictly after the training data, reflecting the natural progression and dependencies inherent in time series data. For k-fold time series validation, the data is divided into $k+1$ partitions. In the kth split, it returns the first k folds as the training set and the $(k+1)$th fold as the testing set. Given that our data spans only two months, a 5-fold or 10-fold cross validation would result in insufficient data for accurate learning of ADLs patterns. Consequently, we perform a 3-fold time series cross validation.

All experiments are evaluated using the mean absolute error (MAE) and the root mean squared error ($RMSE$) measures. MAE, shown in (6.2), and $RMSE$, shown in (6.3) both measure the average magnitude of the errors in a set of predictions, with 0 corresponding to perfect accuracy, while $RMSE$ magnifies the impact of large errors. In these equations, for each predicted value i, $\hat{y}_i$ represents a size-n vector of the predicted values, y_i is the vector of actual values, and n is the number of test instances.

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |\hat{y}_i - y_i| \tag{6.2}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (\hat{y}_i - y_i)^2} \tag{6.3}$$

For example, if the MAE of a model of sleep duration is 1800 seconds, then the model predicts on average 30 minutes more or less sleep than the actual value. A lower MAE indicates a better performing algorithm with 0 corresponding to perfect accuracy. When comparing models trained with datasets from 2 distinct individuals with the same algorithm, a lower MAE score is an indication of less variability in the data and, therefore, a more predictable and steady daily routine.

6.5.3 Deviation detection

After building the predictive model, we can get the predicted value of behaviour markers, which are utilised as a *ground truth* of expected value for deviation detection. In this section, we present our method of detecting short-term deviations and long-term patterns.

6.5.3.1 Short-term deviations

To detect potential deviations based on the predicted value, we calculate the distance between the predicted routine value and the actual value as shown in (6.4).

$$z = \frac{|y_{predict} - y_{true}|}{MAE} \tag{6.4}$$

The z value in (6.4) represents the distance of the actual value from the predicted value as a fraction of MAE. If the calculated z value of a predicted behaviour marker exceeds a chosen threshold, a deviation is detected. The chosen threshold corresponds to a time window proportional to MAE within which we consider the behaviour as normal or routine.

A distinct threshold can be selected for each behaviour marker in the daily routine, to account for the flexibility we want to allow for different activities. For example, activities such as work or reading often exhibit greater flexibility in timing and duration within a person's daily schedule. By setting specific thresholds for each behaviour marker, we can more accurately discern between normal variability and genuine anomalies in behaviour. For instance, if a participant decides to read a book at a considerably different time than usual, we might not want to consider this as a significant deviation due to the flexible nature of the activity. However, sleep timing is more critical as a health indicator (Piovezan et al., 2013), thus smaller deviations in sleep patterns might be more significant and need to have closer attention. To address this, we might set a higher z threshold for reading, allowing greater variability, and a lower z threshold for sleep-related markers to flag even minor deviations.

For example, given the sleep duration model with MAE of 0.5 hours, assume a predicted sleep duration $y_{predict}$ for a particular day is 8 hours. Setting the threshold of z to 1 means that an actual sleep y_{true} of less than 7.5 or more than 8.5 hours will be considered a deviation.

Given that the MAE reflects the variability in an individual's routine, choosing the same z value across all individuals allows us to account for that variability. In the example above, an MAE of 1 hour would lead us to only consider sleep of less than 7 hours or more than 9 hours to be a deviation.

The deviations of an individual across all behaviour markers in a particular day can be quantified in terms of a deviation *cost*. In this, deviations in each behaviour marker may have a different cost, for instance in terms of its potential impact on the person's health. For example, a deviation from the expected sleep duration is likely considered more impactful to health compared to a deviation in the time one chooses to read a book.

For this purpose, we set a customised weight for each behaviour marker to adjust the impact of specific behaviour. This weighting system is designed to be configurable by the individual or their caregivers, enabling a personalised approach to monitoring. By allowing users to set these weights, the system can prioritise certain behaviours more than others based on individual health needs or preferences. For example, if maintaining a regular sleep schedule is crucial for a person's health condition, a higher weight can be assigned to the sleep marker, making any deviations in sleep patterns more prominent in the overall analysis.

For the detected deviations above, we then calculate both absolute total deviation costs and weighted deviation costs. The total deviation cost is the sum of the cost of each deviating behaviour marker, shown in equation (6.5), and the weighted deviation cost is shown in equation (6.6). In these two equations, i represents each behaviour marker, z_i is the z value from (6.4) for each behaviour marker and w_i is the selected custom weight of the marker. The combination of total deviation cost and weighted deviation cost provides a quantification of deviations on an individual day for a person. By comparing the value of total deviation cost and weighted deviation cost, we can measure the extent of deviating behaviour. For example, if the weighted deviation cost is significantly higher than the total deviation cost, it indicates the individual's behaviour has deviated considerably. This may need closer attention.

$$C = \sum_{i}^{n} w_i \tag{6.5}$$

$$C = \sum_{i}^{n} z_i * w_i \tag{6.6}$$

6.5.3.2 Long-term patterns

In addition to short-term deviations over a single day, we also analyse the long-term trend of behaviour markers. This type of analysis can help us detect long-term changes such as reduced mobility (some activities taking longer), reduced or disrupted sleep patterns, etc. that may be linked to health outcomes, such as deterioration, physical and cognitive frailty, and reduced independence.

We calculate long-term trends of behavioural markers based on the individual's normal daily routine. In this context, we consider days with detected deviations through the previous analysis as *abnormal* and filter them out of the dataset. We are then able to focus on the normal behaviour markers to analyse the seasonality for the long-term behaviour trend. The seasonality of the model gives us insights on whether it indicates the deterioration of health. For example, the sleep duration is declining in a long-term period, which may indicate the health condition is deteriorating.

6.5.4 Experimental study

In this section, we present the results we obtained in each stage of our approach. We develop and evaluate our predictive models of behaviour markers by comparing the performance of different learning models. For detected deviations, as we do not have the ground truth, we primarily rely on the visualisation of data patterns. We develop an interactive dashboard for visualising personal temporal profiles of daily living, including detected deviations, daily routines, etc.

6.5.4.1 Building predictive models

In our effort to develop a predictive model for individual ADL routines, we performed an array of experiments to select the best performing algorithm as follows: (i) We compared the performance of a number of different ML regression models. Here, we only use time-series as training features, as mentioned in Section 6.5.2. (ii) We ensemble the top 3 regression models to improve the performance. (iii) We then compare the Prophet algorithm with them.

We choose the data of individual *hh102* in the CASAS datasets as an example to present our results. The *hh102* dataset consists of 62 days of activity-labelled data, ranging from 15/06/2011 to 15/08/2011. The data from 08/08/2011 at 14:38 to 13/08/2011 at 15:31 is missing due to the individual not being at home during this period. This absence means that no activities were recorded in the home environment, leading to a gap in the dataset for these dates and times. Therefore, the total data we have is 58 days.

The performance of different models in terms of the two metrics (*MAE* and *RMSE*) is shown in Table 6.4. The units of these two metrics are both seconds.

As the result shows, Prophet is the best model for forecasting sleep duration. In

Model	MAE	RMSE
Prophet	3683	4532
Random Forest Regressor	4200	5065
K Neighbors Regressor	4306	5394
Extra Trees Regressor	4552	5436
Stacking Regressor (top 3)	5943	6551
Blending Regressor (top 3)	4214	5127

Table 6.4: MAE and $RMSE$ of different predictive models of sleep duration for individual *hh102*. The units of these two metrics are both seconds.

fact, Prophet outperformed the ML algorithms in all behaviour markers across our dataset, so we selected that algorithm for all our predictive models. This came with added benefits of Prophet, such as the calculation of seasonality and long-term trends, which are provided in the Prophet algorithm (see Section 6.5.2).

After the best performing model is selected, the model performance on the behaviour markers of the individual *hh102* is shown in Table 6.5. As an example, the MAE in the prediction of the time when breakfast was cooked is 3745 seconds, so approximately one hour, which is a reasonable level of variability for that activity according to its standard deviation is 4253 seconds. The maximum MAE was observed in predicting the time of eating lunch, with an error of 6120 seconds, approximately 1.7 hours. While this error is slightly higher, it remains within a reasonable range, considering the individual often varies their lunchtime with a similar range (standard deviation is 5374 seconds). Therefore, we consider that all of the obtained results show a reasonable level of accuracy, given the high variability in people's daily lives.

6.5.4.2 Short-term deviations

After developing predictive models for various behaviour markers, we then identify short-term deviations by calculating the z value, which measures the distance between the actual value and the predicted value, as we presented in Section 6.5.3.1. We present the results of our deviation detection approach using individual *hh102* data as an example. In this, we set the threshold of the z value of each behaviour marker to 1.

To quantify deviations in a particular day by a deviation cost, we set weights for different behaviour markers. In our analysis, the weight of sleep duration, bathing duration and leave home duration are set to 2 and the weights of the rest of the behaviour

Behaviour markers	MAE	RMSE
cook_breakfast_time	3745.318	3938.721
eat_breakfast_time	4185.226	4611.391
cook_lunch_time	4232.662	5127.29
eat_lunch_time	6120.533	6651.227
cook_dinner_time	1263.975	1580.311
eat_dinner_time	1687.354	2115.673
sleep_time	2726.950	3160.010
sleep_duration	3875.959	4543.112
take_medicine_time	4615.162	4933.677
morning_medicine_time	2250.793	2825.039
bathe_duration	187.128	274.429
leave_home_duration	3415.511	4944.483

Table 6.5: Performance measures of the models on the behaviour markers of *hh102*. The units of these two metrics are both seconds.

markers are set to 1. These weights can be customised according to personal preferences or tailored based on instructions from caregivers, allowing for specific attention to be given to particular behaviours.

We use the results of sleep duration as an example to illustrate our results, as it is an important health indicator. Since the data ends on 15/08/2011, it is not possible to measure the sleep duration for that day. Therefore, we exclude the data from this day when building the sleep duration model. Figure 6.2 shows the calculated z values for sleep duration on different dates. Negative values mean that the actual sleep duration was longer than the predicted value, while positive values mean the expected sleep duration was longer than the actual value. We detect deviations on 19 out of the 58 days, which we label *abnormal*. Some dates, specifically the 10th and 11th of July, and the 7th and 8th of August, show very high z values. These anomalies are attributed to the individual not sleeping at home on these dates, although they were at home during parts of the day. This contrasts with the period from the 9th to the 13th of August, when the individual was completely out of the home, resulting in no recorded activities.

We also visualise the actual values and predicted values of behaviour markers related to the time of the day and their MAE range in a timetable in Figure 6.3. Red

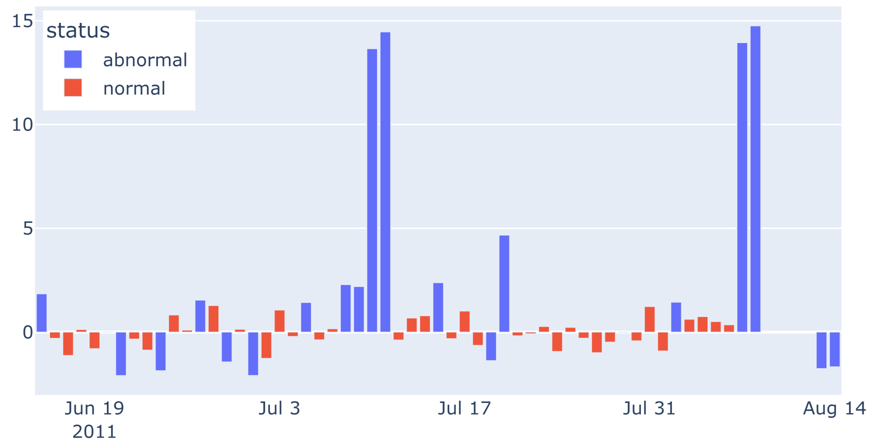


Figure 6.2: Detected deviations of sleep duration for *hh102*.

marks indicate the actual value and the yellow marks indicate the predicted value with a line showing the corresponding fault tolerance range based on the threshold of the z value. We can readily figure out which behaviour markers are likely deviations and the magnitude of each one. For example, we see that participant *hh102* took their morning medicine earlier than the predicted time on this particular day.

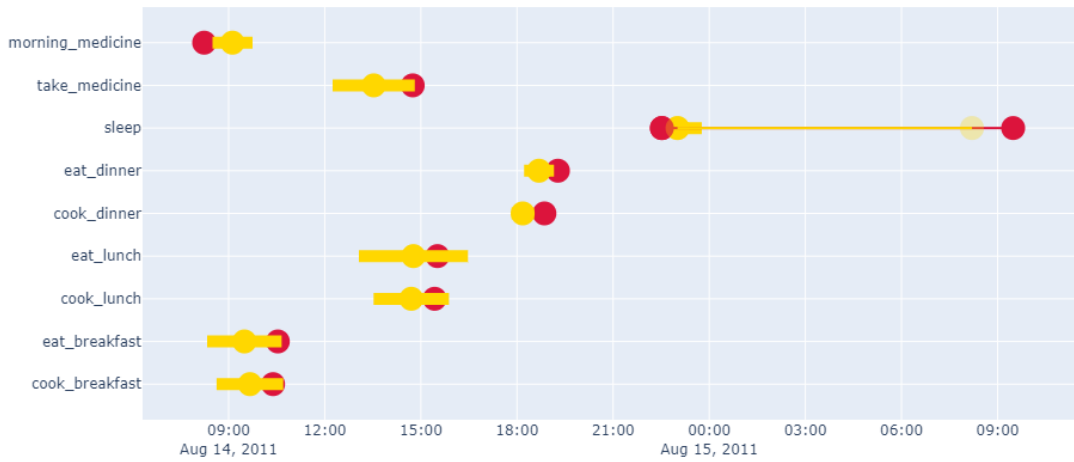


Figure 6.3: The visualisation of actual values and predicted values of behaviour markers. The red marks indicate the actual value and yellow marks mean the predicted value with a line showing the fault tolerance.

We summarise the computed personal temporal profile of the individual, including their daily routines, detected deviations and costs, and long-term trends in an interactive dashboard. The dashboard is developed based on a Python package named Plotly[1].

[1] https://github.com/plotly

Through this interactive dashboard, users can visualise the profiles of different dates and assign different weights to behaviour markers to tailor for individual health needs or preferences (see Section 6.5.3.1). Figure 6.4 shows one of the views of our dashboard that visualises the daily routine, absolute deviation cost, the weighted deviation cost, the number of deviations, the detected deviations, and the predicted ranges (in yellow) and actual values (in red) of behavioural markers related to the time of the day for a particular selected date.

Figure 6.4: The dashboard for visualising the temporal profile related to individual *hh102* on the date of August 14, 2011.

In the dashboard, we can see there are two deviations, one is the time of having

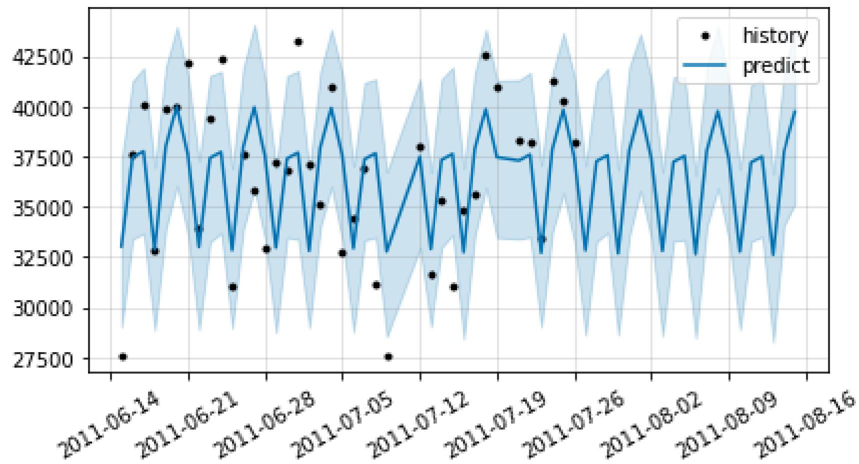


Figure 6.5: The predicted trend of sleep duration for *hh102* using data from June 15 to July 26, 2011. The black dots represent the actual sleep duration of that day, while the blue shaded area shows the prediction range provided by Prophet.

morning medicine, which is slightly earlier than the predicted value. Another deviation is *bed_toilet_duration*, which represents the duration of toilet visits during sleep. This deviation indicates that the actual toilet duration is less than the expected value. We see that the weighted deviation cost (considering the extent of deviations (z value)) is similar to the deviation cost. This similarity suggests that, while these behaviours have deviated from their expected values, they are not significantly different enough to cause major attention. We also visualise the number of deviations. By comparing the number of deviations and the total deviation cost, we can easily figure out whether the behaviours with higher weights are detected as deviations. In this particular case, the number of deviations is equal to the total deviation cost, which means the behaviours with higher weights are not identified as deviations.

This dashboard provides a clear understanding of how the behaviours are performing relative to the expected values, highlighting whether any key areas of interest or concern are deviating from the norm. It can effectively highlight deviations, providing useful insights for monitoring and potential follow-up.

6.5.4.3 Long-term patterns

Based on the 2 months data of the individual *hh102*, we analyse the sleep duration trend to detect whether it is deteriorating. The general trend of sleep duration for *hh102* is shown in Figure 6.5, where the horizontal axis and the vertical axis represent the observed date period and the sleep duration in seconds respectively. The black dots represent the actual sleep duration of that day, while the blue shaded area shows the prediction range (lower bound and upper bound) provided by Prophet. The predicted trend indicates that sleep duration exhibits a relatively stable pattern over an extended period, despite experiencing some daily fluctuations.

We calculate linear trend and weekly seasonality (the relative effect of each day of the week to the predicted value) using all of the data by the Prophet algorithm. The linear trend and weekly seasonality are shown in Figure 6.6 and Figure 6.7, respectively. Next, we filter out the dates with abnormal sleep duration, as identified in Section 6.5.4.2, and investigate the normal pattern over this period. The calculated linear trend and weekly seasonality of our predictive model are shown in Figure 6.6 and Figure 6.7, respectively.

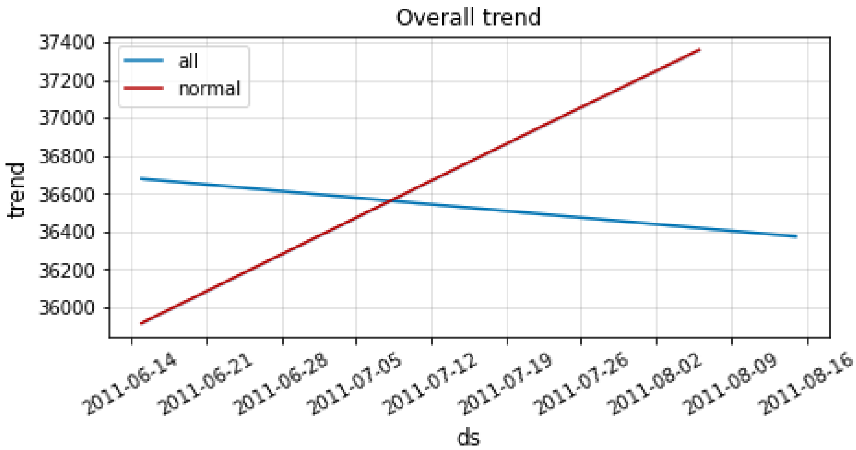


Figure 6.6: The Prophet trend component of sleep duration for *hh102*. The red line represent the "normal" trend, achieved by filtering out the dates with abnormal sleep duration, while the blue line represent the trend across all data.

As shown in Figure 6.6, the two linear trends are visibly different, which demonstrates how deviating behaviour can significantly affect the observed trend. In the context of analysing long-term sleep routines with the aim of exploring implications to health, abnormal days become outliers that skew the results. Instead, we choose to filter those out and focus on the "normal" or routine behaviour and how it evolves through time. The normal linear trend shows that the sleep duration of *hh102* is increasing during this period, suggesting an improving sleep pattern.

The weekly seasonality, as shown in Figure 6.7 shows a similar pattern between the results using all data and those using only normal data. This consistent pattern indicates that the person gets the least sleep on Wednesday and the most sleep duration

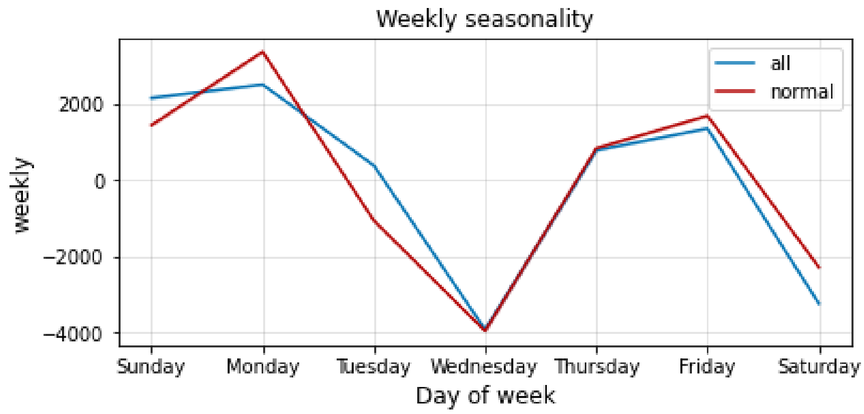


Figure 6.7: The Prophet weekly seasonality component of sleep duration for *hh102*. The red line represent the "normal" trend, achieved by filtering out the dates with abnormal sleep duration, while the blue line represent the trend across all data.

on Monday.

6.6 Process-aware detection of behavioural changes

In this section, we present the probabilistic model checking-based approach for detecting complex behaviour changes over time. We employ PRISM, which is one of the most commonly used probabilistic model checkers (Kwiatkowska et al., 2011).

Our approach first builds user activity models based on Discrete Time Markov Chains (DTMCs) to represent users' daily routines of different periods using activity-labelled data collected by sensors. Next, we detect behavioural changes over time, including i): modelling daily behaviour properties based on Probabilistic Computation Tree Logic (PCTL), ii): leveraging the verification capabilities of probabilistic model checking to assess the probabilities of behaviour properties over different periods, iii) identifying periods that exhibit outlier probabilities of behaviours.

This section first introduces the method of modelling daily routines. Next, we present how to model behaviour properties and use probabilistic model checking along with the user activity model to capture behaviour changes over time. Finally, we present the experimental study of our approach.

6.6.1 Modelling daily routines

Based on the available activity-labelled ADLs data, a sequence of activities performed during a day represents a user's daily routine. We build a user activity model to represent the individual's daily routine in the form of a DTMC as a formal model. In our user activity model, the states represent the activity which is currently being performed, and transitions represent the end of one activity and the start of another one.

Definition 7 (User Activity Model) *The user activity model is defined as* $UAM = (S, Init, T)$*, which is modelled as a DTMC, where:*

- *S represents a set of states $\{s_0, s_1, ..., s_m\}$, that is an activity set with m activities, corresponding to the states that the user is performing one of the activities.*
- *$Init$ represents the initial state of the activity model, indicating the initial activity of the user, where $Init \in S$.*
- *T represents a transition probability matrix ($P_{i,j} \in [0,1]$). If S contains m activities, then T is a $m \times m$ matrix, whose entries are all non-negative and whose rows sum to 1. The interpretation of the number $P_{i,j}$ is the conditional probability, given that the model is in activity i at time t, then the probability of transiting to activity j at time $t+1$ is $P_{i,j}$, i.e.:*

$$P_{i,j} = \mathbb{P}\{s_{t+1} = j \mid s_t = i\}$$

This model transforms the individual's daily activities into a collection of activity states S. For example, a user's state S records a daily activity, such as sleep, taking medicine, eating breakfast, etc. The transition between different activities occurs with different probabilities, recorded in the transition probability matrix T, computed using Algorithm 4. We calculate the transition probabilistic matrix T based on the sequence of activities in the datasets. We extract the possible combinations of two consecutive activities. The first activity denotes the current states s_t, and the second denotes state $s_{(t+1)}$ entered at the next step. We calculate the probability of each transition by employing a frequentist approach, calculating these probabilities based on the observed frequencies of transitions within the data. Algorithm 4 shows the pseudocode for computing activity transition matrix.

An excerpt of the example PRISM code of the user activity model is shown in Listing 1. In this model, we have 33 states representing 33 different activities. For each

Listing 1 An excerpt of the example PRISM code of the user activity model.

```
dtmc
module ActivityModel
s:[0..32] init 0;
[ ]s = 0 -> 0.2889:(s'=0)+ 0.3334:(s'=1)+ ...+ 0.0:(s'=32);
[ ]s = 1 -> 0.9375:(s'=0)+ 0.0625:(s'=1)+ ... + 0.0:(s'=32);
...
...
...
[ ]s = 31 -> 0.0:(s'=0)+ 0.1625:(s'=1)+ ... + 0.1429:(s'=32);
[ ]s = 32 -> 0.0:(s'=0)+ 0.0:(s'=1)+ ... + 0.0:(s'=32);
endmodule
```

state, we specify transition probabilities to other states to model the likelihood of moving from one current activity to another. For example, the probability of transitioning from state 0 to state 1 is 0.3334.

Algorithm 4 Compute activity transition matrix

Require: An set of activities recorded in datasets $A = [a_1, a_2, ..., a_m]$, an activity sequence σ

1: $ConsecutiveSeq \leftarrow$ Extract consecutive binary sequences (s_t, s_{t+1}) from σ
2: $P = \parallel m \times m \parallel$ ▷ Initialise a $m \times m$ transition matrix
3: **for** $a_i \in A$ **do** ▷ Iterate each activity as the current activity
4:     **for** $a_j \in A$ **do** ▷ Iterate each activity as the next activity
5:         $Num_{i,j} = Count((s_t = a_i) \wedge (s_{t+1} = a_j))$
▷ Count the number of $ConsecutiveSeq$ transferring from a_i to a_j
6:         $P_{i,j} = \frac{Num_{i,j}}{Count(s_t = a_i)}$ ▷ Calculate the probability of transitions from a_i to a_j
7:     **end for**
8: **end for**
9: **return** P

6.6.2 Detecting behavioural changes

In this section, we present how to use probabilistic model checking to detect behaviour changes by checking verification properties against constructed user activity models.

First, we introduce the Probabilistic Computation Tree Logic (PCTL) as the formalisation language of verification properties. Then, we formalise a set of properties to define the daily activity behaviours and assess the probability of these behaviours by the probabilistic model checker. Finally, we introduce a statistical approach to compare the probability of behaviours over different periods to detect behaviour changes.

Definition 8 (PCTL Syntax) *The syntax of PCTL formula is defined as follows:*

$$\phi := true \mid false \mid p \mid \phi \wedge \varphi \mid \phi \vee \varphi \mid \phi \rightarrow \varphi \mid \neg\phi \mid \mathbb{P}_{\sim\lambda}(\psi)$$

$$\psi := X\ \phi \mid F\ \phi \mid\ \phi\ U\ \varphi \mid \phi\ U^{\leq k}\ \varphi$$

where:

- *p denotes a finite set of atomic propositions. A state s in a DTMC model corresponds to an atomic proposition p.*
- *φ, ϕ are state formulae.*
- *ψ is a path formula interpreted over the states and paths of the model.*
- *$\neg$, $\wedge$ and $\vee$ are the boolean connectives.*
- *$\mathbb{P}_{\sim\lambda}$ is a probabilistic operator, where $\sim\ \in \{<,\ \leq,\ >,\ \geq\}$ is a comparison operator and λ is a probability threshold. It assesses whether the probability is below ($<$) or above ($>$) the threshold λ.*
- *U is the "until" operator. $\phi\ U\ \varphi$ is true for a path if φ is true in some state of the path and ϕ is true in all preceding states. F is the "finally" operator, which is a special case of "until", represented as true $U\ \phi$. $F\ \phi$ denotes that the state will eventually meet ϕ.*
- *$k \in \mathbb{N}^{+}$ is a positive integer number reflecting the maximum number of transitions needed to reach a certain state.*
- *X is the "next" operator. $X\ \phi$ denotes the next states satisfies ϕ.*

Definition 9 (PCTL Semantic) *Let M be a DTMC, s is a state in the DTMC model, V is a labelling function mapping each state s to a set of propositions. $Path(s)$ is the set of paths starting from s. For a path $\sigma =< s_1, s_2, ..., s_n >$, $\pi_m(\sigma) = \prod_{0 \leq i \leq n} P(s_i, s_{i+1}) =$*

$P(s_0, s_1) \times P(s_1, s_2) \times ... \times P(s_{n-1}, s_n)$ *is defined as the probability. The* $\models$ *symbol is used to denote the satisfaction relationship. The verification property* ρ *is given in the form of PCTL, and the semantics is defined as follows:*

$M, s \models \rho$	*iff* $\rho \in V(s)$.
$M, s \models \neg\rho$	*iff* $M, s \not\models \rho$.
$M, s \models \rho \wedge \varphi$	*iff* $M, s \models \rho$ *and* $M, s \models \varphi$.
$M, s \models \rho \vee \varphi$	*iff* $M, s \models \rho$ *or* $M, s \models \varphi$.
$M, s \models \mathbb{P}_{\sim\lambda}(\psi)$	*iff* $\pi_m(\sigma \in Paths(s)$ *s.t.,* $M, \sigma \models \psi) \sim \lambda$.
$M, s \models X\ \rho$	*iff* $M, \sigma[1] \models \rho$.
$M, s \models F\ \rho$	*iff* $\exists\ i \geq 0,$ *s.t.,* $M, \sigma[i] \models \rho$.
$M, s \models \rho\ U\ \varphi$	*iff* $\exists\ i \geq 0,$ *s.t.,* $M, \sigma[i] \models \varphi$ *and* $(\forall j < i)\ M, \sigma[j] \models \rho$.
$M, s \models \rho\ U^{\leq k}\ \varphi$	*iff* $\exists\ 0 \leq i \leq k,$ *s.t.,* $M, \sigma[i] \models \varphi$ *and* $(\forall j < i)\ M, \sigma[j] \models \rho$.

where $\sim \in \{<, \leq, >, \geq\}$, $0 \leq \lambda \leq 1$ is a probability threshold, ρ and φ are state formulae, and $k \in N$ is a natural number that denotes a time step.

We mainly use $P =_? (\phi)$ to return a probability value for the given formula ϕ. For example, the formula of $P =_? [X(s = eating)]$ means the probability that the next state is eating will be returned.

Typically, when checking a property, PRISM only outputs a value for the initial state of the model by default. In order to ask PRISM to return different values, we introduce the *filter* formula.

Definition 10 (PRISM filter formula) *The filter formula allows you to customise properties of PRISM to obtain different results. The filter formula takes the following form:*

$$filter(op,\ prop,\ states)$$

where `op` *is a filter operator,* `prop` *is any PCTL properties, and* `states` *is a Boolean-valued expression identifying a set of states over which to apply the filter. This formula indicates that applying the operator* `op` *to the values of property* `prop` *for all the states that satisfy* `states`*. Thus, the filter can be used to verify PCTL from different starting states. The commonly used operator* `op` *includes:*

- *min: the minimum value of* `prop` *over states satisfying* `states`*.*
- *max: the maximum value of* `prop` *over states satisfying* `states`*.*

- *count: counts the number of states satisfying* `states` *for which* `prop` *is true.*
- *sum: sums the value of* `prop` *for states satisfying* `states`*.*
- *avg: the average value of* `prop` *over states satisfying* `states`*.*

For example, the formula $filter(max,\ p =_{?} [F\ (s = eating)],\ s = cooking)$ gives the maximum probability starting from state *cooking* when reaching *eating* state.

In the next section, we use PCTL and the filter formula to formalise daily behaviours and evaluate the probability of these behaviours against the user activity model.

6.6.2.1 Modelling daily behaviour properties

To delve into daily behaviours and provide a detailed analysis of daily routines and behaviours, we particularly focus on instrumental daily activities like cooking and cleaning, and essential basic daily activities such as grooming and dressing, as well as medication-related activities. We choose 4 verification properties of interest as an example to model daily behaviours related to these activities. These properties are detailed below.

- **Morning routine efficiency**: This property can measure the probability that the progression from *wake up* to *have breakfast* occurs within a given number of steps. Changes in this probability could indicate alterations in morning routine. An increase in this probability might suggest that the individual has become more efficient in their morning routine, possibly due to lifestyle adjustments, improved time management, etc. A decrease could indicate that the person is spending more time on other morning activities, perhaps due to changes in priorities (like exercising or meditating in the morning), altered preferences (such as taking more time for relaxation or reading), or even health issues that slow down the morning routine.

- **Medication non-adherence**: This property is designed to measure the probability of not having medicine before a meal. This can indicate adherence to medical advice, and changes in health status and medication. An increase in this probability might suggest issues with memory or cognitive functions, which could be due to ageing, stress, sleep deprivation, or health conditions affecting memory. Moreover, improvement in health might lead to a less stringent medication schedule, thereby increasing the probability of not taking medicine before meals. Conversely, worsening health conditions might necessitate more careful adherence to medication schedules.

- **Meal preparation omission**: This property can evaluate the probability of not cooking before having a meal. A rising probability of not cooking could indicate a greater reliance on prepared meals or takeout. This might be due to busier schedules, convenience, or changes in financial circumstances. A decrease in this probability may indicate there may be a growing awareness of healthier eating habits because ingredients and cooking methods can be controlled when cooking by themselves. Financial considerations may also lead individuals to cook at home as a cost-saving measure. Therefore, by monitoring this property over time, it is possible to gain insights into lifestyle change, dietary habits, and even economic conditions.

- **Clean dishes consistency**: This property can measure the probability of washing dishes after a meal within a given number of steps. Variations here might reflect changes in domestic habits or time management skills. An increase in this probability may suggest that the person has become more routine-oriented or efficient in their daily habits. They might be focusing on maintaining a clean and orderly environment by not allowing dishes to pile up. If the person's schedule has become busier, perhaps due to increased work responsibilities or other duties, they may postpone washing dishes, resulting in a decreased probability.

We encoded these properties in PCTL to be able to measure the probability against the constructed user activity model.

Property 1. Morning routine efficiency. What is the probability of having breakfast after waking up within a given number of steps. The initial state is *wake up*.

$$P =_? [true\ U^{<=n}\ (s = s_i)]$$

where n is a variable to define the number of steps between waking up and having breakfast, and s_i represents the state of having breakfast.

Property 2. Medication non-adherence. What is the probability of not having medicine before meal.

$$P =_? [(\neg(s = s_{medicine})\ U\ (s = s_{meal}))]$$

where $s_{medicine}$ denotes the state of medication intake, for example, taking morning medicines or midday medicines. Meanwhile, s_{meal} refers to the state of eating, encompassing meals like breakfast and lunch. It is essential that $s_{medicine}$ and s_{meal} are matched appropriately. For instance, if $s_{medicine}$ refers to taking medicine in the morning, then s_{meal} should be related to eating breakfast. This ensures that the activities being considered are directly related within the same part of the daily routine.

Property 3. Meal preparation omission. What is the probability of not cooking before having a meal.

$$P =_? [(\neg(s = s_{preparation})\ U\ (s = s_{meal}))]$$

where $s_{preparation}$ represents the state of preparing a meal, such as cooking breakfast, etc. s_{meal} represents the state of having a meal. There should be a connection between $s_{preparation}$ and s_{meal}. For example, if $s_{preparation}$ is *cooking breakfast*, then s_{meal} should be the state of *eating breakfast*.

Property 4. Clean dishes consistency. What is the probability of washing dishes after a meal within a given number of steps.

$$filter(max,\ P =_? [true\ U^{<=n}(s = s_{dishes})],\ s = s_{meal})$$

where n is a variable of this property to define the number of steps from having a meal to washing dishes for verification. s_{meal} represents the state of having a meal while s_{dishes} denotes the state of washing dishes. In different instances, s_{meal} can represent different types of meal, i.e., breakfast, lunch, dinner, etc., and s_{dishes} can refer to washing the dishes specific to each of these meals. s_{meal} and s_{dishes} should correspond to each other, e.g., if s_{meal} is *eating breakfast* then s_{dishes} should be *washing breakfast dishes*.

6.6.2.2 Identifying outlier behaviours and deviating periods

To evaluate the probability of behaviour shifts over time, we divide the available data evenly into periods, for example, a period can contain 2 weeks. These periods are arranged in chronological order. We build a separate user activity model for each period following the procedure described in Section 6.6.1. Then we evaluate the probability of the PCTL properties against the user activity model corresponding to each period.

Upon calculating the probabilities of specified properties for each period, our next step involves pinpointing any abnormal probabilities across these periods. We employ two distinct strategies to achieve this.

Firstly, we focus on identifying periods where the probabilities of a specific behaviour deviate from the normal established by other periods. We detect outliers by comparing the probability of each single property across all periods. This provides us insights on significantly higher or lower probabilities for a given behaviour. We use the interquartile range (IQR) method to identify outliers among these probabilities (Dekking, 2005). IQR is defined as the difference between the 75th and 25th percentiles of the data. To calculate the IQR, the dataset is divided into quartiles, which are denoted by $Q1$, referred to as the lower quartile, $Q2$ as the median, and $Q3$, referred to as the upper quartile. The lower quartile corresponds with the 25th percentile and the upper quartile corresponds with the 75th percentile. The IQR is calculated by equation (6.7). The common cut-off for outliers is 1.5 times the IQR and subtract this cut-off from the 25th percentile as lower bound (Dekking, 2005), shown in equation (6.8) and add it to the 75th percentile as upper bound, shown in equation (6.9).

$$IQR = Q3 - Q1 \tag{6.7}$$

$$Lower\ bound = Q1 - IQR \times 1.5 \tag{6.8}$$

$$Upper\ bound = Q3 + IQR \times 1.5 \tag{6.9}$$

Secondly, we conduct a period analysis considering all properties. This strategy involves evaluating each period as a whole by considering the probabilities of all properties altogether. This strategy helps us to determine if a specific period stands out as abnormal when compared to others. Therefore, we can identify periods that diverge from the individual's typical behaviour patterns. The deviating period could potentially indicate lifestyle changes, health issues, or significant life events affecting multiple aspects of daily living. This aims to identify periods that are worth attention for potential intervention.

To identify these periods, we categorise the identified properties into "negative" and "positive" groups based on their implications for lifestyle, health, or efficiency. Positive properties are associated with behaviours where an increase in probability is indicative of positive developments, such as enhanced lifestyle quality, improved health, or greater efficiency in daily routines. Examples of such properties include morning routine efficiency and cleaning dishes consistency. For these positive properties, a higher probability is desirable as it reflects better adherence to healthy or efficient behaviours. Thus, when the probability falls below a certain threshold (the lower bound), it is considered an outlier, indicating a potential decrease in positive behaviour that might need further investigation or intervention. Conversely, negative properties represent those where an increase in probability could indicate a decline in health, well-being or efficiency. Examples include medication non-adherence and meal preparation omission properties. For these properties, a lower probability is preferable because it indicates reduced engagement in behaviours that could negatively impact health or well-being. Therefore, probabilities that exceed a predetermined upper threshold (the upper bound) are flagged as outliers.

After identifying the outliers probabilities for each property, we proceed to categorise the probabilities into two distinct labels: those not identified as outliers are labelled as *normal* and those which are outliers are labelled as *abnormal*. Therefore, for each period, we assign labels indicating whether the probability of each property is normal or abnormal. These labels, representing the state of each property within a period, are then utilised as features to discern periods exhibiting abnormal daily behaviour. To achieve this, we adopt the Isolation Forest algorithm (Liu et al., 2012), introduced in Section 2.2, which is a widely applied unsupervised method for identifying outliers within data.

6.6.3 Experimental study

In this section, we first evaluate our methods on a synthetic dataset, which allows us to define expected outcomes for validating our methods. This synthetic dataset allows us to precisely control the anomalies, which enables an assessment of our methods' performance in a controlled environment. Then we perform a sensitivity analysis based on synthetic data to evaluate the effectiveness of our method on subtle behavioural changes. Finally, we conduct a case study on CASAS datasets, which offers practical insights into how our methods perform in real-world scenarios with genuine daily behaviour data.

6.6.3.1 Evaluation on artificial data

To evaluate the effectiveness of our proposed method and assess its sensitivity for identifying subtle behaviour changes, we generate synthetic data, providing a controlled environment that simulates these behaviour changes. By using synthetic data, we can manipulate the occurrence of anomalies and the extent of behaviour changes, which real data cannot provide. While synthetic data offers the advantage of precise manipulation and controlled testing conditions, it is important to acknowledge that it may not capture the full complexity and unpredictability of real-world data.

We design a single-agent model to generate synthetic daily routines for individuals, taking into account the randomness and variability characteristics in human behaviour. Each agent is designed to embody a distinct individual, complete with a unique daily routine. The daily routine is structured around a core set of daily activities, including relaxation, television watching, and drinking, supplemented by specific routines for morning activities, meal preparations, and medication intake. The generation of the daily routine is dynamic, governed by probabilities that determine the occurrence of various activities, such as medication intake or meal consumption.

Each agent is equipped with a set of predefined activity categories. General daily activities are categorised as `base activities`, including *Toilet*, *Dress*, *Personal Hygiene*, *Relax*, *Nap out of bed*, *Work*, *Watch TV*, *Read*, *Phone*, *Entertain Guests*, *Step Out*, *Groom*, *Bathe*, *Work At Table* and *Drink*. Activities typically performed in the morning are categorised in `morning activities`, including *Toilet*, *Dress*, *Read*, *Bathe*, *Drink*, *Relax* and *Groom*.

Agent's daily routines incorporate randomness through the selection of activities and the decision-making process regarding meals and medication. The daily routine is generated according to a workflow, shown in Figure 6.8. It starts with *Sleep* to represent the agent's state at the beginning of the day. Then, a random number of activities are selected from `morning activities` to fill in their morning routine. We introduce probabilities to certain activities to ensure a realistic variation in daily routines, like the probability of having medicines and meals. The model includes routines for three main meals: breakfast, lunch and dinner, with a probability assigned to each to reflect the likelihood of the agent performing these activities. A similar approach is used for having morning medication before breakfast and having midday medication before lunch, where a random number is generated for comparison with the predefined probability to determine whether the agent taking medication or not before meals. In the intervals between their meals, such as the period after breakfast leading up to lunch, we insert a random selection of activities from the `base activities` to fill in parts of the day. Finally, we ensure each day comprises a random total number of activities.

Each meal routine is further detailed by adding probabilities for cooking and washing dishes. The workflow of each meal routine is shown in Figure 6.9. For each meal activity, including breakfast, lunch and dinner, we assign a probability that determines

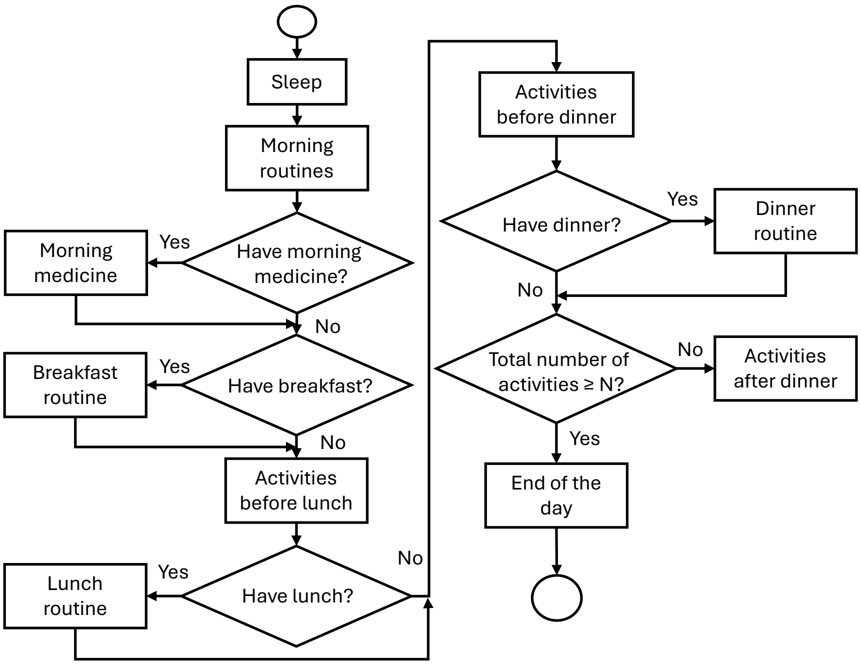


Figure 6.8: The workflow of the agent for generating daily routine.

whether the agent will prepare the meal. The decision-making process for skipping meal preparation is determined by generating a random number between 0 to 1 and comparing it with the pre-defined probability of skipping meal preparation. Following a meal, the agent faces the decision of whether to clean up immediately, delay the task, or skip it altogether. The model introduces probabilities for each of these outcomes, mirroring real-life scenarios where individuals might immediately wash dishes, leave them for later, or neglect the task due to various reasons. In cases where agents delay washing dishes, we introduce a random number of other activities from *base activities* before the task of cleaning dishes is revisited.

Once the day's activities are finished, the model is reset, thereby simulating the start of a new day.

In our study, we conduct a simulation, generating 112 days (8×14 days) of data and dividing the data evenly into eight periods, each of which contains two weeks (14 days) of data. Moreover, we simulate both typical daily routines and routines with intentional disruptions or "noise" to establish a baseline for identifying which periods exhibit abnormal patterns. Within the eight periods of our simulation, we introduce one period of noisy daily routine. This is done to assess whether our methods are capable of accurately identifying the period with irregular or abnormal patterns amidst the standard routines. With just one noisy period, we can effectively test the sensitivity

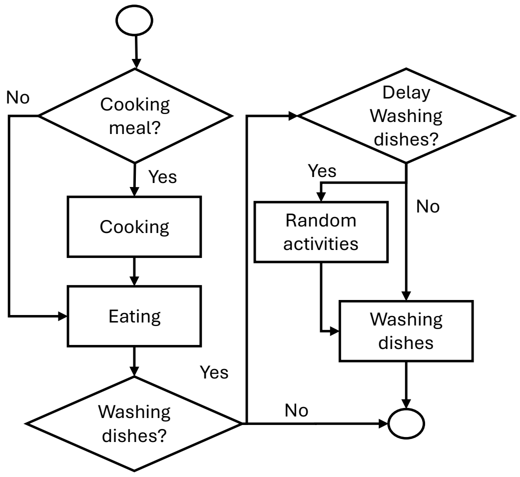


Figure 6.9: The workflow of each meal routine.

and accuracy of our method in detecting the noisy period.

In our simulation of typical daily routines, to make the synthetic data as realistic as possible, we determine the probabilities of various behaviours, such as the likelihood of skipping meal preparation, etc., by analysis the real-life datasets. We derive the frequency of various behaviours from the preprocessed CASAS data, as outlined in Section 6.4. By interpreting the data through the lens of frequency, we establish probabilities that reflect the regular occurrence of specific behaviours. These probabilities are then integrated into our simulation model to construct representations of typical daily routines. These probabilities are detailed as follows:

- The probability of having morning medicine: 0.9.
- The probability of having midday medicine: 0.74.
- The probability of having breakfast: 0.65.
- The probability of having lunch: 0.48.
- The probability of having dinner: 0.86.
- The probability of forgetting cooking: 0.2.
- The probability of forgetting washing dishes: 0.2.
- The probability of delaying washing dishes: 0.5.
- Range of activities between eating and washing dishes: 1 to 3, chosen randomly.

- Range of activities between waking up and cooking breakfast: 1 to 3, chosen randomly.
- Range of total activities: 35 to 40, chosen randomly.

For the simulation of intentionally disrupted daily routines, we conduct two sets of experiments. The first experiment is generating completely random daily routines. This is accomplished by randomly picking activities from the pool of all activities, while ensuring that the total count of activities each day falls within the range of 35 to 40. The selected activities for each day form a sequence that represents a random daily routine. This strategy ensures complete randomness in daily schedules. We insert the generated random daily routines into the third of the eight periods of daily routine data. This insertion is designed to test the model's ability to detect and differentiate between regular and irregular patterns in different periods.

We build the user activity model for different periods. For each period, we evaluate the probability of different PCTL properties. The upper and lower bounds of the probability for each property are also computed. The result is shown in Table 6.6. Probabilities exceeding the computed upper and lower bounds are emphasised in bold. The result shows that there are four outlier probabilities within the third period, intentionally set as the disrupted period. Then Isolation Forest (see Section 2.2) is applied to determine the deviating period based on the features of whether the probabilities of properties are normal or abnormal within each period. The Isolation Forest results confirmed the third period as an outlier, aligning with the designed disruption.

Another experiment is that we simulate scenarios that mimic real-life changes impacting daily routines. To achieve this, we ask the assistance of ChatGPT 4 (OpenAI, 2023) to generate some scenarios where an individual's daily routine undergoes modifications due to a specific change in their life, such as starting a new, demanding job, etc. By employing ChatGPT for this task, we aim to create scenarios that are free from biases which might be consciously or unconsciously introduced by human creators. This approach ensures a diverse and unbiased array of scenarios, reflecting a wide range of real-life situations that individuals might encounter. The prompt we used for generating these scenarios is as follows:

> *Create a scenario that reflects a personality type accustomed to conducting daily activities at home. Typically, this routine includes: a 90% chance of taking morning medicine, 74% for midday medicine, 65% for breakfast, 48% for lunch, 86% for dinner, 20% chance of forgetting to cook, 20%*

	1st	2nd	3rd (Noise)	4th	5th	6th	7th	8th	avg	std	lower	upper
Morning efficiency	0.0639	0.0656	**0.1189**	0.0460	0.0560	0.0708	0.0405	0.0577	0.0649	0.0240	0.0334	0.0870
Medication non-adherence (Morning medicine)	0	0.1411	**0.4098**	0.1473	0.0754	0.1533	0	0.0832	0.1263	0.1299	0	0.2872
Medication non-adherence (Midday medicine)	0.0759	0.2308	**0.5674**	0.0897	0.1648	0.2865	0.1788	0.1791	0.2216	0.1555	0	0.3928
Meal preparation omission (Breakfast)	0.1817	0.2995	**0.5535**	0.1657	0.2269	0	0.1428	0.1429	0.2141	0.1613	0	0.3983
Meal preparation omission (Lunch)	0	0	0.5765	0.1988	0.3345	0.0911	0.2525	0.2825	0.2170	0.1923	0	0.6363
Meal preparation omission (Dinner)	0.3834	0.1997	0.3180	0.1927	0.0881	0.0886	0.3402	0.0730	0.2105	0.1239	0	0.6762
Clean dishes (Breakfast)	0.3834	0.4204	0.1162	0.3418	0.3471	0.1545	0.4366	0.4417	0.3302	0.1264	0.1008	0.6187
Clean dishes (Lunch)	0.6873	0.2128	0.0434	0.2217	0.8339	0.4641	0.1383	0.3034	0.3631	0.2767	0	1.008
Clean dishes (Dinner)	0.4641	0.3899	0.1778	0.3325	0.2703	0.4642	0.5108	0.5540	0.3954	0.1281	0.0786	0.7142

Table 6.6: The probabilities of properties in different periods, including a period (3rd) with an intentionally disrupted random daily routine. Probabilities exceeding the computed upper and lower bounds are highlighted in bold.

> *chance of forgetting to wash dishes, and a 50% chance of delaying dishwashing. The number of activities between eating and washing dishes varies from 1 to 3, as does the number of activities between waking up and cooking breakfast. The number of total activities per day ranges from 35 to 40. Develop a scenario in which this individual's behaviour changes due to specific life events or conditions during one of the eight two-week periods in the simulation. The probabilities of their routine behaviours should alter in this period due to these events or conditions, and return to regular in the other periods. Please detail the changes in probabilities and the reasons behind these adjustments.*

We use ChatGPT 4 to generate our simulated scenarios. In our experiment, we analyse two scenarios that are randomly selected to serve as examples. The first scenario involves an individual who is recovering from a minor surgery. In this scenario, the individual's daily routine in period 5 is significantly impacted. Due to the recovery from surgery, the individual needs to take additional medication, increasing the likelihood of morning and midday medication. Post-surgery recovery often leads to a reduced appetite, hence the decreased likelihood of eating all three meals. The recovery process might lead to increased fatigue or physical discomfort, making it more likely for the individual to forget or delay tasks like cooking and washing dishes. Post the recovery period, the routine gradually returns to its normal pattern. The probability of the behaviours in the affected period is below:

- The probability of taking morning medication increases to 0.95 (from 0.9) due to the additional attention of post-surgery medication.
- The probability of having breakfast decreases to 0.45 (from 0.65) because of a reduced appetite post-surgery.
- The probability of taking midday medicine increases to 0.85 (from 0.74) due to the need for consistent medication post-surgery.
- The probability of having lunch decreases to 0.3 (from 0.48) due to a decreased appetite and perhaps more time spent resting.
- The probability of having dinner decreases to 0.75 (from 0.86) because of the ongoing reduced appetite.
- The probability of not cooking: increases to 0.4 (from 0.2) as the individual might be more tired or in discomfort.
- The probability of not washing dishes increases to 0.4 (from 0.2) due to reduced mobility or energy levels.
- Number of activities between waking up and cooking breakfast: decreases to 1-2, mainly focusing on recovery-related activities.
- Number of total activities: decreases to 33-38.

Another scenario involves an individual who has recently adopted a puppy. The addition of the new pet brings joy but also changes the probability of their regular activities in the third period. The excitement and responsibility of caring for a puppy in the morning led to a slight neglect of personal routines like medication and breakfast. Puppy care requires time and attention, leading to a more erratic schedule and possibly skipping lunch. After a day filled with puppy-related activities, the individual might delay chores like cooking and dishwashing due to exhaustion or being preoccupied with the puppy. The individual might find mealtime a relaxing break from a day of puppy duties, leading to an increased probability of having dinner. After this period, as the individual and the puppy settle into a routine, the probabilities gradually return to their normal pattern. The set of probabilities during the third period is as follows:

- The probability of taking morning medicine decreases to 0.8 (from 0.9), due to the distraction and extra time spent with the puppy in the morning.

- The probability of having breakfast increases to 0.8 (from 0.65) as the individual might feel hungrier after walking the dog.
- The probability of taking midday medicine decreases to 0.6 (from 0.74) due to a disrupted routine caused by puppy care.
- The probability of having lunch decreases to 0.3 (from 0.48) as the individual might be busy training the puppy or taking it for a walk.
- The probability of having dinner increases to 0.95 as the individual might find mealtime a relaxing break from puppy duties.
- The probability of not cooking increases to 0.4 (from 0.2) due to tiredness from caring for the puppy.
- The probability of not washing dishes increases to 0.3 (from 0.2), and delaying washing dishes increases to 0.75 (from 0.5) due to the additional time spent on puppy care.
- The activities between eating and washing dishes are now 2-4 activities, previously 1-3 due to additional activities including cleaning up and feeding the puppy.
- The number of activities between wake up to breakfast is now 2-4 activities, previously 1-3 as adding more activities of walking the dog, feeding and playing.

The detailed parameters of our simulation model to mimic these two scenarios are detailed in Table 6.7. The assessed probability for each property across different periods in these two scenarios is presented in Table 6.8 and Table 6.9, respectively.

Table 6.8 presents that in the minor surgery scenario, due to the decreased number of activities between *wake up* and *having breakfast*, it shows an increased morning efficiency. As the probability of not cooking is increased from 0.2 to 0.4 in the simulation model, the result demonstrates a corresponding increase in the verified probabilities of meal preparation omission properties across all three meals. Specifically, the probability of skipping cooking breakfast is notably higher compared to other typical periods. The probabilities related to medication non-adherence in the fifth period are relatively lower compared to the average probability, aligning with the increased probability of having medication designed in the simulation model. These insights demonstrate that our approach can effectively capture the behaviour change and identify significant

	Typical periods	Noise period of minor surgery	Noise period of puppy adoption
Take morning medicine	0.9	0.95	0.8
Take midday medicine	0.74	0.85	0.6
Have breakfast	0.65	0.45	0.8
Have lunch	0.48	0.3	0.3
Have dinner	0.86	0.75	0.95
Not cooking	0.2	0.4	0.4
Not washing dishes	0.2	0.4	0.3
Delay washing dishes	0.5	0.5	0.75
Range of activities between eating and washing dishes	1 to 3	1 to 2	2 to 4
Range of activities between wake up and cook breakfast	1 to 3	1 to 2	2 to 4
Range of total activities	35 to 40	33 to 38	35 to 40

Table 6.7: Generated parameters of the simulation model in typical periods and intentionally disrupted periods in different scenarios.

	1st	2nd	3rd	4th	5th (Noise)	6th	7th	8th	avg	std	lower	upper
Morning efficiency	0.0699	0.0776	0.0734	0.0866	**0.1143**	0.0433	0.0699	0.0905	0.0657	0.0252	0.0383	0.1048
Medication non-adherence (Morning medicine)	0	0.1532	0	0.2158	0	0	0.1426	0.0771	0.0726	0.0870	0	0.3631
Medication non-adherence (Midday medicine)	0.0840	0.2978	0.1538	0.1725	0.1015	0.0799	0.1788	0.1791	0.1559	0.0711	0	0.3015
Meal preparation omission (Breakfast)	0.2727	0.2003	0.2269	0.1650	**0.6000**	0	0.1428	0.1429	0.2188	0.1726	0	0.3816
Meal preparation omission (Lunch)	0	0.2956	0.3345	0.1417	0.3360	0.1319	0.2525	0.2825	0.2218	0.1192	0	0.5544
Meal preparation omission (Dinner)	0.1924	0.1666	0.0881	0.1828	0.4513	0.3074	0.3402	0.0730	0.2252	0.1306	0	0.5685
Clean dishes (Breakfast)	0.4735	**0.6038**	0.3471	0.4354	0.3406	0.3508	0.4366	0.4417	0.4287	0.0874	0.2002	0.5993
Clean dishes (Lunch)	0.3425	0.5733	0.8339	0.4370	0.0211	0.4641	0.1383	0.3034	0.3892	0.2529	0	0.8353
Clean dishes (Dinner)	0.4048	0.3632	0.2703	0.3190	0.2974	0.4642	0.5108	0.5540	0.3980	0.1038	0.0702	0.7192

Table 6.8: The probabilities of properties in different periods in the minor surgery scenario, including a period (5th) with an intentionally disrupted routine due to minor surgery. Probabilities exceeding the computed upper and lower bounds are highlighted in bold.

	1st	2nd	3rd (Noise)	4th	5th	6th	7th	8th	avg	std	lower	upper
Morning efficiency	0.048	0.0495	0.0453	0.083	0.0599	0.0668	0.0721	0.0541	0.0598	0.0132	0.0206	0.0966
Medication non-adherence (Morning medicine)	0.0665	0.0824	0.3183	0	0.0772	0	0.2171	0.224	0.1232	0.1161	0	0.4723
Medication non-adherence (Midday medicine)	0.0689	0.1696	**0.3699**	0.1647	0	0	0.097	0.1714	0.1302	0.1199	0	0.3476
Meal preparation omission (Breakfast)	0.2300	0.3987	0.4410	0.0909	0.2213	0.1249	0.1995	0.1225	0.2286	0.1286	0	0.4940
Meal preparation omission (Lunch)	0.127	0.5052	0.3943	0.1662	0.1249	0.3998	0	0.1245	0.2302	0.1778	0	0.8020
Meal preparation omission (Dinner)	0.3839	0.0846	**0.4657**	0	0.1021	0.1565	0	0.1594	0.1690	0.1704	0	0.4436
Clean dishes (Breakfast)	0.2331	0.3286	0.2567	0.2017	0.4632	0.2653	0.3095	0.5065	0.3206	0.1096	0.0836	0.5294
Clean dishes (Lunch)	0.3773	0.3642	0.6014	0.1769	0.6481	0.8068	0.5083	0.3856	0.4836	0.1983	0.0155	0.9717
Clean dishes (Dinner)	0.3449	0.4409	0.1754	0.3201	0.424	**0.7706**	0.4779	0.5502	0.4380	0.1759	0.1028	0.7319

Table 6.9: The probabilities of properties in different periods in the puppy adoption scenario, including a period (3rd) with an intentionally disrupted routine due to the impact of adopting a puppy. Probabilities exceeding the computed upper and lower bounds are highlighted in bold.

shifts in behaviours. Moreover, the verified probability of property related to cleaning breakfast dishes within 3 steps in the second period shows a significantly high probability, indicating a more efficient routine. This is attributed to inherent variability allowed within the probability model.

Similarly, the Isolation Forest is applied to identify the period diverging from the individual's typical behaviour patterns, based on the features of normality and abnormality of property probabilities within each period. The result confirms that the fifth period is an outlier, consistent with the period intentionally designed for disruption.

Table 6.9 presents the result in the scenario of puppy adoption. The additional activities between waking up and having breakfast in the third period lead to a lower probability of morning efficiency, aligning with the expectation of decreased morning efficiency. This period also exhibits a significantly higher probability of not taking midday medicine before lunch, in line with the scenario's design to reduce the likelihood of midday medication intake, as detailed in Table 6.7. Regarding the meal preparation omission properties, i.e., evaluating the probability of not cooking meals, the third period sees an increase in the probability of not cooking for all meals, surpassing the average across periods. This is particularly pronounced for dinner, where the probability of not cooking is markedly high, identifying it as an outlier. Conversely, for cleaning dishes within three steps after the meal, the third period records probabilities below the average, with the exception of lunch. This exception is attributed to the probabilistic model's allowance for inherent variability. More precisely, during this period, the individual had lunch on five days out of 14 days, and on each of these 5 days, the activity of washing lunch dishes followed. Notably, on 3 of these occasions, the dishing washing activity occurred immediately after having lunch. This is a rare case, given the probability of delaying washing dishes is 0.75. The expected probability of cleaning lunch dishes within 3 steps after having meal is expected to be lower. Interestingly, the corresponding value of cleaning dinner dishes in the sixth period exceeds the upper bound, indicating a more efficient cleaning routine.

Similarly, the Isolation Forest is applied in the result to identify the deviating period in this scenario. The outcome of the Isolation Forest confirms that the third period stands out as an outlier, in alignment with the period intentionally set for disruption.

After evaluating the ability of our method to identify periods with behavioural changes, we next investigate the extent of behaviour changes our method can detect by conducting a sensitivity analysis.

6.6.3.2 Sensitivity analysis

In order to evaluate the extent of behaviour changes our method can detect, we conduct a sensitivity analysis based on synthetic data, which allows us to intentionally introduce varying degrees of changes. This helps us to understand how responsive our method is to varying extents of behaviour changes.

We use the example of morning medication non-adherence property to demonstrate this analysis. Following the same procedure as outlined in Section 6.6.3.1, we simulate both typical daily routines and those with deliberate disruptions, as noisy routines. We generate data for eight periods, with each period representing one month (30 days) of data. Among these eight periods, one period is designed to show intentional disruption in daily behaviour against the seven periods of typical daily routines.

In simulating the typical daily routines, we maintain the probability of taking morning medication at 0.9, consistent with the typical routine probability outlined in Section 6.6.3.1. To simulate intentional disruptions, we reduce this probability by 5% at each step. For each decrement in probability, we generate 50 sets of data to conduct a quantitative analysis of the probability change.

Similarly, we construct the user activity model for each period and compute the probability of the morning medication non-adherence property. The probabilities of the intentionally disrupted period for each change step are visualised in Figure 6.10. It shows that as the probability of taking morning medication decreases, the probability of morning medication non-adherence probability increases, aligning with expectations. Therefore, the greater the change in probability during the noisy period, the easier it becomes to identify the probability in this noisy period as an outlier.

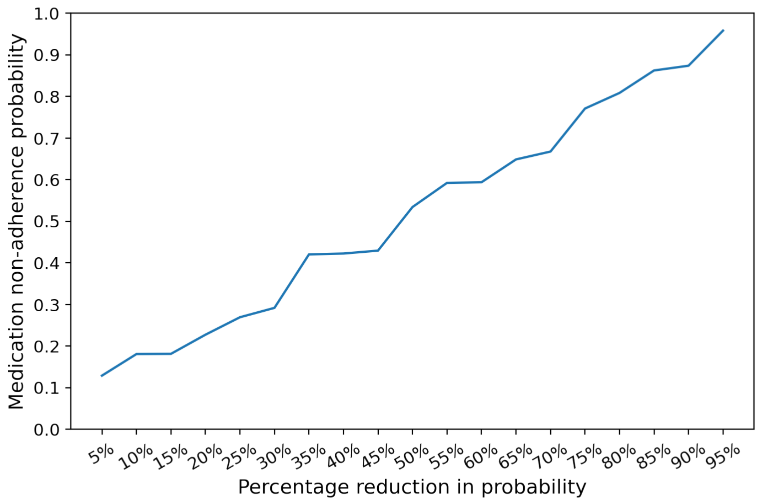


Figure 6.10: Probability of morning medication non-adherence property in the intentionally disrupted period for each change step.

Outliers are detected using the IQR method (see Section 6.6.2.2). The sensitivity of our approach is determined by its ability to accurately identify the probability of the property in the intentionally disrupted period as an outlier. Therefore, we calculate the sensitivity by equation (6.10), where TP denotes true positives, meaning the instances where the probability in the intentionally disrupted period is correctly identified as an outlier, and FN represents false negatives, indicating instances where the probability of noisy period is incorrectly identified as not an outlier. For each decrement step, we conduct 50 experiments, represented as N in equation (6.10), and the overall sensitivity is calculated as the average value from these experiments.

$$Sensitivity = \frac{1}{N}\sum_{1}^{N}\frac{TP}{TP+FN} \tag{6.10}$$

The result of overall sensitivity for each change step is visualised in Figure 6.11. It shows that as the probability of taking morning medication decreases, there is a sharp rise in sensitivity. When the percentage of probability change hits 25%, which is from 0.90 to 0.68 the sensitivity reaches a level of 0.7. Sensitivity reaches 1, indicating perfect sensitivity, when the percentage of probability change is at or exceeds about

35%, i.e., from 0.90 to 0.59. This suggests that any probability change above 35% will consistently be identified correctly as an outlier.

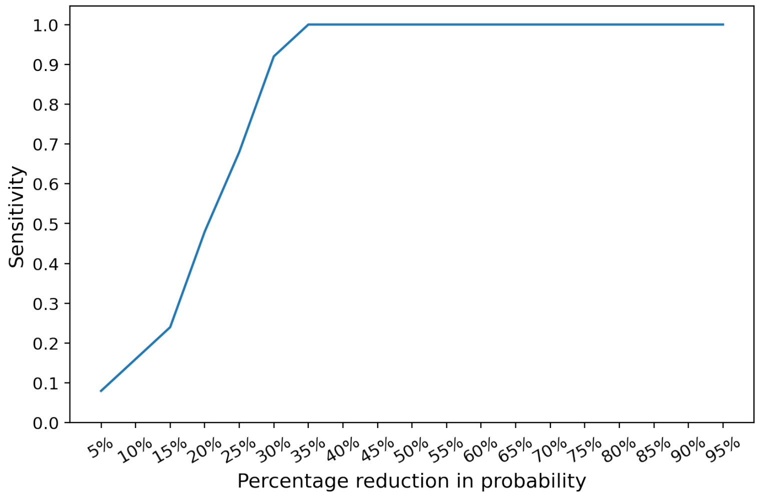


Figure 6.11: Overall sensitivity for each change step.

6.6.3.3 Case study on a real-life dataset

Following the evaluation of our methods based on synthetic datasets, we proceed to perform an analysis on a real-life dataset to explore the applicability of our methods. However, a notable change in this phase is the absence of ground truth regarding which probabilities of each property are considered outliers and which periods are abnormal. This limitation constrains our ability to precisely evaluate the outliers we identified. Therefore, our analysis focuses on observing changes in daily behaviour over time through measured probabilities of different properties in different periods. This provides us with insights into shifts in daily behaviour over time in real-world contexts.

We use the CASAS datasets described in Section 6.3. Following the data preprocessing procedure in Section 6.4, we obtain the data labelled with activities. In this analysis, we continue to use the data from the same individual (*hh102*) that was discussed in Section 6.5.4 to demonstrate the result. The *user activity model* is constructed based on the sequence of activities in the processed data.

In this dataset, there are 33 types of daily activities, leading to the creation of the *user activity model* that comprises 33 states. The dataset contains data spanning over 2 months from June 15, 2011 to August 15, 2011. First, we divide this timeframe into weekly periods, so that 8 periods of data are obtained. We then construct the user activity model for each of these separate periods. We visualise the activity transition matrix in the user activity model of the first week in Figure 6.12. The y-axis of this figure represents the current activity, the x-axis represents the next activity, and the value represents the transition probability from the current activity to the next activity. For example, it shows this individual has probability 1 to wash lunch dishes after having lunch.

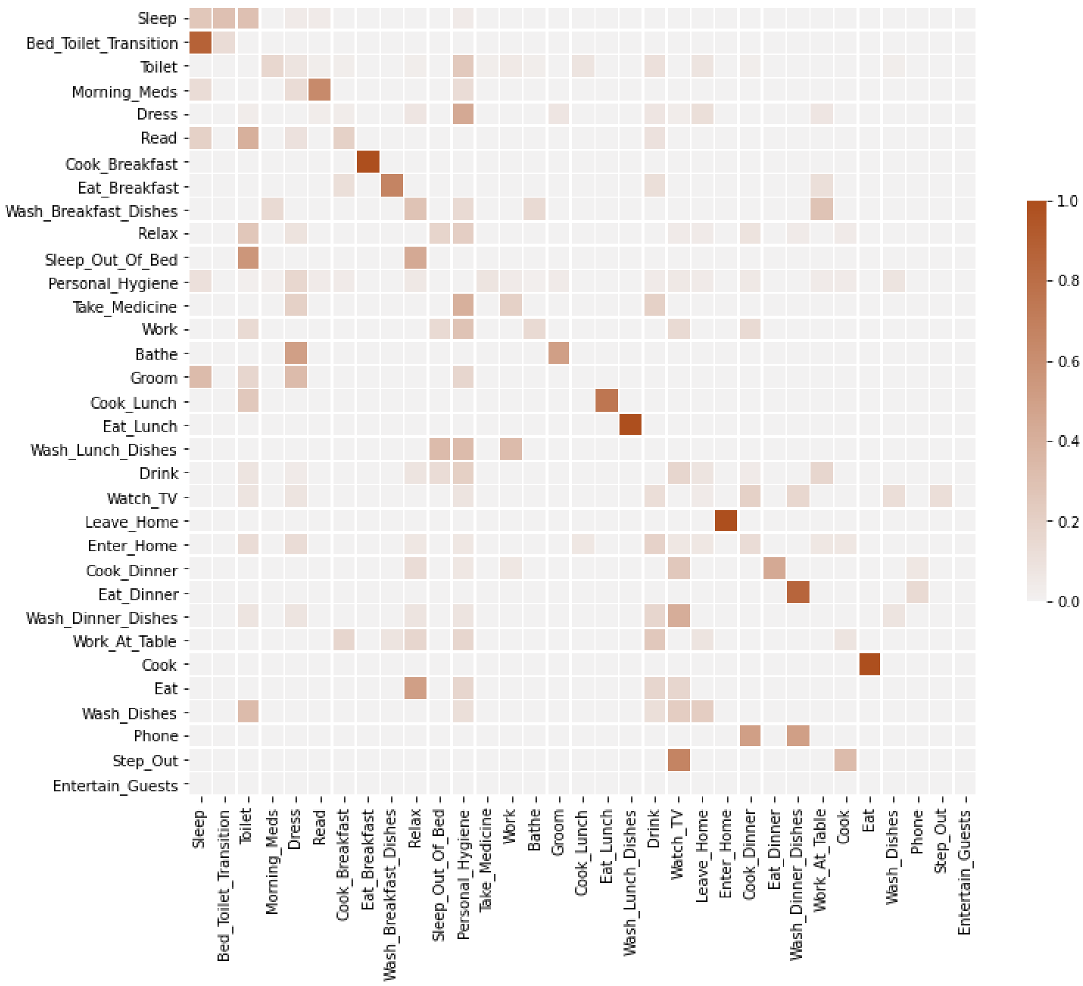


Figure 6.12: Visualisation of activity transition matrix in the user activity model of the first week.

With the constructed user activity model for each period and modelled properties, we measure the probability of each property in each period. The result is shown in Table 6.10. The probabilities exceeding the calculated lower and upper bounds are emphasised in bold.

The result shows there is a high probability that the individual tends to forget their midday medication before lunch in the 6th period. This observation suggests potential disruptions or shifts in the individual's midday routine, possibly due to increased ac-

	1st	2nd	3rd	4th	5th	6th	7th	8th	avg	std	lower	upper
Morning efficiency	0.0253	0.0329	0.0053	0.0178	0.0059	0.0191	0.0229	0.0300	0.0199	0.0102	0.0000	0.0439
Medication non-adherence (Morning medicine)	0.4401	0.3919	0.3967	0.1478	0.3323	0.3970	0.3700	0.3307	0.3508	0.0896	0.2346	0.4941
Medication non-adherence (Midday medicine)	0.4013	0.3433	0.3707	0.1695	0.4252	**0.5780**	0.3106	0.4206	0.3774	0.1158	0.2052	0.5517
Meal preparation omission (Breakfast)	0.0000	0.0000	0.0000	0.0000	0.0000	**0.1274**	**0.1346**	0.0000	0.0327	0.0607	0.0000	0.0796
Meal preparation omission (Lunch)	0.0000	0.0000	**0.1854**	0.0000	**0.1669**	0.0000	0.0000	0.0000	0.0440	0.0817	0.0000	0.1043
Meal preparation omission (Dinner)	0.0000	0.0000	0.0000	0.0000	**0.0510**	0.0000	**0.0847**	0.0000	0.0170	0.0327	0.0000	0.0319
Clean dishes (Breakfast)	0.8571	0.6364	0.4200	0.0000	0.8176	0.6724	0.4073	0.7682	0.5724	0.2858	0.0000	1.0000
Clean dishes (Lunch)	1.0000	0.5000	0.2500	0.5000	0.6034	0.8000	1.0000	0.0000	0.5817	0.3505	0.0000	1.0000
Clean dishes (Dinner)	1.0000	0.7482	0.5689	0.4826	0.7880	0.7126	0.4850	0.5783	0.6705	0.1768	0.2326	1.0000

Table 6.10: The probability of properties in 8 periods, each period containing one week of data. Probabilities exceeding the computed upper and lower bounds are highlighted in bold.

tivities, changes in the daily schedule, or simply forgetfulness, underscoring the need for interventions or reminders to enhance medication non-adherence.

During the 6th and 7th periods, a high probability of breakfast preparation omission shows this individual is more likely to not cook breakfast during these two periods. This pattern might reflect changes in morning priorities, possibly due to early morning commitments, or a shift towards opting for quicker, possibly less nutritious breakfast options.

A similar pattern is observed with probabilities related to dinner preparation omission in the 5th and 7th periods, indicating a decrease in cooking evening meals. This could be indicative of lifestyle changes, such as increased evening activities, a preference for social dining experiences outside the home, or a higher reliance on pre-prepared meals.

There are two highlighted probabilities related to meal preparation omission in the 5th and 7th periods. It reveals a high probability of the individual opting not to cook lunch and dinner during the 5th period, while in the 7th period, the individual is likely to skip preparing breakfast and dinner. These patterns indicate a shift in the individual's meal preparation habits, possibly due to various factors, such as increased personal commitments, or a preference for convenience.

The probabilities associated with dishwashing after meals within three steps ex-

	1st period	2nd period	3rd period	4th period	avg	std
Morning efficiency	0.0255	0.0123	0.0138	0.0269	0.0196	0.0076

Table 6.11: The probabilities of the morning efficiency property in 4 periods.

hibit significant fluctuations, ranging between 0 and 1, highlighting variability in daily routines. A probability of 0 indicates that the individual did not clean dishes within three steps after meals, whereas a probability of 1 indicates that the individual cleaned dishes immediately following meals. For instance, a high probability observed in the 1st period suggests adherence to a disciplined schedule for dish cleaning post-meals. Conversely, in the 4th period, the individual shows a probability 0 of washing dishes after breakfast. This means the individual did not clean dishes after breakfast within 3 steps, which could suggest a notably busier or more occupied schedule during that period. Similarly, the individual did not clean dishes within 3 steps after lunch during the 8th period.

Next, we conduct an analysis at a different level of granularity. The available dataset is segmented into 4 distinct periods, each spanning a duration of two weeks. Given the limited number of periods and lack of ground truth of outlier probabilities of each property and abnormal periods, we do not calculate lower and upper bounds based on the IQR method. Instead, we visually represent the probabilities associated with each period, which allows us to effectively illustrate the shifts in probabilities across time. Similarly, the user activity models are constructed for each of these separate periods. We measure the probability of each property in each period.

For **morning efficiency properties**, the number of steps from waking up to having breakfast is set as 3 for our analysis. The result of 4 periods is shown in Table 6.11. The result is visualised as a line chart, shown in Figure 6.13. This visualisation reveals a consistent morning routine throughout the four periods, as evidenced by a low standard deviation of 0.0076.

For **medication non-adherence** properties, we take into account two types of medicine, i.e., morning and midday medicines. Therefore, this type of property will encompass measuring probabilities related to not taking morning medicine before having breakfast, as well as not taking midday medicine before having lunch. The result of different periods for two types of medicine is shown in Table 6.12. The line chart is shown in Figure 6.14. The 2nd period displays the lowest probabilities regarding medication, suggesting that in most instances, the individual tended to take their medicine

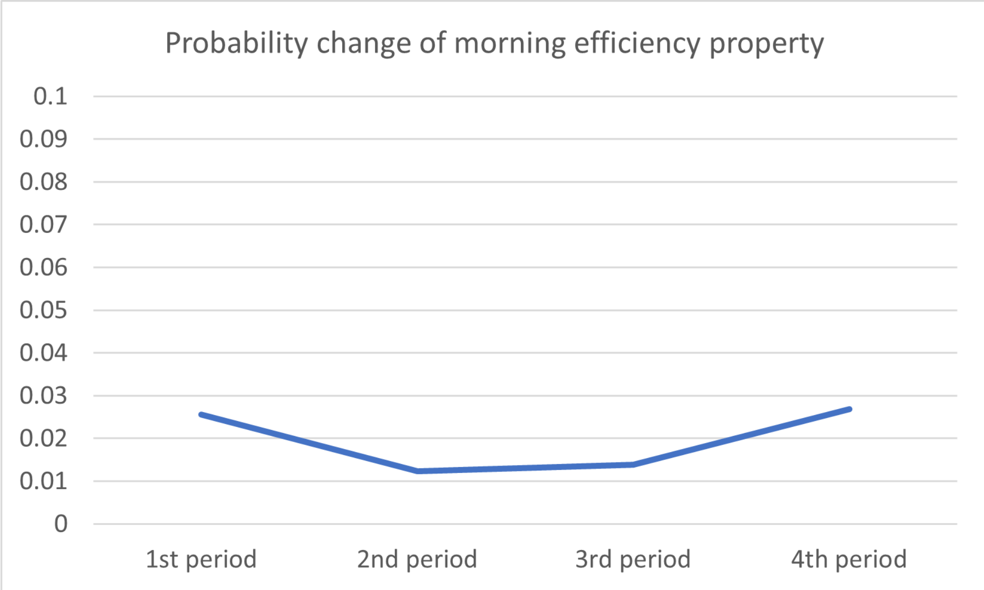


Figure 6.13: Probabilities shift of morning efficiency property over 4 periods.

prior to meals. Conversely, the 3rd period shows the highest probability, indicating that the individual was more likely to not take their medicine before meals. These insights are in harmony with the findings presented at a weekly granularity in Table 6.10. Specifically, the 2nd period in biweekly analysis corresponds to the 3rd and 4th periods at weekly granularity, showing lower probabilities of skipping medication before meals. Similarly, the 3rd period in biweekly analysis corresponds to the 5th and 6th periods in weekly granularity, indicating an increased probability of not taking medication before meals.

	1st period	2nd period	3rd period	4th period	avg	std
Morning medicine	0.4098	0.2867	0.4061	0.3416	0.3611	0.0507
Midday medicine	0.3939	0.3138	0.4689	0.3170	0.3734	0.0637

Table 6.12: The probability of medication non-adherence properties in four periods.

For **meal preparation omission** properties, we consider 3 types of meals: breakfast, lunch and dinner. The result of probabilities of these three properties in different periods is shown in Table 6.13. The change of probabilities over four periods is shown in a line chart in Figure 6.15. The result shows the highest probability that the individual ate lunch without cooking it beforehand in the 2nd period. There is a strong likelihood that the individual did not cook before meals in the 3rd period, signifying a noticeable deviation from other periods. This insight aligns with the findings at the

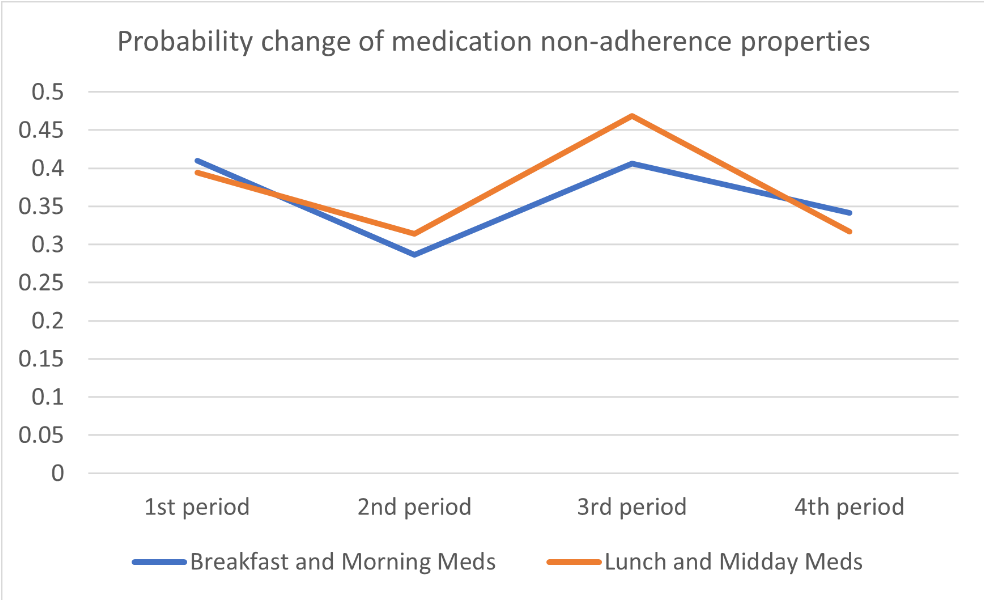


Figure 6.14: Probabilities change of properties related to medication non-adherence over four periods.

weekly granularity in Table 6.10, where the 5th and 7th periods show an increased probability of skipping cooking meals.

	1st period	2nd period	3rd period	4th period	avg	std
Breakfast	0	0	0.1257	0	0.0314	0.0544
Lunch	0	0.1357	0.0859	0	0.0554	0.0581
Dinner	0	0	0.0341	0.0360	0.0175	0.0175

Table 6.13: The probability of meal preparation omission properties in four periods.

For the properties related to **clean dishes consistency**, the number of steps from meal to washing dishes is set as three for our analysis. The result is shown in Table 6.14. We visualise the result in a line chart, shown in Figure 6.16. The 2nd period has the lowest probabilities for all three meals. This suggests that the individual often postponed dishwashing after meals, which could imply a busier schedule or a reduction in discipline during these times.

The 3rd period is particularly noteworthy, with an increased probability across several properties, including skipping both morning and midday medication, skipping preparation of breakfast and lunch, and increasing the efficiency of cleaning dishes. This may indicate a temporary period when the individual has increased personal commitments and efficiency with less time at home or a preference for convenience

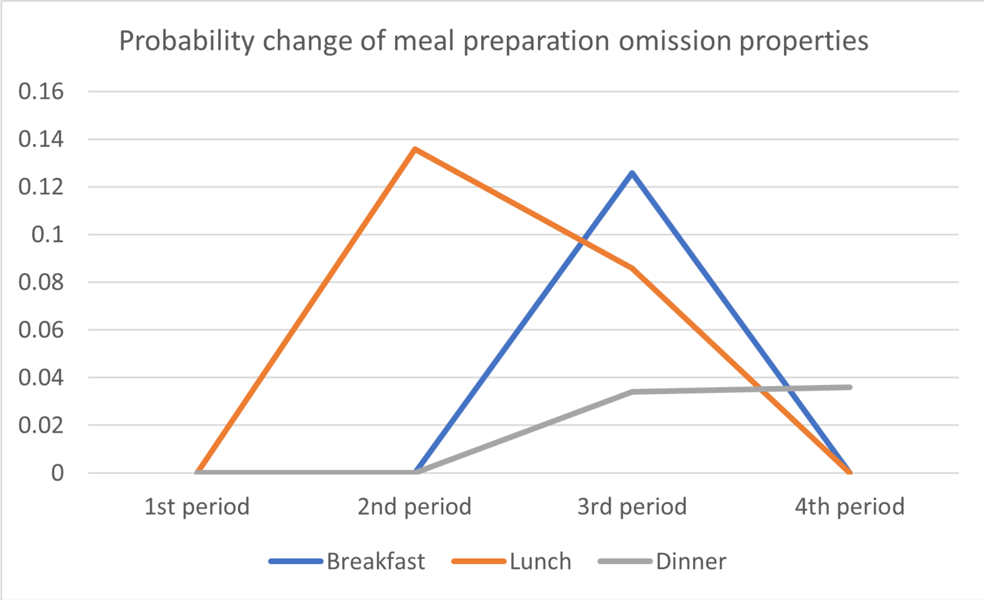


Figure 6.15: Probabilities change of properties related to meal preparation omission over four periods.

options. Conversely, the 2nd period exhibits better adherence to morning and midday medication, consistency of meal preparation, except lunch, and decreased efficiency of cleaning dishes. It perhaps indicates a more structured routine or a period with an enhanced focus on health and meal preparation.

In conclusion, our comprehensive analysis of real-life datasets provides insightful findings into daily behavioural changes in real-world contexts. These analyses highlight specific periods where significant deviations in behaviours occur, pointing towards potential lifestyle changes or adjustments in daily routines. For example, the changes in probabilities regarding skipping the preparation of key meals could have implications for the individual's nutritional intake. Identifying these patterns provides an opportunity for targeted interventions, such as meal planning assistance. By conducting evaluations at different levels of granularity, we have been able to maintain consistency in our outcomes, reinforcing the reliability of our observations. The consistency across our analytical approaches underscores the robustness of our methods and the validity of the identified behaviour changes.

	1st period	2nd period	3rd period	4th period	avg	std
Breakfast	0.7303	0.4023	0.6401	0.7921	0.6412	0.1481
Lunch	0.6679	0.4176	0.7159	0.4433	0.5612	0.1321
Dinner	0.8376	0.5459	0.7958	0.5828	0.6905	0.1277

Table 6.14: The probability of cleaning dishes properties in 4 periods.

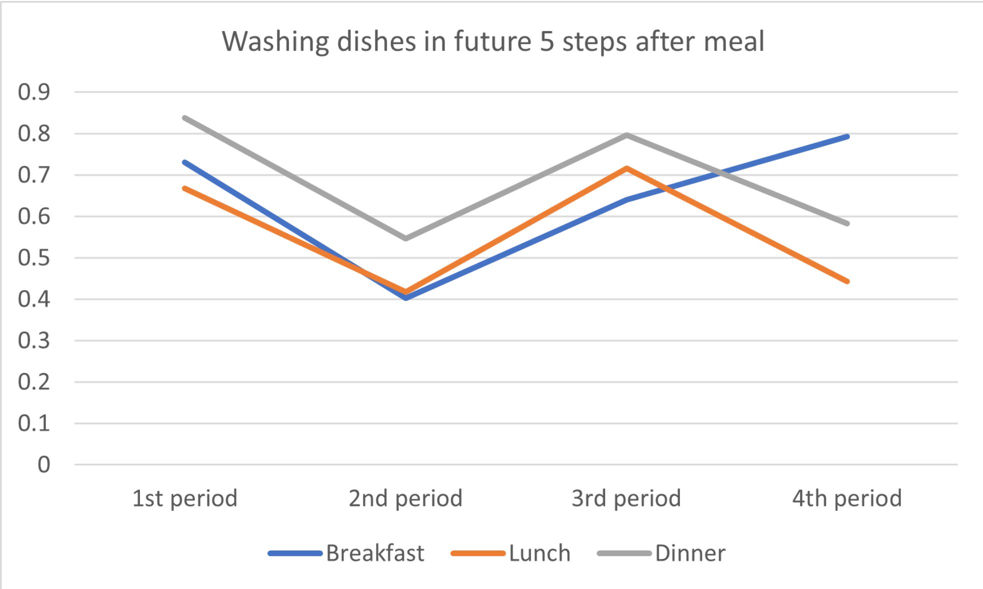


Figure 6.16: Probability change of properties related to clean dishes consistency over four periods.

6.7 Discussion

In this chapter, we present a method for identifying short-term deviations and long-term patterns of behaviour involving single activities. We also introduce a method for detecting changes of complex behaviours over time based on sensor collected ADL data.

Specifically, due to the dynamic nature of human behaviour and the variability between different people, we build personalised behavioural profiles and set an adjustable deviation threshold for each behaviour marker. This method ensures that the model is finely tailored to each individual's unique patterns and daily activities. We can modify the weights of each behaviour marker in order to obtain an aggregate score that reflects the importance of deviations in a particular day, for instance in terms of health outcomes. Moreover, we develop an interactive dashboard which visualises the various information of human behaviour, such as daily routine, potential deviations, trend of sleep duration, etc. The dashboard can provide both the person involved and their caregivers with key behavioural insights.

Furthermore, we present a probabilistic conformance checking-based method for detecting complex behaviour changes over time. We model complex behaviours as properties based on PCTL, which provides us with a formal way to model complex relationships between interdependent activities. We model daily routines of different periods as user activity models based on DTMCs to represent the sequence of daily activities. A probabilistic model checker is used to assess the probability of behaviour properties over different periods. We adopt a statistical algorithm to pinpoint the periods where behaviours exhibit outlier probabilities compared to other periods, which signifies a potential behavioural change.

To evaluate whether our method can accurately identify these changes, we conduct several experiments based on synthetic scenarios generated by ChatGPT. These experiments are designed to test the ability of our method to pinpoint intentionally disrupted periods as outliers. The results demonstrate our method can accurately detect these disruptions. Beyond this, it is important to assess how this method responds to varying degrees of behavioural changes. Therefore, we conduct a sensitivity analysis. This analysis helps us understand the method's responsiveness to different scales of behavioural changes, from subtle to significant changes.

Moreover, to ensure that our approach is effective in real-world settings, a case study on a real-life dataset demonstrates the effectiveness of our approach in observing behaviour changes in real environments.

In summary, our proposed methods address the challenges of the variability of individual routines by building personalised temporal profiles and user activity models for each individual. When detecting anomalies, we focus on different behaviour markers and properties, accounting for the varying sensitivity of activities. Additionally, to tackle the complexity of modelling behaviours, we employ logic formulas, enabling us to model complex behaviours in a formalised way.

This work underscores the significant value of data collected by sensors in providing continuous monitoring of human behaviours and health. Based on these captured real-time and continuous data, we have a deep understanding of normal and abnormal patterns in daily activities, which can be crucial for the early detection of health issues. Understanding these patterns can be beneficial for individuals, as it can provide them with their regular daily patterns and abnormal behaviours. Our work can help individuals actively monitor their own health and activities. For example, it helps them recognise when their behaviour deviates from their typical patterns, which might be indicative of potential health issues or the need for adjustments in their rehabilitation plan. The deviation scores of particular days have the potential to be used as behaviour performance scores for care givers when monitoring the daily living routine of an older adult at a glance. It can also promote proactive health management. For example, if an elderly person notices a decreasing trend in sleep duration, they might consult their healthcare provider to adjust their sleep hygiene practices or medication. Furthermore, these detected patterns and anomalies can also offer healthcare providers valuable information to tailor care plans more effectively.

Despite these contributions, future challenges still exist in dealing with inherent noise in the data and variability of individuals' routines. As we discussed in Section 6.1.1, different modalities of sensors have inherent limitations in accurately identifying specific activities. This introduces noises in terms of measuring activity intervals and modelling daily activity sequences. We therefore believe that future improvements in ADL data collection will bring better and more comprehensive insights. For example, integrating ambient sensors with wearable devices could collect more detailed information about activities. Through fusing data from different modalities, we could reduce some uncertainties and noise of data collected by single modality. Therefore, adapting these data into our proposed methods could provide more accurate insights into anomaly behaviours.

Moreover, wearable devices can serve as a complementary to ambient sensors, enabling the collection of data on activities conducted outdoors, such as exercise and walking. They also offer the capability to collect data related to vital health indicators, such as heart rate, blood pressure, etc. Such comprehensive data can be pivotal in understanding an individual's daily activities and health. By adapting these data into our analytic frameworks, we can develop a more nuanced understanding of how daily activities impact overall health and potentially detect early signs of health issues. These insights are crucial for preventing adverse health outcomes and enabling proactive interventions at an early stage.

As we discussed in Section 6.1.2, the limited amount of data also affects the result we obtained, because the smaller sample size may not fully capture the range and variability of typical behaviour patterns. Further experiments based on longer periods of data are likely to improve the quality of our approach and lead to new types of insights, particularly for identifying long-term patterns.

Furthermore, detected anomalies and behavioural changes can be further contextualised to the needs and lifestyles of the individual participants. We can gain a deeper understanding of what exactly causes these changes and anomalies. By tailoring the analysis to the specific contexts of each participant, the accuracy of the findings can be significantly enhanced. For example, customising anomaly detection methods based on an individual's environment, such as weather, and specific health conditions, such as mobility issues, can greatly enhance the relevance and accuracy of the insights derived from their behaviour patterns. For example, consider individuals with mobility issues, the layout of their home can significantly influence their activity patterns. An anomaly detection system could be tailored to understand the layout of the individual's home, such as the presence of stairs or the distance between critical areas like the bedroom and bathroom.

6.8 Conclusion

This chapter introduces methods for identifying temporal patterns and detecting deviations in unstructured domains, particularly focusing on the context of ADLs. The significance of our methods lies in their capacity to function within environments where expected behaviours are not clearly defined, compounded by high degrees of variability and complexity of human behaviours. Addressing this, we present a predictive method for the establishment of expected behaviour patterns in unstructured domains, and the subsequent detection of deviations from these patterns while accounting for the inherent variability of behaviours. We not only focus on identifying anomalies in behaviours with a single activity but also consider changes of complex behaviours that involve multiple interdependent activities. We evaluate our approach in both synthetic datasets to ensure its effectiveness in controlled scenarios and real-life datasets to ensure its applicability and practicability, and offer valuable insights in real environments.

In the next chapter, we will conclude the thesis and outline potential avenues for future research and exploration.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

In this thesis, we explore how process information can be incorporated to help interpret patterns and detect anomalies in different domains, spanning from structured domains, like manufacturing, all the way to unstructured ones, like ADLs. To leverage the process information in different domains, we are facing many challenges, such as the varying structure of processes in different domains, noisy and uncertain data, etc., as we discussed in Section 1.3.

We investigate different methods to address challenges that arise in different domains. Next, we briefly recapitulate the key contributions and limitations of our work.

7.1.1 Workflow recognition

In structured domains, we introduce a conformance checking-based approach for workflow recognition. This approach involves complex event processing for handling noisy and low-level event data collected by sensors, modelling workflows, and employing conformance checking to determine which workflow the generated data most closely matches. Our approach is designed to automatically identify the specific workflow that corresponds to the generated event sequences by sensors. This work not only enhances the capability of monitoring operational process, but also supports subsequent analysis leveraging process information, such as detecting process deviation in Chapter 4.

Despite these contributions, several challenges and constraints still remain to enhance its effectiveness and scalability. Our approach relies on the quality and granularity of generated data. Misrecorded, or improperly ordered events can lead to misidentification of workflows, particularly in scenarios where workflows are similar. This means that any errors in data collection can impact the outcomes of the approach.

7.1.2 Deviation detection over probabilistic events

In structured domains, we present a deviation detection approach over probabilistic events, named `ProbCost`. We extend the alignment-based conformance checking to function under probabilistic event data. One of the challenges in handling probabilistic data is that it is difficult to discern whether a detected deviation is caused by the uncertainty of event data or a real deviation. We introduce a weighted trace model to denote a probabilistic event trace and a weighted alignment cost function along with a parameter ε to control the level of confidence on the event log vs. process model. The evaluation results show that our approach can better tolerate uncertainties in event logs by leveraging process information, and achieve lower false positives and false negative rates in deviation detection.

The selection of the parameter ε of our approach relies on empirical evaluations on training data. Therefore, a limitation arises in scenarios where there are no available training data to guide the selection of an appropriate value of ε. Further exploration of the parameter's influence and how to select this value based on the domain knowledge would be useful.

7.1.3 Process-aware human activity recognition

In semi-structured domains, we propose a framework that incorporates process information to enhance the activity recognition performance of traditional machine learning models. We automatically discover the process information from data. We adopt our proposed `ProbCost` approach to achieve an alignment between process models and the probabilistic output of machine learning models. We can consider activity classes with lower probabilities but better align with process models to create the alignment. An experimental study is conducted to compare the performance of our approach with original deep learning models for activity recognition based on a dataset recording eating behaviour. The results show our approach can achieve better performance in both accuracy and macro F1-score metrics.

This work underscores the effectiveness of combining machine learning models with domain-specific process knowledge to enhance performance. By integrating these two elements, the approach not only leverages the predictive power of machine learning but also incorporates the knowledge that domain-specific processes provide.

When considering scaling this integration framework into other domains, one potential challenge lies in discovering and modelling process information given the varying degrees of process structures across different domains. Further evaluation of our framework on different scenarios will be key to broadening its applicability across a range of domains.

7.1.4 Anomaly detection in unstructured domains

In unstructured domains, we not only focus on detecting anomaly behaviours involving single activities but also consider complex behaviours involving multiple activities. Firstly, we propose a predictive approach for identifying temporal patterns based on historical data and an anomaly detection approach based on the identified patterns. Specifically, we extract daily behaviour markers according to the features of daily activities, which are high level representations of activities, encompass rich temporal information and are potentially linked to health indicators. A personalised behaviour profile is constructed based on the extracted behaviour markers. We introduce a deviation score which is tailored to personalised adjustments for different behaviour markers according to their relevance to health. These strategies effectively account for the variability of daily activities to different activities and different individuals.

Secondly, we propose a probabilistic model checking-based approach for detecting changes of complex behaviours. This approach focuses on behaviours that involve temporal dependencies between multiple activities, such as having medication before a meal. Focusing on these complex behaviours provides deeper insights into an individual's routine. We use DTMCs to model daily routines in different periods as user activity models. A set of behaviour properties modelled by temporal logic is introduced to assess the probabilities of behaviours by checking against the user activity model in different periods. We use a statistical method to identify the outliers of probabilities of the properties, marking them as deviations.

Our approaches are tested through comprehensive evaluations, spanning synthetic tests, sensitivity analyses, and real-world applications. These evaluations demonstrate the capability of our method to function reliably under a variety of scenarios, making it a valuable tool for detecting anomalies and behavioural changes.

This work provides insights into extracting daily patterns and abnormal behaviours.

Therefore, it can contribute to self-management and activity monitoring for individuals. The ability to detect changes in behaviour patterns helps both individuals and care givers to involve early, potentially preventing adverse health events.

Detected anomalies and behavioural changes from our work rely heavily on the quality of activity data collected by sensors. Enhancements in data collection and cleaning can make our approach provide more accurate insights. For instance, integrating multiple sensing modalities for data collection can refine our input data, which can lead to a more meaningful understanding of daily routines and accurate detection of anomalies. Besides, detecting anomalies in ADLs continues to face several challenges including limited data availability for analysing long-term patterns, lack of ground truth of anomalies for the evaluation of proposed methods in real scenarios, etc.

7.2 Directions of future work

Incorporating process information to interpret patterns and detect anomalies across various domains is a massive undertaking. This thesis focuses on specific objectives to demonstrate the value of process context in addressing these challenges and pave the way for future research. In this section, we discuss potential directions in which this thesis could be extended.

7.2.1 Efficient online workflow recognition

Although our proposed workflow recognition approach is capable of functioning in online scenarios, it may face potential performance challenges in situations where the process model is highly complex, or the event log contains a vast number of events. This is particularly crucial in domains where response time is a critical factor. Our approach is based on the alignment-based conformance checking algorithm, which, despite its effectiveness, needs large memory space and may incur significant computational delays when constructing alignment between complex process models and large event logs (Lee et al., 2018).

As part of future work, exploring strategies to enhance computational and memory efficiency would be invaluable. One promising direction involves extending the existing online conformance checking techniques, such as (van Zelst et al., 2019; Lee et al., 2020) with new methods for calculating the optimal alignment. For example, decomposing the process model into subprocesses allows for computing alignment on relevant parts of the process. By focusing on creating alignment on these segments rather than the entire model, it can reduce the overall computation time and memory usage. Computing decomposed alignment may result in local optimal problems, where the alignment is optimal for individual subprocesses but not for the entire process model. Therefore, it is also important to propose a new metric to better measure the overall fitness score by aggregating alignments between these subprocesses and the incomplete event log.

7.2.2 Deviation detection under uncertain data

Our current proposed approach, detailed in Chapter 4, can function under probabilistic events with a categorical probability distribution. A custom threshold parameter ε in the cost function is proposed to control the level of confidence on the event log.

One possible extension of the current approach is to introduce a nuanced method to managing confidence levels of different activities in the event log. Specifically, the adoption of a customisable threshold parameter within the cost function could be refined to vary across different activities. Incorporating domain-specific knowledge, it becomes possible to assign distinct threshold parameters that reflect our varying degrees of confidence in different activities in the event log. For example, in cases where we believe that the event log is a highly reliable indication of occurrences of certain activities, setting a higher threshold would indicate greater trust in the event log. Therefore, even if another activity might align better with the process model but is deemed less probable, it could be classified as a deviation. This extension could potentially further reduce the number of false positives and false negatives, and enhance the accuracy of deviation detection.

Moreover, considering other types of uncertainty in using conformance checking for deviation detection is a valuable direction. For example, we could explore other types of uncertainty of event data, e.g., repeating or missing events. Furthermore, the consideration of uncertainties in both processes and event data presents a promising research direction. One method to model uncertain processes is through the use of Markov Chains. In this method, each process step or transition is associated with a probability that reflects the likelihood of moving from one activity to another.

7.2.3 Process-driven activity recognition

Our current work, detailed in Chapter 5, underscores the effectiveness of incorporating process information in semi-structured domains to improve human activity recognition (HAR). It opens up avenues for further exploration of how varying degrees of process structures can be leveraged to improve the HAR.

A pivotal area of this exploration involves examining the various representation ways of process models and their implications in different scenarios. Process models can range from highly prescriptive, procedural models that describe the precise ordering of activities, to more flexible, declarative models that specify rules and constraints governing activities without strictly defining their order (Pichler et al., 2011). There are also hybrid models that combine elements of both procedural and declarative models (van Dongen et al., 2020). Future work can focus on developing methodologies for integrating these diverse process model representations into ML-based HAR systems, with the goal of enhancing their adaptability and accuracy. A promising research direction involves integrating hybrid process models (Di Ciccio and Montali, 2022), to model semi-structured processes. A novel framework or algorithm can be designed for reasoning ML outputs in the context of the constraints specified by hybrid process models.

Moreover, the research on integrating process information to enhance ML models offers broad opportunities to impact various domains, not only in the domains of HAR. This future work could involve developing methodologies for effectively capturing and integrating process information, exploring the interplay between process structures and ML predictions, and validating these models across diverse real-world scenarios. For example, in the healthcare domain, data-driven models designed to predict patient risks or the likelihood of hospital admissions can be significantly improved by incorporating semi-structured process information related to patient flows and care routines. We could investigate how patient treatment trajectories can be incorporated to enhance the predictions of patient risks. One possible approach is that process information can be directly integrated as additional features for training ML models. For instance, when predicting a patient's mortality, we can incorporate the patient's historical treatment paths as additional features.

7.2.4 Anomaly detection in unstructured domains

Anomaly detection in unstructured domains faces many challenges, due to there being no baseline for comparison, lack of ground truth, data limitations, etc., as we discussed in Section 6.7. Our work makes key contributions in this direction under some assumptions, but these challenges provide many worthwhile opportunities for further work beyond ours:

- **Building baseline by incorporating domain knowledge**: Collaborating with domain experts to manually review and validate identified patterns and deviations can provide a form of ground truth. One possible strategy is incorporating domain-specific knowledge, such as healthy living or fire security guidelines. These guidelines can be utilised to construct static constraints that define expected behaviours. This approach can provide us with a benchmark of normality and address the fundamental issue of ground truth scarcity. For example, we can identify anomalies where activities are deviating from specific guidelines, such as "do not leave cooking unattended on the hob or grill" (London Fire Brigade, 2024).

- **Data cleaning and enhancement by integrated sensing modalities**: To overcome the limitations of individual sensing technologies and enhance the accuracy of behaviour monitoring, future research could focus on integrating data from diverse sensors. By integrating data from motion sensors, which detect movement and location, with data from wearable sensors that can monitor physiological responses (e.g., heart rate, blood pressure), we can construct a more holistic and precise view of individual behaviours. This integrated approach helps mitigate the limitations of any single sensing technology, such as the ambiguity of motion sensors alone, and reduce the uncertainty of detected activities, such as the exact time of falling asleep.

- **Exploring the correlation between daily activities and health status**: With an enriched dataset that includes both activity and physiological data, we can further investigate the correlations between daily activities and various health outcomes. This could involve studying how specific patterns of movement relate to changes in physiological states or how deviations from typical activity patterns could signal emerging health issues. Understanding these correlations can significantly improve health monitoring and intervention strategies, making it possible to offer timely and targeted responses based on a person's unique activity profile and health status.

- **Continuous model refinement**: In order to capture the gradual behaviour changes, an iterative approach needs to be adopted to update the model continuously as new data becomes available. Combining models with varying levels of sensitivity also could provide a more comprehensive view of behaviour changes, where more sensitive models detect subtle behaviour changes and less sensitive models confirm longer-term trends.

- **Incorporating contextual analysis**: Discerning between temporary fluctuations and meaningful deviations requires a comprehensive understanding of contextual information. This necessity arises because human activities are inherently complex, and behavioural changes can be influenced by multiple factors, including weather changes, health issues, and other external conditions. One possible direction is enhancing models with the capability to analyse the context surrounding a behavioural change, such as analysing the correlation between behaviour changes and seasonal and contextual factors, like holidays, weather conditions, and environmental and societal changes. Here are some examples illustrating how such contextual information can be applied:

  - Weather conditions: For instance, during unusually hot weather, it is common for outdoor activities to decrease. By factoring in weather data, the model can adjust its expectations and not mistakenly flag reduced outdoor activity as an anomaly. Conversely, increased indoor activity during cold seasons can be similarly contextualised, preventing misinterpretations of behavioural data.
  - Health conditions: Changes in health status can lead to significant alterations in behaviours. Integrating health monitoring data, such as medication adherence, or physiological data from wearable devices, can provide important context. For example, if a person with arthritis has increased medication usage or reports higher pain levels, a corresponding decrease in physical activity could be expected.
  - Personal lifestyles: High levels of physical activity or unusual heart rate patterns might be normal for athletes in training. By incorporating their training schedules and physical demands, detecting anomalies and potential

interventions can be more appropriately targeted, and their conditioning and recovery can be better monitored.

7.2.5 Ethical considerations for deployment of sensory technologies

It is essential to address ethical implications associated with the deployment of sensory technologies, especially in personal homes for tracking personal behaviours. This issue is significant because, similar to how insurance companies use car sensor data to monitor driving behaviour and emotional statuses (Duncan, 2024), the collection and analysis of personal behaviour through sensors can be misused or taken out of context without proper safeguards. Such practices highlight the complexity of ethical concerns in the deployment of these technologies, where data can be used in ways that may not fully respect the privacy or intentions of the individuals involved. Recent research and initiatives have brought these issues to the forefront, emphasising the need for robust ethical frameworks in the application of these technologies (Chung et al., 2016; Fallatah et al., 2023). Although these aspects are beyond the scope of this thesis, they represent crucial considerations for the productive and ethical use of these techniques in practice. It is crucial to ensure that all data is collected with consent and handled in accordance with data protection policies to maintain the privacy of individuals. Furthermore, data collection and processing must be conducted with transparency, provenance, and clearly defined goals and outcomes to ensure that these technologies serve the welfare of individuals without compromising their autonomy or privacy.

7.3 Concluding remarks

Incorporating process information to recognise patterns and detect anomalies in human activities is a worthwhile but highly complex problem with numerous variables, settings, and goals. This thesis makes significant contributions in this direction and sets the stage for further research.

We categorise different domains based on the structure of processes into three types, spanning from structured and semi-structured to unstructured. We focus on specific objectives and present a set of methods to explore how process information can be captured and integrated to help interpret patterns and identify anomalies in different domains. Our work investigates how well-defined processes can guide workflow recognition for process monitoring and anomaly detection under uncertain data. We demonstrate the effectiveness of incorporating semi-structured or unstructured process information to enhance machine learning models for classification and recognising human behaviour patterns and changes.

Reflecting on the comprehensive research of integrating process information across various domains, several key lessons have been drawn that are instrumental for further research and practical application. One of the most critical lessons is the pivotal role of understanding and capturing process information plays in the effectiveness of anomaly detection systems. Interpreting the specific constraints of each domain enables more accurate and context-aware anomaly detection. For example, looking back to the example of using indoor localisation techniques for manufacturing monitoring, as discussed earlier, we highlight the importance of integrating process information to recognise workflows, navigate sensor data uncertainties and minimise false positive alerts concerning deviations from expected behaviour.

This example also emphasises that high-quality data is essential for effectively capturing patterns and enhancing anomaly detection accuracy. Similarly, in exploring the recognition of human activities and identifying behavioural changes, we encounter different challenges associated with various sensors, such as video and unobtrusive ambient sensors. Each sensor type comes with its own set of limitations and noise issues. Looking forward, the potential for enhancing and extending our research in the future with more high-quality data, such as integrating multiple sensing modalities for data collection, is substantial. This could lead to more accurate models and deeper insights across a range of applications, from industrial process monitoring, healthcare and behaviour monitoring to anomaly detection in manufacturing and daily activities.

Overall, we believe that this thesis serves as a foundational stone in advancing the understanding of recognising patterns and detecting anomalies in various domains from a process perspective.

Bibliography

Aalst, W. (1998). The Application of Petri Nets to Workflow Management. *Journal of Circuits, Systems, and Computers*, 8:21–66.

Adriansyah, A., van Dongen, B., and van der Aalst, W. (2011). Conformance Checking Using Cost-Based Fitness Analysis. In *2011 IEEE 15th International Enterprise Distributed Object Computing Conference*, pages 55–64.

Akila, K. and Chitrakala, S. (2019). Highly refined human action recognition model to handle intraclass variability & interclass similarity. *Multimedia Tools and Applications*, 78(15):20877–20894.

Albanese, A. M., Bartz-Overman, C., Parikh, MD, T., and Thielke, S. M. (2020). Associations Between Activities of Daily Living Independence and Mental Health Status Among Medicare Managed Care Patients. *Journal of the American Geriatrics Society*, 68(6):1301–1306.

Alberdi Aramendi, A., Weakley, A., Aztiria Goenaga, A., Schmitter-Edgecombe, M., and Cook, D. J. (2018). Automatic assessment of functional health decline in older adults based on smart home data. *Journal of Biomedical Informatics*, 81:119–130.

Alshammari, T. O., Alshammari, N. O., Sedky, M. H., and Howard, C. (2018). Evaluating machine learning techniques for activity classification in smart home environments. *Journal of information and communication convergence engineering*, 12:72–78.

Aminikhanghahi, S. and Cook, D. J. (2019). Enhancing activity recognition using CPD-based activity segmentation. *Pervasive and Mobile Computing*, 53:75–89.

Aminikhanghahi, S., Wang, T., and Cook, D. J. (2019). Real-Time Change Point Detection with Application to Smart Home Time Series Data. *IEEE Transactions on Knowledge and Data Engineering*, 31(5):1010–1023.

Amit, Y., Felzenszwalb, P., and Girshick, R. (2020). Object detection. *Computer Vision: A Reference Guide*, pages 1–9.

Anderson, J. L., Rosen, L. N., Mendelson, W. B., Jacobsen, F. M., Skwerer, R. G., Joseph-Vanderpool, J. R., Duncan, C. C., Wehr, T. A., and Rosenthal, N. E. (1994). Sleep in fall/winter seasonal affective disorder: Effects of light and changing seasons. *Journal of Psychosomatic Research*, 38(4):323–337.

Arifoglu, D. and Bouchachia, A. (2019). Detection of abnormal behaviour for dementia sufferers using Convolutional Neural Networks. *Artificial Intelligence in Medicine*, 94:88–95.

Asim, Y., Azam, M. A., Ehatisham-ul-Haq, M., Naeem, U., and Khalid, A. (2020). Context-Aware Human Activity Recognition (CAHAR) in-the-Wild Using Smartphone Accelerometer. *IEEE Sensors Journal*, 20(8):4361–4371.

Baier, T., Di Ciccio, C., Mendling, J., and Weske, M. (2015). Matching of Events and Activities - An Approach Using Declarative Modeling Constraints. In *Enterprise, Business-Process and Information Systems Modeling*, Lecture Notes in Business Information Processing, pages 119–134, Cham. Springer International Publishing.

Baier, T., Di Ciccio, C., Mendling, J., and Weske, M. (2018). Matching events and activities by integrating behavioral aspects and label analysis. *Software & Systems Modeling*, 17(2):573–598.

Baier, T., Rogge-Solti, A., Weske, M., and Mendling, J. (2014). Matching of Events and Activities - An Approach Based on Constraint Satisfaction. In *The Practice of Enterprise Modeling*, Lecture Notes in Business Information Processing, pages 58–72, Berlin, Heidelberg. Springer.

Banovic, N. (2017). Method for Understanding Complex Human Routine Behaviors from Large Behavior Logs. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '17, pages 254–258, New York, NY, USA. Association for Computing Machinery.

Barlow, H. B. (1989). Unsupervised Learning. *Neural computation*, 1(3):295–311.

Bergami, G., Maggi, F. M., Montali, M., and Peñaloza, R. (2021). Probabilistic Trace Alignment. In *2021 3rd International Conference on Process Mining (ICPM)*, pages 9–16.

Bergmeir, C. and Benítez, J. M. (2012). On the use of cross-validation for time series predictor evaluation. *Information Sciences*, 191:192–213.

Berti, A., van Zelst, S. J., and van der Aalst, W. M. P. (2019). Process mining for python (pm4py): Bridging the gap between process- and data science. In *ICPM Demo Track 2019*, volume 2374, pages 13–16.

Bloemen, V., van Zelst, S. J., van der Aalst, W. M. P., van Dongen, B. F., and van de Pol, J. (2018). Maximizing Synchronization for Aligning Observed and Modelled Behaviour. In *Business Process Management*, Lecture Notes in Computer Science, pages 233–249, Cham. Springer International Publishing.

Bogdanov, E., Cohen, I., and Gal, A. (2022). Conformance Checking over Stochastically Known Logs. In *Business Process Management Forum*, Lecture Notes in Business Information Processing, pages 105–119, Cham. Springer International Publishing.

Bogdanov, E., Cohen, I., and Gal, A. (2023). SKTR: Trace Recovery from Stochastically Known Logs. In *2023 5th International Conference on Process Mining (ICPM)*, pages 49–56.

Botchkarev, A. (2018). Performance metrics (error measures) in machine learning regression, forecasting and prognostics: Properties and typology. *arXiv preprint arXiv:1809.03006*.

Bouchabou, D., Nguyen, S. M., Lohr, C., LeDuc, B., and Kanellos, I. (2021). A Survey of Human Activity Recognition in Smart Homes Based on IoT Sensors Algorithms: Taxonomies, Challenges, and Opportunities with Deep Learning. *Sensors*, 21(18):6037.

Brady, R., Brown, W. J., and Mielke, G. I. (2023). Day-to-day variability in accelerometer-measured physical activity in mid-aged Australian adults. *BMC Public Health*, 23(1):1880.

Burattin, A., van Zelst, S. J., Armas-Cervantes, A., van Dongen, B. F., and Carmona, J. (2018). Online Conformance Checking Using Behavioural Patterns. In *Business Process Management*, pages 250–267. Springer, Cham.

Carmona, J., van Dongen, B., Solti, A., and Weidlich, M. (2018). *Conformance Checking: Relating Processes and Models*. Springer International Publishing, Cham.

Carrasco, U., Urbina Coronado, P. D., Parto, M., and Kurfess, T. (2018). Indoor location service in support of a smart manufacturing facility. *Computers in Industry*, 103:132–140.

Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys*, 41(3):15:1–15:58.

Chen, C., Chen, Y., Lai, H.-Q., Han, Y., and Liu, K. R. (2016). High accuracy indoor localization: A WiFi-based approach. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6245–6249.

Chen, C. Y., Fu, J. H., Sung, T., Wang, P.-F., Jou, E., and Feng, M.-W. (2014). Complex event processing for the Internet of Things and its applications. In *2014 IEEE International Conference on Automation Science and Engineering (CASE)*, pages 1144–1149.

Chen, L., Nugent, C. D., Mulvenna, M., Finlay, D., Hong, X., and Poland, M. (2008). A Logical Framework for Behaviour Reasoning and Assistance in a Smart Home. *International Journal of Assistive Robotics and Mechatronics*, 9(4):20–34.

Chen, L., Nugent, C. D., and Wang, H. (2012). A Knowledge-Driven Approach to Activity Recognition in Smart Homes. *IEEE Transactions on Knowledge and Data Engineering*, 24(6):961–974.

Chen, Y., Zhang, Z., Yuan, C., Li, B., Deng, Y., and Hu, W. (2021). Channel-wise topology refinement graph convolution for skeleton-based action recognition. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV*, pages 13339–13348.

Chung, J., Demiris, G., and Thompson, H. J. (2016). Ethical considerations regarding the use of smart home technologies for older adults: An integrative review. *Annual Review of Nursing Research*, 34(1):155–181.

Civitarese, G., Sztyler, T., Riboni, D., Bettini, C., and Stuckenschmidt, H. (2021). POLARIS: Probabilistic and Ontological Activity Recognition in Smart-Homes. *IEEE Transactions on Knowledge and Data Engineering*, 33(1):209–223.

Cleland, I., Han, M., Nugent, C., Lee, H., McClean, S., Zhang, S., and Lee, S. (2014). Evaluation of prompted annotation of activity data recorded from a smart phone. *Sensors*, 14(9):15861–15879.

Cohen, I. and Gal, A. (2021). Uncertain process data with probabilistic knowledge: Problem characterization and challenges. In *2021 Business Process Management Workshops*, volume 2938, pages 51–56.

Cook, D. J., Crandall, A. S., Thomas, B. L., and Krishnan, N. C. (2013a). CASAS: A Smart Home in a Box. *Computer*, 46(7):62–69.

Cook, D. J., Krishnan, N. C., and Rashidi, P. (2013b). Activity Discovery and Activity Recognition: A New Partnership. *IEEE Transactions on Cybernetics*, 43(3):820–828.

Cook, D. J. and Schmitter-Edgecombe, M. (2021). Fusing Ambient and Mobile Sensor Features Into a Behaviorome for Predicting Clinical Health Scores. *IEEE Access*, 9:65033–65043.

Crandall, A. S. and Cook, D. J. (2009). Coping with multiple residents in a smart environment. *Journal of Ambient Intelligence and Smart Environments*, 1(4):323–334.

Crawford, M., Khoshgoftaar, T. M., Prusa, J. D., Richter, A. N., and Al Najada, H. (2015). Survey of review spam detection using machine learning techniques. *Journal of Big Data*, 2(1):1–24.

Cugola, G. and Margara, A. (2012). Processing flows of information: From data stream to complex event processing. *ACM Comput. Surv.*, 44(3).

Czempiel, T., Sharghi, A., Paschali, M., Navab, N., and Mohareri, O. (2023). Surgical Workflow Recognition: From Analysis of Challenges to Architectural Study. In *Computer Vision – ECCV 2022 Workshops*, Lecture Notes in Computer Science, pages 556–568, Cham. Springer Nature Switzerland.

Dash, T., Chitlangia, S., Ahuja, A., and Srinivasan, A. (2022). A review of some techniques for inclusion of domain-knowledge into deep neural networks. *Scientific Reports*, 12(1):1040.

Dawadi, P. N., Cook, D. J., and Schmitter-Edgecombe, M. (2016). Automated Cognitive Health Assessment From Smart Home-Based Behavior Data. *IEEE Journal of Biomedical and Health Informatics*, 20(4):1188–1194.

De Medeiros, A. A., van der Aalst, W. M., and Weijters, A. J. (2008). Quantifying process equivalence based on observed behavior. *Data & knowledge engineering*, 64(1):55–74.

de Zambotti, M., Cellini, N., Goldstone, A., Colrain, I. M., and Baker, F. C. (2019). Wearable Sleep Technology in Clinical and Research Settings. *Medicine and science in sports and exercise*, 51(7):1538–1557.

Deep, S. and Zheng, X. (2019). Leveraging CNN and Transfer Learning for Vision-based Human Activity Recognition. In *2019 29th International Telecommunication Networks and Applications Conference (ITNAC)*, pages 1–4.

Dekking, F. M. (2005). *A Modern Introduction to Probability and Statistics: Understanding why and how*. Springer.

Di Ciccio, C. and Montali, M. (2022). Declarative Process Specifications: Reasoning, Discovery, Monitoring. In *Process Mining Handbook*, Lecture Notes in Business Information Processing, pages 108–152. Springer.

Duncan, M. (2024). Telematics + In-Car Sensing = Unique Ethical Issues . https://www.ethicsandinsurance.info/telematics-and-in-car-sensing-create-unique-ethical-issues/. Online; accessed 07 June 2024.

Fallatah, K. U., Barhamgi, M., and Perera, C. (2023). Personal Data Stores (PDS): A Review. *Sensors (Basel, Switzerland)*, 23(3):1477.

Felli, P., Gianola, A., Montali, M., Rivkin, A., and Winkler, S. (2022). Conformance Checking with Uncertainty via SMT. In *Business Process Management*, pages 199–216.

Gao, H., Zhou, L., Kim, J. Y., Li, Y., and Huang, W. (2023). Applying Probabilistic Model Checking to the Behavior Guidance and Abnormality Detection for A-MCI Patients under Wireless Sensor Network. *ACM Transactions on Sensor Networks*, 19(3):48:1–48:24.

Ghahramani, Z. (2015). Probabilistic machine learning and artificial intelligence. *Nature*, 521(7553):452–459.

Goldstein, M. and Uchida, S. (2016). A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. *PloS one*, 11(4):e0152173.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017). On Calibration of Modern Neural Networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1321–1330.

Guo, S., Wang, D., Feng, Z., Chen, J., and Guo, W. (2023). Di-CNN: Domain-Knowledge-Informed Convolutional Neural Network for Manufacturing Quality Prediction. *Sensors (Basel, Switzerland)*, 23(11):5313.

Gupta, A. and Davis, L. S. (2007). Objects in Action: An Approach for Combining Action Understanding and Object Perception. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8.

Gupta, H. D. and Sheng, V. S. (2020). A Roadmap to Domain Knowledge Integration in Machine Learning. In *2020 IEEE International Conference on Knowledge Graph (ICKG)*, pages 145–151.

Gupta, N., Gupta, S. K., Pathak, R. K., Jain, V., Rashidi, P., and Suri, J. S. (2022). Human activity recognition in artificial intelligence framework: A narrative review. *Artificial Intelligence Review*, 55(6):4755–4808.

Hansson, H. and Jonsson, B. (1994). A logic for reasoning about time and reliability. *Formal aspects of computing*, 6:512–535.

Hayward, S. J., van Lopik, K., Hinde, C., and West, A. A. (2022). A Survey of Indoor Location Technologies, Techniques and Applications in Industry. *Internet of Things*, 20:100608.

Ho, S. L. and Xie, M. (1998). The use of ARIMA models for reliability forecasting and analysis. *Computers & Industrial Engineering*, 35(1):213–216.

Howedi, A., Lotfi, A., and Pourabdollah, A. (2020). An Entropy-Based Approach for Anomaly Detection in Activities of Daily Living in the Presence of a Visitor. *Entropy*, 22(8):845.

Hu, H., Cheng, K., Li, Z., Chen, J., and Hu, H. (2020). Workflow recognition with structured two-stream convolutional networks. *Pattern Recognition Letters*, 130:267–274.

Ibidunmoye, O., Hernández-Rodriguez, F., and Elmroth, E. (2015). Performance anomaly detection and bottleneck identification. *ACM Computing Surveys*, 48(1).

Ilyas, I. F. and Chu, X. (2019). *Data Cleaning*. Association for Computing Machinery.

James, G., Witten, D., Hastie, T., Tibshirani, R., and Taylor, J. (2023). *An Introduction to Statistical Learning: with Applications in Python*. Springer.

Jindal, S., Sachdeva, M., and Kushwaha, A. K. S. (2022). A Systematic Analysis of the Human Activity Recognition Systems for Video Surveillance. In *IoT and Analytics for Sensor Networks*, Lecture Notes in Networks and Systems, pages 345–354.

Kanti Majumdar, K. (2014). Relationship of Activity of Daily Living with Quality of Life. *British Medical Bulletin*, pages 757–764.

Kim, Y.-J., Kang, B.-N., and Kim, D. (2015). Hidden Markov Model Ensemble for Activity Recognition Using Tri-Axis Accelerometer. In *2015 IEEE International Conference on Systems, Man, and Cybernetics*, pages 3036–3041.

Kirikkayis, Y., Gallik, F., Seiger, R., and Reichert, M. (2023). Integrating IoT-Driven Events into Business Processes. In *Intelligent Information Systems*, Lecture Notes in Business Information Processing, pages 86–94.

Koorneef, M., Solti, A., Leopold, H., and Reijers, H. A. (2018). Automatic Root Cause Identification Using Most Probable Alignments. In *Business Process Management Workshops*, Lecture Notes in Business Information Processing, pages 204–215.

Kubat, M., Matwin, S., et al. (1997). Addressing the curse of imbalanced training sets: one-sided selection. In *ICML*, volume 97, page 179. Citeseer.

Kwiatkowska, M., Norman, G., and Parker, D. (2011). PRISM 4.0: Verification of probabilistic real-time systems. In *Proc. 23rd International Conference on Computer Aided Verification (CAV'11)*, pages 585–591.

Kwiatkowska, M., Norman, G., and Parker, D. (2018). Probabilistic model checking: Advances and applications. *Formal System Verification: State-of the-Art and Future Trends*, pages 73–121.

Lawler, G. F. (2018). *Introduction to stochastic processes*. Chapman and Hall/CRC.

Lee, W. L. J., Burattin, A., Munoz-Gama, J., and Sepúlveda, M. (2020). Orientation and conformance: A HMM-based approach to online conformance checking. *Information Systems*, page 101674.

Lee, W. L. J., Verbeek, H. M. W., Munoz-Gama, J., van der Aalst, W. M. P., and Sepúlveda, M. (2018). Recomposing conformance: Closing the circle on decomposed alignment-based conformance checking in process mining. *Information Sciences*, 466:55–91.

Lee, X. K., Chee, N. I., Ong, J. L., Teo, T. B., van Rijn, E., Lo, J. C., and Chee, M. W. (2019). Validation of a Consumer Sleep Wearable Device With Actigraphy and Polysomnography in Adolescents Across Sleep Opportunity Manipulations. *Journal of Clinical Sleep Medicine : JCSM : Official Publication of the American Academy of Sleep Medicine*, 15(9):1337–1346.

Leemans, S. J., Fahland, D., and Van Der Aalst, W. M. (2014). Process and deviation exploration with inductive visual miner. In *12th International Conference on Business Process Management, BPM 2014*, pages 46–50.

Lever, J. (2016). Classification evaluation: It is important to understand both what a classification metric expresses and what it hides. *Nature methods*, 13(8):603–605.

Liciotti, D., Bernardini, M., Romeo, L., and Frontoni, E. (2020). A sequential deep learning application for recognising human activities in smart homes. *Neurocomputing*, 396:501–513.

Liu, F. T., Ting, K. M., and Zhou, Z.-H. (2012). Isolation-Based Anomaly Detection. *ACM Transactions on Knowledge Discovery from Data*, 6(1):1–39.

Liu, R., Ramli, A. A., Zhang, H., Henricson, E., and Liu, X. (2022). An Overview of Human Activity Recognition Using Wearable Sensors: Healthcare and Artificial Intelligence. In *Internet of Things – ICIOT 2021*, Lecture Notes in Computer Science, pages 1–14.

Liu, Y., Dillon, T., Yu, W., Rahayu, W., and Mostafa, F. (2020). Noise removal in the presence of significant anomalies for industrial iot sensor data in manufacturing. *IEEE Internet of Things Journal*, 7(8):7084–7096.

London Fire Brigade (2024). Cooking - Fire safety at home. https://www.london-fire.gov.uk/safety/the-home/cooking/. Online; accessed 26 May 2024.

Lu, W., Li, J., Wang, J., and Qin, L. (2021). A CNN-BiLSTM-AM method for stock price prediction. *Neural Computing and Applications*, 33:4741–4753.

L'Yvonnet, T., De Maria, E., Moisan, S., and Rigault, J.-P. (2021). Probabilistic model checking for human activity recognition in medical serious games. *Science of Computer Programming*, 206:102629.

Maclin, R. and Opitz, D. (1999). Popular Ensemble Methods: An Empirical Study. *Journal of Artificial Intelligence Research*, 11:169–198.

Mannhardt, F. (2018). *Multi-perspective Process Mining*. PhD thesis, Eindhoven University of Technology.

Mannhardt, F., de Leoni, M., Reijers, H., Aalst, W., and Toussaint, P. (2016). From Low-Level Events to Activities - A Pattern-Based Approach. In *International Conference on Business Process Management*, pages 125–141.

Margara, A., Cugola, G., and Tamburrelli, G. (2014). Learning from the past: Automated rule generation for complex event processing. In *Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems*, pages 47–58.

Meier, J., Dietz, A., Boehm, A., and Neumuth, T. (2015). Predicting treatment process steps from events. *Journal of Biomedical Informatics*, 53:308–319.

Mendling, J., Neumann, G., and van der Aalst, W. (2007). Understanding the Occurrence of Errors in Process Models Based on Metrics. In *On the Move to Meaningful Internet Systems 2007: CoopIS, DOA, ODBASE, GADA, and IS*, pages 113–130.

Munoz-Gama, J., Martin, N., Fernandez-Llatas, C., Johnson, O. A., Sepúlveda, M., Helm, E., Galvez-Yanjari, V., Rojas, E., Martinez-Millana, A., Aloini, D., Amantea, I. A., Andrews, R., Arias, M., Beerepoot, I., Benevento, E., Burattin, A., Capurro, D., Carmona, J., Comuzzi, M., Dalmas, B., de la Fuente, R., Di Francescomarino, C., Di Ciccio, C., Gatta, R., Ghidini, C., Gonzalez-Lopez, F., Ibanez-Sanchez, G., Klasky, H. B., Prima Kurniati, A., Lu, X., Mannhardt, F., Mans, R., Marcos, M., Medeiros de Carvalho, R., Pegoraro, M., Poon, S. K., Pufahl, L., Reijers, H. A., Remy, S., Rinderle-Ma, S., Sacchi, L., Seoane, F., Song, M., Stefanini, A., Sulis, E., ter Hofstede, A. H. M., Toussaint, P. J., Traver, V., Valero-Ramon, Z., van de Weerd, I., van der Aalst, W. M. P., Vanwersch, R., Weske, M., Wynn, M. T., and Zerbato, F. (2022). Process mining for healthcare: Characteristics and challenges. *Journal of Biomedical Informatics*, 127:103994.

Murphy, K. P. et al. (2002). Dynamic bayesian networks. *Probabilistic Graphical Models, M. Jordan*, 7:431.

Niemann, F., Lüdtke, S., Bartelt, C., and ten Hompel, M. (2022). Context-Aware Human Activity Recognition in Industrial Processes. *Sensors*, 22(1):134.

OpenAI (2023). ChatGPT. https://chatgpt.com/. Online; accessed 15 December 2023.

Palmer, M. G. (2018). Experimenter presence in human behavior analytic laboratory studies: Confound it? *Behavior Analysis: Research and Practice*, 19(4):303.

Papapanagiotou, P., Vaughan, J., Smola, F., and Fleuriot, J. D. (2021). A real-world case study of process and data driven predictive analytics for manufacturing workflows. In *54th Hawaii International Conference on System Sciences*, pages 1001–1010.

Parzen, E. (1999). *Stochastic processes*. Society for Industrial and Applied Mathematics.

Pazhoumand-Dar, H., Armstrong, L. J., and Tripathy, A. K. (2020). Detecting deviations from activities of daily living routines using kinect depth maps and power consumption data. *Journal of Ambient Intelligence and Humanized Computing*, 11(4):1727–1747.

Pegoraro, M., Uysal, M. S., and van der Aalst, W. M. P. (2020). Efficient Construction of Behavior Graphs for Uncertain Event Data. In Abramowicz, W. and Klein, G., editors, *Business Information Systems*, Lecture Notes in Business Information Processing, pages 76–88, Cham. Springer International Publishing.

Pegoraro, M., Uysal, M. S., and van der Aalst, W. M. P. (2021). Conformance checking over uncertain event data. *Information Systems*, 102:101810.

Perkowitz, M., Philipose, M., Fishkin, K., and Patterson, D. J. (2004). Mining models of human activities from the web. In *Proceedings of the 13th International Conference on World Wide Web*, page 573–582.

Pesic, M. (2008). *Constraint-Based Workflow Management Systems : Shifting Control to Users*. Phd Thesis, Eindhoven University of Technology.

Peterson, J. L. (1977). Petri nets. *ACM Computing Surveys (CSUR)*, 9(3):223–252.

Pichler, P., Weber, B., Zugal, S., Pinggera, J., Mendling, J., and Reijers, H. A. (2011). Imperative versus Declarative Process Modeling Languages: An Empirical Investigation. In *Business Process Management Workshops*, pages 383–394.

Pierleoni, P., Belli, A., Palma, L., Pellegrini, M., Pernini, L., and Valenti, S. (2015). A High Reliability Wearable Device for Elderly Fall Detection. *IEEE Sensors Journal*, 15(8):4544–4553.

Piovezan, R. D., Poyares, D., and Tufik, S. (2013). Frailty and sleep disturbances in the elderly: possible connections and clinical implications. *Sleep Science*, 6(4):175–179.

Pratap, A., Homiar, A., Waninger, L., Herd, C., Suver, C., Volponi, J., Anguera, J. A., and Areán, P. (2022). Real-world behavioral dataset from two fully remote smartphone-based randomized clinical trials for depression. *Scientific Data*, 9(1):522.

Rahmani, A. M., Babaei, Z., and Souri, A. (2021). Event-driven IoT architecture for data analysis of reliable healthcare application using complex event processing. *Cluster Computing*, 24(2):1347–1360.

Raza, M. A., Chen, L., Nanbo, L., and Fisher, R. B. (2023). EatSense: Human centric, action recognition and localization dataset for understanding eating behaviors and quality of motion assessment. *Image and Vision Computing*, 137:104762.

Richardson, M. and Domingos, P. (2006). Markov logic networks. *Machine learning*, 62:107–136.

Roberts, D. M., Schade, M. M., Mathew, G. M., Gartenberg, D., and Buxton, O. M. (2020). Detecting sleep using heart rate and motion data from multisensor consumer-grade wearables, relative to wrist actigraphy and polysomnography. *Sleep*, 43(7):zsaa045.

Russell, S. J. and Norvig, P. (2016). *Artificial intelligence: a modern approach*. Pearson.

Saba, T., Mohamed, A. S., El-Affendi, M., Amin, J., and Sharif, M. (2020). Brain tumor detection using fusion of hand crafted and deep learning features. *Cognitive Systems Research*, 59:221–230.

Šabić, E., Keeley, D., Henderson, B., and Nannemann, S. (2021). Healthcare and anomaly detection: using machine learning to predict anomalies in heart rate data. *AI & SOCIETY*, 36(1):149–158.

Saives, J., Pianon, C., and Faraut, G. (2015). Activity discovery and detection of behavioral deviations of an inhabitant from binary sensors. *IEEE Transactions on Automation Science and Engineering*, 12(4):1211 – 1224.

Sathya, R., Abraham, A., et al. (2013). Comparison of supervised and unsupervised learning algorithms for pattern classification. *International Journal of Advanced Research in Artificial Intelligence*, 2(2):34–38.

Scholl, P. M., Wille, M., and Van Laerhoven, K. (2015). Wearables in the wet lab: A laboratory system for capturing and guiding experiments. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '15, pages 589–599.

Schönig, S., Ackermann, L., Jablonski, S., and Ermer, A. (2020). IoT meets BPM: A Bi-directional Communication Architecture for IoT-Aware Process Execution. *Software and Systems Modeling*, pages 1443–1459.

Sepesy Maučec, M. and Donaj, G. (2021). Discovering Daily Activity Patterns from Sensor Data Sequences and Activity Sequences. *Sensors*, 21(20):6920.

Shao, D., Zhao, Y., Dai, B., and Lin, D. (2020). Finegym: A hierarchical video dataset for fine-grained action understanding. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2616–2625.

Sheng, T. and Huber, M. (2020). Unsupervised embedding learning for human activity recognition using wearable sensor data. In *The Thirty-Third International Flairs Conference*.

Shi, L., Zhang, Y., Cheng, J., and Lu, H. (2019). Two-Stream Adaptive Graph Convolutional Networks for Skeleton-Based Action Recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12026–12035.

Suhothayan, S., Gajasinghe, K., Loku Narangoda, I., Chaturanga, S., Perera, S., and Nanayakkara, V. (2011). Siddhi: A second look at complex event processing architectures. In *Proceedings of the 2011 ACM Workshop on Gateway Computing Environments*, GCE '11, pages 43–50.

Sztyler, T., Carmona, J., Völker, J., and Stuckenschmidt, H. (2016). Self-tracking Reloaded: Applying Process Mining to Personalized Health Care from Labeled Sensor Data. *Transactions on Petri Nets and Other Models of Concurrency XI*, pages 160–180.

Tabira, T., Hotta, M., Maruta, M., Ikeda, Y., Shimokihara, S., Han, G., Yamaguchi, T., Tanaka, H., Ishikawa, T., and Ikeda, M. (2022). Characteristic of process analysis on instrumental activities of daily living according to the severity of cognitive impairment in community-dwelling older adults with Alzheimer's disease. *International Psychogeriatrics*, pages 1–12.

Tax, N., Sidorova, N., Haakma, R., and van der Aalst, W. M. P. (2018). Event Abstraction for Process Mining using Supervised Learning Techniques. In *Proceedings of SAI Intelligent Systems Conference*, volume 15, pages 251–269.

Taylor, S. J. and Letham, B. (2017). Forecasting at scale. Technical report, PeerJ Inc.

Teplitzky, B. A., McRoberts, M., and Ghanbari, H. (2020). Deep learning for comprehensive ecg annotation. *Heart rhythm*, 17(5):881–888.

Thakur, N. and Han, C. Y. (2021). A Review of Assistive Technologies for Activities of Daily Living of Elderly. *arXiv:2106.12183 [cs]*.

Theodoridis, S. and Koutroumbas, K. (2006). *Pattern recognition*. Elsevier.

Tufano, M., Lasschuijt, M., Chauhan, A., Feskens, E. J. M., and Camps, G. (2022). Capturing Eating Behavior from Video Analysis: A Systematic Review. *Nutrients*, 14(22):4847.

van der Aa, H., Leopold, H., and Reijers, H. A. (2020). Efficient Process Conformance Checking on the Basis of Uncertain Event-to-Activity Mappings. *IEEE Transactions on Knowledge and Data Engineering*, 32(5):927–940.

van der Aalst, W. (2016). *Process Mining*. Springer.

van der Aalst, W., Adriansyah, A., and Van Dongen, B. (2012). Replaying history on process models for conformance checking and performance analysis. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(2):182–192.

van der Aalst, W., Weijters, T., and Maruster, L. (2004). Workflow mining: Discovering process models from event logs. *IEEE transactions on knowledge and data engineering*, 16(9):1128–1142.

van Der Aalst, W. M. P. and Carmona, J., editors (2022). *Process Mining Handbook*, volume 448 of *Lecture Notes in Business Information Processing*. Springer International Publishing.

van Dongen, B. F., De Smedt, J., Di Ciccio, C., and Mendling, J. (2020). Conformance checking of mixed-paradigm process models. *Information Systems*, page 101685.

van Zelst, S., Bolt Iriondo, A., and van Dongen, B. (2017). Tuning alignment computation : An experimental evaluation. In *Proceedings of the International Workshop on Algorithms & Theories for the Analysis of Event Data (ATAED 2017)*, pages 6–20.

van Zelst, S. J., Bolt, A., Hassani, M., van Dongen, B. F., and van der Aalst, W. M. P. (2019). Online conformance checking: Relating event streams to process models using prefix-alignments. *International Journal of Data Science and Analytics*, 8(3):269–284.

Wang, X., Liu, J., Moore, S. J., Nugent, C. D., and Xu, Y. (2023). A behavioural hierarchical analysis framework in a smart home: Integrating HMM and probabilistic model checking. *Information Fusion*, 95:275–292.

Weijters, A., Aalst, V., and Medeiros, A. K. a. D. (2006). Process mining with the HeuristicsMiner algorithm. *Eindhoven University of Technology*.

White, S. A. (2004). Introduction to BPMN. *IBM Cooperation*, 2(0):0.

Wu, Y.-H., Chen, Y., Shirmohammadi, S., and Hsu, C.-H. (2022). AI-Assisted Food Intake Activity Recognition Using 3D mmWave Radars. In *Proceedings of the 7th International Workshop on Multimedia Assisted Dietary Management*, pages 81–89.

Yahaya, S. W., Lotfi, A., and Mahmud, M. (2019). A Consensus Novelty Detection Ensemble Approach for Anomaly Detection in Activities of Daily Living. *Applied Soft Computing*, 83:105613.

Yahaya, S. W., Lotfi, A., and Mahmud, M. (2021a). Detecting Anomaly and Its Sources in Activities of Daily Living. *SN Computer Science*, 2(1):14.

Yahaya, S. W., Lotfi, A., and Mahmud, M. (2021b). Detecting Anomaly and Its Sources in Activities of Daily Living. *SN Computer Science*, 2(1):14.

Yahaya, S. W., Lotfi, A., and Mahmud, M. (2021c). Towards a data-driven adaptive anomaly detection system for human activity. *Pattern Recognition Letters*, 145:200–207.

Yamada, N., Sakamoto, K., Kunito, G., Isoda, Y., Yamazaki, K., and Tanaka, S. (2007). Applying Ontology and Probabilistic Model to Human Activity Recognition from Surrounding Things. *Ipsj Digital Courier*, 3:506–517.

Yan, S., Xiong, Y., and Lin, D. (2018). Spatial Temporal Graph Convolutional Networks for Skeleton-Based Action Recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Yang, L., Dong, X., Xing, S., Zheng, J., Gu, X., and Song, X. (2019). An abnormal transaction detection mechanim on bitcoin. In *2019 International Conference on Networking and Network Applications (NaNA)*, pages 452–457.

Yang, Y., Hu, P., Shen, J., Cheng, H., An, Z., and Liu, X. (2024). Privacy-preserving human activity sensing: A survey. *High-Confidence Computing*, 4(1):100204.

Yaya, S., Idriss-Wheeler, D., Sanogo, N. A., Vezina, M., and Bishwajit, G. (2020). Self-reported activities of daily living, health and quality of life among older adults in South Africa and Uganda: A cross sectional study. *BMC Geriatrics*, 20(1):402.

Zhang, J., Zhang, Q., Li, D., Zhao, R., and Wang, D. (2017). RFlow-ID: Unobtrusive Workflow Recognition with COTS RFID. In *Proceedings of the 14th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, pages 333–342.

Zheng, J. and Papapanagiotou, P. (2022). Predictive Behavioural Monitoring and Deviation Detection in Activities of Daily Living of Older Adults. In *15th International Joint Conference on Biomedical Engineering Systems and Technologies, Volume 5 - HEALTHINF*, pages 899–910. SCITEPRESS.

Zheng, J., Papapanagiotou, P., and Fleuriot, J. (2024). Alignment-based conformance checking over probabilistic events. In *Proceedings of the 57th Hawaii International Conference on System Sciences*.