Enhancement of Process Mining with machine learning for identification of patterns in human behaviour

Georgia Theodoropoulou

Thèse de Doctorat

UNIVERSITE DE LIMOGES

ECOLE DOCTORALE SCIENCE - TECHNOLOGIE

FACULTE DES SCIENCES & TECHNIQUES

Laboratoire XLIM

Thèse

pour obtenir le grade de

DOCTEUR DE L' UNIVERSITÉ DE LIMOGES

Discipline / Spécialité : Informatique

présentée et soutenue par

Georgia THEODOROPOULOU

le [06 2024]

Enhancement of process mining with machine learning for identification of patterns in human behaviour

Université de Limoges

Thèse dirigée par le Professeur Georgios MIAOULIS

et le Professeur Djamchid GHAZANFARPOUR

Président du jury Professeur Paris MASTOROCOSTAS

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ
UNIVERSITY OF WEST ATTICA

Acknowledgements

I want to express my gratitude to Professor George Miaoulis, my supervisor, for inspiring and encouraging me to commence the PhD journey. I am thankful for his assistance, support, kindness, generosity, and the knowledge he imparted to me throughout this enriching experience.

I am also thankful to Professor Djamchid Ghazanfarpour for his essential support and guidance throughout these years.

I would like to convey my appreciation and thankfulness to Dr. Alexandros Bousdekis, Adjunct Lecturer and Researcher at the "Systems Modelling and Process Intelligence" research group of the Department of Informatics and Computer Engineering at the University of West Attica, for dedicating his precious time and providing constant support throughout my academic journey. His assistance was crucial and greatly contributed to the successful completion of my research and writing process for this thesis.

I want to thank my close friend Ifigenia, who has been there for me throughout, providing encouragement to persevere and tackling the challenges together in our joint effort to complete the PhD.

Also, I would like to extend my sincere appreciation to my dear family for their continuous moral and practical support during my PhD and in life overall.

Finally, I wish to convey my appreciation to V.J.A. who motivated me to embark on my PhD's degree journey, bringing me to the delightful realization of fulfilling a long-held dream. This accomplishment wouldn't have been attainable without her steadfast support.

Abstract

The study of human behavior is a broad and interdisciplinary field. A variety of research methods has been employed in order to gain insights into the complexities of human actions. The process of identifying and classifying specific activities or actions that individuals perform based on data collected from sensors is a rapidly evolving field that benefits from several technologies which provide the necessary tools for collecting, processing, and analyzing data related to human movements and behaviors.

The combination of sensor data and Machine Learning algorithms, enable analyzing data using Human Activity Recognition (HAR) techniques, which yield valuable insights into behavior, preferences, and daily routines as well as identify human behavior disorders, anomalies and patterns. This is particularly crucial for promoting healthier and more independent living, aiding those with disabilities, and providing support to caregivers and medical professionals.

A field that has contributed significantly to the effort to achieve these goals is the Process Mining field. Process mining relies on event logs, enables the analysis of them, offering insights into the workflow and sequence of activities.

As human behavior is not structured enough to be represented using a process model, more robust techniques need to be incorporated within process mining. Change Point Detection emerges as a potential tool for identifying disorders in human behavior and along with Clustering methods which are effectively applied to categorize and group diverse human activities seem to be a solution to the aforementioned challenge.

This research work aims to develop an advanced framework that combines Process Mining techniques, Change Point Detection, and Clustering methods to achieve a holistic analysis of human behavior patterns. The study explores the integration of these diverse techniques to enhance the understanding of complex human activities, with a specific focus on identifying patterns and deviations indicative of potential disorders.

Human activities are represented in a sequential manner. By using also unsupervised Machine Learning techniques, which enable handling and analyzing the inherent variability in human behavior, makes it possible to identify and detect changes throughout the dynamic process.

The experiments reveal that the proposed framework effectively identifies anomalies or disorders in human behavior and offers insights into the workflow and sequence of activities. Furthermore, it facilitates the discovery and modeling of behavioral patterns, along with the identification of deviations in human activities, which is crucial for analyzing unstructured human behaviors.

Resumé

L'étude du comportement humain est un domaine vaste et interdisciplinaire. Une variété de méthodes de recherche a été utilisée afin d'obtenir des perspectives sur les complexités des actions humaines. Le processus d'identification et de classification d'activités ou d'actions spécifiques que les individus effectuent sur la base de données collectées à partir de capteurs est un domaine en évolution rapide qui bénéficie de plusieurs technologies fournissant les outils nécessaires pour collecter, traiter et analyser les données liées aux mouvements et comportements humains.

La combinaison de données de capteurs et d'algorithmes d'apprentissage automatique permet d'analyser les données à l'aide de techniques de reconnaissance d'activité humaine (HAR) qui fournissent des informations précieuses sur le comportement, les préférences et les routines quotidiennes, ainsi que sur l'identification des troubles du comportement, des anomalies et des schémas humains. Ceci est particulièrement crucial pour promouvoir une vie plus saine et plus indépendante, aider les personnes handicapées et fournir un soutien aux aidants et aux professionnels de la santé.

Un domaine qui a contribué de manière significative aux efforts pour atteindre ces objectifs est le domaine de la Process Mining. La Process Mining s'appuie sur des journaux d'événements, permet leur analyse et offre des informations sur le flux de travail et la séquence des activités. Comme le comportement humain n'est pas suffisamment structuré pour être représenté par un modèle de processus, des techniques plus robustes doivent être incorporées dans la Process Mining. La détection de points de changement émerge comme un outil potentiel pour identifier les troubles du comportement humain, associée à des méthodes de regroupement qui sont efficacement appliquées pour catégoriser et regrouper diverses activités humaines semblent être une solution au défi susmentionné.

Ce travail de recherche vise à développer un cadre avancé qui combine les techniques de Process Mining, les méthodes de détection de points de changement et de clustering pour obtenir une analyse holistique des modèles de comportement humain. L'étude explore l'intégration de ces diverses techniques pour améliorer la compréhension des activités humaines complexes, en mettant l'accent sur l'identification de modèles et d'écarts révélateurs de troubles potentiels.

Les activités humaines sont représentées de manière séquentielle. En utilisant également des techniques d'apprentissage automatique non supervisées qui permettent de gérer et d'analyser la variabilité inhérente au comportement humain, il est possible d'identifier et de détecter des changements tout au long du processus dynamique.

Les expériences révèlent que le cadre proposé identifie efficacement les anomalies ou troubles du comportement humain et offre des informations sur le flux de travail et la séquence des activités. De plus, il facilite la découverte et la modélisation des schémas comportementaux, ainsi que l'identification des déviations dans les activités humaines, ce qui est crucial pour analyser les comportements humains non structurés.

Table of Contents

Table of Figures

Table of Tables

Chapter 1

Introduction

1.1 Motivation

The study of human behavior is a broad and interdisciplinary field that encompasses various branches of psychology, sociology, anthropology, economics, and other disciplines. Researchers in this field seek to understand, explain, and predict the ways in which individuals and groups behave in different contexts [1].

Researchers in the field of human behavior often employ a variety of research methods, including experiments, surveys, observations, and data analysis, to gain insights into the complexities of human actions and interactions [2]. The interdisciplinary nature of this field allows for a holistic understanding of human behavior from multiple perspectives.

Human Activity Recognition (HAR) can be seen as a specific application within the broader study of human behavior, contributing insights into the physical aspects of human actions in specific contexts [3]. HAR is a field of study that refers to the process of identifying and classifying specific activities or actions that individuals perform based on data collected from sensors [4]. The primary goal is to develop algorithms and systems that can automatically recognize and interpret human activities based on data collected from various sensors. This field has practical applications in diverse areas such as healthcare [5], sports [6], smart homes [7], and human-computer interaction [8].

HAR is a rapidly evolving field that benefits from several technologies, which provide the necessary tools for collecting, processing, and analyzing data related to human movements and behaviors [9].

Key technologies that have played a crucial role in advancing HAR include inertial sensors, like accelerometers and gyroscopes commonly found in smartphones [10], fitness trackers [11], and wearable devices [12]. These sensors are fundamental for capturing motion-related data for activity recognition. Computer vision technologies [13], utilizing cameras and depth sensors, are employed for visual activity recognition. They can capture and analyze human movements, gestures, and interactions, contributing to applications in security, healthcare, and human-computer interaction. Wireless sensor networks, including Radio-Frequency Identification (RFID) technology [14], are used to track the movement of objects and individuals in logistics, healthcare, and controlled environments. Bluetooth beacons, small low-energy devices, strategically placed in environments, track the location and movement of individuals carrying

Bluetooth-enabled devices [15], contributing to indoor positioning systems for activity recognition.

Moreover, environmental sensors [16], such as temperature, humidity, and light sensors, provide additional context to activity recognition systems. Changes in environmental conditions may influence human behavior and are considered in the recognition process. Audio sensors, like microphones, capture sounds and speech [17], contributing to applications such as voice activity recognition in smart home systems and voice-controlled devices.

Machine learning [18] and data analytics play also a crucial role, along with deep learning [19,20] models applied to analyze and classify activity patterns based on collected sensor data. The Internet of Things (IoT) [21] contributes through the integration of devices and platforms, facilitating seamless communication and data exchange between various sensors and systems for comprehensive activity recognition solutions. The integration of these technologies has led to the development of sophisticated and accurate HAR systems, applicable in healthcare, fitness tracking, smart homes, security, and industrial settings.

The combination of sensor data and advanced algorithms [22] allows for a better understanding of human behavior in various contexts particularly within the realm of smart homes. The integration of smart home technologies into daily life has created opportunities to collect rich data on human activities within the home environment. Analyzing this data using Human Activity Recognition (HAR) techniques yields valuable insights into behavior, preferences, and daily routines. A primary goal in HAR is the identification of human behavior disorders, the detection of anomalies and patterns [23], and the enhancement of individuals' quality of life. This is particularly crucial for promoting healthier and more independent living, aiding those with disabilities, and providing support to caregivers and medical professionals.

Smart environments carry the potential to deliver in-home support services, fostering the well-being, independence, and healthcare of their residents. This focus is especially pertinent for the elderly and individuals requiring assistance. By leveraging the capabilities of smart technologies, these environments contribute to extended independence and improved quality of life, aligning with the broader objective of supporting individuals with diverse needs [24].

A field that has contributed significantly to the effort to achieve these goals is the Process Mining field. Process Mining emerged as a novel discipline in the business domain around the beginning of the 21st century focusing initially on the automatic discovery of business process models. However, over time the scope of process mining has expanded in various directions [25]. Notably, one of the areas where process mining has found substantial applications is the analysis of human behavior [26].

In the realm of Human Activity Recognition (HAR), the contribution of process mining proves invaluable. Process mining relies on event logs, which capture sequences of events within a system. In the context of Human Activity Recognition (HAR), these events can represent human activities recorded through sensors. The application of process mining enables the analysis of these logs, offering insights into the workflow and sequence of activities. Furthermore, it facilitates the discovery and modeling of behavioral patterns, along with the identification of deviations or anomalies in human activities. Through the assessment of process performance and the correlation of sensor data with process logs, it becomes feasible to pinpoint activities contributing to delays or inefficiencies, thereby supporting process optimization endeavors.

Additionally, process mining tools generate visual representations of processes, aiding in the interpretation and communication of intricate human-centric workflows. They offer the capability to provide user-centric insights by scrutinizing the paths and sequences of activities conducted by individuals. This, in turn, contributes to the creation of personalized models that more accurately capture individual behavior patterns.

A fundamental question arises in the application of process mining to human behavior: Is human behavior structured enough to be represented using a process model [27]? To address the unstructured and complex contexts, more robust techniques need to be incorporated within process mining. These complexities can make it challenging to accurately monitor and predict human actions, especially given individual variations. Identifying deviations from established patterns becomes crucial for analyzing unstructured behaviors. Change Point Detection emerges as a potential tool for identifying disorders in human behavior. Many applications of Change Point Detection in human activity revolve around detecting transitions in activities using sensor data, thereby segmenting human daily activities into distinct actions and accurately identifying each action [28]. In contrast, less emphasis has been placed on processing activities as text data to uncover change points in the sequence of activities. While Change Point Detection algorithms have been applied to time series data, human activity durations tend to vary, and activities do not necessarily occur at specific time intervals. As such, preserving human activities as sequences over time is preferable to constraining them to specific time intervals in which activities are repeated or multiple activities occur. Hence, the monitoring of context aware system is a real world problem which requires sequential analysis of time series data to identify and detect change during the whole dynamic process [29].

Human activity datasets often exhibit small sizes and high variability, a consequence of the inherent nature of human behavior. Deep learning techniques, known for their efficacy, typically require substantial amounts of data to yield accurate results. In contrast, machine learning techniques demonstrate versatility by effectively managing both small and large datasets

characterized by high variability. This adaptability enables machine learning methods to discern and identify patterns in human behavior across a spectrum of dataset sizes and complexities.

To the best of this research current knowledge a scientific gap also lies in the limited exploration of unsupervised Machine Learning techniques applied to text data representing human activities in a sequential manner. The need for techniques that can handle the inherent variability in human behavior and analyze it in a context-aware system requires sequential analysis of data to identify and detect changes throughout the dynamic process. This gap highlights the potential for advancements in techniques that can effectively address the complexities of unstructured human behavior data for more accurate monitoring and analysis.

1.2 Research questions

The research questions of this work are summarized in the following table:

How can existing Process Mining techniques be applied to accommodate human behavior analysis? (Section 4.1)	• How does the integration of human activity data with process mining tools contribute to a more comprehensive analysis of human-centric workflows? • How Conformance Checking can improve process models of human behavior in conjunction to human activity event log? • In what ways can performance analysis identify disorders in the durations of human activities? • How can Process Mining techniques behave when handling unstructured event logs in the Human activity domain space?
How to address the complexity of process mining techniques in human behavior in order to provide interpretable results? (Section 4.2)	• How visual analytics can enhance the understanding of human behavior? • In what ways can process mining be integrated into a web interactive tool, adopting a holistic approach?
How to develop and implement an advanced process mining framework focused on analyzing human behavior, particularly with the aim of identifying patterns and disorders?	• How can text data of human activity from smart environments can be utilized to feed modern techniques in order to find human disorders?

(Section 4.3)	• Can process mining be effectively employed to detect deviations and anomalies in human activities event logs? • How can change point detection methods be effectively integrated into HAR systems that can autonomously learn to identify patterns, transitions and abrupt changes (disorders) in human activities? • How can unsupervised clustering techniques be effectively applied to categorize and group diverse human activities?

1.3 Thesis Structure

Chapter 2 explores the theoretical background of the techniques examined and employed in this research, covering Process Mining (section 2.1), Data Encoding (section 2.2), Change Point Detection (section 2.3), and Clustering methods (section 2.4). Meanwhile, Chapter 3 offers a comprehensive overview of related works in the field.

Moving on to Chapter 4, the methodology of the PhD thesis is scrutinized. Initially, in section 4.1, Process Mining algorithms are explored in the domains of human activity and financial administration. Subsequently, section 4.2 delves into the significance of achieving explainability, leading to the development of visualizations within the realm of Process Mining techniques. It also explores an application facilitating a more profound analysis of event logs within the context of Process Mining techniques. In section 4.3, a framework is presented, developed for the analysis of human behavior and the identification of disorders through unsupervised techniques based on Change Point Detection and Clustering algorithms.

Chapter 5 delves into the experiments conducted in the current research and presents the obtained results which are derived from each section of chapter 4 respectively.

Finally, Chapter 6 encapsulates the limitations, overall conclusions, and future challenges in the field.

Chapter 2

Background

2.1 Process Mining

Process mining is a field which involves analysis, discovery, monitoring and improvement of real processes by means of knowledge extracted from event logs [30]. Each time a process runs a new process instance (trace) is recorded with unique id named trace id. Each trace consist of a sequence of activities. Event logs are files that record sequences of activities that have been performed with time stamps.

There are several key process mining techniques used to analyze and understand processes. Each technique serves a specific purpose and provides unique insights into process performance. Some common process mining techniques are:

Process Analysis: Processes can be analyzed from different viewpoints [31]:

- Control-flow, which focuses on the activities that are executed and the relationships of precedence among them (in terms of preconditions and post-conditions).
- Organizational, which focuses on the actors that are involved, on their roles, and on how they are mapped/assigned to the activities.
- Temporal, which focuses on the temporal aspect of process executions, such as their duration, the temporal relations between activities, frequency of specific activities, and is typically considered for the detection of bottlenecks and the estimation of performance indicators, such as throughput and average duration of process executions.
- Data, which focuses on the data used and generated during the process execution, including the attributes that characterize the given enactment, and the attributes that characterize the triggered events

Process Discovery: A discovery technique takes an event log and produces automatically a process model. One of the oldest algorithm as well as the first official algorithm of process mining is Alpha Miner [32]. The Alpha Miner was one of the first process discovery algorithms that could adequately deal with concurrency. However, the Alpha Miner should not be seen as a very practical mining technique as it has problems with noise, infrequent/incomplete behavior, and complex routing constructs. The Alpha Miner is simple and many of its ideas have been embedded in more complex and robust techniques. Hence, the algorithms which were developed hereafter were designed to reduce the complexity of the model such as heuristic-based mining algorithm [33], genetic process mining [34], region-based mining [35], fuzzy based algorithm [36], and episode mining algorithm [37]. Inductive Miner [38], is known for its flexibility and

adaptability. It can handle a wide range of process types, including those with various complexities, such as loops, parallelism, and choice. This adaptability makes it a good choice for real-world processes that often exhibit such complexities. Also noise tolerance makes Inductive Miner robust when it comes to handling noisy event logs, which may contain irregularities, errors, or incomplete data.

Conformance Checking: Conformance checking evaluates how well the observed process aligns with a predefined or discovered process model. It takes as input the event log and the process model and reveals deviations, bottlenecks, and the compliance degree of the process [39]. The output of conformance checking consists of diagnostic information showing differences and commonalities between the model and its log. Two main Conformance Checking algorithms are Trace Alignment [40] and Replay Log [39]. Conformance checking compares a process model with respect to the event log with four main quality dimensions: fitness, simplicity, precision, and generalization [41].

For Replay Log, the process model is represented as a Petri net and traces in the event log are replayed on the model. Petri net consist of places (empty round circles) and transitions (squares that includes activities). If the trace indicates that an activity needs to take place, the corresponding transition is executed. If this is not possible due to an empty input place, a so-called missing token is added. Tokens that are never consumed are called remaining tokens. The numbers of missing and remaining tokens relative to the numbers of consumed and produced tokens indicate the severity of the conformance problem [42].

In contrast to computing alignments, token-based replay is relatively efficient but doesn't consistently generate valid paths through the process model. Alignments offer more detailed and precise diagnostics as each observed trace is meticulously mapped onto a model behavior that closely resembles the observed sequence. Alignments highlight common behavior, as well as skipped and inserted events, providing clearer insights into deviations compared to the challenges associated with interpreting missing and remaining tokens. However, for large event logs and complex processes, alignment computations can become impractical. Additionally, the existence of numerous optimal alignments introduces a non-deterministic aspect to the diagnostics. [42].

Performance Analysis: The goal of performance analysis is process enhancement. Adding further data perspectives to the process model using event data can extend the knowledge gained for the process [43]. By projecting time information from the event log to process model we can uncover bottlenecks, deviation, service times, throughput time, waiting times and execution times of the process.

As research advances, process mining has expanded to encompass a broader range of techniques such as:

- Machine Learning: Integrating machine learning into process mining techniques enables the identification of patterns, anomaly detection, classification, noise reduction, data transformation and prediction based on event logs. Machine learning can offer advanced insights and automation, making process mining more effective in handling complex and large-scale datasets [44].
- Predictive Analytics: Predictive analytics leverages historical process data to make forecasts about future process performance. This technique can anticipate potential issues, bottlenecks, and trends, allowing to take proactive measures and make informed decisions.

Since process mining is an emerging topic, several commercial and open source software tools have been developed:

ProM [45] is an open source software tool (GNU Public License (GPL)) developed with Java. It is an extensible framework that supports a wide variety of process mining techniques in the form of plug-ins that can be loaded on demand of the user. Although ProM has many capabilities, it does not address the business users and has been mainly used for academic and research projects.

Disco (Fluxicon) [46] is a commercial tool with academic license providing fewer capabilities. For example, it does not discover process model types like Petri-Nets, BPMNs, etc., while it does not support conformance checking. Disco provides many visualizations and an easy user interface for filtering event logs.

PMTK [47] is a web-based toolkit, provided as stand-alone application, built on top of the open-source python library for process mining pm4py, which was developed by Fraunhofer Institute for Applied Information Technology (FIT). It is user friendly, with file conversion capabilities, extensive filtering functionalities, visualizations for event log exploration, performance analysis and automatic discovery of process maps and BPMN models.

Many other companies such as Software AG [48] and Celonis [49] provide commercial solutions to enterprises for their business transformation including process mining techniques. In [50] a detailed presentation and comparative analysis among several process mining tools, is conducted.

2.2 Data Encoding

In recent years, a growing body of literature has examined the state of the art of AAL domain by different points of view. Human Activity Recognition from AAL environments has reached a high level of accuracy. So the objective of this work is not to recognize activities from sensor values but start in the level that the sequences of activities are known and labeled. In this research human activity data exist in textual form. Text encoding is a process that converts meaningful text into numerical or vector representations, preserving the context and relationships between words and sentences. This enables machines to discern patterns within the text and understand the context of sentences. A variety of encoding techniques have been developed to facilitate the processing of textual data within models. In this study, we have opted to utilize Keras Tokenizer and Label Encoding techniques as scalar transformations for our data, while One Hot Encoding and Bert Tokenizer have been selected for a more comprehensive exploration of vector transformations. The quality of data significantly influences the outcomes of any research; thus, it is crucial to preprocess it effectively and examine many options.

Human activity event logs in this work are encoded with Keras Tokenizer, Label Encoding, One-Hot-Encoding or Bert Tokenizer. The Keras Tokenizer changes activities into numbers based on frequency. It's a powerful tool in the Keras library that works smoothly with Theano and TensorFlow by Google. This makes it easy and fast to use because of its straightforward capabilities. Label Encoder technique is implemented in scikit-learn library of python and Encode activities with value from 0 to number of activities minus one. One Hot Encoding converts activities to a vector. In One Hot Encoding, each unique activity or word is represented by a vector where the dimensionality is equal to the size of the vocabulary (number of distinct activities). The vast majority of elements in this vector are zeros, making it sparse, and only the element corresponding to the specific activity is assigned a value of one. One Hot Encoding can result in high-dimensional sparse vectors, especially for event logs with high number of distinct activities [51]. BERT (Bidirectional Encoder Representations from Transformers) Tokenizer, is a state-of-the-art NLP technique introduced by authors in [52], which outperforms previous methods on various NLP tasks and datasets. BERT Tokenizer is used on models that belongs to the family of transformers, the current state-of-the-art models dealing with sequences. BERT Tokenizer converts activities to vectors based on a pre-trained vocabulary, and it typically produces vectors with smaller dimensions compared to one-hot encoding. Unlike one-hot encoding, which creates high-dimensional sparse vectors, BERT Tokenizer uses pre-trained word embeddings that capture semantic relationships between activities.

2.3 Change Point Detection

Change point detection is the problem of estimating the point at which the statistical properties of a sequence of observations change. The challenge in multiple change point detection is identifying the optimal number and location of change points as the number of solutions increases rapidly with the size of the data [24].

Detection methods are expressed as the combination of the following three elements [53].

- Cost function. The cost function is a measure of “homogeneity”. Its choice encodes the type of changes that can be detected - low value if the signal “homogeneous” (meaning that it does not contain any change point), and large if the signal is “heterogeneous” (meaning that it contains one or several change points).
- Search method. Resolution procedure for the discrete optimization problems associated with finding knowing or unknowing number of change points (in an exact or approximate fashion). Each method strikes a balance between computational complexity and accuracy.
- Constraint (on the number of change points). When the number of changes is unknown, a constraint is added, in the form of a complexity penalty (pen), to balance out the goodness-of-fit. The choice of the complexity penalty is related to the amplitude of the changes to detect: with too “small” a penalty (compared to the goodness-of-fit), many change points are detected, even those that are the result of noise. Conversely, too much penalization only detects the most significant changes, or even none.

Change point detection techniques are divided into two main branches: online methods that aim to detect changes as soon as they occur in a real-time setting, and offline methods that detect changes after all samples have been collected [54].

Change point detection has been a subject of study for many years. An early technique, referred to as CUSUM [55], is capable of identifying changes in univariate time series data. However, it relies on the assumption that the data adheres to a normal distribution with known parameters, and its scope is limited to detecting changes in the mean. With the help of advanced search algorithms , new change point detection algorithms based on cost functions can detect multiple (rather than single) change points. In [56], Dynamic Programming is introduced as a method for determining the optimal segmentation of data into segments with homogeneous statistical properties. In [57], PELT (Pruned Exact Linear Time) is presented, an efficiency algorithm that can identify change points in time series data, and it offers advantages such as its ability to operate in linear time complexity. However, PELT assumes a known distribution and is primarily suited for univariate time series data, where it focuses on changes in the mean. Another change point

detection algorithm is Binary Segmentation (BinSeg) [58], which identifies structural changes in time series data. Binary Segmentation is known for its simplicity and effectiveness in detecting change points by recursively dividing the time series into segments and testing for abrupt shifts within each segment. However, it's important to acknowledge that its performance can be influenced by factors like segment length and the choice of change point criteria.

2.3.1 Pelt

Pelt algorithm relies on a pruning rule because the enumeration of all possible partitions is impossible. Many indexes are discarded, greatly reducing the computational cost while retaining the ability to find the optimal segmentation. In addition, under certain conditions on the change point repartition, the average computational complexity is of the order of O(CKn), where K is the number of change points to detect, n the number of samples and C the complexity of calling the considered cost function on one sub-signal. Consequently, piecewise constant models (model=l2) are significantly faster than linear or autoregressive models. To reduce the computational cost, only a subsample of possible change point indexes can be considered. PELT is an improvement of the dynamic programming approach. Its idea is that, if the cost function satisfies some properties, then some iterations can be skipped, and this makes the algorithm much faster. Implementing Pelt with python library called ruptures [54] 3 main parameters exist:

- Penalty: This parameter is used to control the trade-off between the number of detected change points and the goodness of fit (cost). Essentially, it influences the algorithm's sensitivity to detecting change points. A higher penalty value will encourage the algorithm to detect fewer change points, favoring simpler models with larger, more homogeneous segments. On the other hand, a lower penalty value will allow the algorithm to detect more change points, potentially capturing smaller variations in the data.
- Min Size: This parameter determines the minimum distance between change points.
- Jump: This parameter controls the grid of possible change points; for example, if jump=k, only changes at k, 2k, 3k, etc. are taken into consideration.

2.3.2 Dynamic Programing

Dynamic Programming determines the exact minimum of the sum of costs by evaluating the cost of all subsequences within a given signal. The term "dynamic programming" is attributed to the ordered exploration of all potential segmentations using a dynamic programming approach. To utilize this method, users are required to specify the number of changes they intend to detect in advance. The complexity of the dynamic programming approach is on the order of $O(CKn^2)$, where K represents the number of change points to be detected, n is the number of samples, and C is the complexity associated with calling the specified cost function on a single sub-signal.

Consequently, models employing piecewise constant behavior (model=l2) exhibit significantly faster computation compared to linear or autoregressive models.

To mitigate computational costs, users have the option to consider only a subset of potential change point indexes. This can be achieved by adjusting the min_size and jump parameters during the instantiation of Dynp:

- min_size determines the minimum distance between change points; for instance, setting min_size=10 ensures that all change points are at least 10 samples apart.
- jump controls the grid of possible change points; for example, if jump=k, only changes at k, 2k, 3k,... are taken into consideration.

2.3.3 Binary Segmentation

Binary change point detection facilitates rapid signal segmentation. This approach follows a sequential process: initially, a single change point is identified in the entire input signal, the series is then partitioned around this change point, and the operation is reiterated on the resulting sub-signals. The advantages of binary segmentation encompass its low complexity (on the order of O(Cnlogn), where n is the number of samples, and C the complexity of calling the specified cost function on one sub-signal), its capability to extend any single change point detection method for detecting multiple change points, and its adaptability to scenarios where the number of regimes is either known in advance or not. Implementing Binary Segmentation with python library called ruptures 3 main parameters exist:

- Penalty: This parameter is used to control the trade-off between the number of detected change points and the goodness of fit (cost). Essentially, it influences the algorithm's sensitivity to detecting change points. A higher penalty value will encourage the algorithm to detect fewer change points, favoring simpler models with larger, more homogeneous segments. On the other hand, a lower penalty value will allow the algorithm to detect more change points, potentially capturing smaller variations in the data.
- Min Size: This parameter determines the minimum distance between change points.
- Jump: This parameter controls the grid of possible change points; for example, if jump=k, only changes at k, 2k, 3k, etc. are taken into consideration.

2.4 Clustering

Clustering [59], a fundamental concept in data analysis and machine learning, plays a pivotal role in organizing and interpreting large datasets. At its core, clustering involves grouping similar data points together based on certain criteria, thereby revealing inherent patterns and structures that

might not be immediately apparent. This unsupervised learning technique has found application in diverse fields, from marketing and biology to finance and image analysis.

Clustering is used in order to gain important insights from data by observing what groups (or clusters) the data points fall into when they apply a clustering algorithm to the data. By definition, unsupervised learning is a type of machine learning that searches for patterns in a data set with no pre-existing labels and a minimum of human intervention. Clustering can also be used for anomaly detection to find data points that are not part of any cluster, or outliers.

In process mining [60] clustering serves a wide range of applications, from understanding process behavior and performance to facilitating root cause analysis and guiding process improvements. It is a versatile technique that enhances the extraction of meaningful insights from complex processes.

2.4.1 Kmeans

K-means [61] is a widely utilized clustering algorithm in the field of data analysis and machine learning. The primary goal of K-means is to partition a dataset into K distinct clusters, with each cluster represented by a centroid, which serves as the mean of the data points within that cluster. The algorithm follows an iterative process, starting with the random selection of K data points as initial centroids. It then proceeds through alternating steps of assigning each data point to the cluster with the nearest centroid and updating the centroids based on the current assignment. This process repeats until convergence, where the assignment of data points to clusters stabilizes.

A crucial aspect of implementing K-means is the determination of the number of clusters, denoted as K [62]. This can significantly impact the clustering results. Various methods are employed for this purpose, such as the elbow method, where the within-cluster sum of squares (WCSS) is plotted against the number of clusters to identify an "elbow" point indicating an optimal K. Silhouette analysis is another method that assesses the cohesion and separation of clusters to gauge the appropriateness of a given K.

Centroids, representing the center of each cluster, play a pivotal role in the K-means algorithm. The calculation of distances between data points and centroids, often using Euclidean distance, determines the assignment of points to clusters. Convergence is reached when subsequent iterations result in minimal or no changes in the assignment of data points to clusters.

K-means is employed in anomaly detection, identifying outliers or anomalies in a dataset by considering data points that deviate significantly from cluster centroids.

2.4.2 DBSCAN

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [63] is a widely employed clustering algorithm in data analysis and machine learning, particularly valued for its ability to identify clusters based on the density distribution of data points in the feature space. Unlike algorithms such as K-means, DBSCAN doesn't necessitate the prior specification of the number of clusters, offering adaptability to datasets with varying cluster densities and shapes.

The fundamental premise of DBSCAN lies in the classification of data points into core points, border points, and noise points. A core point is one with a minimum number of neighboring points (MinPts) within a specified distance (Epsilon, ε), designating it as part of a dense region. Border points are in proximity to core points but do not meet the density criterion, while noise points are those that do not belong to any cluster.

One of the key strengths of DBSCAN is its flexibility in identifying clusters with irregular shapes, making it well-suited for datasets where traditional algorithms might struggle. It's also robust to outliers, effectively categorizing them as noise points.

In practical applications, DBSCAN finds utility in various domains. Spatial data analysis benefits from its capability to identify clusters with varying spatial densities. The algorithm's proficiency in detecting outliers makes it valuable for anomaly detection in datasets where noise points may signify irregularities. Furthermore, DBSCAN is applied in image segmentation, where it can segment images based on pixel similarity, aiding computer vision tasks.

DBSCAN stands out as a robust and flexible clustering algorithm, well-suited for datasets with irregularly shaped clusters and varying densities.

Chapter 3

3.1 State of the Art analysis

The research area of ambient assisted living has led to the development of activity recognition systems (ARS) based on human activity recognition (HAR). These systems improve the quality of life and the health care of the elderly and dependent people. A significant amount of work has been done on human activity recognition and researchers have leveraged different approaches [64], such as sensor based approaches [65], wearable device based [66], vision based [67], to recognize human activities.

While considerable research has been conducted in the realm of human activity recognition, there remain unresolved issues that warrant further attention and investigation. The need for a broader contextual understanding of human activities within complex processes made Process

Mining essential to be used in human activity recognition even though it was first introduced as a novel discipline in the business domain [68,69,70,71,72]. The idea of process mining was firstly introduced as workflow mining by Cook and Wolf in [73] who conducted a research on how to extract a sequence of activities in software development environment. In this experiment, process mining was used to describe the human activity model in a whole day based on the recorded action sequences. Process Mining brings a sequential understanding by visualizing how individual activities relate to each other in the context of larger processes, offering insights into the flow and order of activities [74]. Process Mining allows for the identification of bottlenecks and inefficiencies in processes [75], which can be applied to HAR to recognize suboptimal sequences or areas in human activity execution that need improvement. Process Mining excels in identifying patterns and variations in the execution of processes. By applying similar analyses to HAR, it becomes possible to recognize changes and variations in human activities, contributing to more adaptive models. In [74] the eMotiva process mining algorithms are introduced. These algorithms serve to filter, deduce, and depict workflows. The workflows are deduced from samples generated by an indoor location system, which records the whereabouts of a resident in a nursing home. The visualization tool is proficient in comparing and accentuating behavioral patterns, aiming to enhance the comprehension of human behavior by experts.

Encompassing techniques, methodologies, and tools, process mining involves the analysis and enhancement of processes across various applications. This includes automated knowledge discovery from data, compliance verification with regulations, and the selection of the most appropriate model to represent intricate processes based on event sequences, timing, and involved resources. The process mining approach facilitates the identification of patterns concealed within extensive datasets and assesses process performance, aiming to drive improvements and generate predictions. Over the past decade, process mining techniques have found widespread applications in the realm of human activity recognition [76] and healthcare processes [77]. These applications span a diverse range, including the discovery of process models, analysis of social interactions, and conformance checking.

Process mining can be seamlessly applied to the realm of Activities of Daily Living (ADL), offering a transformative approach to understanding and optimizing individual routines. In the context of healthcare and assisted living, process mining involves the extraction of valuable insights from data collected through sensors, wearables, or monitoring devices, capturing the intricacies of daily activities. By creating an event log that represents each ADL as a distinct entry, process mining techniques can then unravel patterns, sequences, and variations in how individuals perform routine tasks.

There is a number of ADL-based AAL systems that have been developed to deal with the challenge of elderly wellbeing monitoring. In [78] authors developed a framework for ADL monitoring for

elderly subjects affected by diabetes and from ADL analysis, recognizing the Wellness level. In [79] authors offered a Centinela system, which monitors activity related to gait movements with the application of mobile and other wearable devices. In [80] an exceptional system based on heterogeneous wireless sensors networks investigated and designed to detect the behavioral pattern and forecast the anomaly in behavior. A multi-layered healthcare platform developed in [81] to monitor the activities associated with depression and diabetes, named the long term monitoring research system.

Existing solutions are focused on recognizing activities which are normal daily life activities such as sitting, standing, sleeping, walking, and eating. Recognition of these activities is important but that is not the whole purpose of activity recognition, especially for those applications which intend to identify abnormal activities. Detection of abnormal activities is of great importance in applications like security and healthcare. Additionally, there are many activities which are composite and consist of multiple simple activities. Moreover, human movements can vary widely among individuals performing the same activity, where activities may occur in unstructured environments with varying levels of complexity.

As human behavior is quite complex, unpredictable, and inherently dynamic, there is a need for smart solutions which can simplify complex models by leveraging visual representations to convey intricate details in a more accessible and intuitive manner. Addressing this, visual analytics holds the potential to enhance process mining, focusing on explainability, interpretability, and trustworthiness to support human decision-making. In [82], a method is presented which leverages visual analytics within the realm of process mining, aiming to deliver analytics results for business processes that are both explainable and interpretable. In [83], an innovative visual analytics approach is introduced with the goal of fostering a comprehensive understanding of user behavior by analyzing sequences of user activities. This approach involves a unique amalgamation of "action space" analysis, pattern mining, and the interactive visual analysis of multiple sequences. These combined techniques represent a pioneering effort to lay the groundwork for a more thorough comprehension of user behavior. The authors in [84] present a visual approach designed to assist process analysts in their efforts, contributing valuable perspectives on enhancing processes. Their work introduces visual tools and methodologies that aim to facilitate the identification and exploration of process improvement opportunities. The aim of research in [82] is to explore the applicability of production process visualization methods using process mining techniques for production process analysis. The authors delve into the intricate intersection of these techniques, providing insights into how the fusion of visual analytics and process mining can yield enhanced understanding and actionable insights from complex process data.

The complexity of data and results, coupled with communication requirements, has amplified the interest in enhancing usability and understandability for all participants.

Both Natural Language Processing (NLP) and Human Activity Recognition (HAR) domains within smart homes handle data in the format of sequences. In the context of smart homes, sensors produce a continuous flow of events that follow a sequential and ordered pattern, akin to words in a text. Authors in [85] introduces a comprehensive framework for recognizing daily activities in smart homes, integrating a time series classifier with Natural Language Processing (NLP) word encoding. The authors specifically employed frequency-based encoding coupled with word embedding to augment feature learning. This approach facilitated the incorporation of domain knowledge, ultimately enhancing the performance of activity recognition. In [86] a system for inferring user medication through the application of Natural Language Processing (NLP) techniques is presented. The authors framed the issue as a ranking task, wherein standard medication names (SMN) are mapped to descriptive medication phrases (DMP) by arranging the medications listed in a patient's prescription from pharmacies. Additionally, they leveraged the outputs of intermediate layers and conducted medication-related analyses.

For enabling a more nuanced analysis of human behavior, Change Point Detection and Clustering techniques have been applied in the research field. A change-point analysis exhibits the capability to identify multiple changes in [55], offering comprehensive insights for each alteration. It furnishes detailed information for each change, encompassing a confidence level that signifies the probability of a change occurrence and a confidence interval pinpointing when the change transpired. Notably flexible, this change-point analysis procedure can be applied to diverse types of time-ordered data, encompassing attribute data, non-normally distributed data, challenging data sets like particle counts and complaint data, and data containing outliers. In [87], a technique is introduced for acquiring optimal change point segmentations in data sequences across a continuous range of penalty values. This enables the assessment of diverse segmentations, aiding in the identification of a judiciously parsimonious penalty choice. A real-time machine learning approach is presented in [28], for the automatic segmentation and recognition of continuous human daily activities. We identify transitions between activities and seamlessly integrate a change point detection algorithm with smart home activity recognition. This integration allows for the precise segmentation of human daily activities into distinct actions, ensuring accurate identification of each action. The authors in [88] introduce a strategy for online change point detection (OCPD) aimed at segmenting continuous multivariate time-series smartphone sensor data. The application of this strategy is demonstrated in the context of a transition-aware activity recognition framework. The OCPD strategy proposed here operates on the principle of hypothesis-and-verification. Following the segmentation of the online data stream using this OCPD strategy, feature engineering is conducted to preserve the crucial

features. To detect changes in text structures, a multinomial change-point model is introduced in [89].

In this study, our primary focus revolves around the challenge of offline change-point detection, wherein the entire dataset is accessible to the analyst in advance. Numerous methodologies in this domain have been thoroughly examined over the past few decades. In [90], the authors demonstrate the automatic generation of novel offline detection methods by training a neural network for identifying change-points in data. The article also introduces theoretical foundations that quantify the error rate associated with this approach and elucidates its dependence on the volume of training data. Moving to [54], Anomaly Detection emerges as a pivotal focus in time series data analysis, with widespread applications across various domains. Acknowledging the dynamic nature of real-world environments and the temporal evolution of data distribution, commonly referred to as concept drift, is essential. In [91], a method employing change point detection is implemented to estimate the time instances of statistically significant changes in terrorism-related time series. This approach utilizes a set of indicators for a comprehensive analysis of trends and variations in a criminal context. In the present study, terrorism-related content and the manifestation of hate speech are identified using Convolutional Neural Network, serving as inputs in the Change Point Detection algorithm. Notably, all these methodologies outlined are not specifically applied to text data in human activity recognition.

Approaching the issue from a different angle, the task of change point detection can be conceptualized as a clustering problem, irrespective of whether the number of clusters is known or unknown. Within this framework, observations within clusters exhibit identical distributions, while observations between adjacent clusters do not. The identification of a distinct cluster for a data point at time stamp "t" compared to the data point at time stamp "t + 1" signifies the presence of a change point between these two observations.

In [92], a novel scheme for multi-sensor activity recognition, termed Wavelet Tensor Fuzzy Clustering Scheme (WTFCS), is introduced. Meanwhile, in [93], the focus is on unsupervised human activity recognition, aiming to deduce activities from unlabeled datasets without relying on domain knowledge. The proposed solution in this study involves an end-to-end multi-task deep clustering framework designed to address the problem at hand.

Chapter 4

PhD Methodology

4.1 Process Mining for Activities of Daily Living in Smart Homecare

4.1.1 Introduction

The aim of this section at hand is to apply process mining algorithms for ADL in a smart homecare context addressed to residents in order to capture patterns representing their physical or mental health conditions, to recognize when activity patterns begin to deviate from the norm, to make useful conclusions about their daily living, and to interpret the results.

4.1.2 Methodology

In terms of human behavior, a process is a sequence of activities performed by human on daily basis. Following the process mining background theory [30], the adopted methodology consists of five steps: Extract Event Log, Filter Event Log, Discover Process Model, Conformance Checking, and Performance Analysis (see Figure 4.1).

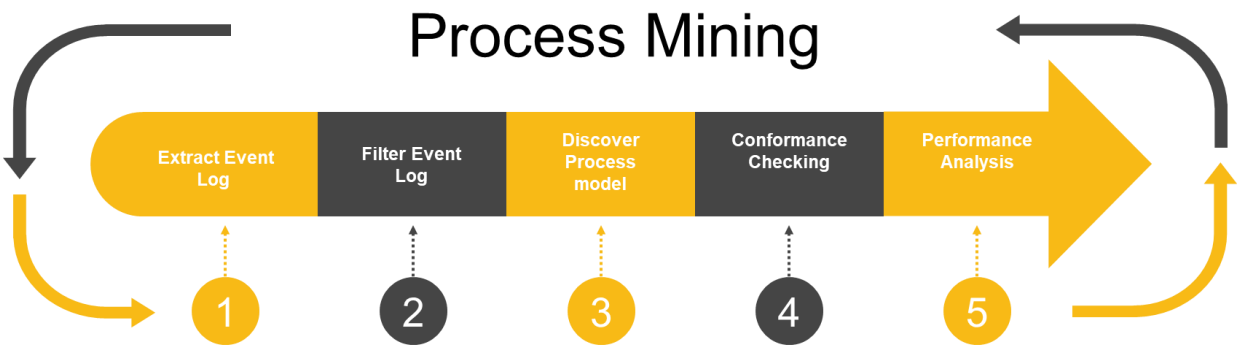


Figure 4.1 Process Mining approach on ADL in smart homecare methodology

Extract Event Logs: The starting point is to extract the event logs related to the ADLs monitored by the home-equipped sensors. Event log files are not always in the desired format. The event log must include a unique number for each day, the activity that was performed by the resident and the timestamp.

Filter Event Logs: The second step is the event log filtering. Filtering is conducted for two main reasons: cleaning the data or narrowing down the analysis. In smart homes, ADLs are identified through various sensors, which may produce noisy data due to malfunctions or hardware failures, resulting in data quality issues (e.g. missing or erroneously recorded attributes or events).

Discover Process Model: A discovery technique takes an event log and produces a process model without using any a-priori information in order to reproduce the traces observed in the event log. In the adopted research methodology, we implement and compare Alpha Miner, Heuristic Miner, Fuzzy Miner, and Inductive Miner.

Conformance Checking: Conformance checking measures the differences between the performed process and the process model specifications. The main aim of this process is to identify the areas that need improvement using the information gained from the actual process [43]. To measure the quality of discovery in process mining algorithms, it is necessary to establish a balance between four competitive criteria [94]: fitness, simplicity, precision, and generality.

Performance Analysis: The goal of performance analysis is process enhancement. Time information is projected from the event log to process model to uncover bottlenecks, deviation, service times, throughput time, waiting times and execution times of the process.

4.2 Develop visualizations and software application to analyze event logs

4.2.1 Visual Analytics in Process Mining for Supporting Process Improvement

4.2.1.1 Introduction

The increasing amounts of data have significantly impacted conceptual modeling within the research field, with the integration of data analytics for generating process models. In this context, process mining encompasses a set of techniques aimed at deriving a process schema from an event log generated during process execution. While automatic algorithms are essential for mining and analyzing processes, filtering out irrelevant data and producing initial results, the proper interpretation of these results requires visual inspection, domain knowledge, human judgment, and creativity. Additionally, process discovery often yields complex and extensive process models that may pose comprehension challenges for users. Addressing this, visual analytics holds the potential to enhance process mining, focusing on explainability, interpretability, and trustworthiness to support human decision-making.

4.2.1.2 Methodology

This section aims at identifying bottlenecks in processes by taking advantage of the large amounts of event logs becoming available through various smart environments. In order to tackle the complexity of the process models extracted by the event logs, we propose a visual analytics approach that facilitates interpretability and explainability of the results in order to support human behavior analysis. In this way, the human expert is in the loop having the capability of taking informed decisions. The proposed approach incorporates an incremental and iterative way

and provides interactive visualization capabilities in order to facilitate the interaction with the human. The proposed approach consists of the following steps:

(a) Event Log Extraction: The starting point is to extract the event logs. In smart homes data comes from sensors and must be labeled correctly in order to use them. In order to monitor activities of daily living the trace id will be the day id that comes from timestamps, the event will be the activity and the timestamp is the mandatory attributes that is needed.
(b) Filtering event logs: To improve the data quality, it is essential to eliminate faulty and anomalous activities. The process of filtering serves two primary purposes: data cleansing and refining the scope of analysis. Filtering can involve actions such as eliminating a process instance (case), adding events, removing events, or altering events.
(c) Traces and Activities: In this step the visualization results offer insights into the activities performed over time, along with the associated traces. Based on these outcomes, the user gains a comprehensive overview of the dataset. Colors can highlight patterns on daily basis or on specific periods, such as non-working days.
(d) Duration of the activities: In this step, the visualization results provide information about the duration of activities over time. The duration of activities is a crucial parameter for imparting the correct significance to the entire process. The shortest and the longest duration can reveal useful information about problematic traces and help improve the process performance.
(e) Mean Duration of activities: In this step, the visualization results provide information about the mean duration of each activity. This visualization provides insights into how long, on average, it takes to complete a particular activity, allowing for an assessment of whether this duration falls within the expected or acceptable range.
(f) Trace Duration: In this step, the visualization results detect the total time of completion of the traces in order to identify the most long-lasting traces as well as their patterns over the time period under examination. Long-lasting days, which can be exhausting for the occupants of the household, have the potential to adversely affect their health.
(g) User cognition: In the final step, the user aggregates the aforementioned visualization results in order to make decisions about the efficiency of the processes as well as about actions for improvement. The human decisions are supported by the results of visual analytics making capable of taking advantage of the large amounts of event logs generated by information systems without requiring advanced data analytics and machine learning skills.

4.2.2 Smyrida: A web application for process mining and interactive visualization

4.2.2.1 Introduction

Despite the continuous demand for the integration of advanced data analytics into a comprehensive framework, this requirement has yet to be met. There is a necessity for a modular software system that can adapt to new techniques and visualizations suitable for process mining.

"Smyrida" is a modular software system in the form of a web application with open APIs making it adaptive to new techniques that can be applied for process mining and capable of being integrated to information systems producing event logs. From the user perspective, there is the need for providing interactive intuitive visualization capabilities, and allowing users without advanced process mining skills to extract process models and evaluate their processes.

Smyrida covers the whole process mining lifecycle, i.e. importing events logs, descriptive statistics about the event log (e.g. number of traces, number of events, and list of events with the corresponding frequency), visualizations, process discovery, conformance checking, and performance analysis. The objective of this software tool is twofold. From the user's perspective, its user interface is designed with a focus on usability, since it is capable of producing interactive visualizations and draggable process models. From a developer's perspective, Smyrida embraces a modular architecture, facilitating extensibility with the inclusion of new process mining algorithms. Furthermore, it operates as an open-source tool, ensuring transparency in the implementation of all algorithms and techniques, thus enabling algorithm customization.

In section 2.2 we analyzed Process Mining Tools. In Table 4.1, we provide a brief comparative analysis between Smyrida and the most relevant not commercial tools. We must note that we compare to PROM[1], academic Disco[2] and PMTK[3] stand-alone version.

Table 4.1 Comparison between Smyrida and the most relevant not commercial tools.

	PROM	**Disco**	**PMTK**	**Smyrida**
File conversions	✓	✓	✓	✓
Filtering	✓	✓	✓	X
Client–Server	X	X	X	✓

[1] https://promtools.org/
[2] https://fluxicon.com/disco/
[3] https://pmtk.fit.fraunhofer.de/

API/Web service support	X	X	X	✓
Conformance checking	✓	X	X	✓
Visualizations	✓	✓	✓	✓
Statistics	✓	✓	✓	✓

4.2.2.2 Software Description

"Smyrida" incorporates process mining algorithms and provides intuitive interfaces for configuration and interactive visualization. In this way, it covers the whole process mining pipeline. Specifically, it uses event logs to extract information from the process and produces automatically process models that depict the real process. Event logs include at least case id (process instance), event that occurred and timestamp of the event end (event log). Optionally more attributes of the event can be recorded (such as start timestamp of event, resource, category, user, etc.). Process discovery currently incorporates three algorithms: Alpha miner, Heuristics miner, and Inductive miner. Alpha miner was one of the first process discovery algorithms that could adequately deal with concurrency. However, it suffers from sensitivity to noise, infrequent/incomplete behavior, and complex routing constructs. Heuristics miner takes frequencies of events and sequences into account and therefore it can filter out a noisy or infrequent behavior. It is also able to detect short loops and to allow skipping of single activities. Inductive miner provides flexibility, formal guarantees and scalability. It works recursively with divide and conquer strategy: split log, construct part of process tree, and proceed with handling split parts of log separately.

Using event log and the produced process model, conformance checking techniques were developed to analyze how much the discovered process model is compliant with the event log so as to find deadlocks or congestions. Then, performance analysis techniques optimize process and eliminate execution delays. Evaluation metrics can measure simplicity, generalization, fitness and precision of the process models. In addition to that, visualizations from event log contribute to process explainability. Smyrida operational architecture is depicted in Figure 4.2

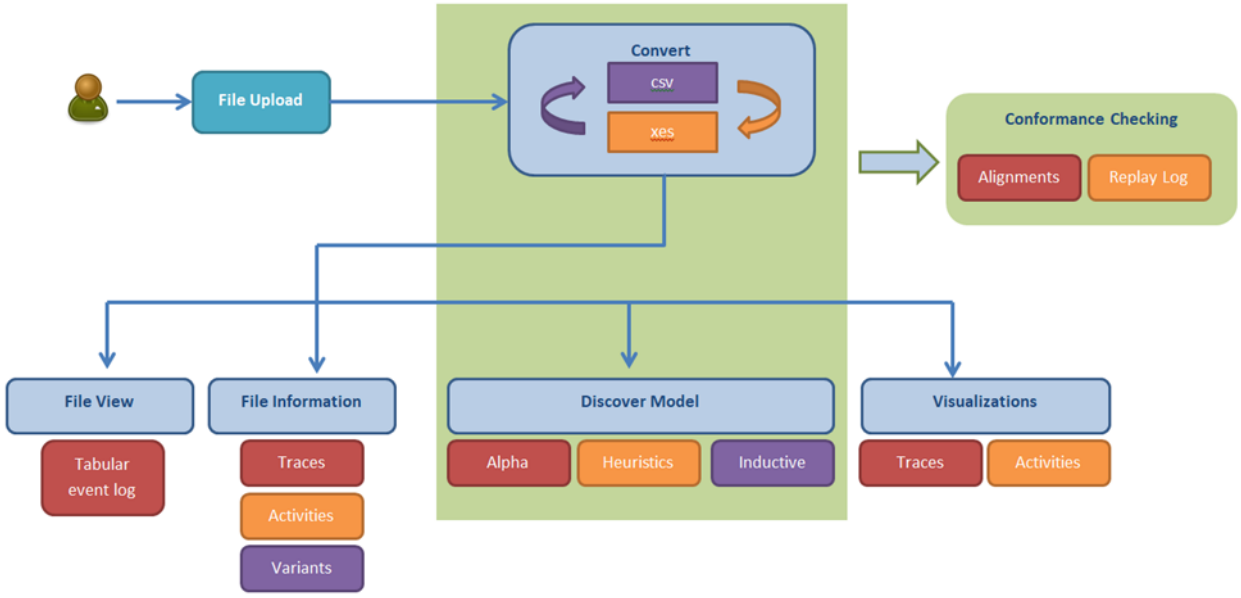


Figure 4.2 Software Operations

4.2.2.3 Software Architecture

The technical architecture of the "Smyrida" web application is depicted in Figure 4.3. Server side consist of a Flask (python) API backend with React js frontend. Sqllite3 database is used for user credentials. The application can be deployed in Ubuntu Server. Flask API is served by gunicorn. Gunicorn (Green Unicorn) is a Python WSGI HTTP Server for UNIX. Gunicorn is waiting for requests on the socket file in the project directory. Because Gunicorn cannot face the web we need Nginx web server to pass web requests to that socket. Nginx functions as a reverse proxy and also serve static files. Since the application is going to run on a production server, we have to make sure that it is always running. For this reason we used systemd, which is a process monitoring system of Ubuntu that is dedicated to the purpose of keeping services running. For each user action, a request is sent from React js to flask API endpoint (either with POST or GET request) and the API responds to React js with the corresponding data. React js application present data to the user.

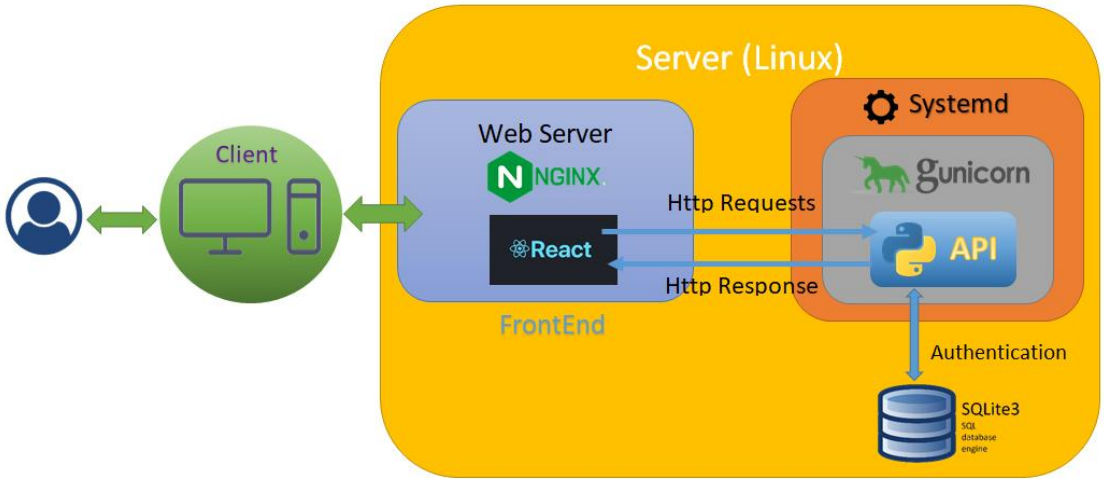


Figure 4.3 The "Smyrida" Software Architecture.

Flask API uses various python libraries. Pm4py for process mining, Plotly express for visualization, Pandas for data manipulation, Sqlite3 database connectivity for user storage. Flask CORS are enabled in order the API can be used from different source. React js provides the user interface. As the user navigate through the application menu and submit data, React js sends request to the Flask API. The API responds with data (mainly in json format) and React present them to the user. Figure 4.4 depicts some indicative examples of http requests and responses.

Smyrida transforms data structures to json format to meet API requests and web interface requirements. To do this, we built the process model outline, the interactive visualization, the user authentication, the data representation, and the communication mechanisms in React js. The produced process models (implemented on React js from raw data) are draggable (transitions and places individually and all together) and can be zoomed in and out.

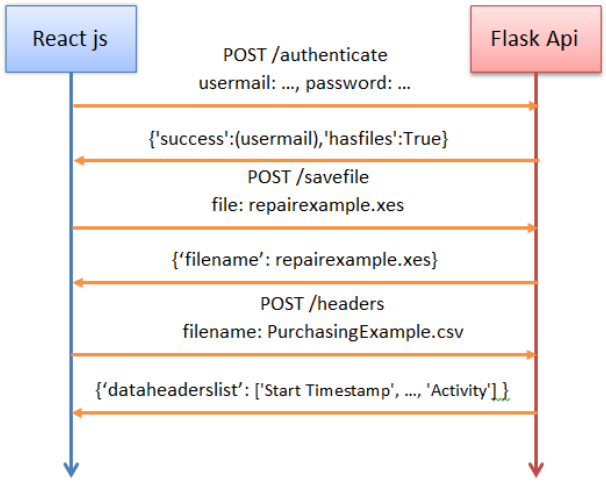


Figure 4.4 Requests and responses between React js and Flask Api.

4.2.2.4 Software Functionalities

The main functionalities of the "Smyrida" software tool are[4]:

User authentication: The user must first sign-up and be authenticated to be able to use the application.

File upload: The user can upload and retain (on server) event log in csv, txt, xes or xes.gz format

File conversion: The user can convert a xes (or xes.gz) file to csv and vice versa.

Event Log exploration: The user can view the content of the uploaded files and various information about the event log such as: number of traces, number of events, occurrences of each event, start and end evets, trace variances.

Process discovery: Currently, the algorithms that have been implemented for automatically discovering process models from event logs are: Alpha (+) Miner, Heuristics Miner and Inductive Miner.

Conformance Checking: Token-based replay and alignments techniques are embedded for performing conformance checking.

Evaluation: Measurement of fitness, precision, generalization and simplicity of process models.

Visualization graphs: The graphs that are exposed to the user constitute the outcomes of visual analytics in the process mining context. Such approaches are capable of providing explainable and interpretable analytics results for business processes in an intuitive and non-intrusive way [75]. "Smyrida" provides the following graphs:

- Activities: Bar plot with event frequencies, Bar plot with mean duration of each activity, Bar plot with mean duration of each activity and frequency, Scatter plot with events throughout time (colored by trace), Bubble plot of activities duration throughout time (colored by activities).
- Traces: Traces variants (colored by activities), Horizontal bar plot with traces variants frequency, Bubble plot with trace duration throughout time (colored by trace)
- Other: Scatter plot with activities and traces

[4] https://smpi-islab-uniwa.github.io/Smyrida/

Smyrida is comparable to open source tools and prevails the others software due to its architecture. The open-source approach contributes to the creation of a collaborative eco-system that allows researchers and practitioners to share code and results with the process mining community.

Arguably, the most relevant tool is PMTK toolkit. Similarly to Smyrida, it is a web-based toolkit, provided as stand-alone application, built on top of the open-source python library for process mining pm4py, which was developed by Fraunhofer Institute for Applied Information Technology (FIT). It is user friendly, with file conversion capabilities, extensive filtering functionalities, visualizations for event log exploration, performance analysis and automatic discovery of process maps and BPMN models. PMTK in contrast to Smyrida does not support xes.gz file types which results in the loading of large files being delayed. In comparison to Smyrida, PMTK does not include additional process discovery algorithms (such as Alpha Miner, Heuristic Miner and Inductive Miner). In addition Smyrida builds the process models in a level-based manner in order to be more readable and structural understandable. PMTK lacks the ability of including Conformance Checking techniques, whereas Smyrida implements Replay Log and Alignments Conformance Checking techniques. Also, Smyrida is particularly flexible and extensible because it can accept API calls so as anyone can build his/her own user interface as he/she wishes. Also, Smyrida is a server–client architecture application that can be deployed on an Ubuntu server, totally open-source in conjunction to PMTK that has only a standalone application free to try and the on-premise version that is not free.

4.3 Develop framework to detect changes in human behavior

4.3.1 Introduction

Analyzing human behavior is a complex undertaking, but it holds significant importance for individuals living alone, particularly for the elderly and dependent individuals. Early prediction of human disorders can save lives and instill confidence in individuals living alone. Detecting potential health issues in advance allows for timely intervention and support, contributing to overall well-being and safety.

In the upcoming section, we delve into the analysis of a framework designed to identify disorders in human behavior. It is a Process Mining approach that takes advantage of Change Point Detection and Clustering techniques.

4.3.2 Method framework

In this framework a dataset that contains human activities in text format is extracted as event log. This event log is pre-processed in order to filter-out unwanted behavior and format

appropriately some fields of the event log. Then the n activities of the event log are grouped together as a point. In the next step each point is encoded in order to feed various algorithms. The encoded points feed change point detection algorithms in order to detect change points (List A). Also the encoded points feed clustering algorithms and then if the cluster change from one point to the next one is considered a change point (List B). Then from the combination of List A and List B the final change points are produced from the initial event log, to denote human behavior disorders. Figure 4.5 illustrates the steps of the framework, while Table 4.2 outlines the input and output for each step.

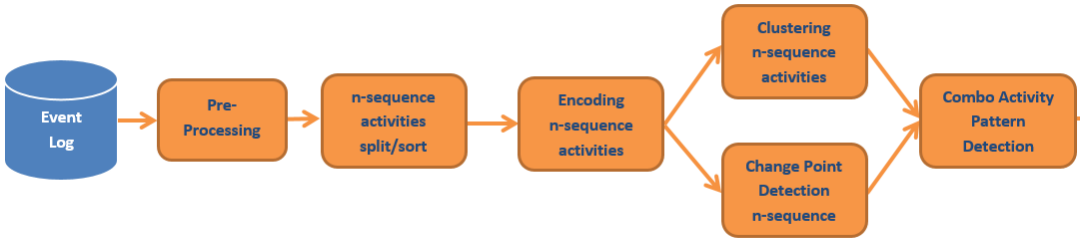


Figure 4.5 Framework Architecture

Table 4.2 Inputs and outputs of each step in the framework

STEP	ACTIONS	Input	Output
Extract event log	A dataset from smart home is converted to event log	Dataset from smart home	Event log
Pre-processing	The event log is filtered and the fields are formatted	Event Log	Transformed event log
N-sequences of activities split/sort	Every n activities is grouped as one point and they are sorted	Transformed event log	Dataframe with n-activities in each row (one point)

Encoding n-sequence of activities	Each point is encoded with an encoding algorithm	Dataframe with point	List with encoded points
Change Point Detection	Change points are detected on the list of the encoded points	List with encoded points	List with change points from change point detection algorithm (List A)
Clustering n-sequence of activities	List of encoded points are clustered. If from one point to the next one the cluster is changed, a change point is considered	List with encoded points	List with change points from change of cluster (List B)
Combo Activity Pattern Detection	From List of encoded points only those that included in List A and in List B are considered as change points	List A and List B	List with final change points

Each step of the framework is analyzed in the following sections.

4.3.2.1 Event Logs

In smart environments, datasets are derived from sensors. In human activity recognition systems using sensors, data are annotated with the associated activity and timestamp. Event logs are files containing recorded event data. Specifically, for each event, they document information about the process instance (trace ID), event name, and the end timestamp of the event. Additional attributes about each event, such as start timestamp and resources, may also be included. In Process Mining, in the context of human activity domain space, an event log consists of activities, including at least the end timestamp of the activity and a day ID, which serves as the trace ID for the event log. In Table 4.3, a segment of an event log is illustrated. An activity has a duration, whereas an event does not. Therefore, when recording an event in the event log, it can signify

either the start or the end of an activity. By definition, we record the end of an activity as an event in the event log.

Table 4.3 A segment of an event log

DayId	Activity	Status	Timestamp
12	snack	complete	2011-11-28 20:20:59+01:00
22	watchingtv	complete	2011-11-29 02:06:51+01:00
22	sleeping	complete	2011-11-29 11:31:04+01:00
22	toilet	complete	2011-11-29 11:37:27+01:00
22	washing	complete	2011-11-29 11:48:29+01:00
22	shower	complete	2011-11-29 11:51:13+01:00

4.3.2.2 Event Log preprocessing

Event logs, like all datasets, may require preprocessing to enhance data quality. In Process Mining, recorded event logs could contain incomplete instances, such as those initiated before the examination start time or not completed by the examination end time. Additionally, data may be filtered to include only a subset of events within the process. Another preprocessing step is to convert timestamp of event log to the appropriate format. The final preprocessing step involves sorting the event log chronologically before passing the data to the next stage. Our framework reads the event log, whether in XES or CSV format, filters incomplete traces, converts timestamps into a suitable format, and eliminates undesired activities from the event log.

4.3.2.3 N-sequences of activities split/short

The data within the event log represents the sequence of activities performed each time the process is executed. A distinct identifier (trace ID) is assigned to each instance of the process run. The sequences of activities in a trace can either be identical or differ from one another. Variants

refer to the unique sequences of activities within the process trace. In structured processes, the number of variants is typically less than the number of instances (traces). However, in unstructured processes, the number of variants may be even equal to the number of instances. For instance, in a well-organized business with stringent policies, the process consistently follows specific paths with each run. Conversely, in the case of human activities, daily routines may involve similar activities but could occur in a different order during various time zones. To effectively address human behavior, a different approach and perspective are required.

Every morning, when a person wakes up, they typically go to the bathroom to wash their face, brush their teeth, and comb their hair. The sequence of these three activities is consistently part of their morning routine, although the order may vary. It's important to note that a change in the order of these activities doesn't necessarily indicate a change in human behavior; all these activities are integral to the morning routine regardless of their sequence. Our approach involves dividing the data into n-sequences (where n is an integer greater than or equal to 1) of activities, treating them as equivalent irrespective of the order in which they occurred. Thus, the event log is segmented into n sequences of activities and arrange each sequence to have the activities in the same order. Each sequence (including n activities) is treated as a point. For example, if we have sequences of activities like "activity1-activity2-activity3" and "activity2-activity1-activity3," they are considered equivalent and sorted as "activity1-activity2-activity3." A log with 15 activities, grouped into sets of 3 (3 sequences of activities), consists of 5 points.

4.3.2.4 Encoding n-sequences of activities

The n-sequences of activities is initially in categorical format. In order to apply Change Point Detection and Clustering techniques each sequence (composed of n activities and considered as a point) must be transformed to numerical values. Each encoding approach has its own trade-offs and impact on Change Point Detection and Clustering. The key is to use an encoding method that effectively captures the relationships between activities, aiding in the identification of changes. In the following section we analyze the effectiveness of encoding techniques concerning the number of change points. Four distinct encoding techniques are applied to diverse human behavior datasets: Keras Tokenizer, Label Encoder, One-Hot Encoder and Bert Tokenizer (bert-base-cased). One-Hot Encoder and Bert Tokenizer specifically focus on activity relationships, representing each point (n-sequence) as a vector. In contrast, Keras Tokenizer and Label Encoder assign a numerical value to each point.

4.3.2.5 Change Point Detection of encoded n-sequence of activities

The encoded n-sequences of activities (points) are fed into various Change Point Detection Algorithms in order to identify changes in human behavior. As the n value increases, the number of points decreases. Various Change Point Detection techniques, such as Pelt, Binary

Segmentation, and Dynamic Programming, are applied in this phase. Fine-tuning the algorithm parameters based on the data is crucial at this stage to derive meaningful results. Sensitivity analysis plays a pivotal role in guiding the selection of appropriate parameter values.

Pelt algorithm advantage is its exceptional speed under specific conditions, making it an excellent choice for handling large datasets. Another positive aspect is its ability to accommodate various penalty functions, even those with curvy and concave shapes. This versatility allows its application in diverse situations. However, in certain scenarios where pruning is not applied, PELT might exhibit a slowdown, resulting in a computational cost that increases quadratically with the number of data points. Hence, it's not universally flawless. Additionally, concerning the use of curvy concave penalty functions, the straightforward method employed by Pelt to update the penalty constant may not always yield the optimal number of change points. While it performs well in some cases, a more sophisticated search approach might be necessary to ensure the most accurate results in others.

Dynamic Programming algorithm employs a dynamic programming approach to efficiently sequence the exploration of all potential segmentations, enhancing the optimization process and yielding precise results. It involves systematically evaluating every conceivable segmentation of a given signal to identify the minimum sum of costs associated with each segmentation. However, a crucial prerequisite for utilizing Dynamic Programming is that users must specify the number of change points in advance. While this requirement may pose limitations in certain scenarios, the method's adaptability and accuracy establish it as a valuable tool for change point detection, particularly when the number of change points is either known or must be specified. The computational complexity of Dynamic Programming serves as another potential constraint, particularly when handling extensive datasets or more intricate cost functions. The algorithm's efficiency may diminish or become impractical for certain applications due to its elevated computational cost.

Binary segmentation operates through a straightforward process: initially, it identifies a single change point within the entire signal. After pinpointing this point, the signal is divided into two segments, and the process is iteratively applied to each of these segments. This continues until no additional change points are discovered or a predetermined stopping criterion is satisfied. The method boasts a low complexity, implying that it demands minimal time and computing power, rendering it suitable for handling large datasets. However, there are drawbacks. Binary segmentation may occasionally overlook change points or erroneously detect them, especially when changes are closely spaced or when the signal is characterized by noise. Furthermore, as a greedy algorithm, it makes decisions based on the most favorable immediate choice without considering the overall impact on the final result. This characteristic can lead to suboptimal solutions in certain cases.

4.3.2.6 Clustering n-sequences of activities

Concurrently with change point detection the encoded n-sequences of activities are fed into various clustering algorithms to ascertain the cluster to which each n-sequence belongs. Over time, as points (n-sequences) transition between clusters, change points are identified when a given point shifts to a different cluster compared to the preceding one. For instance, if at time "t" a 2-sequence (activity1-activity2) belongs to cluster 0 and at time "t+1" the subsequent 2-sequence (activity3-activity4) belongs to cluster 1, a change point is detected. If both 2-sequences belongs to the same cluster no change point is identified. In this phase, the careful selection of clustering parameters is crucial, and a sensitivity analysis is necessary to fine-tune these parameters. Our framework utilizes Kmeans and DBSCAN algorithms for this purpose.

Kmeans is a centroid-based clustering algorithm that aims to partition data into K clusters, where K is a user-defined parameter. DBSCAN is a density-based clustering algorithm that groups together data points that are close to each other and have a sufficient number of neighboring points within a specified radius (eps) and a minimum number of points (minPts) to form a cluster. DBSCAN is robust to outliers since it classifies points with low density as noise points. By using both K-means and DBSCAN clustering algorithms, we can address a wide range of clustering needs. K-means is suitable for scenarios where we have well-separated, isotropic, and convex clusters, while DBSCAN excels in handling datasets with irregular shapes and varying densities, automatically identifying the number of clusters. Together, these two models complement each other, offering solutions for various types of data and clustering requirements, ensuring comprehensive coverage for clustering tasks.

4.3.2.7 Combo activity pattern detection

By combining the outcomes from Change Point Detection (CPD) algorithm and clustering techniques we can detect changes in human behavior. Changes in both CPD and Clustering algorithms assure the alternation in human behavior. The change point detection in our framework is determined through a fusion of change points detected by applied algorithms.

Chapter 5

Results

5.1 Process Mining for Activities of Daily Living in Smart Homecare

5.1.1 Walkthrough example

The proposed methodology in section 4.1 was applied in a real-life dataset that was published by the University of Mannheim in Germany [95].

Extract Event Logs: In our context, the data must be labeled sensor data, comprising activities (i.e. ADLs in smart homecare) along with corresponding timestamps. Indicative activities that exist in the event logs are: watching TV, washing, toilet, grooming, snack, outdoors, sleeping, prepare breakfast, start of the day, eating breakfast, end of the day, shower, eating lunch, prepare lunch, eating dinner, prepare dinner. Each time a process is executed, a process instance is created. Each process instance is called a trace. Each trace describes the life-cycle of a particular case (i.e., a process instance) in terms of the activities executed [96]. In our case the trace id is the day id (case:concept:name) that is extracted from timestamp. A small part of the event log is depicted in Table 5.1. Activity (concept:name) alongside with status (lifecycle:transition) denote an event.

Table 5.1 Part of event log file.

case:concept:name	concept:name	lifecycle:transition	time:timestamp
21	Start	complete	2012-11-12 00:48:38+01:00
21	washing	complete	2012-11-12 00:50:12+01:00
21	watchingtv	complete	2012-11-12 01:52:12+01:00
21	toilet	complete	2012-11-12 01:53:22+01:00
21	washing	complete	2012-11-12 01:53:35+01:00

21	toilet	complete	2012-11-12 01:53:57+01:00

Filter Event Logs: The dataset utilized in our case consistently commences with the activity "Start," indicating the beginning of the day, and concludes with the activity "End," signifying the end of the day. Consequently, traces that lack both the initiation and conclusion represented by the specified activities are excluded through filtering.

Discover Process Model: The filtered event log is used to discover the process model. Here, Alpha Miner, Heuristic Miner, Fuzzy Miner and Inductive Miner are implemented and compared with the use of the ProM open source tool.

The Alpha Miner emerged as an early process discovery algorithm capable of handling concurrency effectively. However, it is crucial to acknowledge that the Alpha Miner may not be considered a highly practical mining method due to its limitations in handling noise, infrequent or incomplete behaviors, and intricate routing structures. While the Alpha Miner is straightforward, several of its concepts have been incorporated into more advanced and resilient techniques. As shown in Figure 5.1 the Petri-net generated by Alpha Miner fails to capture loops and the resulting process model is not “sound” as it contains deadlocks and lacks the proper initiation with the "Start" activity and conclusion with the "End" activity.

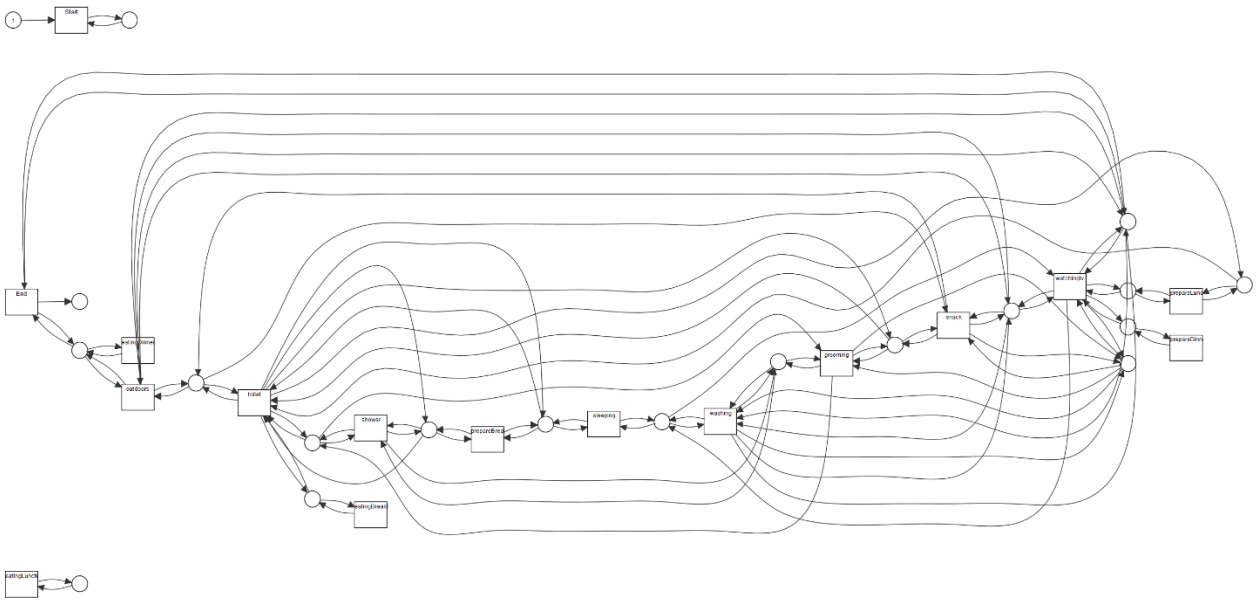


Figure 5.1 Petri-net process model discovered with alpha++ miner algorithm

Figure 5.2 illustrates the Petri net generated by the Heuristics Miner algorithm. The Petri net consists of many arcs and silent transitions (empty activities utilized for routing) resulting in a complex structure. Consequently, it becomes challenging to discern the actual behavior. The advantage of Heuristic Miner in comparison to Alpha Miner, is its capability to filter out infrequent behavior. Additionally, it can identify short loops and the skipping of activities. Based on the results of historical datasets, the dependency value of the relation between two activities to be taken into account was set to 90.0.

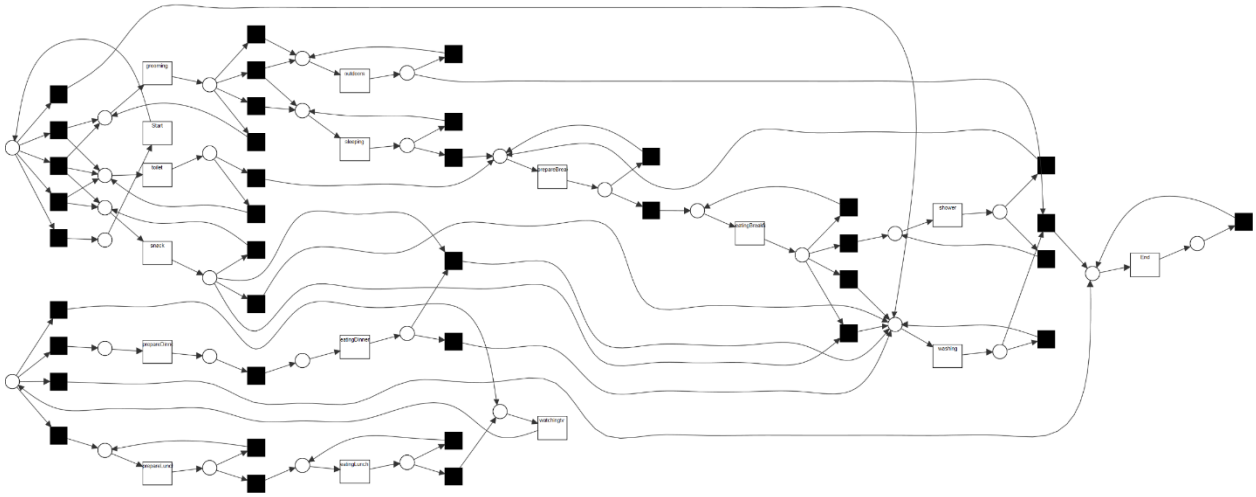


Figure 5.2 Petrinet process model discovered with Heuristics miner algorithm

Figure 5.3 displays the Petri net automatically discovered by Inductive Miner. The produced process model is more straightforward than the others and produces a “sound” process model. Through trial-and-error, the “Noise threshold” was set to 0.20 in order to capture infrequent behavior. Generally, a higher "Noise threshold" enhances precision but lowers fitness values and slows down execution. In comparison to the Heuristic Miner, this model is more legible with less silent nodes. It includes only 14 activities out of 16, as the noise threshold was set to 0.20, filtering out infrequent activities (prepareDinner and eatingDinner). However, the process model remains intricate due to the unstructured nature of human behavior. It is evident that process mining requires more adaptable techniques to address real-life processes that lack a structured format.

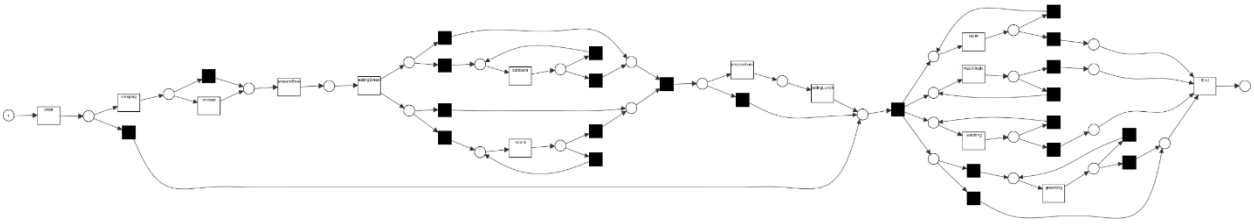


Figure 5.3 Petri net process model discovered with Inductive miner algorithm

Fuzzy Miner depicted in Figure 5.4 excels in discovering process graphs rather than Petri nets and does not distinguish between parallelism and choice. The thickness and darkness of the edges indicate the frequency level, while yellow square nodes represent event classes, and their significance (with a maximum value of 1.0) is provided below each node's event class name. Despite its inability to differentiate between parallelism and choice, Fuzzy Miner proves effective in mining unstructured human behavior processes. Nevertheless, Fuzzy Miner is well-suited for the extraction of less-structured processes characterized by a substantial presence of unstructured and conflicting behaviors.

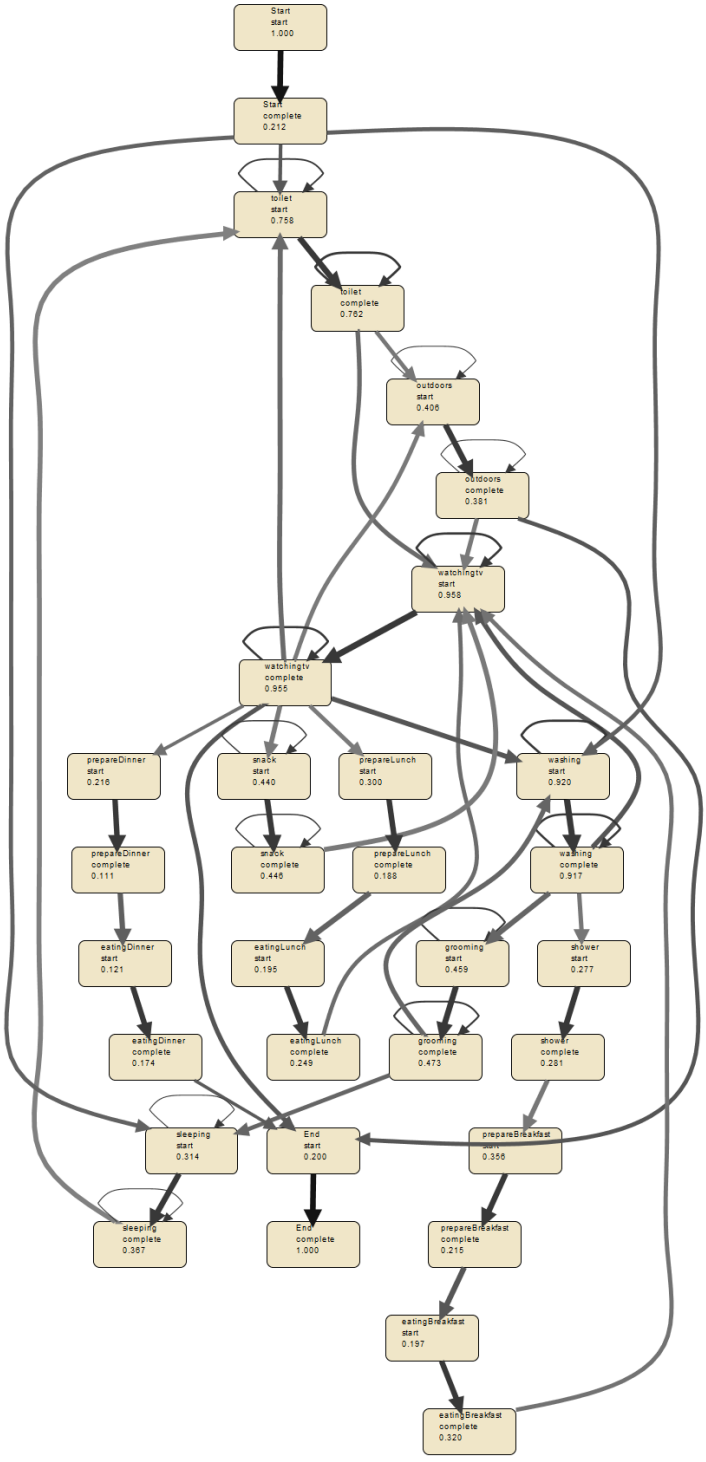


Figure 5.4 Process graph model discovered with Fuzzy miner algorithm

Conformance Checking: Conformance checking measures the differences between the performed process and the process model specifications. More specifically, the behavior in the event log should align with the behavior depicted in the process model. When an activity is

recorded in the event log, its execution must follow a corresponding path in the process model for each trace. As we progress through an activity in the event log (move on log), we concurrently advance along the corresponding path in the process model (move on model).

Figure 5.5 shows the result of conformance checking for Inductive Miner algorithm. Inductive Miner was used because it is the only algorithm that produces a sound process model. Darker shades of blue are used to represent activities that occur more frequently.

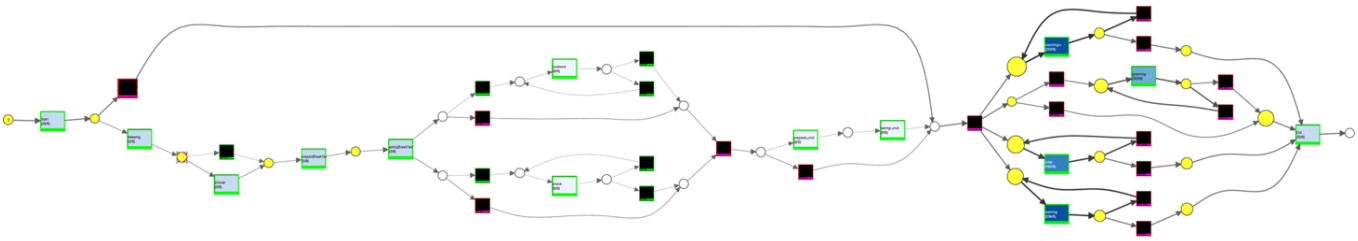


Figure 5.5 Conformance Checking result of Inductive Miner algorithm

The conformance checking results of the elements statistics for the Inductive Miner of a sample event log indicate that: from 23 traces, there were 830 times synchronous move to event log and model, 850 times move only on the model and 464 times move only on the log. The results of the Global Statistics for the Inductive Miner on the overall process are presented in Table 5.2

Table 5.2 Global Statistics of Conformance Checking for the Inductive Miner

Statistics for each property					
Traversed Arcs	19,033.17	Max Move - Log Cost	56.26	Move Model Fitness	1.0
Calculation Time (ms)	316.43	Num. States	5,006.65	Move Log Fitness	0.63
Raw Fitness Cost	20.17	Trace Fitness	0.67	Max Fitness Cost	61.26
Trace Length	56.26	Traces	23	Queued States	7,226.26

Performance Analysis: The objective of performance analysis is to enhance the process. Time information from the event log was projected onto the process model to identify bottlenecks, deviations, service times, throughput time, waiting times, and execution times within the process. In Figure 5.6 darker shades of red are used to represent activities that take more time to complete.

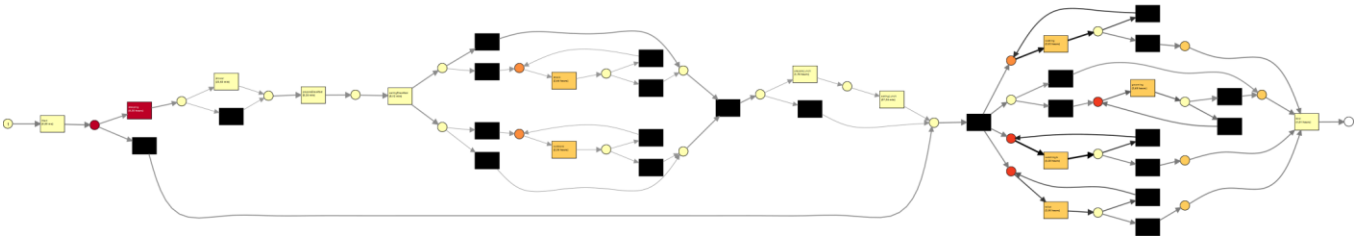


Figure 5.6 Performance analysis results with Inductive Miner algorithm

Performance analysis of the sample event log indicates that, on average, the individual spends 9.38 hours sleeping, allocates minimal time for preparing and eating breakfast, with watching TV being the most frequent activity (3.28 hours). Additionally, the person spends 2.36 hours outdoors.

5.1.2 Extensive experiments

Additional experiments were conducted utilizing the pm4py Python library (version 1.2.12) [97], on 8 real-life event logs from the dataset in [95] which contains Activities of Daily Living (ADLs) of various individuals. For each event log, we applied four mining algorithms: Alpha Miner, Heuristic Miner, Inductive Miner and Directly Follow Graph (DFG) Miner. The evaluation and comparison of the discovery processes were based on five metrics: Log fitness (Figure 5.7a), precision (Figure 5.7b), generalization (Figure 5.7c), simplicity (Figure 5.7d) and F-score (Figure 5.7e). Additionally, we assessed the computational complexity by comparing the time required for each algorithm to produce the process model. These results are depicted in Figure 5.7f, while further details are presented in Table 5.3.

Table 5.3 Comparison of Time Needed to Produce the Process Model

	log1	log2	log3	log4	log5	log6	log7	log8
Number of traces	6	7	10	18	18	21	25	43
Number of events	368	420	488	1152	1728	1390	1392	4200
Time to produce the process model (in sec)								
Alpha Miner	0.0009	0.0010	0.0014	0.0023	0.0036	0.0026	0.0037	0.0096
Heuristic Miner	0.0035	0.0050	0.0046	0.0129	0.0156	0.0122	0.0118	0.0313
Inductive Miner	0.0105	0.0098	0.0106	0.0140	0.0154	0.0147	0.0138	0.0210

Dfg Miner	0.0014	0.0017	0.0016	0.0030	0.0046	0.0037	0.0037	0.0087

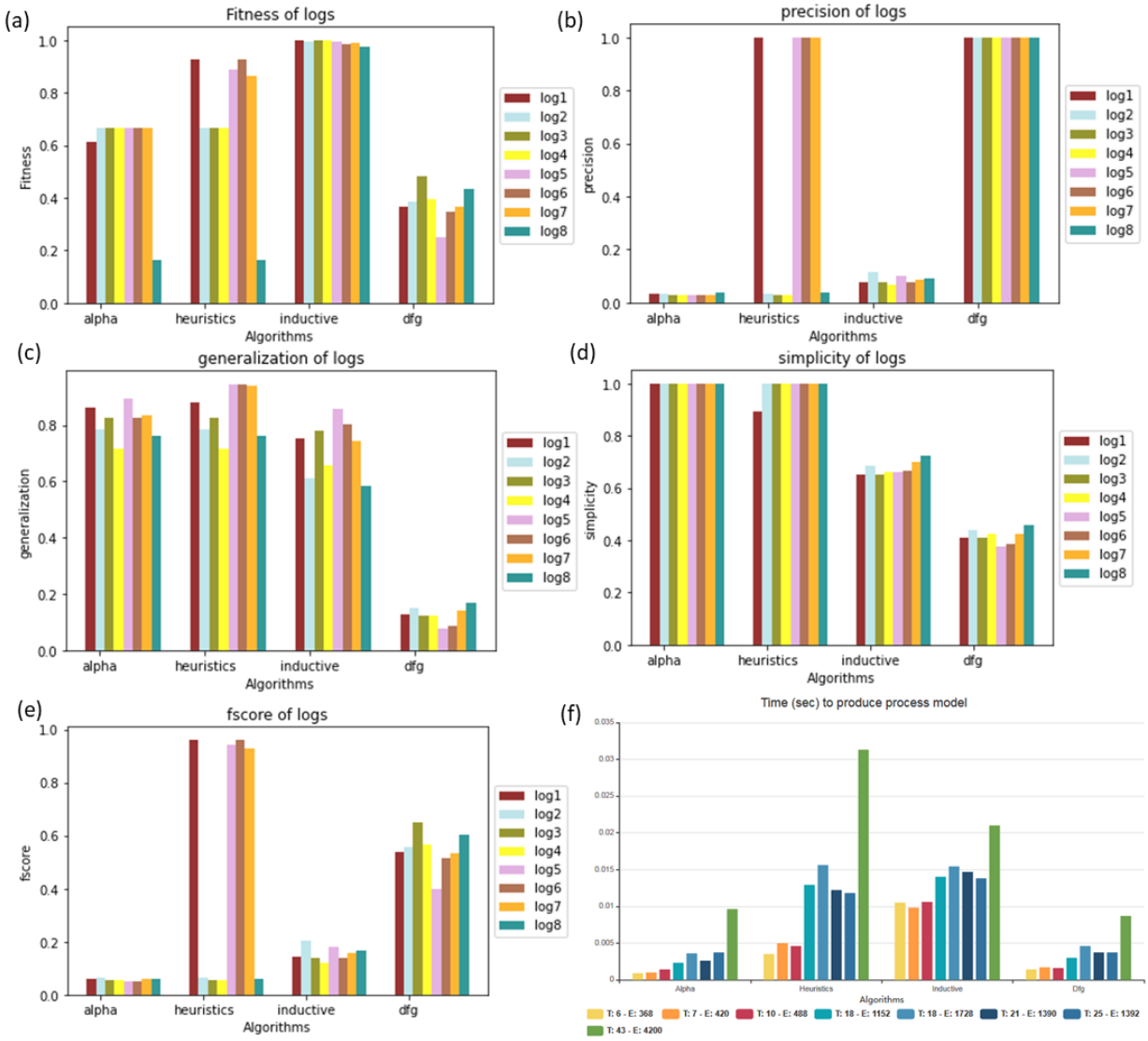


Figure 5.7 Experimental results for: (a) log fitness; (b) precision; (c) generalization; (d) simplicity; (e) f-score; and, (f) time needed to produce the process model.

5.1.3 Conclusions

The resulting models exhibit complexity, reflecting the unstructured nature of human daily routines. Among the algorithms utilized, the Inductive Miner stands out as particularly effective in capturing behavior from event logs.

Conformance analysis assesses the alignment of behavior described by a process model with the behavior recorded in an event log. Performance analysis aims to offer insights for optimizing processes. However, in instances where processes are intricate, involving numerous activities and complex case routings, conventional process models often struggle to provide valuable insights.

5.2 Develop visualizations and software application to analyze event logs

5.2.1 Visual Analytics in Process Mining for Supporting Process Improvement

5.2.1.1 Application to human activity dataset

The widespread integration of IoT devices and sensors in smart environments has spurred growing research interest in real-time data analytics. The presented approach was implemented on a human activity dataset, specifically addressing activities related to daily living.

Initially, following the proposed approach, the event log is extracted (a in Section 4.2.1) and filtered (b in Section 4.2.1) to structure it in a manner conducive to subsequent visual analysis. The dataset comprises Activities of Daily Living (ADLs) of an individual over several days, and Table 5.1 provides a sample from the extracted event log. The pertinent attributes for the proposed approach include the Case ID (Day Id), the Start Timestamp (activity start time), the Complete Timestamp (activity end time), and the Activity. The Start Timestamp offers additional information related to the Activity, enabling further analysis.

Figure 4.1 illustrates the process model derived from the aforementioned event log (Figure 5.3 presents the Petri net discovered with Inductive Miner). However, this model exhibits high complexity, making it impossible for users to comprehend the process. Nevertheless, in human behavior analysis, achieving explainability and transparency is crucial for drawing conclusions about resident health and safety. This can be accomplished through a visual analytics approach.

Following the subsequent step (c in Section 4.2.1) outlined in the proposed approach, a holistic perspective of the visualization can unveil patterns and trends within the event log. Figure 5.9 depicts the activities over time, color-coded by the trace to which they belong. This visualization allows users to discern the start and end dates of the recorded event log. It is interactive, empowering users to zoom in/out to specific dates or activities. Additionally, the user can ascertain if the activities are performed every day or not and their frequency throughout the day.

In accordance with step d (in Section 4.2.1), the proposed approach employs a bubble plot in order to provide representation of activity durations over time. Bubble plots, as an alternative to bar graphs, reduce visual clutter and enhance readability. Figure 5.10 illustrates the duration of various activities relative to their start and end timestamps, with each data point color-coded

according to the corresponding activity. This visualization proves highly beneficial for identifying the root causes of bottlenecks and resource inefficiencies. In the realm of human behavior, it can offer valuable insights; for example, prolonged periods spent watching television without engaging in physical activity may pose health risks, especially for individuals with underlying heart issues.

Moreover, Figure 5.11 illustrates the mean duration of each activity, facilitating the identification of the most time-consuming activities. The visualization indicates that the resident sleeps well and spends more time outdoors than watching television.

In step f (in Section 4.2.1) as depicted in Figure 5.12, trace duration is visualized to determine the total time taken to complete traces aiding in the identification of the most prolonged traces and their patterns over the examined time period. Extended durations, particularly on specific days, have the potential to be physically taxing for household occupants, potentially impacting their health. In our scenario, most days exhibit a consistent duration range, except for one that stands out as an outlier, warranting further investigation.

As outlined in step g (in Section 4.2.1), the user's decision-making process is guided by the visual analytics presented earlier in the process. These decisions encompass both typical patterns and potential health issues.

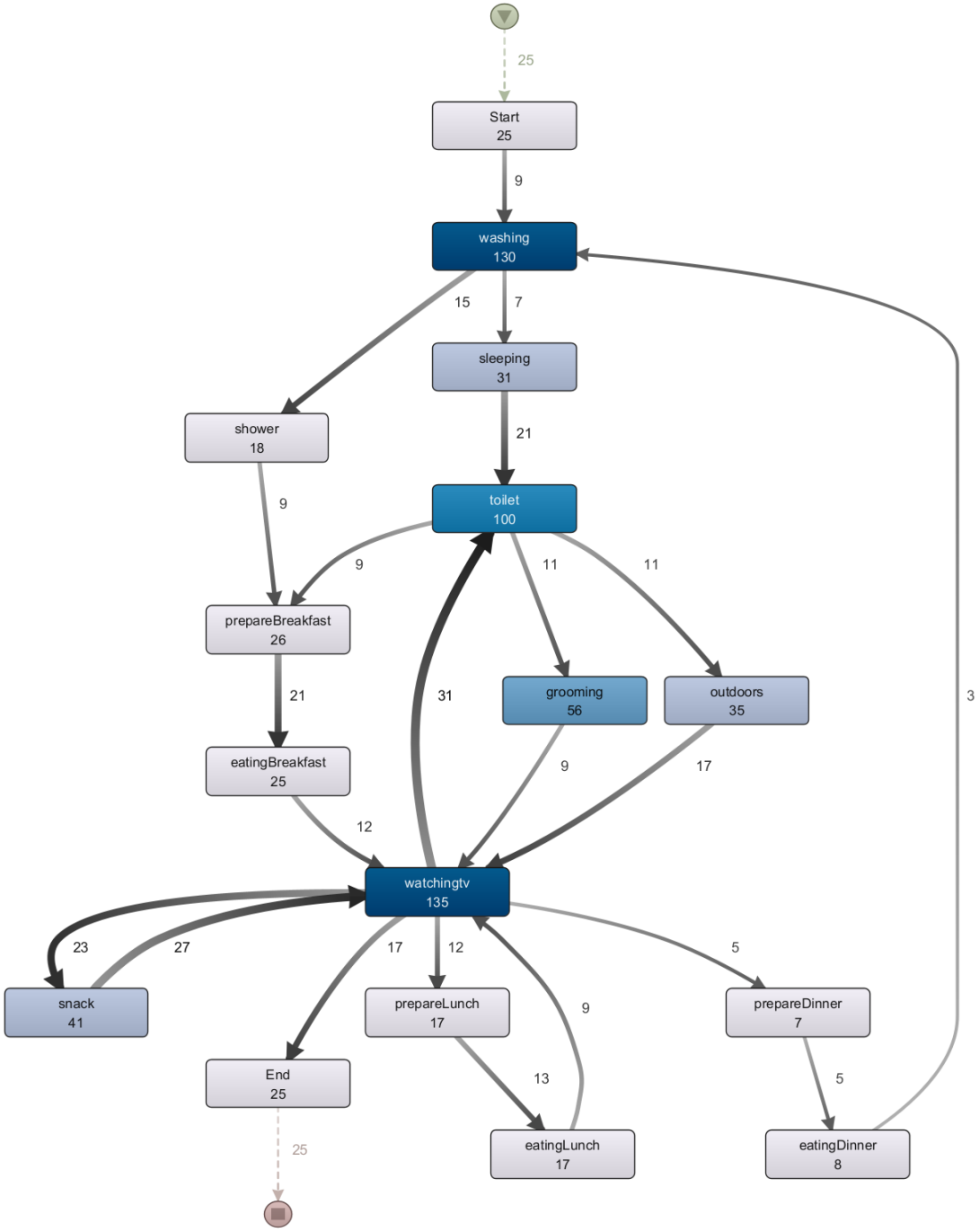


Figure 5.8 Process map for human activity dataset

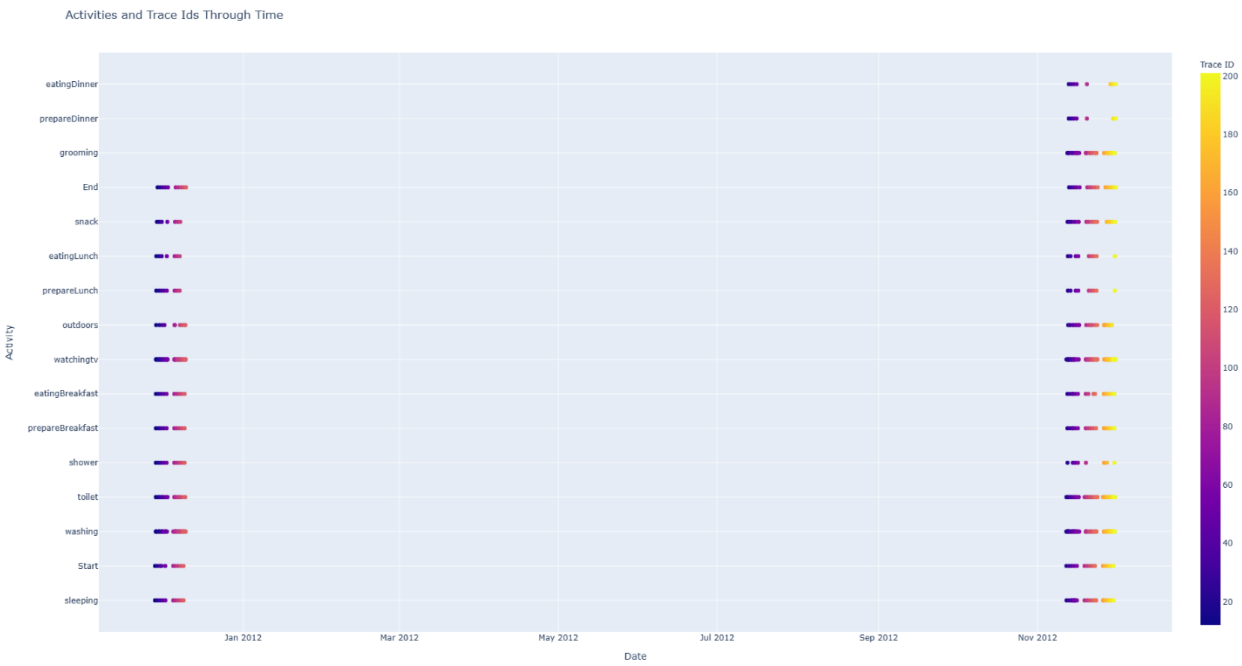


Figure 5.9 Activities performed over time, colored by traces

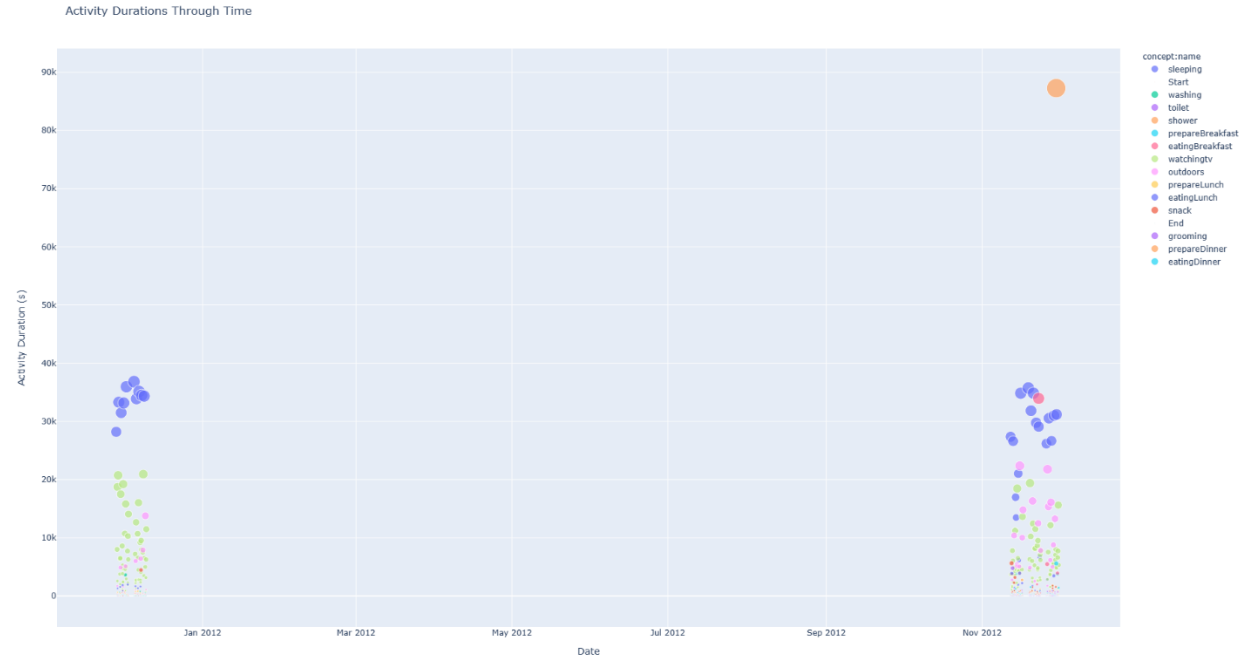


Figure 5.10 Duration of activities over time, colored by activity

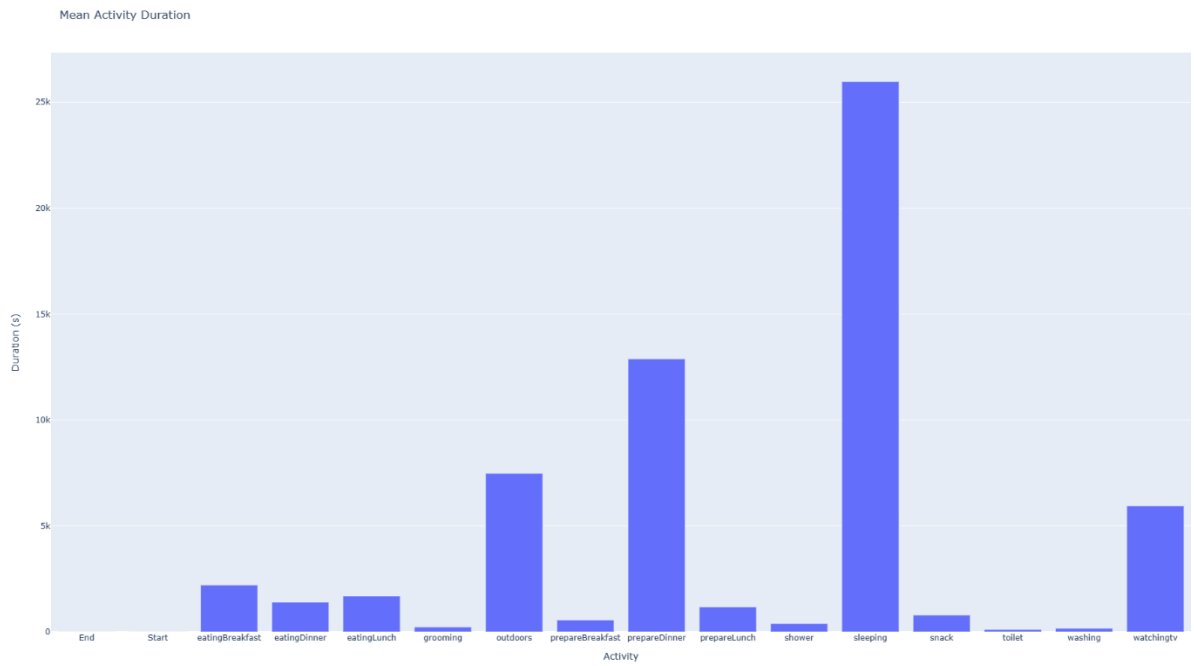


Figure 5.11 Mean duration of activities

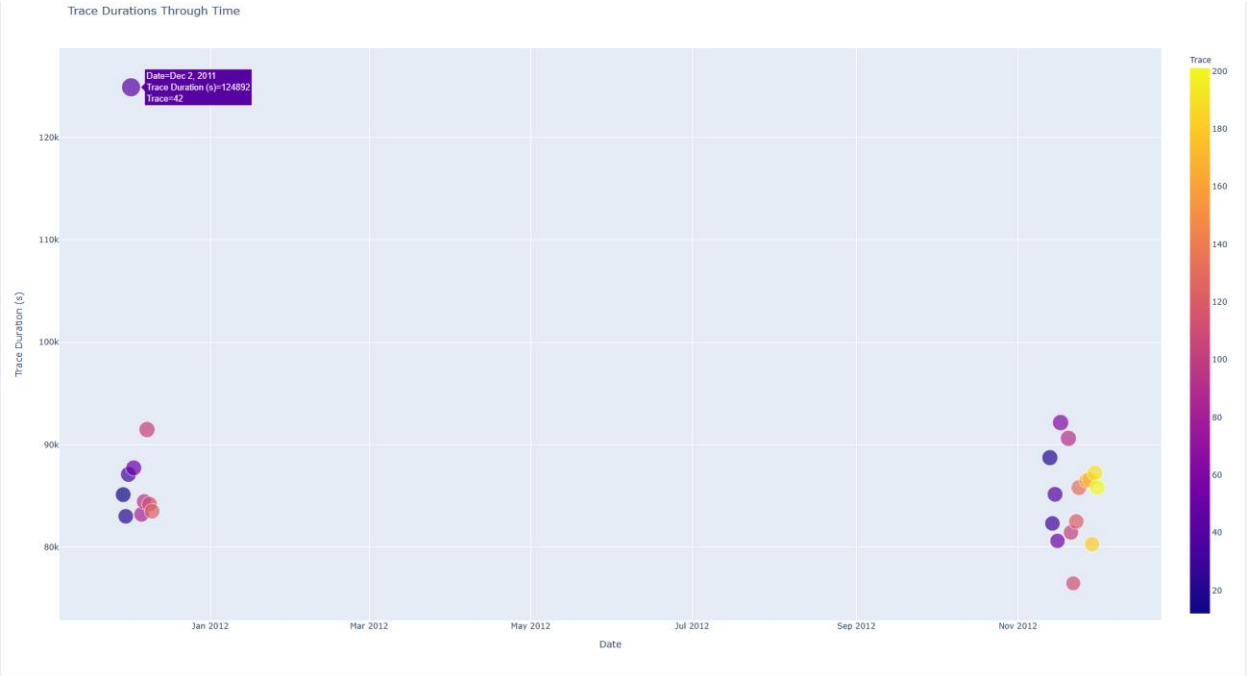


Figure 5.12 Duration of traces over time, colored by trace

5.2.1.2 Conclusions

While automated algorithms play a crucial role in the initial stages of process mining and analysis by filtering out extraneous data and generating preliminary results, the ultimate comprehension and interpretation of these findings necessitate visual inspection, domain knowledge, human judgment, and creativity. Notably, the process discovery on an event log often yields intricate and sizable process models that may pose challenges for users to grasp easily. In addressing this, visual analytics emerges as a powerful tool to enhance process mining by fostering explainability, interpretability, and trustworthiness, thereby facilitating informed human decision-making. The findings presented in the referenced study underscore the capability of visual analytics within the context of process mining, demonstrating its effectiveness in identifying bottlenecks and performance-related issues. Significantly, this information is communicated to users in an intuitive and non-intrusive manner, reinforcing the role of visual analytics in supporting business process improvement.

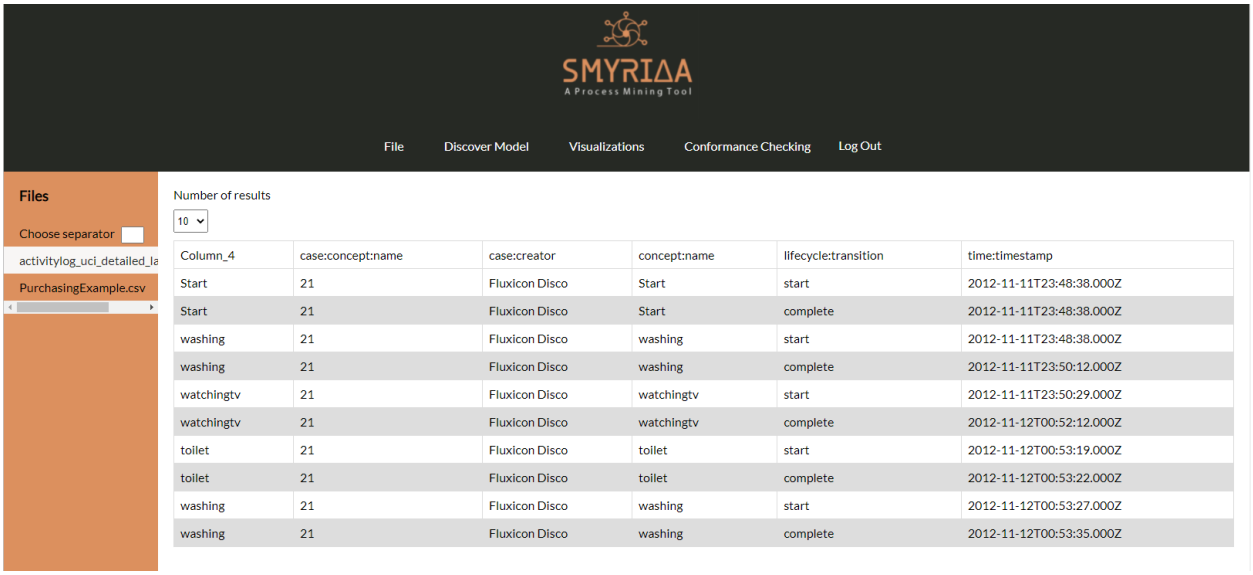
5.2.2 Smyrida: A web application for process mining and interactive visualization

5.2.2.1 Walkthrough example

To start process mining with Smyrida, the user must signup/login and upload at least one file (event log) either in xes (or xes.gz) or in csv (or txt) format. A csv event log can be converted to xes and vice versa.

5.2.2.1.1 File Menu

From the File menu by selecting “View” the user can see the contents of the event log in tabular format (Figure 5.13).



Column_4	case:concept:name	case:creator	concept:name	lifecycle:transition	time:timestamp
Start	21	Fluxicon Disco	Start	start	2012-11-11T23:48:38.000Z
Start	21	Fluxicon Disco	Start	complete	2012-11-11T23:48:38.000Z
washing	21	Fluxicon Disco	washing	start	2012-11-11T23:48:38.000Z
washing	21	Fluxicon Disco	washing	complete	2012-11-11T23:50:12.000Z
watchingtv	21	Fluxicon Disco	watchingtv	start	2012-11-11T23:50:29.000Z
watchingtv	21	Fluxicon Disco	watchingtv	complete	2012-11-12T00:52:12.000Z
toilet	21	Fluxicon Disco	toilet	start	2012-11-12T00:53:19.000Z
toilet	21	Fluxicon Disco	toilet	complete	2012-11-12T00:53:22.000Z
washing	21	Fluxicon Disco	washing	start	2012-11-12T00:53:27.000Z
washing	21	Fluxicon Disco	washing	complete	2012-11-12T00:53:35.000Z

Figure 5.13 Contents of the event log in tabular format

In the presented dataset (activitylog_uci_detailed_labour.xes) the event log consists of 6 attributes (presented as columns):

- Column_4: (copy of concept:name)
- case:concept:name: The trace id of the event log (day id)
- case:creator: The creator of the event
- lifecycle:transition: The activity state - if the activity started in the specific timestamp or completed
- time:timestamp: The time the activity was started or completed

This is highly beneficial for users in comprehending the file's structure and the nature of the values it contains.

From the File menu, by selecting "Info," the user can access valuable information about the event log (Figure 5.14). Specifically, the user can gather details on:

- The number of traces and unique events in the log. In this context the event log consists of 25 traces, representing 25 recorded days of human activity in the house.
- Information about the events, their frequency, and the percentage of appearance in the event log.
- The start and end activities of the event log, crucial for tasks such as filtering incomplete traces of needed.
- The variants of the event log representing unique traces. These variants provide insights into the distinct paths within the event log, offering valuable information about the

duration of each path during each process execution. Additionally, they help assess the level of structure or lack thereof in the process. For instance, in a human activity event log, there are 25 unique traces, each different from the others, highlighting the complexity and unstructured nature of the process. This complexity may necessitate diverse approaches to effectively deconstruct and analyze it effectively.

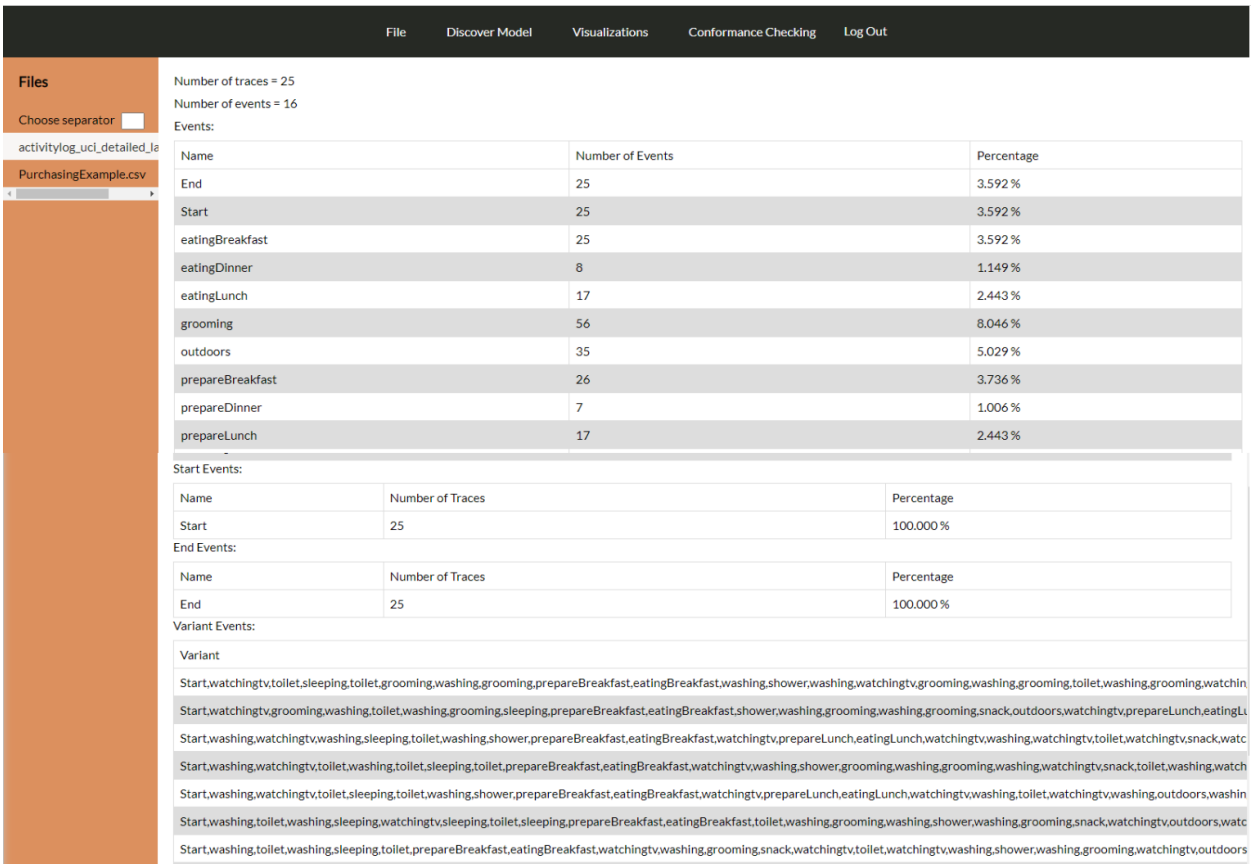


Figure 5.14 Info menu to explore information of event log

5.2.2.1.2 Discover Model Menu

The user has the option to reveal a process model by employing one of the algorithms, such as Alpha Miner (depicted in Figure 5.15), Heuristics Miner (depicted in Figure 5.16), or Inductive Miner (depicted in Figure 5.17). The resulting process model can be zoomed in and out, and it allows the user to drag and reposition places and activities within the generated Petri net. The process models are constructed in a hierarchical approach, aiming to enhance their readability and structural comprehensibility. Each model comes with evaluation metrics presented, including Log Fitness, Log Precision, Generalization, and Simplicity.

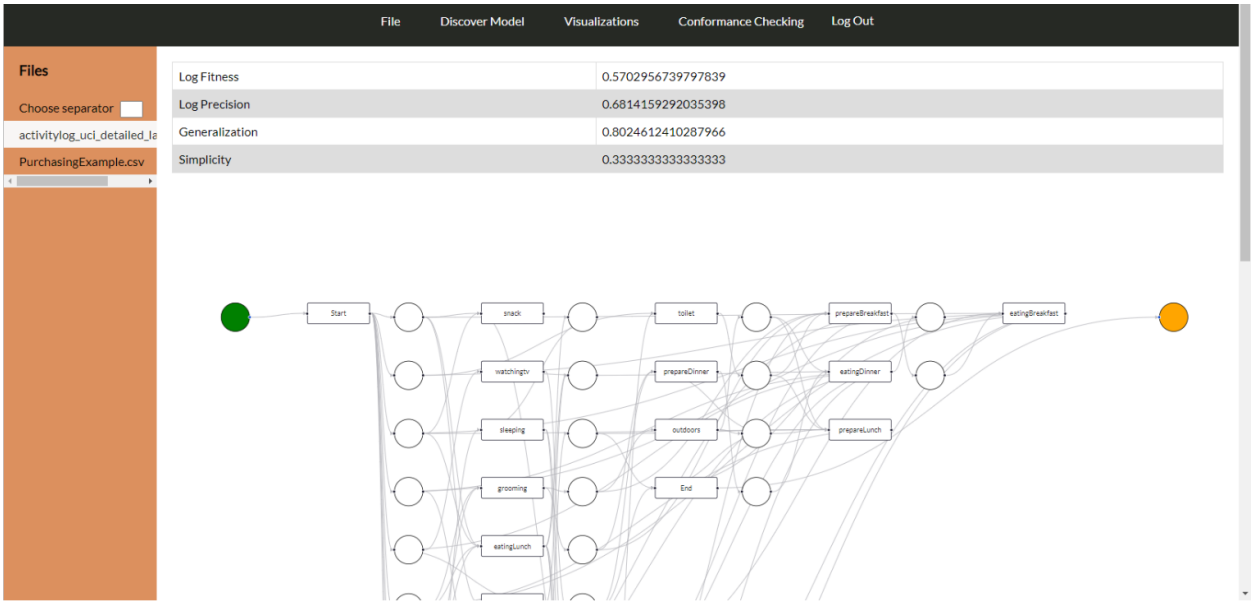


Figure 5.15 Alpha Miner Process Model

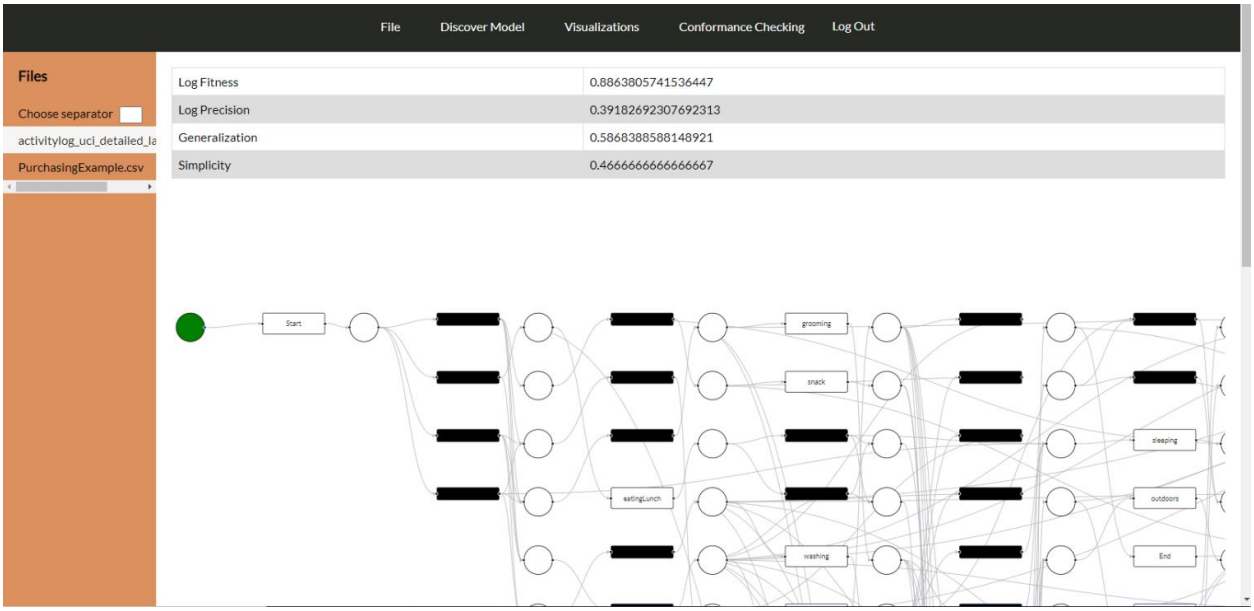


Figure 5.16 Heuristics Miner Process Model

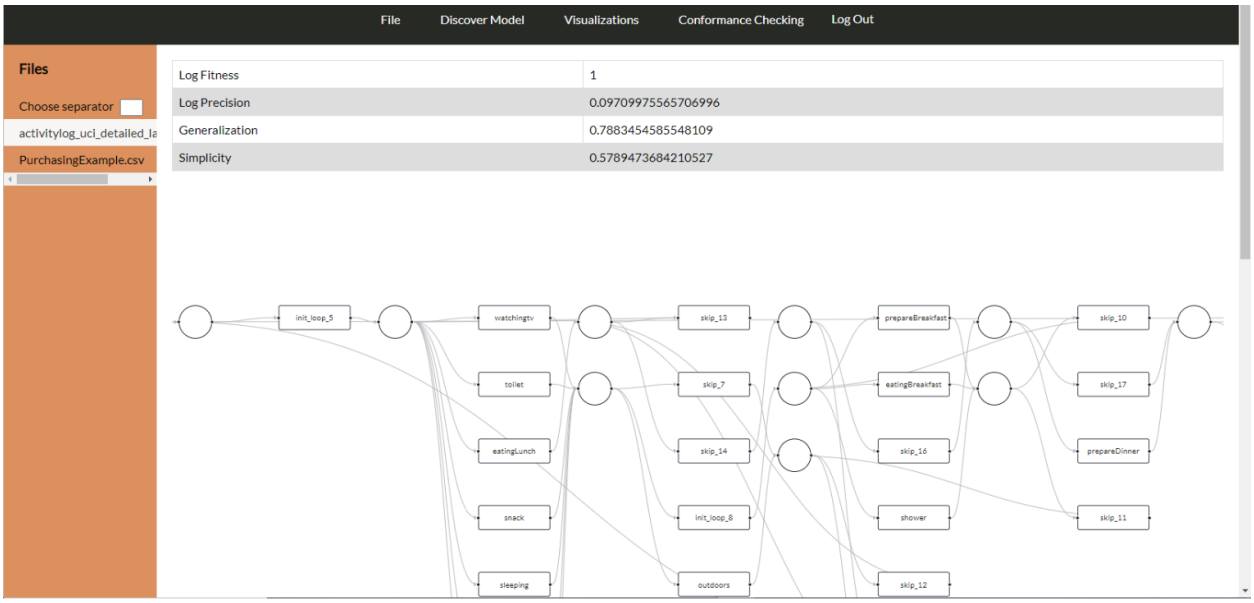


Figure 5.17 Inductive Miner Process Model

Models alone are insufficient for a thorough understanding of the process. In combination with models, advanced techniques must be applied to conduct in-depth analyses of processes. Visualization plays a crucial role in all these techniques, significantly enhancing their overall effectiveness.

5.2.2.1.3 Visualizations Menu

In addition to the initial visualizations detailed in section Application to human activity dataset, Smyrida provides enhanced and more specific plots.

To gain insights into the frequencies of process activities and their occurrences, users can utilize the "Activities Frequency" option within the Visualizations menu. As depicted in Figure 5.18, the event log includes 16 activities, with the most frequently occurring ones being "toilet," "washing", and "watching TV".

Figure 5.18 Frequency of each activity in the event log

Frequency alone lacks context awareness; duration plays a significant role in this regard. To understand the duration of process activities, users can choose "Activities Duration" from the Visualizations menu. As depicted in Figure 5.19 below, the event log includes 16 activities, with the longest durations observed in activities such as "outdoors," "prepare dinner," and "sleeping".

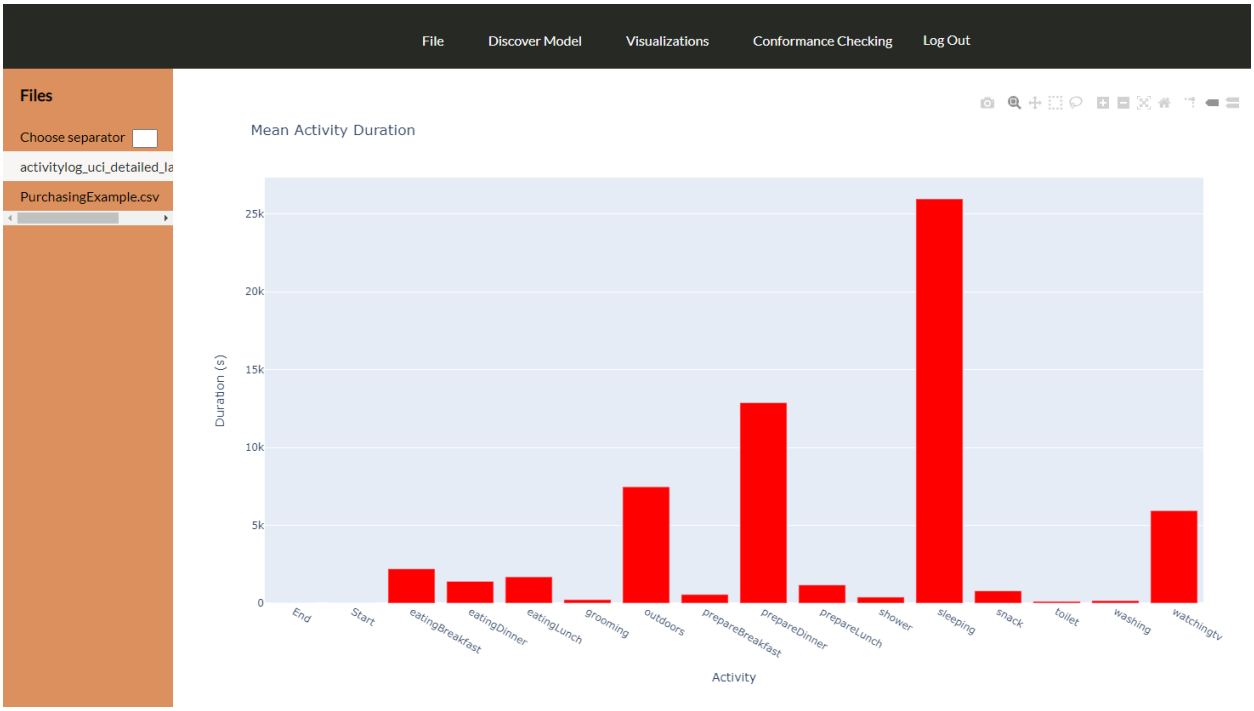


Figure 5.19 Duration of each activity in the event log

In Figure 5.20 each distinct trace (referred to as a variant) is illustrated with the sequence of activities performed. The colored squares represent activities with each activity having a distinct color. The frequency of each variant is also presented as a percentage. This visualization proves highly useful for identifying patterns in activity sequences, such as the same start and end activities in each trace. Additionally, users have the option to filter out infrequent traces, enhancing the understanding of the core behavior. In intricate processes, isolating the main process is crucial for ensuring readable results.

Figure 5.20 Distinct traces with sequence of activities that was performed, colored by activities

Figure 5.21 depicts the frequency of each variant (distinct trace) with each variant appearing only once in this scenario. This implies that the resident performed each activity in a different order every day. It emphasizes the need for a different approach to human activity event logs. Human behavior, as mentioned earlier, is characterized by variability, yet individuals maintain a daily routine. For instance, even if they perform three different activities in the morning with a different order each day, they consistently engage in those activities daily.

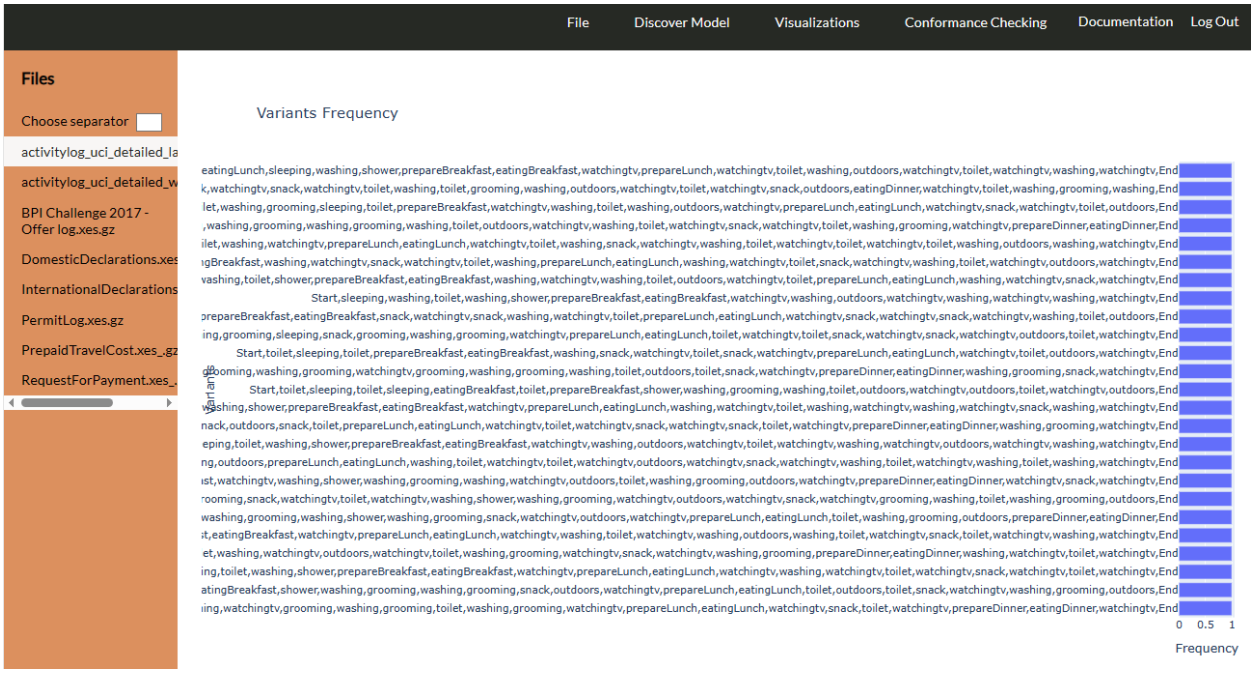


Figure 5.21 Distinct traces with sequence of activities that was performed, colored by activities

In Figure 5.22 the activities performed in the corresponding trace are illustrated. This proves useful for identifying activities that are not carried out on a daily basis. For instance, food preparation and eating may not occur every day, indicating the likelihood that the resident dined outdoors. Additionally, users can assess the regularity of the resident's daily showers, which is crucial for hygiene and, furthermore, mental well-being.

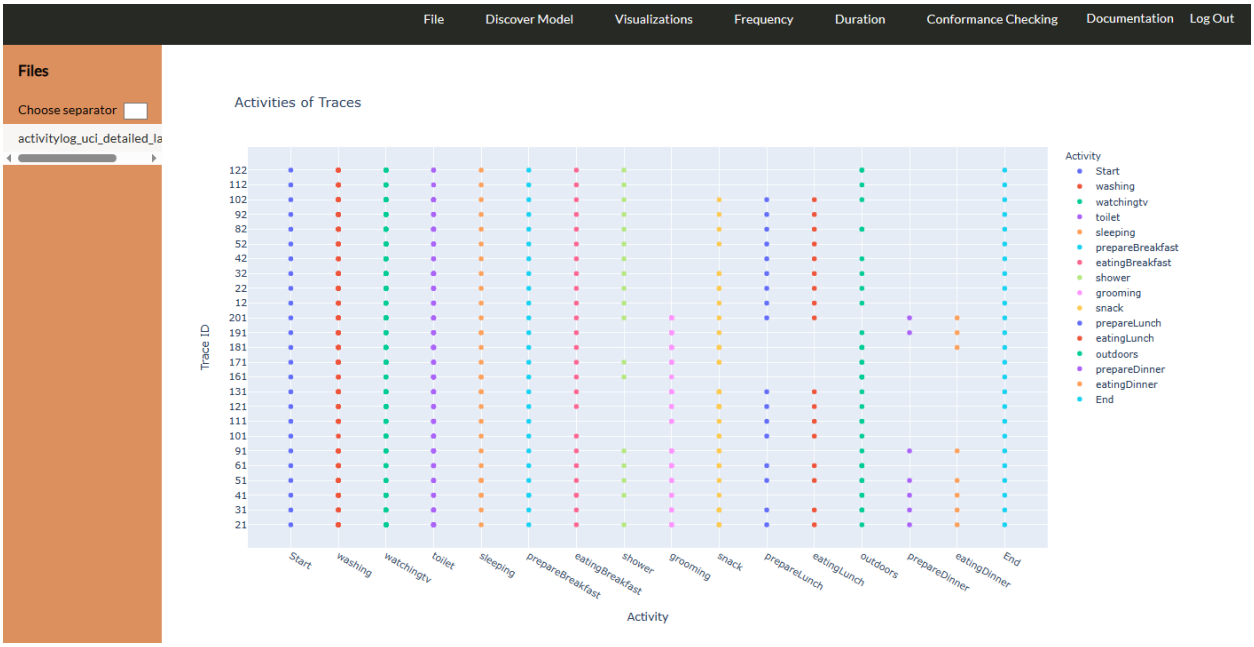


Figure 5.22 Plot of activities that performed in respective traces, colored by activity

5.2.2.1.4 Conformance Checking

Smyrida implements the two primary concepts of conformance checking: Trace Alignment and Replay Log. Conformance checking utilizes the event log and the process model (in this case, the model discovered automatically with Inductive Miner) as input to identify deviations between them. Figure 5.23 shows the outcome of Conformance Checking using the Alignments technique. As evident, the event log and the process model align perfectly.

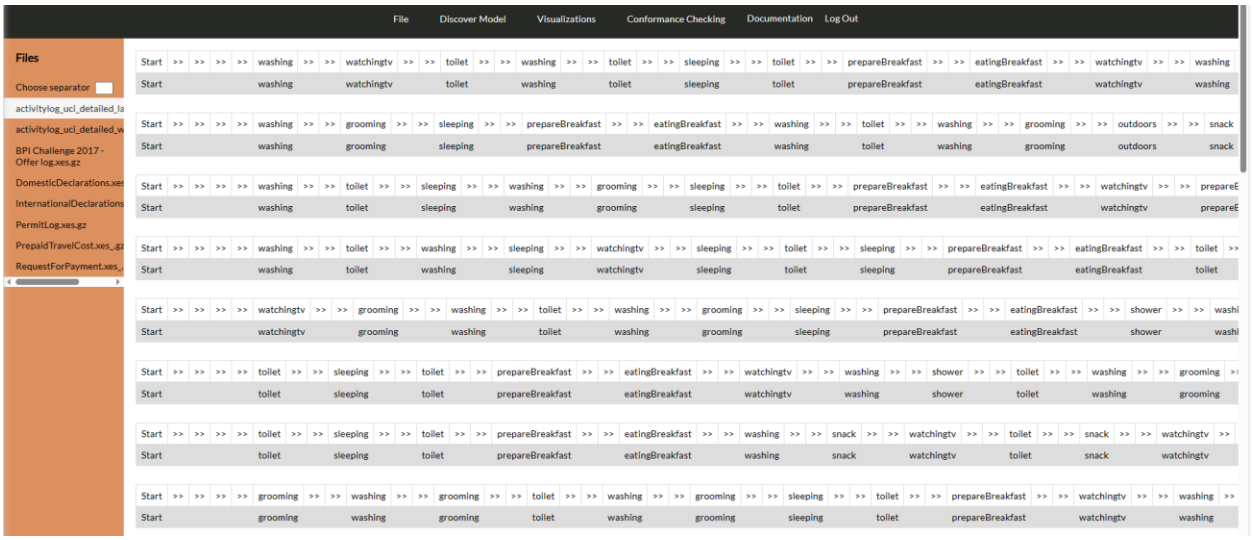


Figure 5.23 The outcome of Conformance Checking using the Alignments technique.

5.2.2.2 Conclusions

"Smyrida" emerges as a versatile modular software system presented in the format of a web application, equipped with open APIs that facilitate adaptability to emerging process mining techniques. Its inherent flexibility allows seamless integration with information systems, generating comprehensive event logs. From a user standpoint, a pivotal aspect involves the provision of interactive and intuitive visualization capabilities. Notably, "Smyrida" addresses the imperative need for users, even those lacking advanced process mining skills, to effortlessly extract intricate process models and conduct thorough evaluations of their processes. This amalgamation of adaptability, user-friendly features, and integration prowess positions "Smyrida" as a robust solution in the realm of process mining.

5.3 Develop framework to detect changes and predict human behavior

5.3.1 Datasets

Human activity datasets consist of the activity and the end timestamp or start timestamp too. In order to be transformed in a suitable format the date forms the trace Id, the completion of activity forms the event and the timestamp the activity completed forms the time.

We used several datasets of human activities of daily living that are summarized in Table 5.4.

Table 5.4 Event Logs with activities of daily living

Id	Name	Reference	Traces(Days)	Distinct Activities
Event Log 1	Activity_uci_labour	[95]	25	16
Event Log 2	Base Kasteren	[98]	25	8
Event Log 3	Activity 3	[99]	10	13
Event Log 4	OrdoreZoneB	[100]	22	10

5.3.2 Experiments

5.3.2.1 Encoding

In this section we will analyze how encoding affects the framework.

In Table 5.5 and Table 5.6 a part of event log that was encoded with numeric (Keras Tokenizer and Label Encoder) and vector (One Hot Encoding and Bert Tokenizer) encoding techniques are depicted for 1-sequence and for 2-sequence of activities.

Table 5.5 Encoding techniques on event log for 1-sequence of activities

Day id	concept:name	Keras Tokenizer	Label Encoder
12	Start	9	1
12	sleeping	6	11
12	washing	2	14
12	toilet	3	13
12	shower	12	10

12	prepareBreakfast	8	7
12	eatingBreakfast	10	2

(a) Numeric encoding

Day id	concept:name	One Hot Encoding	Bert Tokenizer
12	Start	[0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0]	[101 15599 102 0 0 0]
12	sleeping	[0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0]	[101 5575 102 0 0 0]
12	washing	[0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0]	[101 13445 102 0 0 0]
12	toilet	[0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0]	[101 12356 102 0 0 0]
12	shower	[0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0]	[101 5946 102 0 0 0]
12	prepareBreakfast	[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0]	[101 7034 2064 22362 27704 102]
12	eatingBreakfast	[0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0]	[101 5497 2064 22362 27704 102]

(b) Vector encoding

Table 5.6 Encoding techniques on event log for 2-sequence of activities

Day id	concept:name	Keras Tokenizer	Label Encoder
12	Start sleeping	17	9
12	toilet washing	2	53
12	prepareBreakfast shower	18	37
12	eatingBreakfast washing	26	17
12	washing watchingtv	1	56
12	outdoors toilet	13	34
12	toilet watchingtv	3	54

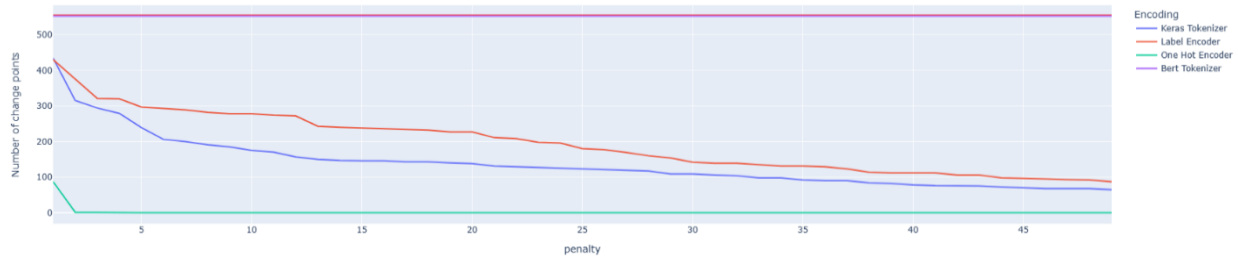
(a) Numeric encoding

Day id	concept:name	One Hot Encoding	Bert Tokenizer
12	Start sleeping	[0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]	[101 15599 5575 102 0 0 0 0 0 0]
12	toilet washing	[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]	[101 12356 13445 102 0 0 0 0 0 0]
12	prepareBreakfast shower	[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]	[101 7034 2064 22362 27704 5946 102 0 0 0]
12	eatingBreakfast washing	[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]	[101 5497 2064 22362 27704 13445 102 0 0 0]
12	washing watchingtv	[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0]	[101 13445 2903 1204 1964 102 0 0 0 0]
12	outdoors toilet	[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]	[101 23178 12356 102 0 0 0 0 0 0]
12	toilet watchingtv	[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0]	[101 12356 2903 1204 1964 102 0 0 0 0]

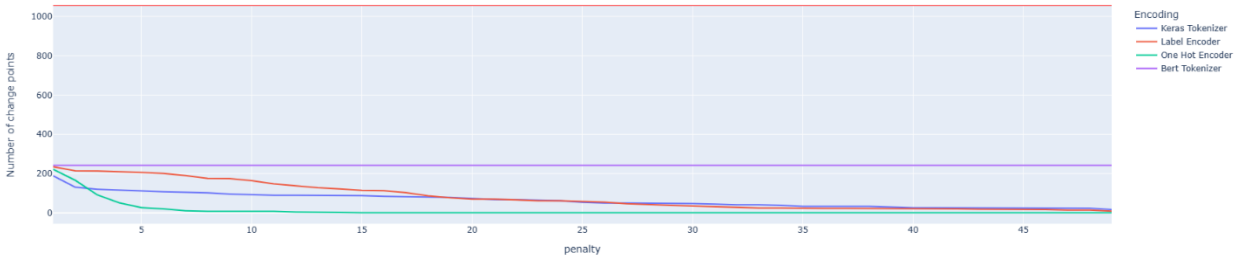
(b) Vector encoding

Figure 5.24 presents how penalty value affects the number of change points with Pelt change point detection algorithm for each encoding method. X-axis represents penalty values between 1 and 49 and Y-axis represents the number of change points for each encoding method. Red line on top is the number of all points in the file, in order to be clear the level of number of change points depending on file size.

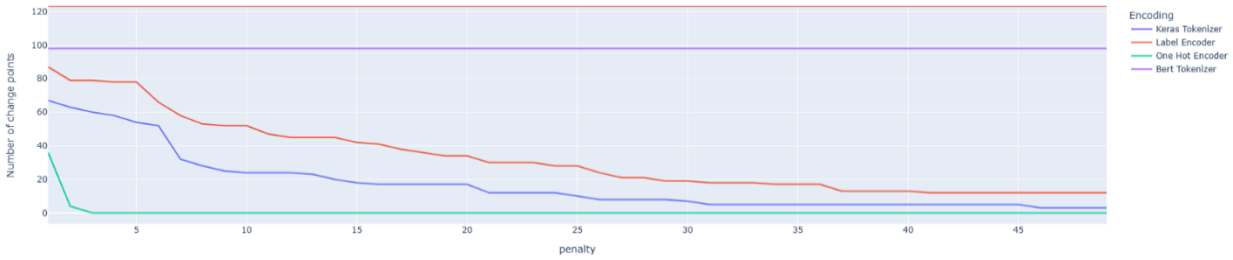
Encoding method plays a significant role in the number and the position of change points. In Label Encoding and Keras Tokenizer techniques, as the penalty value increases from 1 to 50, there is a declining trend in the number of change points (on all event logs). In One Hot Encoding, the penalty value significantly reduces the number of change points around penalty 1. In the Bert Tokenizer, the number of change points appears to remain consistent regardless of the penalty value. This occurs because the encoding values from Keras Tokenizer and Label encoding are within the same range as the penalty value. In the context of One-Hot Encoding, it is preferable to choose values falling between 0 and 2 (refer to Figure 5.25). Conversely, Bert Tokenizer remains unaffected relative to the penalty value, potentially because of the extensive range of values within the vector.



(a) Event log 1



(b) Event Log 2



(c) Event log 3

Figure 5.24 Number of change points calculated with Pelt algorithm for each encoding depending on penalty value (a) for event log 1 (b) for event log 2 (c) for event log 3.

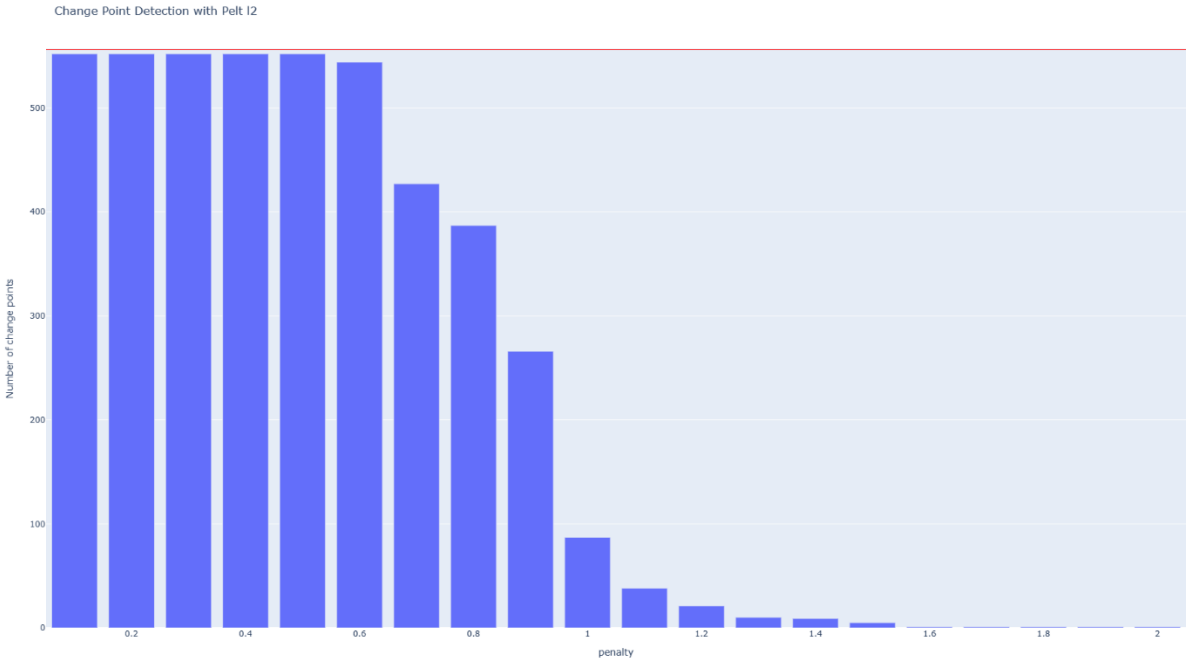
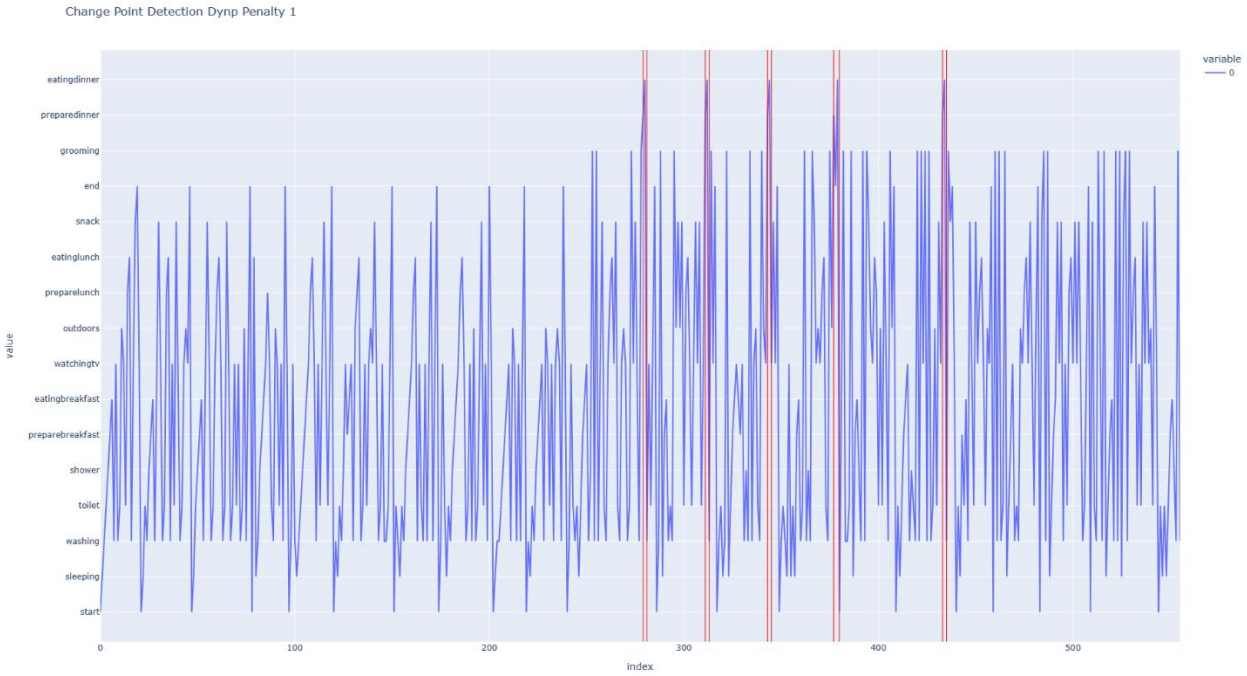


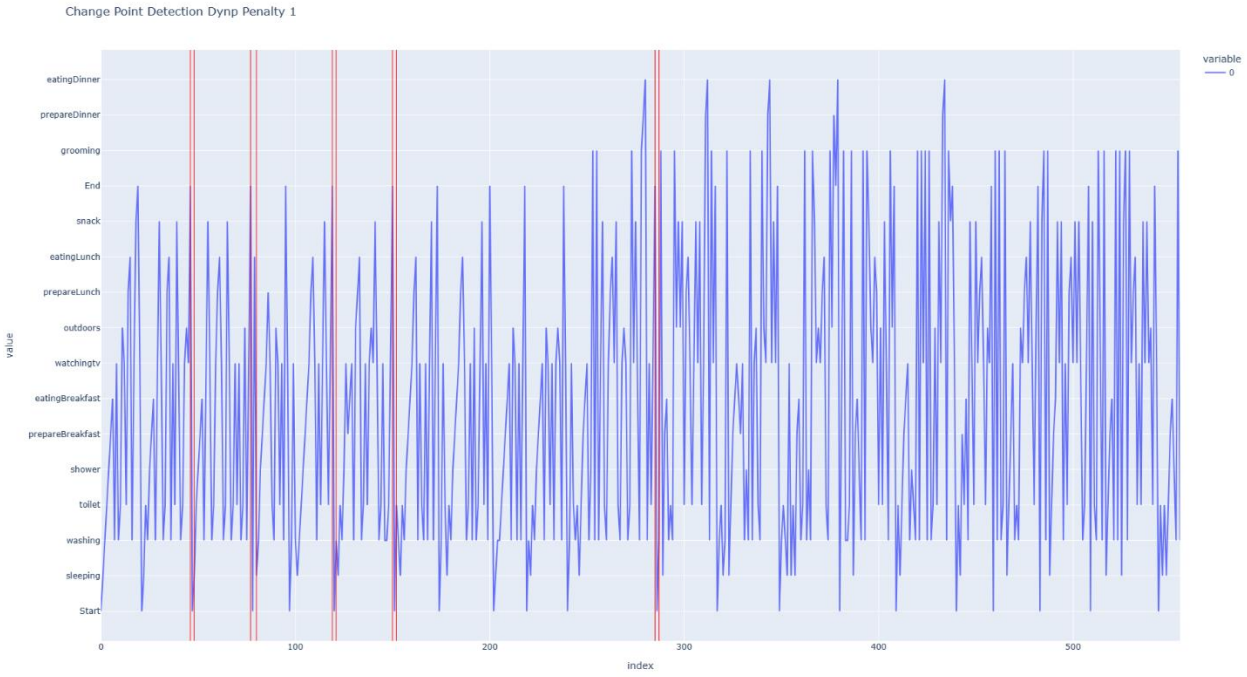
Figure 5.25 Number of change points calculated with Pelt algorithm for One Hot Encoding depending on penalty value (event log 1). Penalty values ranging from 0.1 to 2.

Figure 5.26 illustrates the location of 10 change points in event log 1 identified using the Dynamic Programming algorithm across all four encoding methods. The X-axis denotes time, while the Y-axis represents the corresponding activities. The vertical red lines indicate the time positions of the change points. Notably, the encoding method influences the detected positions of the change

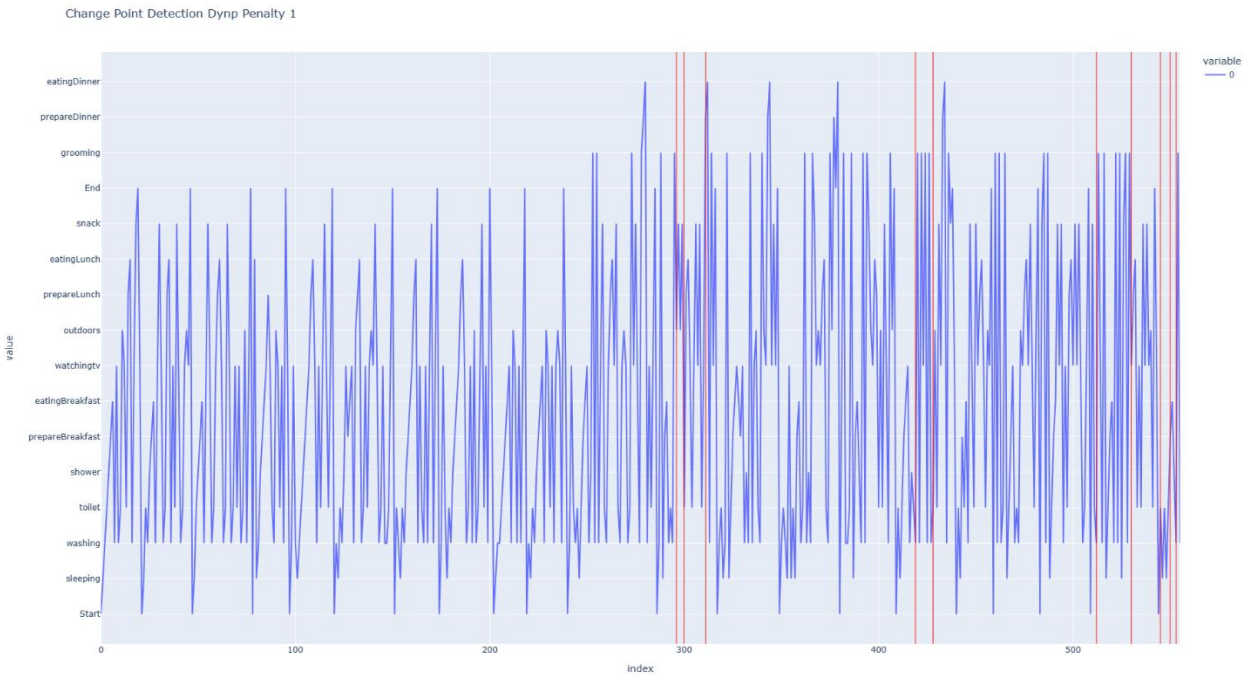
points.



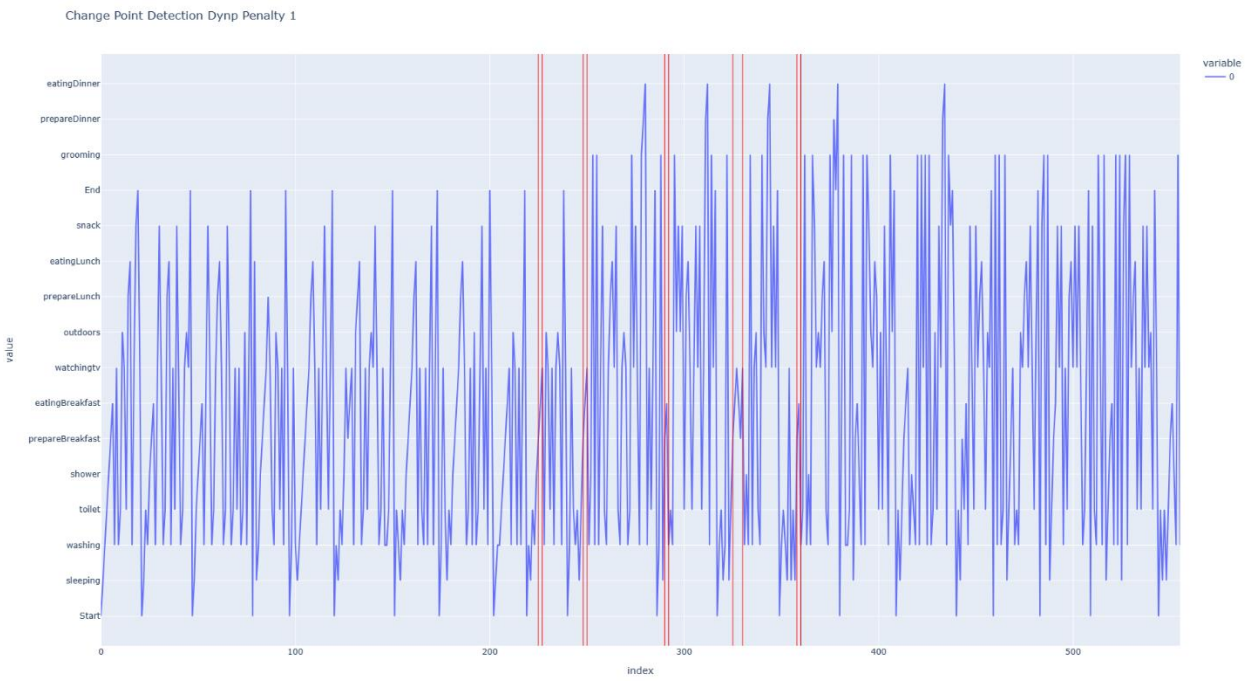
(a) Keras Tokenizer



(b) Label Encoder



(c) One Hot Encoding



(d) Bert Tokenizer

Figure 5.26 Locations (vertical red lines) of 10 change points detected with Dynamic Programming algorithm for event log 1 with (a) Keras Tokenizer (b) Label Encoder (c) One Hot Encoding (d) Bert Tokenizer.

Table 5.6 demonstrates how the selection of encoding method influences the time required for applying the change point detection method. In this table is presented the time required for implementing the Pelt Change Point Detection algorithm with all 4 encoding methods on 3 different files. The table also provides information on the number of samples in the event log (initial points), the ratio of the number of variants to the number of samples, indicating the level of unstructuredness in the input file, the count of activities comprising a point (n-sequences), and the number of detected change points. The conclusion drawn from these tables is that the completion time of Pelt depends more on the number of change points rather than the chosen encoding method.

Table 5.7 Pelt algorithm completion time for 3 different event logs for all 4 encoding techniques

Pelt (penalty=50)	**#samples**	**#variants/#samples**	**n-sequence**	**encoding**	**#change points**	**Time (sec)**
Event log 1	556	0.029	1	Keras Tokenizer	63	5.24

				Label Encoder	**85**	**9.09**
				One Hot Encoding	0	11.23
				Bert Tokenizer	552	460.30
	284	0.20	2	Keras Tokenizer	129	20.40
				Label Encoder	199	49.88
				One Hot Encoding	0	3.95
				Bert Tokenizer	277	101.48
	194	0.39	3	Keras Tokenizer	128	20.33
				Label Encoder	140	24.70
				One Hot Encoding	0	2.12
				Bert Tokenizer	191	45.87

(a) Event log 1

Pelt (penalty=50)	**#samples**	**#variants/#samples**	**n-sequence**	**Encoding method**	**#change points**	**Time (sec)**
Event log 2	1055	0.00758	1	Keras Tokenizer	17	3.46
				Label Encoder	9	4.22
				One Hot Encoding	1	37.57
				Bert Tokenizer	242	74.54
	532	0.051	2	Keras Tokenizer	**82**	**8.53**

				Label Encoder	120	18.23
				One Hot Encoding	0	11.65
				Bert Tokenizer	302	122.61
	357	0.14	3	Keras Tokenizer	111	15.18
				Label Encoder	148	27.58
				One Hot Encoding	0	6.32
				Bert Tokenizer	257	84.62

(b) Event log 2

Pelt (penalty=50)	**#samples**	**#variants/#samples**	**n-sequence**	**encoding**	**#change points**	**Time (sec)**
Event log 3	123	0.11	1	Keras Tokenizer	3	0.35
				Label Encoder	12	0.45
				One Hot Encoding	0	0.67
				Bert Tokenizer	98	11.88
	63	0.41	2	Keras Tokenizer	12	0.40
				Label Encoder	27	1.10
				One Hot Encoding	0	0.43
				Bert Tokenizer	58	4.35

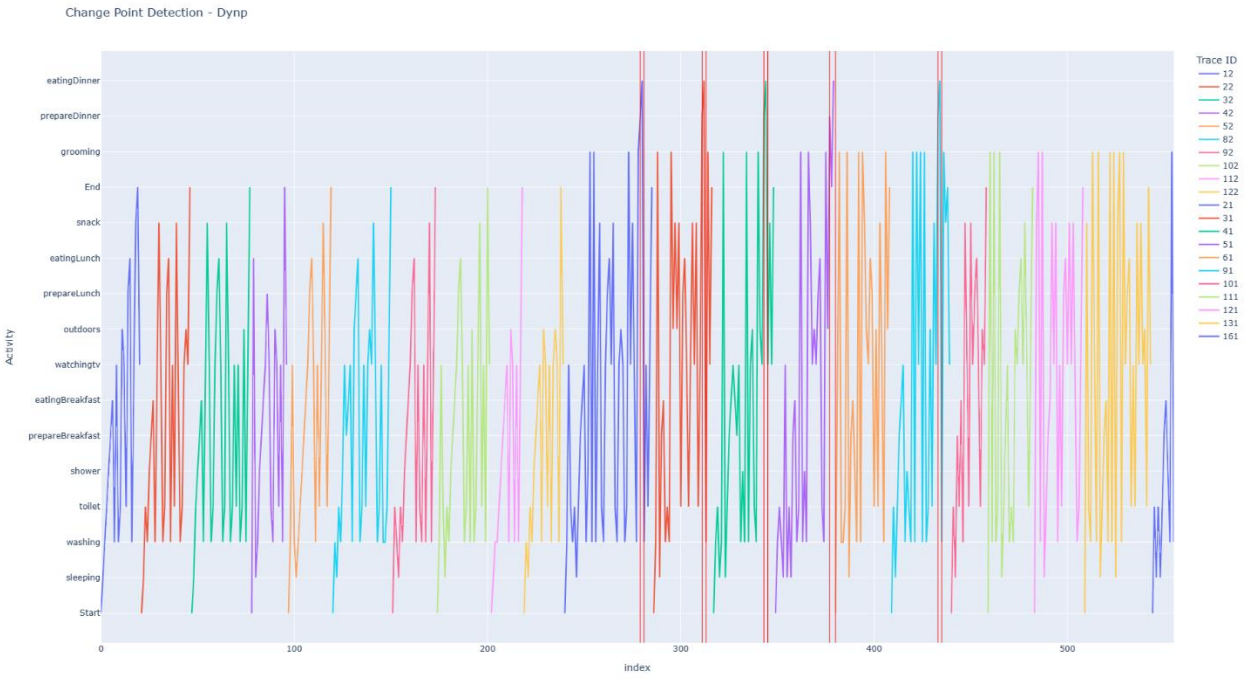
	44	0.68	3	Keras Tokenizer	17	0.58
				Label Encoder	20	0.70
				One Hot Encoding	0	0.35
				Bert Tokenizer	42	2.43

(c) Event log 3

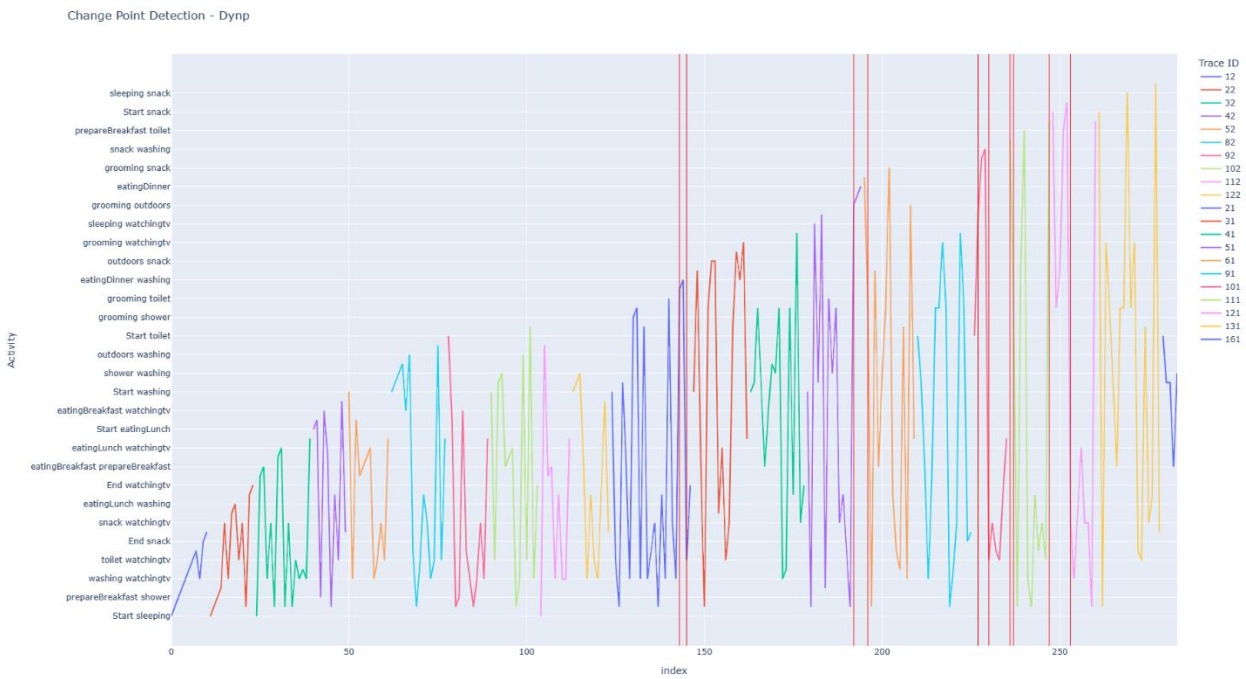
5.3.2.2 N-sequence of activities

In this section we will analyze how the number of activities in a sequence affects our framework.

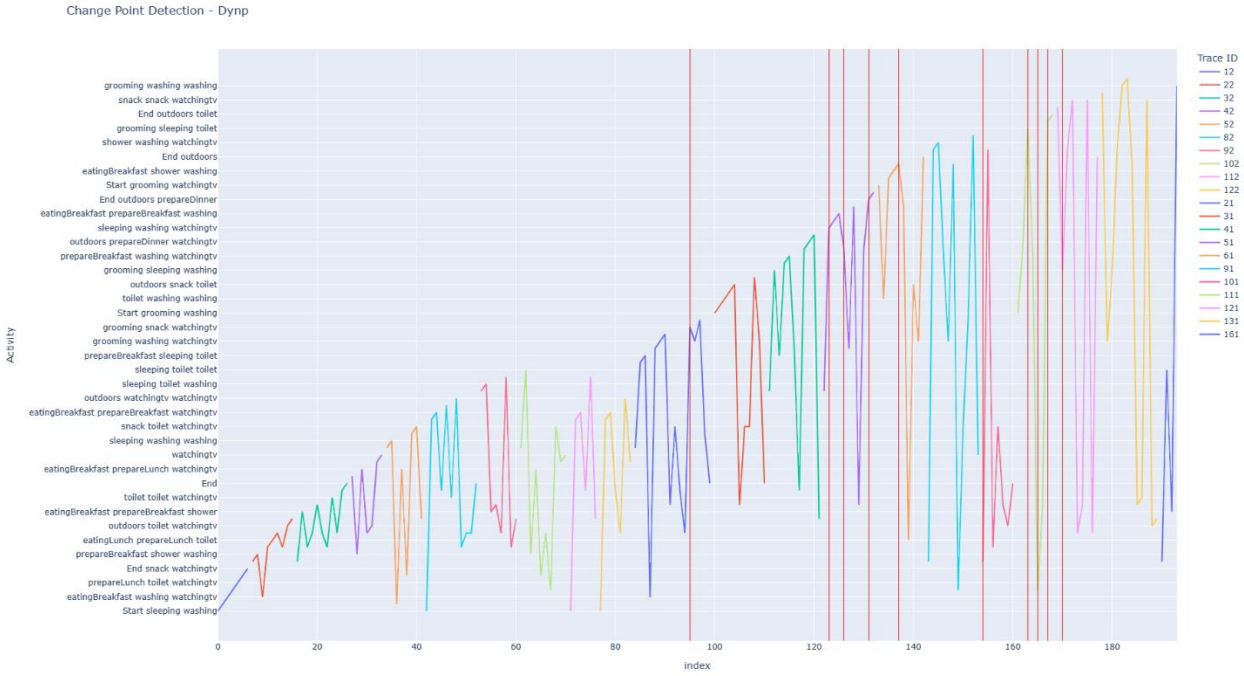
Figure 5.27 presents the position of 10 change points detected with Dynamic Programming for different number of activities in each n-sequence in event log 1. X-axis represents time, y-axis represents n-sequences of activities (either 1, 2, 3 or 4 activities in each sequence). Each day is represented with different color.



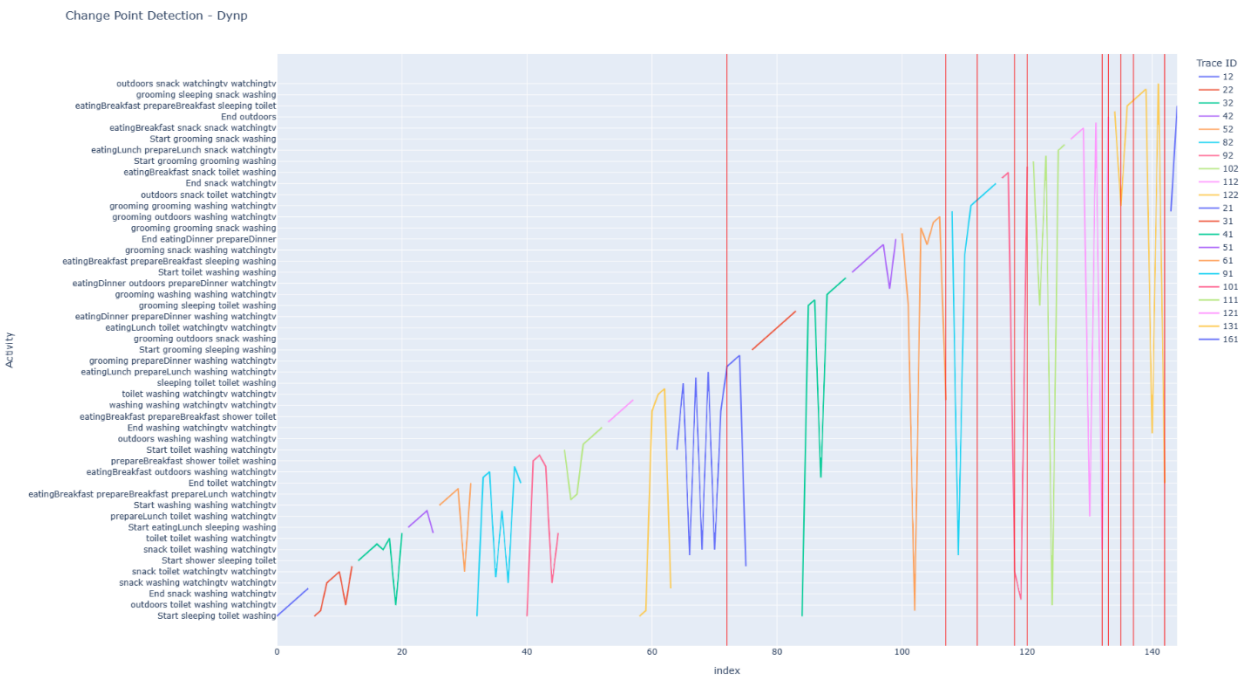
(a) 1-sequence of activities



(b) 2-sequence of activities



(c) 3-sequence of activities



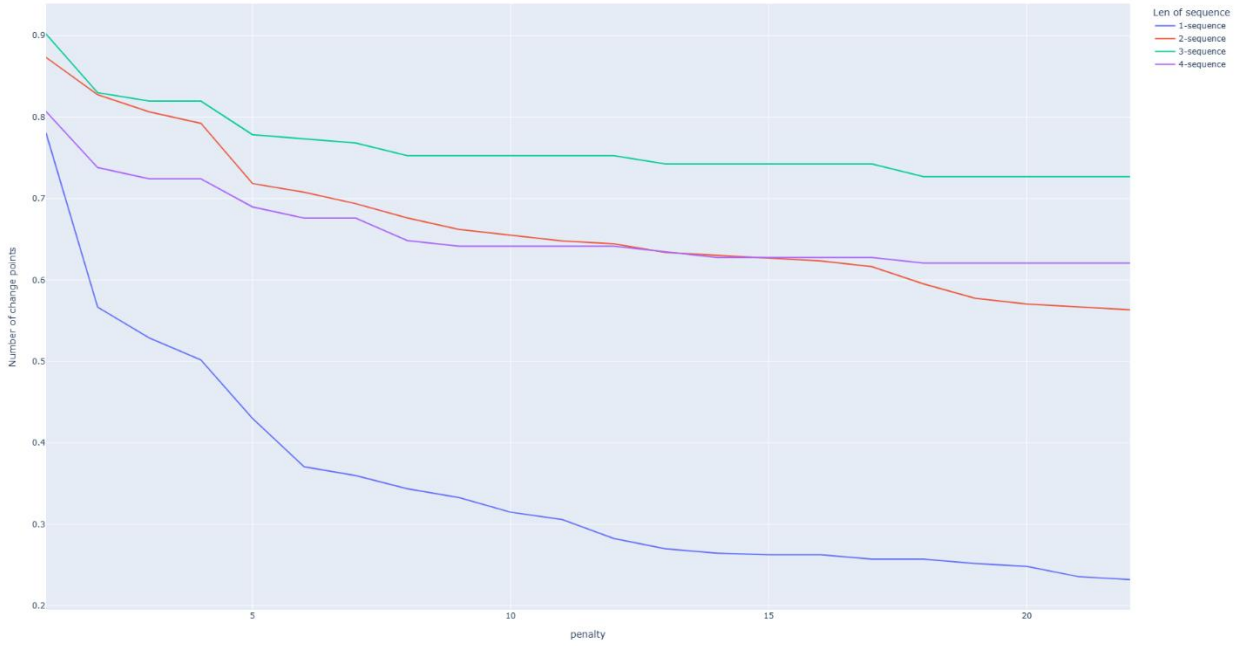
(d) 4-sequence of activities

Figure 5.27 Change points detected with Dynamic Programming algorithm for activities grouped into (a) 1-sequences (b) 2-sequences (c) 3-sequences (d) 4-sequences of activities.

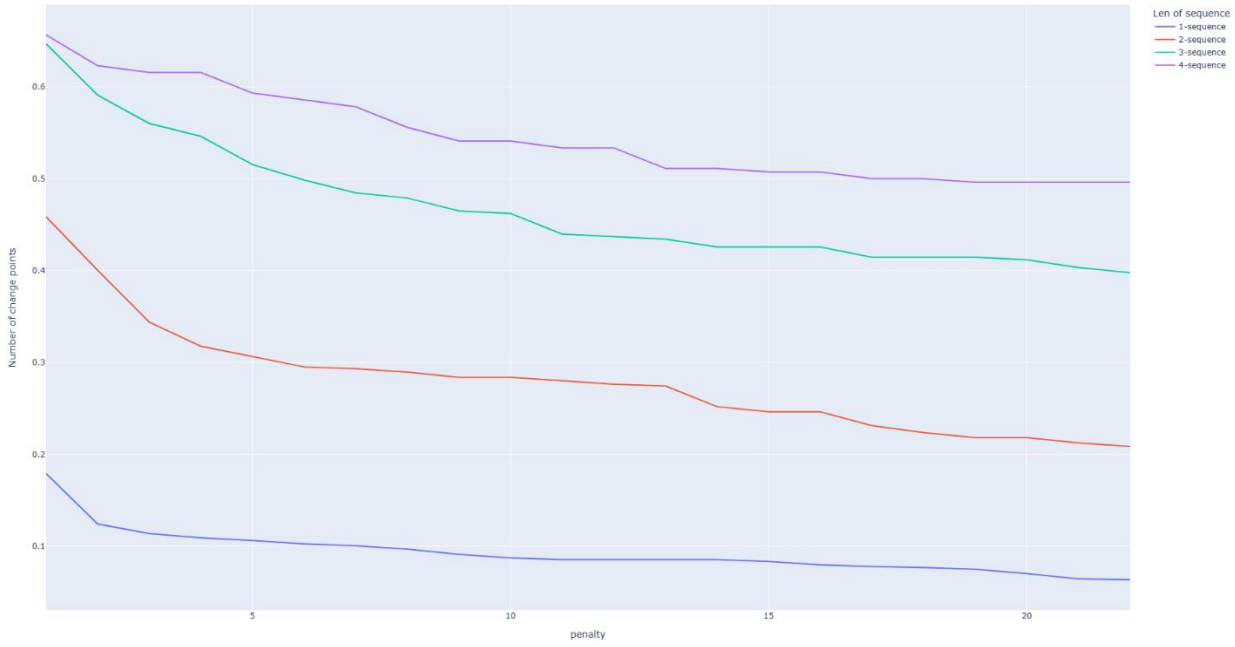
A conclusion that can be drawn is that no change points were detected in the first days for all four instances. Also, as the n increases in n-sequence, change points shift forward. This occurs because by categorizing activities into groups, the framework avoids confusion in identifying a change point when residents alter the order of activities within the same time zone.

As shown on the y-axes of Figure 5.27, the number of points increases with the growth of n. This results in an expansion of vector dimensions for both One-Hot Encoded and Bert Tokenizer encoding techniques, as illustrated in Table 5.5 and Table 5.6 in Section 5.3.2.1.

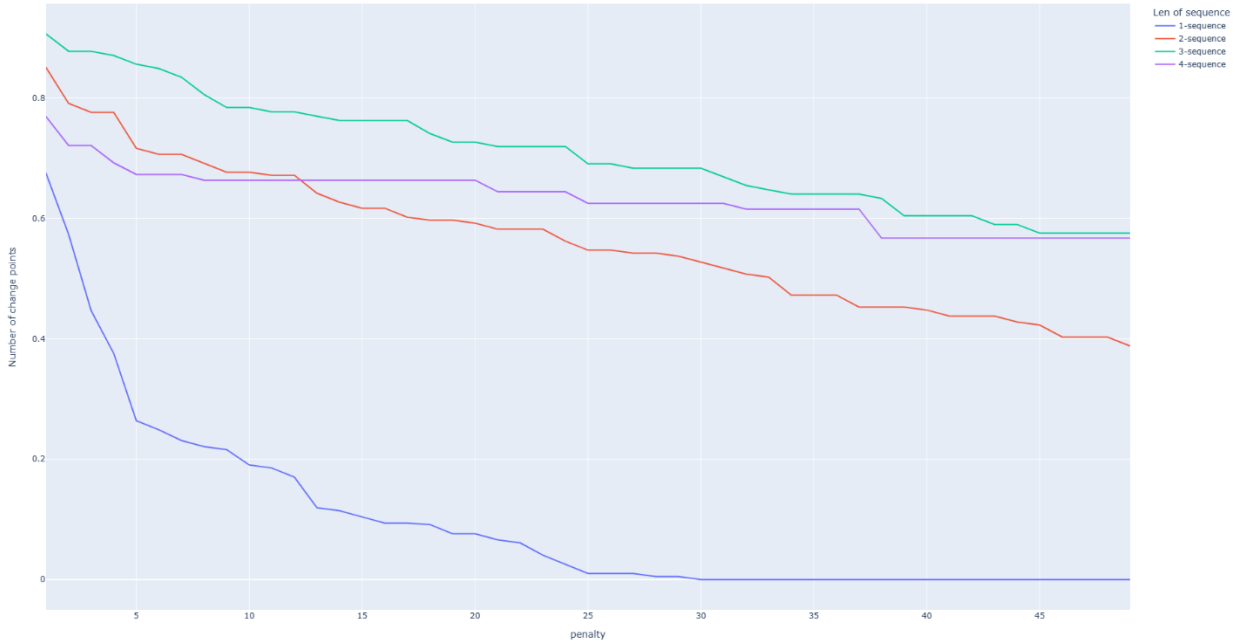
Figure 5.28 showcases the number of change points detected with Pelt (divided with the number of points to achieve normalization), varying the penalty value across different event logs and for various numbers of activities in each n-sequence.



(a) Event log 1



(b) Event log 2



(c) Event Log 4

Figure 5.28 The number of change points detected with Pelt (divided with the number of points to achieve normalization), varying the penalty value across different event logs and for various numbers of activities in each n-sequence

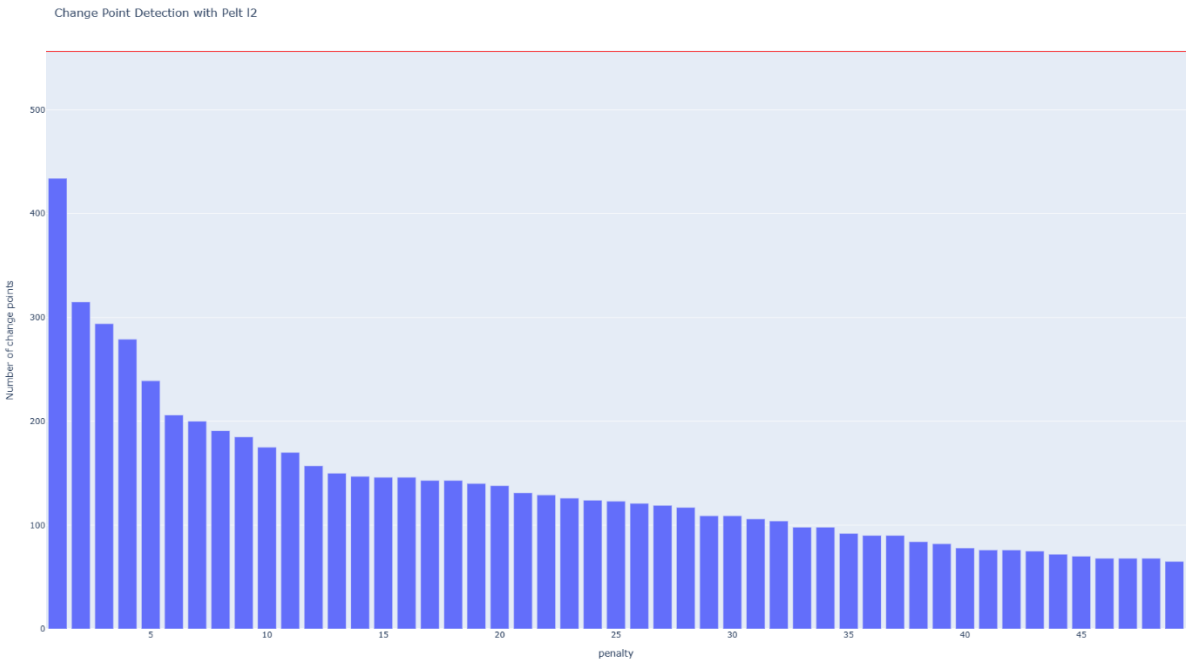
As concluded from Figure 5.28, 1-sequences of activities produce less change points. The logical deduction is that with an increase in the number of grouped activities, the number of change points is likely to decrease. This is because the routine in a time zone of the resident remains unchanged. However, as the number of activities within the group increases, the variants divided by samples also increase. Therefore, the graph in Figure 5.28 can assist us in selecting the optimal number of "n" in n-sequences.

5.3.2.3 Applying Change Point Detection Algorithms

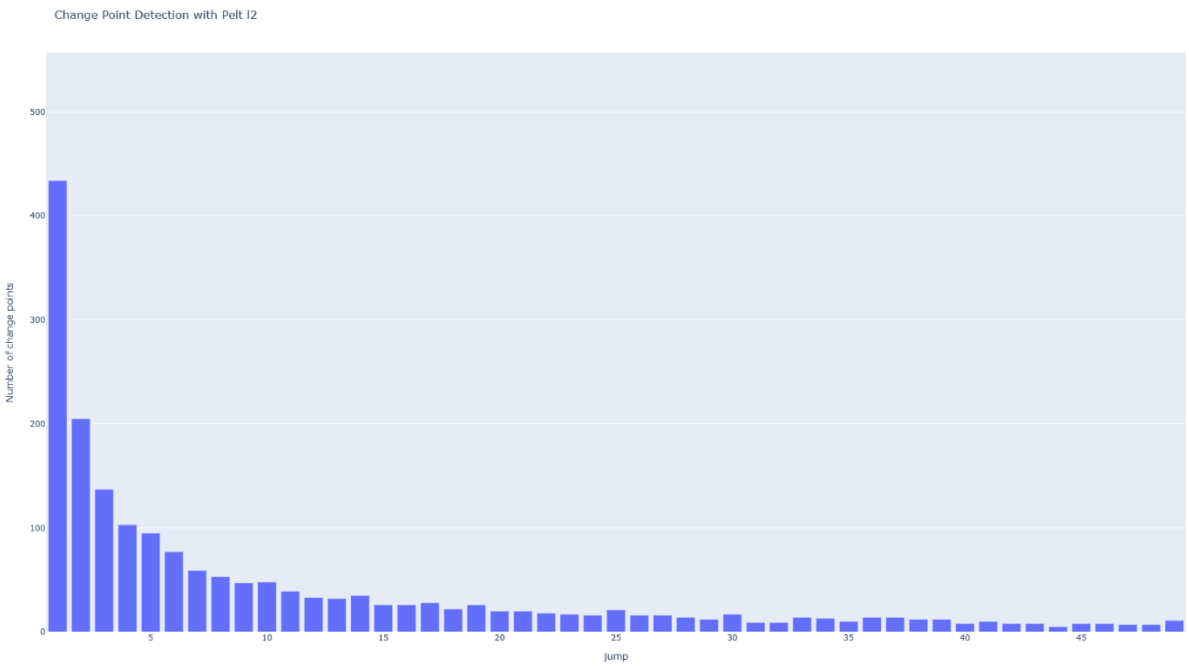
Change point detection in human activity refers to the process of identifying moments or instances when there is a significant shift or alteration in patterns, behaviors, or characteristics related to human actions. This can be applied across various domains such as healthcare, sports, behavior analysis, and more. The goal is to pinpoint moments of change or transition in the observed human activities, enabling a better understanding of shifts in behavior or circumstances. In this section, we examine the outcomes derived from the application of Change Point Detection algorithms to event logs related to human activity.

5.3.2.3.1 Pelt Change Point Detection

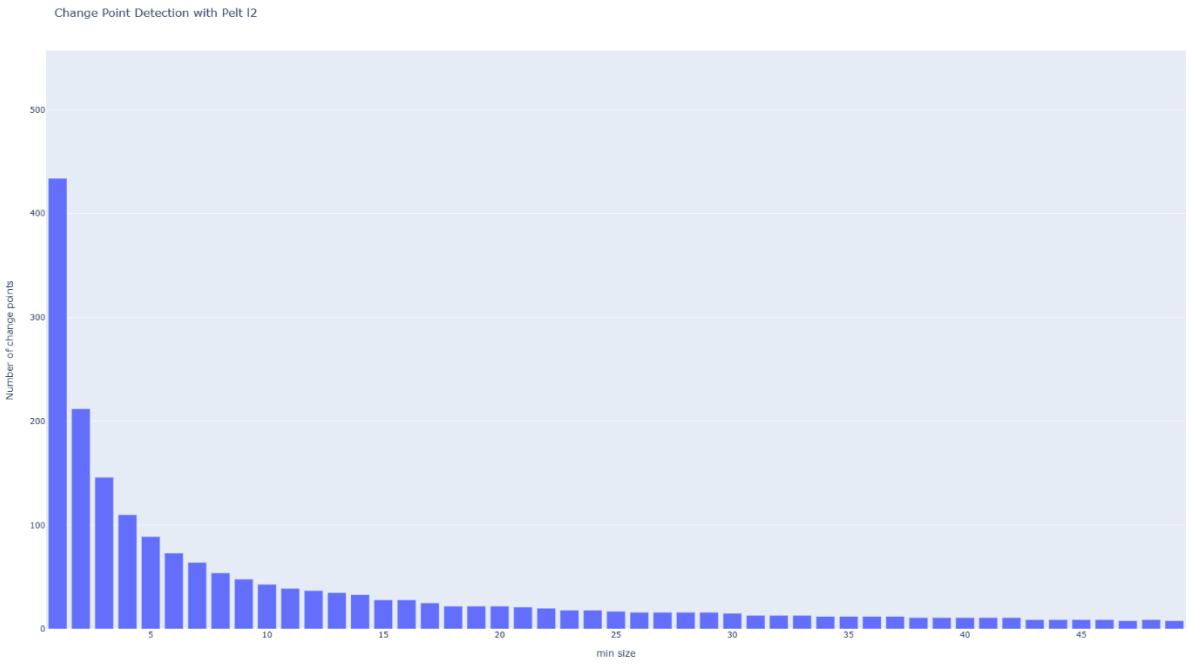
The Pelt change point detection algorithm has three main parameters, as discussed in section 3.2.1: penalty value, jump value, and minimum size value. These parameters influence the number of change points, as illustrated in Figure 5.29.



(a) Penalty parameter



(b) Jump parameter



(c) Minimum size parameter

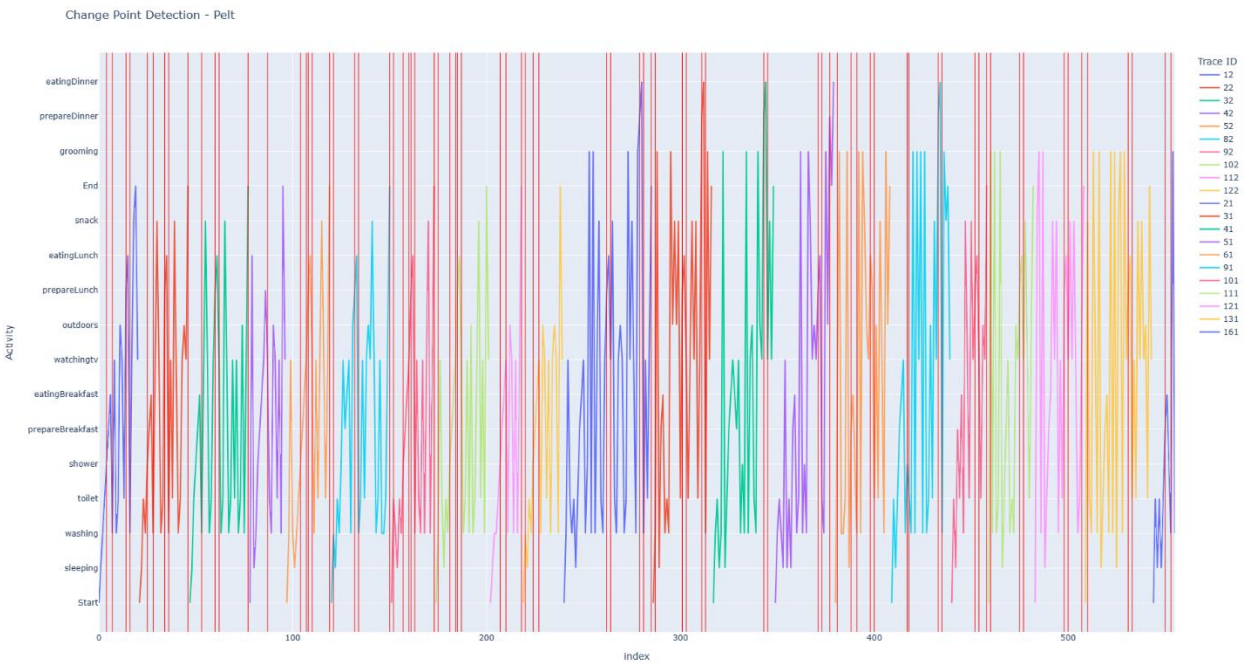
Figure 5.29 Number of change points with Pelt Change Point Detection algorithm on event log 1 related to (a) penalty value (b) jump value (c) min size value.

These parameters assist the user in deciding how to approach the study of human behavior. For instance, if the user aims to identify only significant changes in human life, increasing the penalty value is advisable. If the user intends to investigate daily fluctuations in human behavior and identify changes within specific time zones, adjusting the jump parameter becomes necessary. If the user wants to specify the minimum distance between changes, the min size parameter must be set.

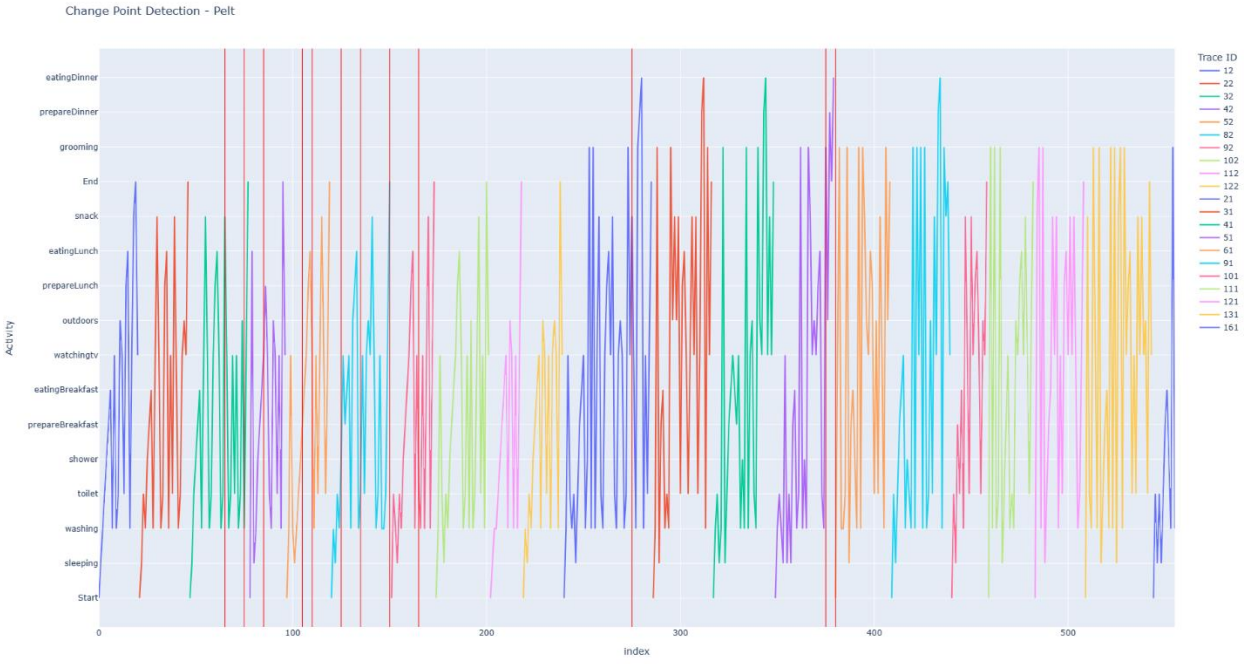
In Figure 5.30 the change points that was detected with Pelt algorithm for specific values of penalty, jump and min size parameters on event log 1 is depicted. The encoding method that was used is Keras Tokenizer. As it can be seen Pelt algorithm with penalty value 40 produce 78 change points. If jump parameter set with value of 5, 12 change point detected in multiple of 5 locations. If min size parameter set to 5, 31 change points detected that is no closer than 5 activities. The results are summarized in Table 5.8.

Table 5.8 Pelt Change Point Detection on event log 1

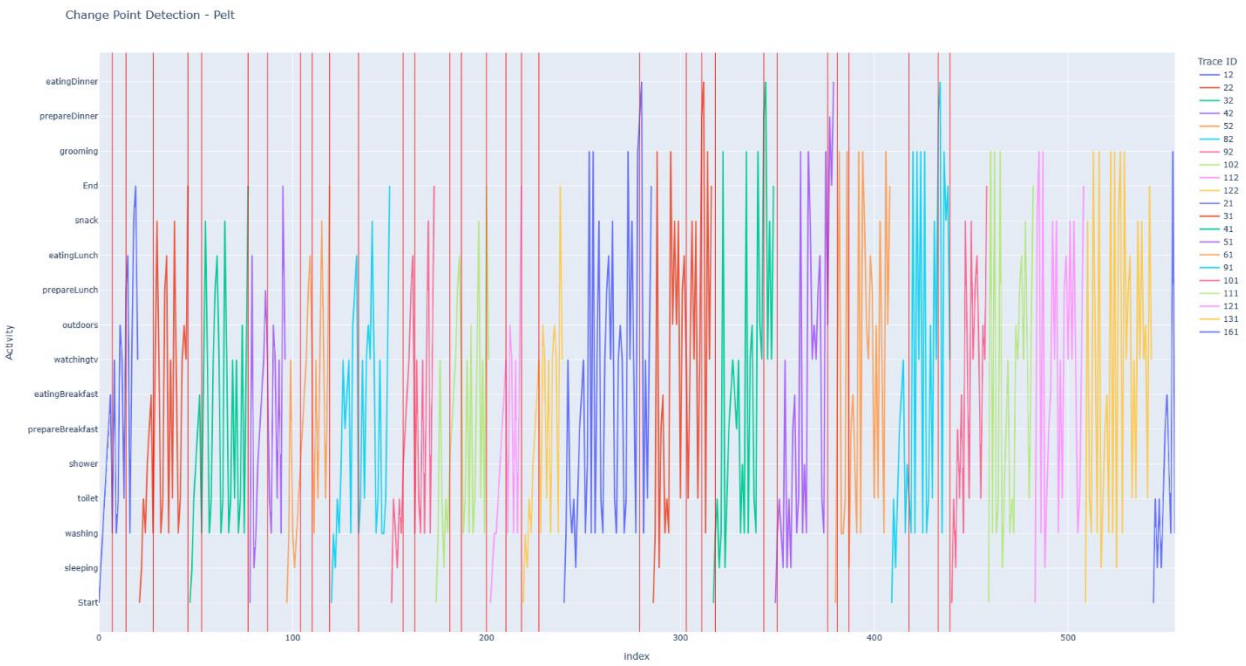
Pelt		
	#change points	**time (sec)**
Penalty 40	78	8.20
Penalty 40 – Jump 5	12	0.46
Penalty 40 – Min size 5	31	1.85



(a) penalty 40



(b) penalty 40, jump 5



(c) penalty 40, min size 5

Figure 5.30 Pelt algorithm on event log 1 with penalty value 40, jump value 5 and minimum size value 5.

5.3.2.3.2 Dynamic Programming Change Point Detection

In the Dynamic Programming algorithm, the user needs to predefine the number of changes to detect. To minimize computational costs, it is possible to focus on only a subset of potential change point indexes by adjusting the min size and jump arguments. In Table 5.9, a summary of the change points identified using the Dynamic Programming algorithm is presented (employing Keras Tokenizer for encoding). It can be inferred that Dynamic Programming requires significantly more time to detect the same change points compared to Pelt. However, the detection time experiences a considerable reduction when utilizing the jump and min size parameters.

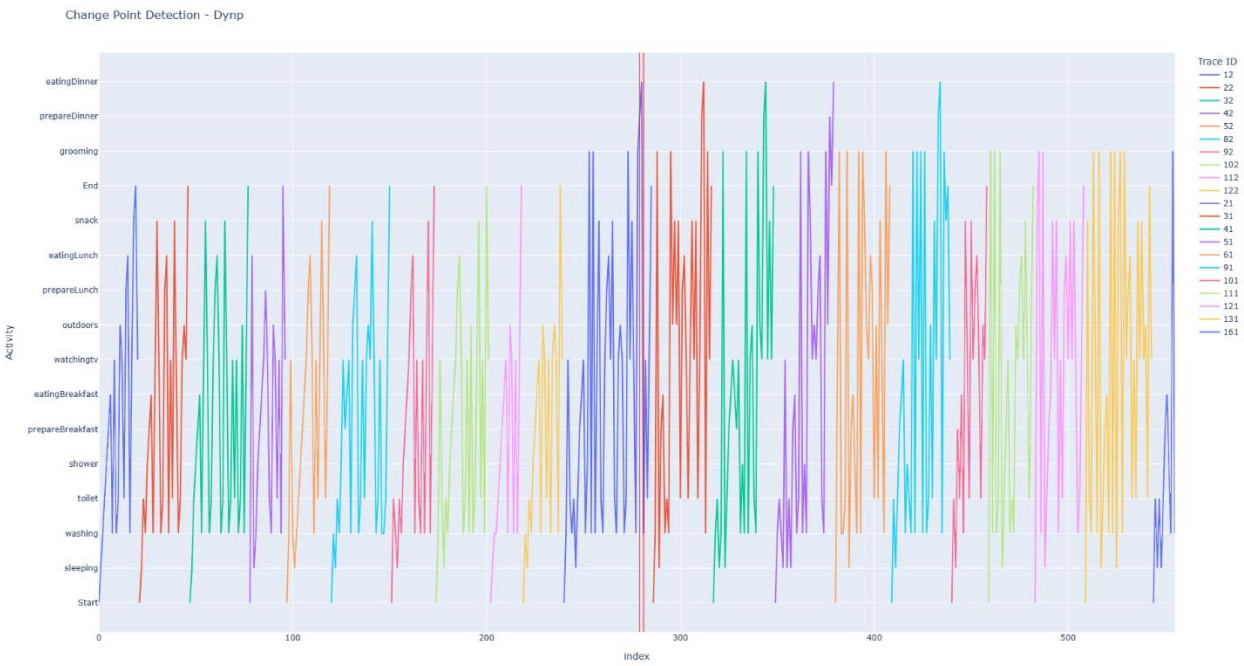
Table 5.9 Summary of the change points identified using the Dynamic Programming algorithm

Dynamic Programming		
	#change points	**time (sec)**
Nbkps 78	78	102.92
Nbkps 78 – Jump 5	78	8.47

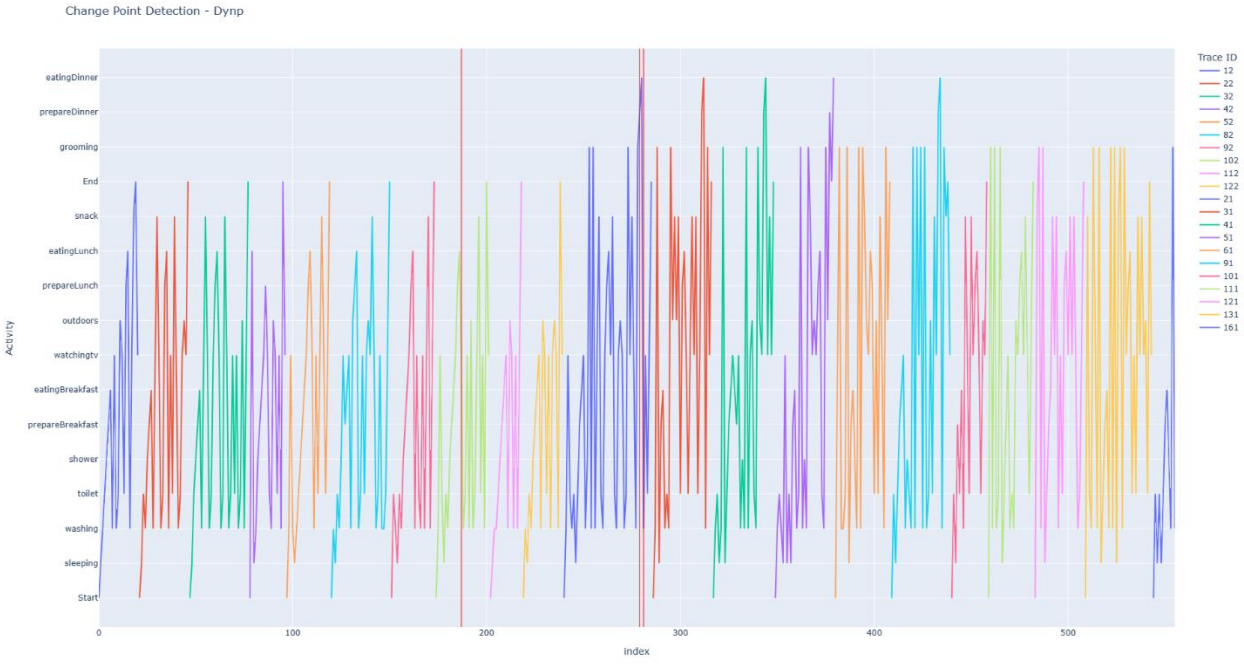
Penalty 40 – Min size 5	78	23.30

Dynamic Programming could be useful to find the one most significant change in human behavior. In Figure 5.31, 1, 2 and 3 most significant changes are shown, using Dynamic Programming algorithm.

(a) One change point



(b) Two change points

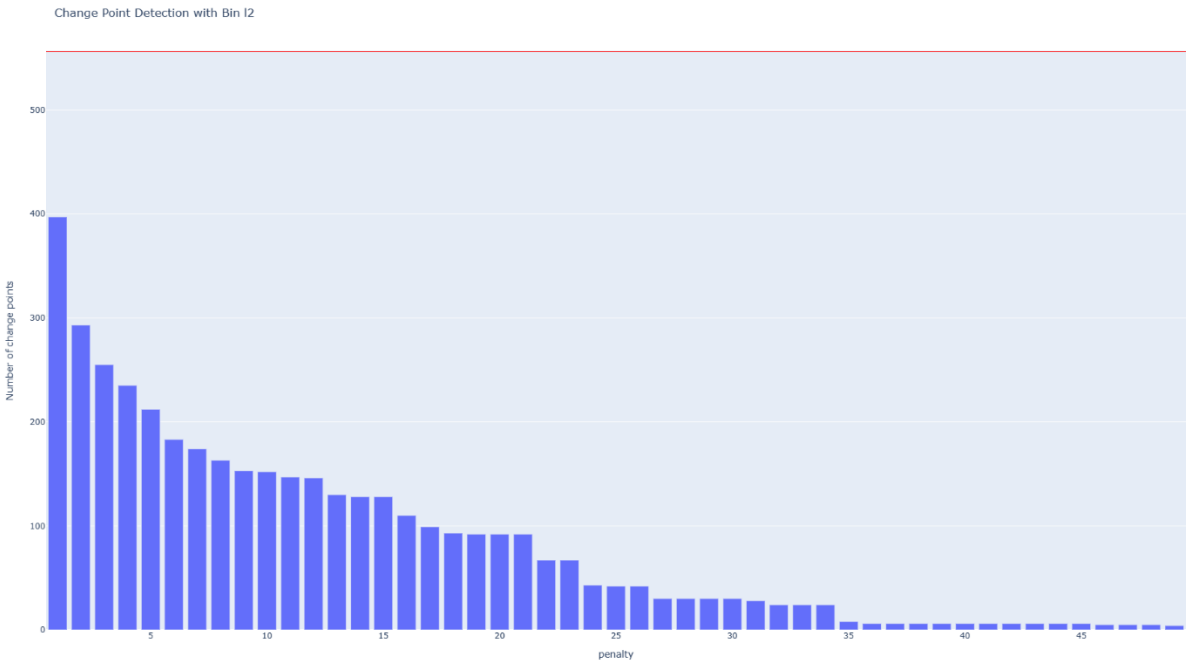


(c) Three change points

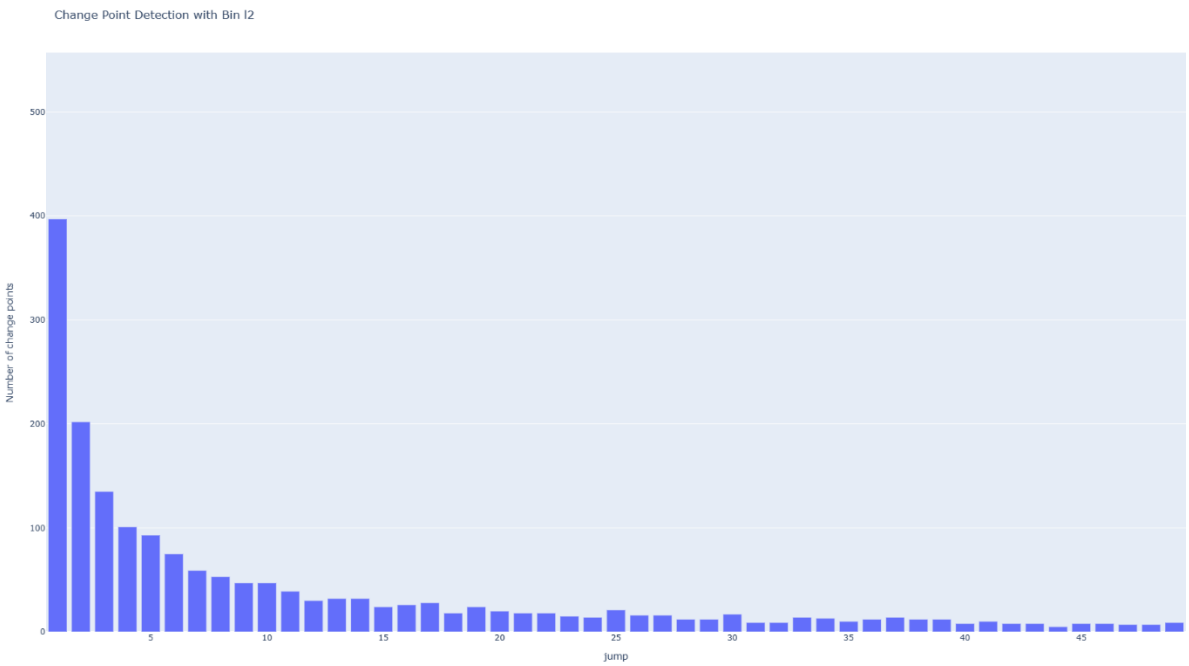
Figure 5.31 The most significant change points detected with Dynamic Programming algorithm

5.3.2.3.3 Binary Segmentation Change Point Detection

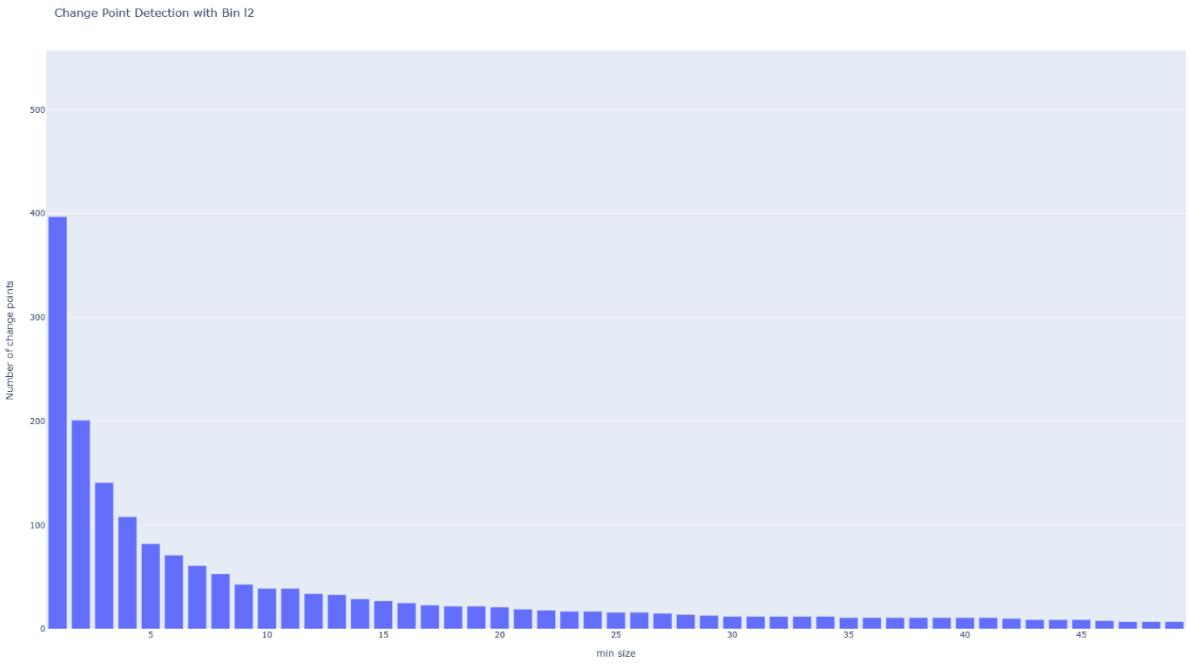
The Binary Segmentation change point detection algorithm has three main parameters, as discussed in section 3.2.3: penalty value, jump value, and minimum size value. These parameters influence the number of change points, as illustrated in Figure 5.32.



(a) Penalty parameter



(b) Jump parameter



(c) Minimum size parameter

Figure 5.32 Number of change points with Binary Segmentation Change Point Detection algorithm on event log 1 related to (a) penalty value (b) jump value (c) min size value.

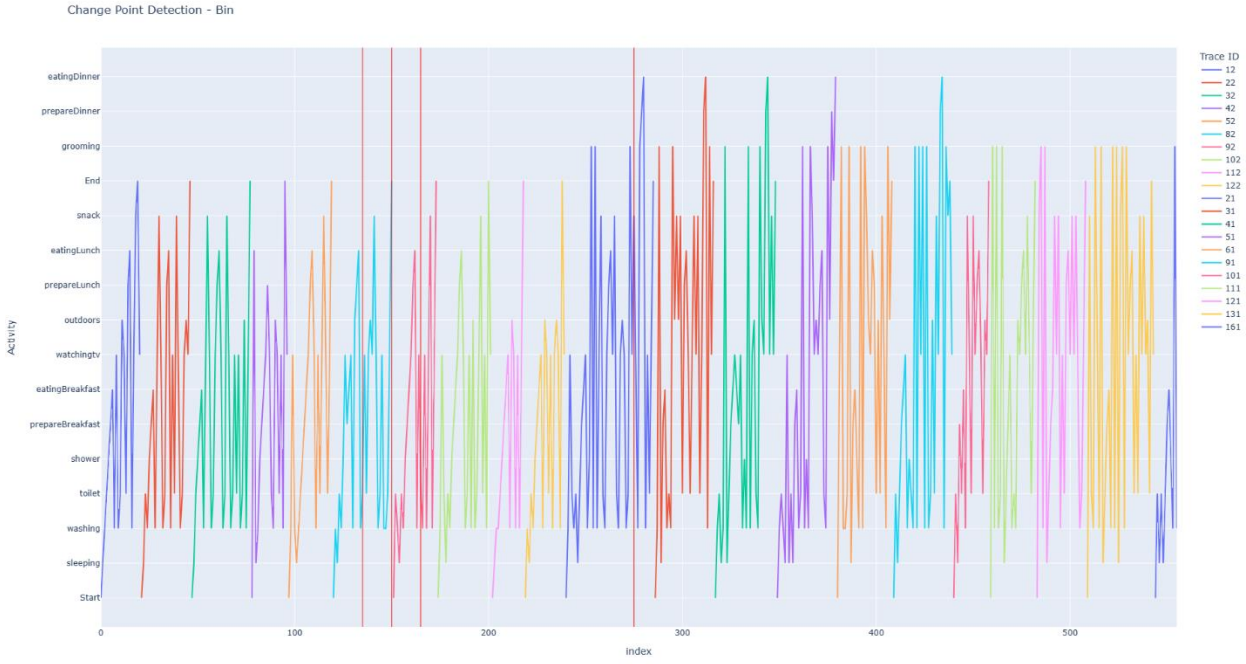
These parameters aid users in customizing their approach to studying human behavior. For example, elevating the penalty value is recommended if the goal is to detect only substantial changes in human life. Adjusting the jump parameter becomes essential when investigating daily variations and identifying changes within specific time zones. Setting the min size parameter is necessary if the user wishes to specify the minimum distance between changes.

In Figure 5.33 the change points that was detected with Binary Segmentation algorithm for specific values of penalty, jump and min size parameters on event log 1 is depicted. The encoding method that was used is Keras Tokenizer. As it can be seen Binary Segmentation algorithm with penalty value 40 produce 78 change points. If jump parameter set with value of 5, 12 change point detected in multiple of 5 locations. If min size parameter set to 5, 31 change points detected that is no closer than 5 activities. The results are summarized in Table 5.10

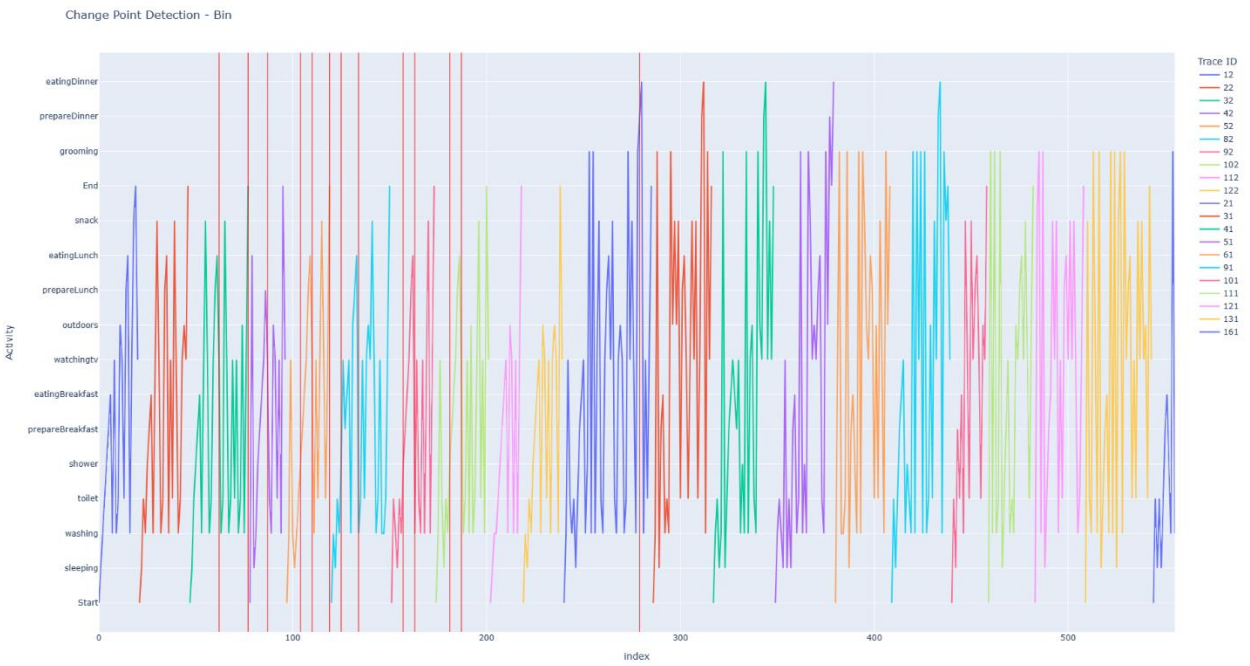
Table 5.10 Pelt Change Point Detection on event log 1

Binary Segmentation		
	#change points	#time (sec)
Penalty 40	6	0.50
Penalty 40 – Jump 5	4	0.24
Penalty 40 – Min size 5	13	0.71

(a) Penalty 40



(b) Penalty 40, jump 5



(c) Penalty 40, min size 5

Figure 5.33 Binary Segmentation algorithm on event log 1 with penalty value 40, jump value 5 and minimum size value 5.

In comparison to Pelt, Binary Segmentation identified fewer change points with the respective penalty parameter values. Figure 5.34 illustrates the corresponding time of detecting change points for various penalty values in both Pelt and Binary Segmentation, along with the number of change points. The plot incorporates two y-axes; the left y-axis represents time, while the right y-axis represents the number of detected change points. The x-axis indicates the penalty values for both Pelt and Binary Segmentation algorithms. The blue and red lines depict that Pelt requires more time to detect change points for the specified penalty values but detects more change points, as indicated by the green and purple lines.

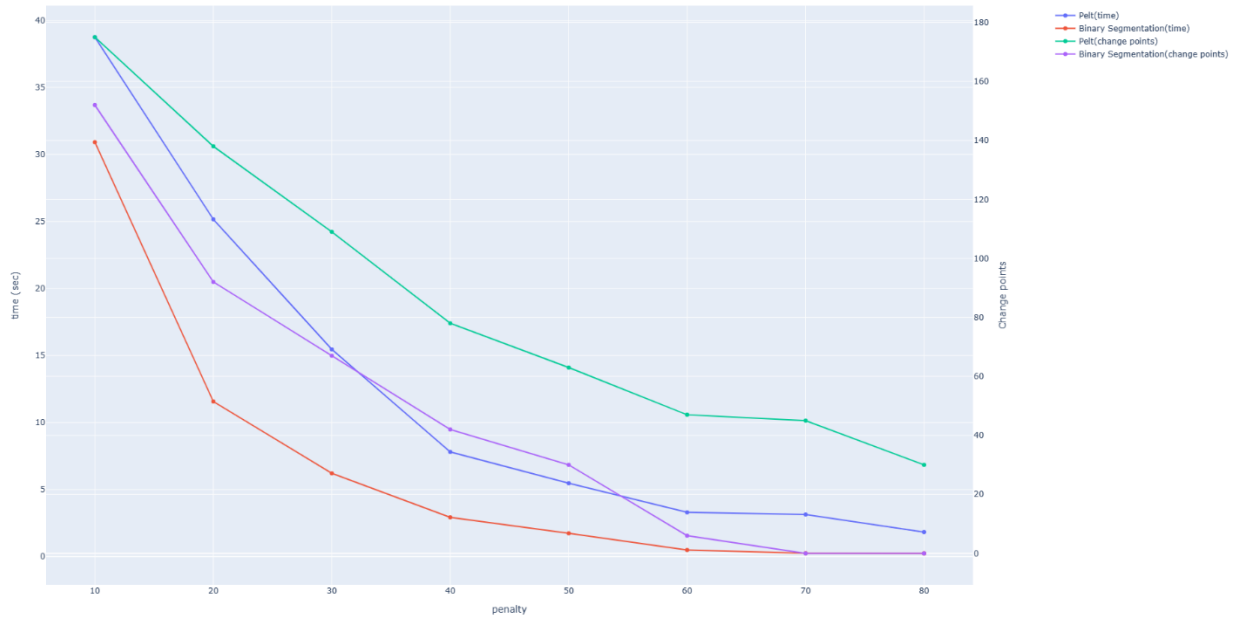


Figure 5.34 Time and number of change points detected using Pelt and Binary Segmentation algorithms with respect to penalty value. The plot consist of 2 y-axis. The left y-axis represent time and the right y-axis represents the number of change points. X-axis represents penalty parameter values.

5.3.2.4 Applying Clustering Algorithms

Clustering techniques in human activities encompass the utilization of algorithms and methodologies to categorize or group patterns, behaviors, and data points associated with human actions. The objective is to discern inherent structures or dissimilarities within datasets, facilitating the comprehension and analysis of intricate patterns in human activity. Change points exist as transitions from one similarity to another. In this section, we examine K-Means and DBSCAN clustering techniques designed to identify alterations in event logs related to human activities.

5.3.2.4.1 K-Means Clustering

In the framework K-Means algorithm is used as an unsupervised technique to categorize each n-sequence of activities (point) into a category. Then a change point is considered each time the cluster change from one n-sequence (point) to the next one. The number of clusters can be either defined by the user or automatically by elbow method which finds the optimal number of clusters.

Figure 5.35 presents the optimal number of clusters (red line) using elbow method for event log 1 (the value is 5).

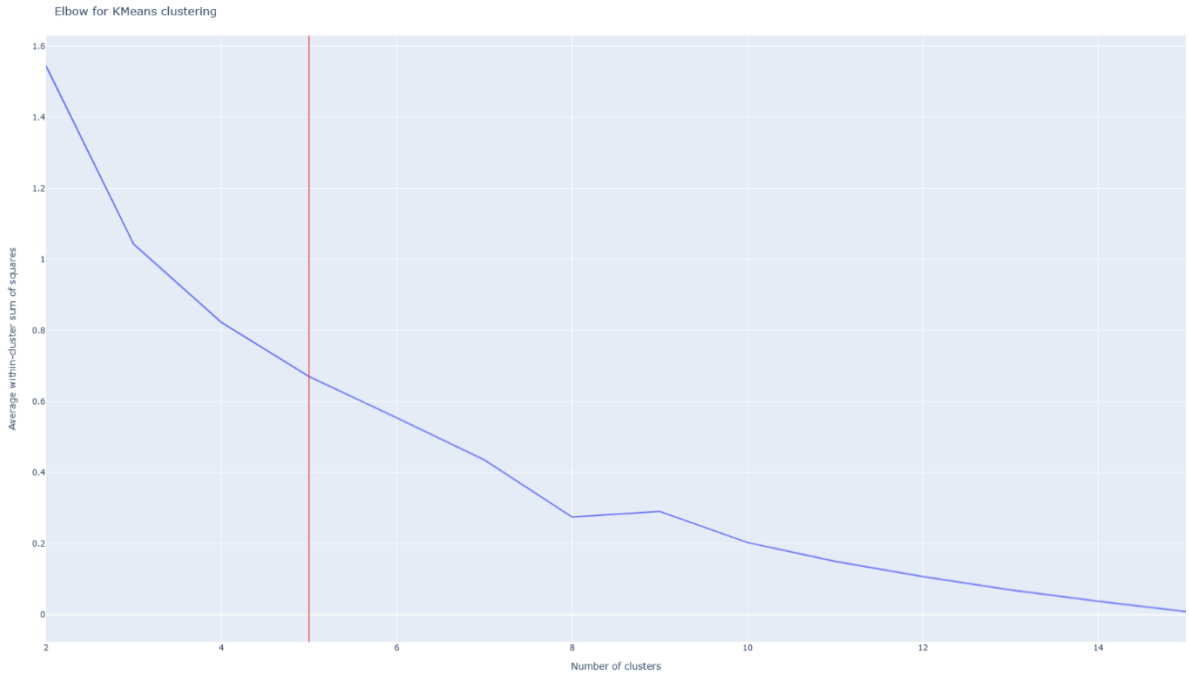


Figure 5.35 Optimal number of clusters based on elbow method

In Figure 5.36 the clusters of data points is depicted for event log 1. In this case K-Means applied with 2 clusters (blue and yellow points).

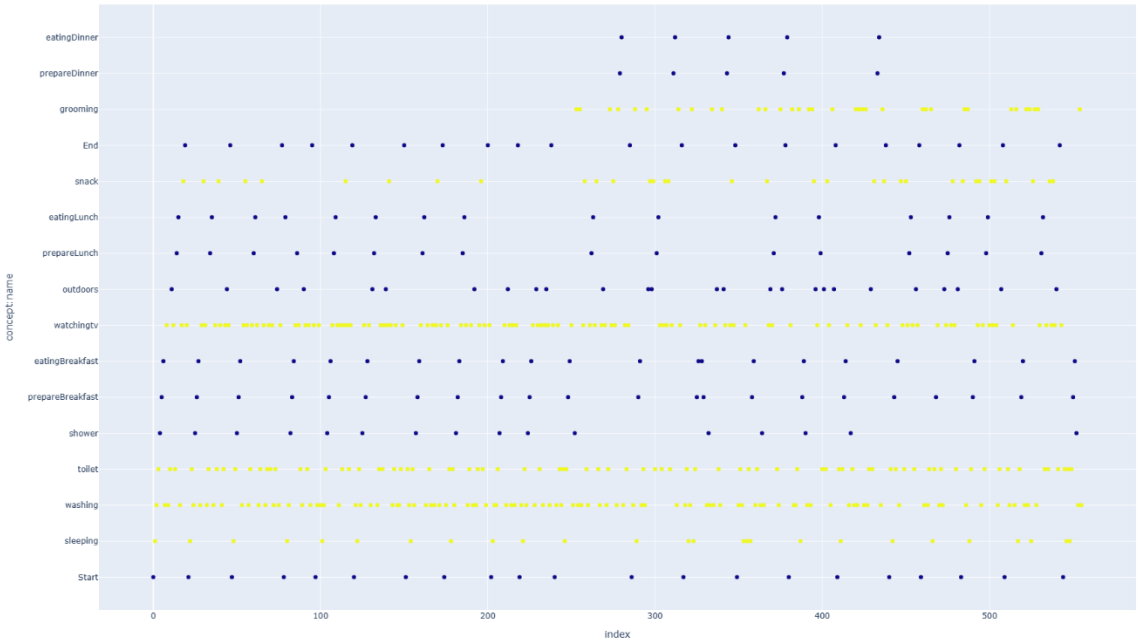


Figure 5.36 K-Means clustering on event log 1 with 2 clusters

After applying K-Means algorithm on dataset the framework consider change points each time the cluster of each point (n-sequence) change from one point to the next one.

5.3.2.4.2 DBSCAN Clustering

DBSCAN requires only two parameters: *epsilon (eps)* and *min samples*. *Epsilon* is the radius of the circle to be created around each data point to check the density and *minPoints* is the minimum number of data points required inside that circle for that data point to be classified as a **Core** point. Figure 5.37 shows number of clusters derived from various values of eps and minPoints.

Figure 5.37 Number of clusters derived from various values of eps and minPoints

The user has the flexibility to choose the number of clusters by adjusting different eps and min sample values. Figure 5.38 depicts event log 1, encoded using the Keras Tokenizer method, as a scatter plot with points colored according to their cluster values. The minimum cluster value calculated across various eps and min sample values is 2.

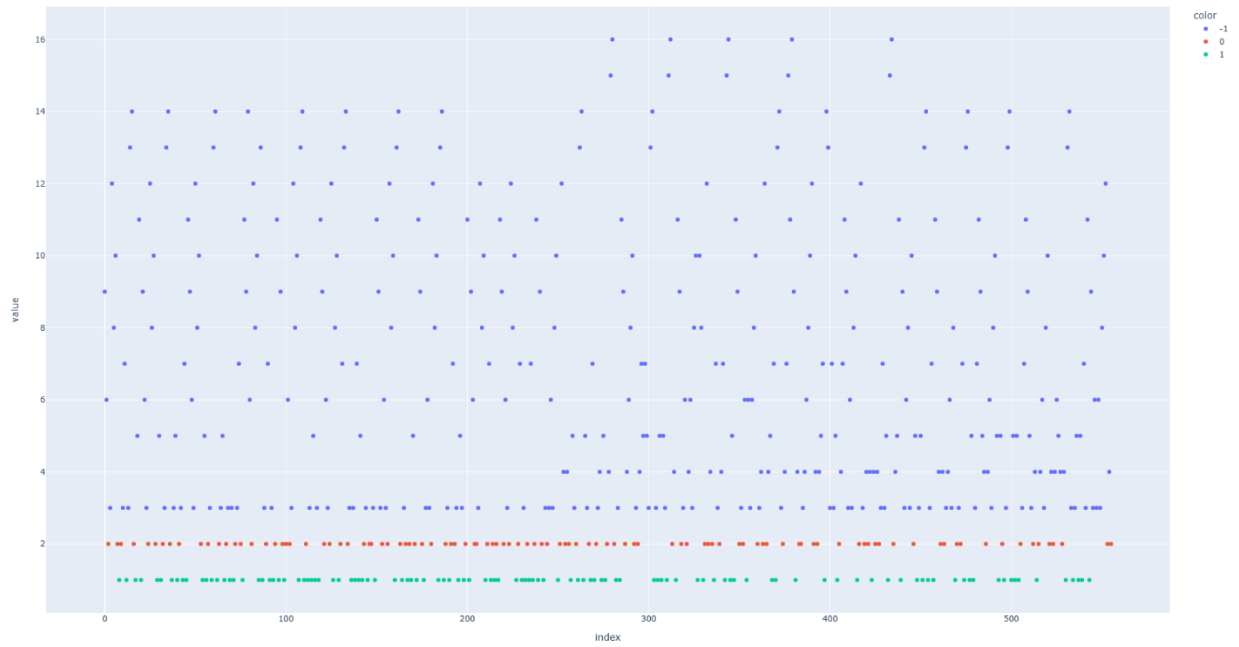


Figure 5.38 DBSCAN clustering on event log 1

After applying DBSCAN algorithm on dataset the framework consider change points each time the cluster of each sample (n-sequence) change from one sample to the next one.

5.3.2.5 Combo activity pattern detection

Following the identification of change points through a change point detection algorithm and clustering techniques, the framework combines the detected change points and acknowledges a change point in the event log only when it is detected by all the applied techniques.

Figure 5.39 presents change points detected with Pelt algorithm (red lines) and change points detected with Pelt and Kmeans with 2 clusters (green lines) as scatter plot colored based on kmeans clusters. The length of each n-sequence of activities is 2 and Keras Tokenizer encoding technique was used. Figure 5.40 presents change points detected with Pelt algorithm (red lines) and change points detected with Pelt and DBSCAN with 2 clusters (green lines) as scatter plot colored based on DBSCAN clusters. The length of each n-sequence of activities is 2 and Keras Tokenizer encoding technique was used.

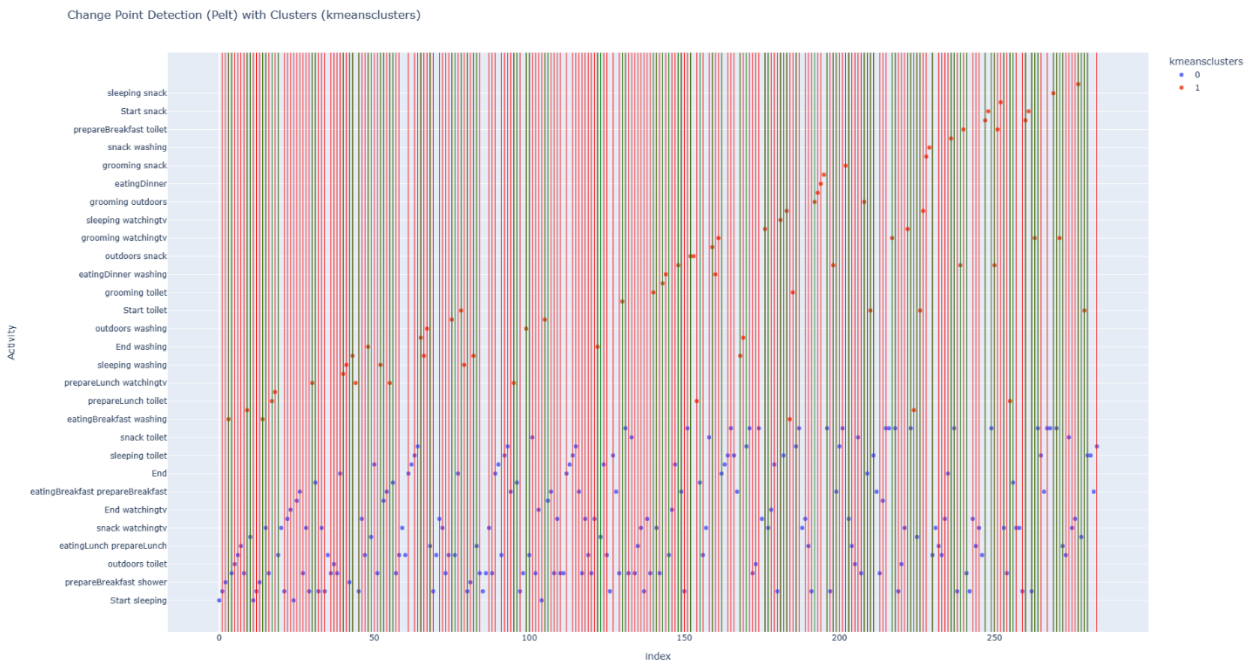


Figure 5.39 Change points detected with Pelt algorithm (red lines) and change points detected with Pelt and Kmeans with 2 clusters (green lines) as scatter plot colored based on KMeans clusters. The length of each n-sequence of activities is 2 and Keras Tokenizer encoding technique was used.

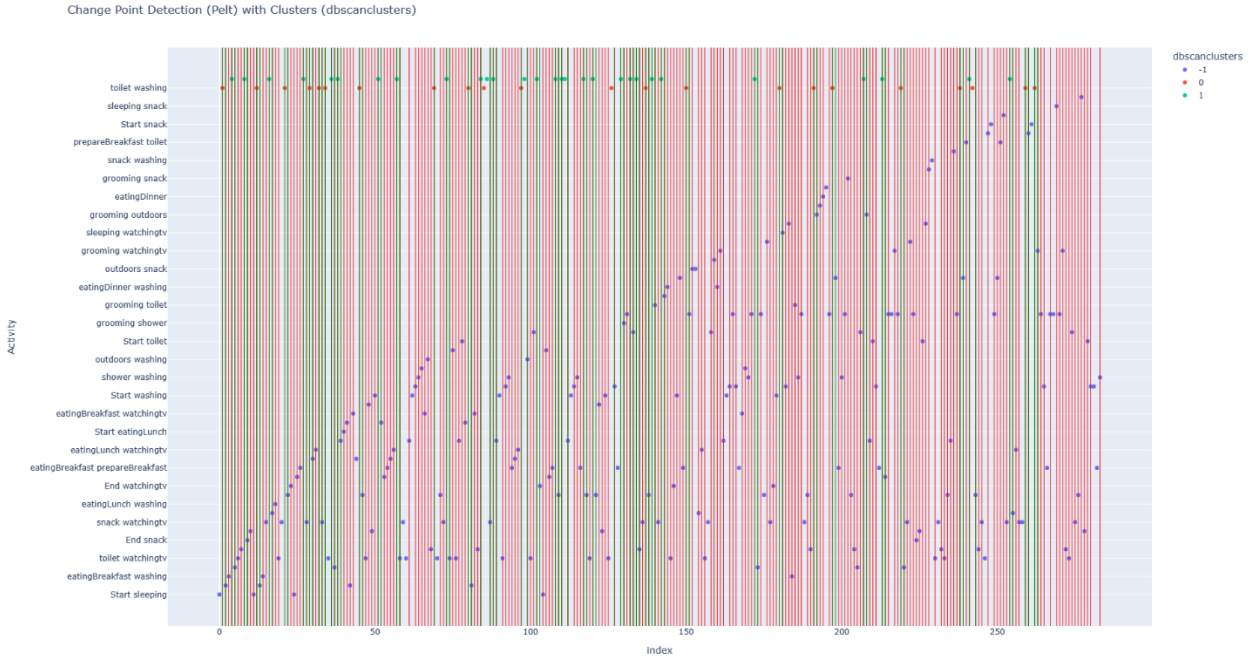


Figure 5.40 Change points detected with Pelt algorithm (red lines) and change points detected with Pelt and DBSCAN with 2 clusters (green lines) as scatter plot colored based on DBSCAN clusters. The length of each n-sequence of activities is 2 and Keras Tokenizer encoding technique was used.

Figure 5.41 presents 10 change points detected with Dynamic Programming algorithm (red lines) and change points detected with Dynamic Programming and KMeans with 2 clusters (green lines) as scatter plot colored based on KMeans clusters. The length of each n-sequence of activities is 2 and Keras Tokenizer encoding technique was used.

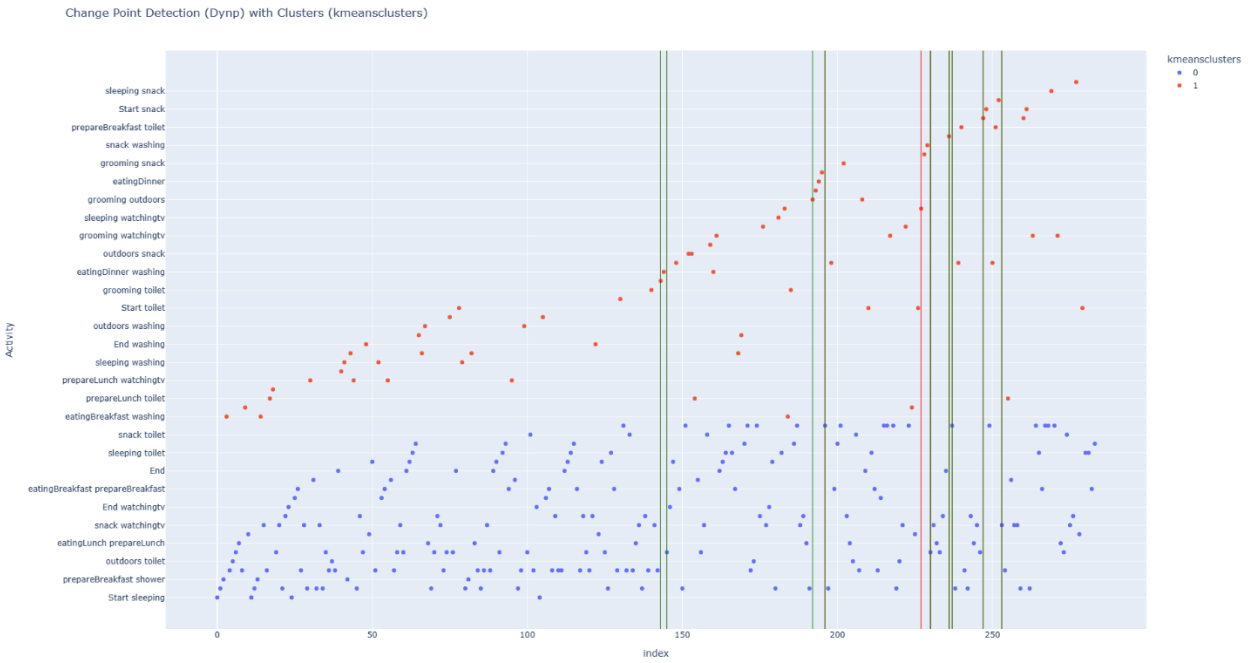


Figure 5.41 Ten change points detected with Dynamic Programming algorithm (red lines) and change points detected with Dynamic Programming and KMeans with 2 clusters (green lines) as scatter plot colored based on KMeans clusters. The length of each n-sequence of activities is 2 and Keras Tokenizer encoding technique was used.

Figure 5.42 presents 10 change points detected with Dynamic Programming algorithm (red lines) and change points detected with Dynamic Programming and DBSCAN with 2 clusters (green lines) as scatter plot colored based on DBSCAN clusters. The length of each n-sequence of activities is 2 and encoding technique was used.

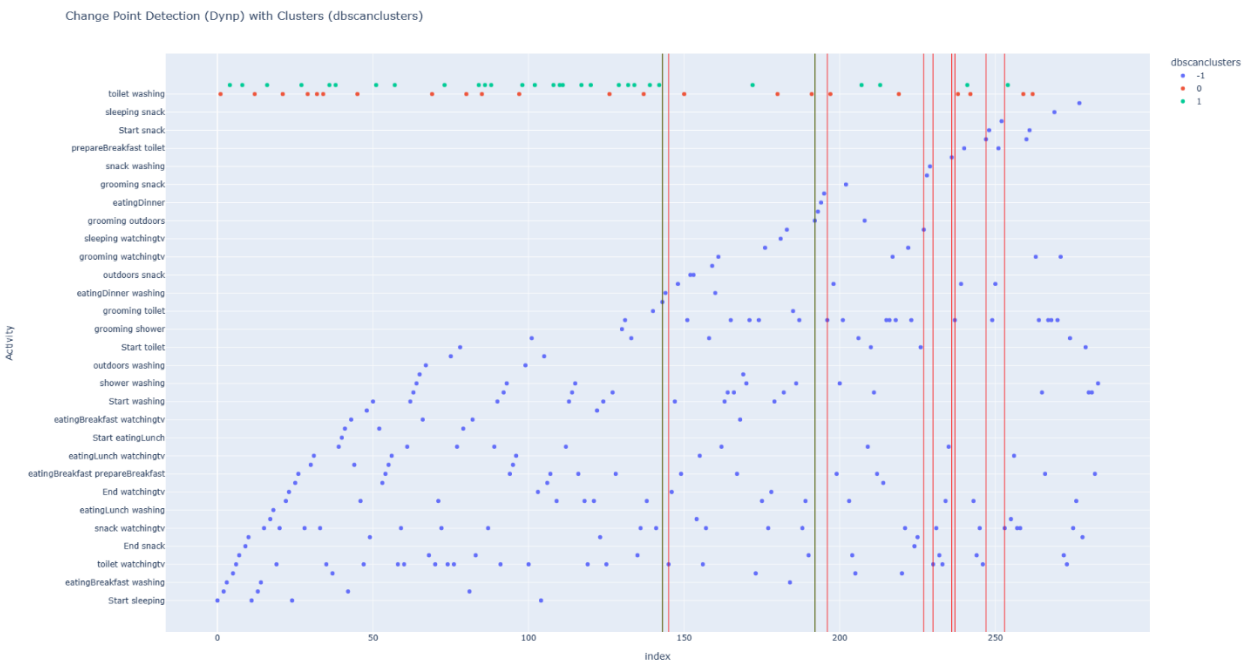


Figure 5.42 Change points detected with Dynamic Programming algorithm (red lines) and change points detected with Dynamic Programming and DBSCAN with 2 clusters (green lines) as scatter plot colored based on DBSCAN clusters. The length of each n-sequence of activities is 2 and Keras Tokenizer encoding technique was used.

Figure 5.43 presents change points detected with Binary Segmentation algorithm (red lines) and change points detected with Binary Segmentation and Kmeans with 2 clusters (green lines) as scatter plot colored based on kmeans clusters. The length of each n-sequence of activities is 2 and Keras Tokenizer encoding technique was used. Figure 5.44 presents change points detected with Binary Segmentation algorithm (red lines) and change points detected with Binary Segmentation and DBSCAN with 2 clusters (green lines) as scatter plot colored based on DBSCAN clusters. The length of each n-sequence of activities is 2 and Keras Tokenizer encoding technique was used.

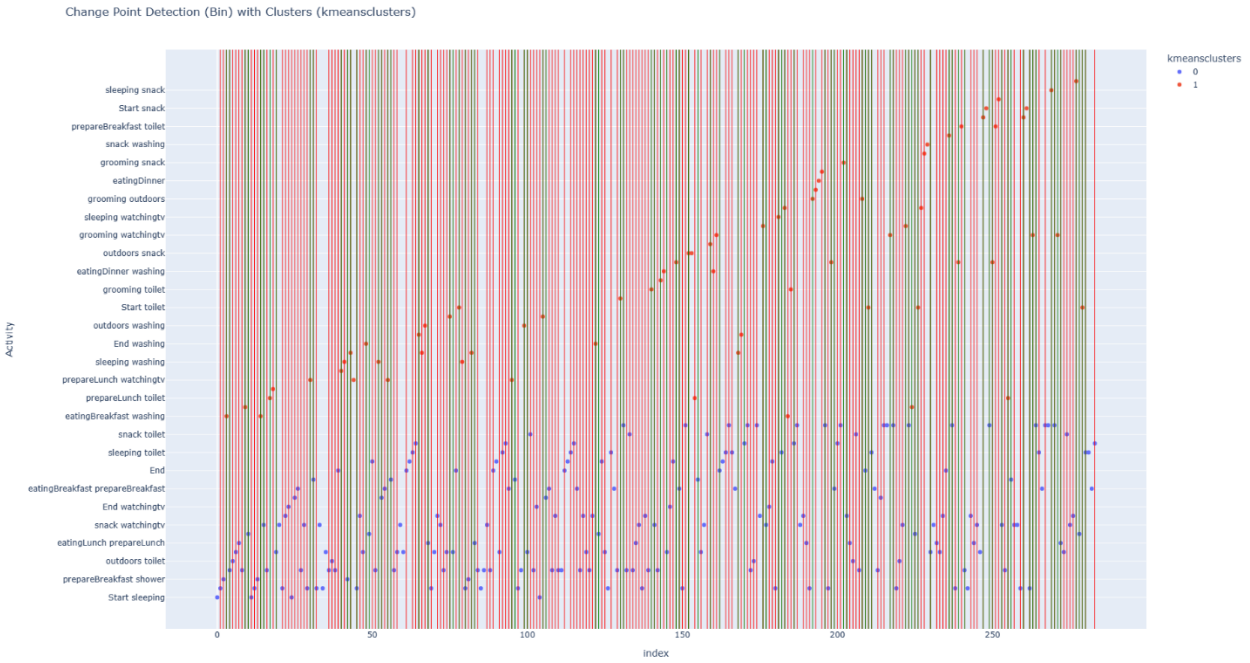


Figure 5.43 Change points detected with Binary Segmentation algorithm (red lines) and change points detected with Binary Segmentation and Kmeans with 2 clusters (green lines) as scatter plot colored based on KMeans clusters. The length of each n-sequence of activities is 2 and Keras Tokenizer encoding technique was used.

Figure 5.44 Change points detected with Binary Segmentation algorithm (red lines) and change points detected with Binary Segmentation and DBSCAN with 2 clusters (green lines) as scatter plot colored based

on DBSCAN clusters. The length of each n-sequence of activities is 2 and Keras Tokenizer encoding technique was used.

As illustrated in the figures, the integration of Change Point Detection techniques with Clustering techniques has the capability to reduce the identified change points to those that are truly significant.

5.3.3 Conclusions

The aim of the developed framework is to detect disorders in human behavior.

Encoding methods that were applied in human activity event logs play a significant role in the number and the position of change points. Penalty values differentiate between encoding techniques based on type (scalar or vector) and context.

It is essential to consolidate multiple activities into a single point to prevent inaccuracies in assessing changes in human behavior solely based on alterations in the order of activities within a specific time zone. The experiments indicate that an increase in the number of grouped activities is likely to result in a decrease in the number of identified change points. This occurs because the routine of the resident remains consistent within a time zone. However, with a rise in the number of activities within the group, the variations divided by samples also increase. Consequently, visualization proves valuable in determining the optimal number of grouped activities.

Choosing the most suitable Change Point detection algorithm is contingent on the chosen approach to understanding human behavior. For instance, if the goal is to pinpoint only substantial changes in human life, it is recommended to raise the penalty value. When investigating daily fluctuations in human behavior and identifying changes within specific time zones, it becomes essential to adjust the jump parameter. Setting the min size parameter becomes imperative if there is a need to specify the minimum distance between changes. In cases where the number of changes in human behavior to be detected is known, Dynamic Programming must be selected. The completion time of Pelt depends more on the number of change points rather than the chosen encoding method. Dynamic Programming requires significantly more time to detect the same change points compared to Pelt. However, the detection time experiences a considerable reduction when utilizing the jump and min size parameters. Dynamic Programming could be useful to find the one most significant change in human behavior. In comparison to Pelt, Binary Segmentation identified fewer change points with the respective penalty parameter values.

Combining Change Point Detection techniques with Clustering techniques can effectively narrow down the identified change points to those that hold genuine significance.

Chapter 6

6.1 Discussion and Conclusions

The implementation of the Process Mining approach with human activity data is viable but requires modifications due to the intricate and diverse nature of human behavior. The inherent complexity and variability in human actions necessitate adjustments in the process model, especially considering the unstructured nature of daily human routines. The Inductive Miner algorithm proves to be particularly effective in capturing behavior from event logs.

Notably, the process discovery on an event log often results in complex and sizable process models, which may pose challenges for users to easily comprehend. A holistic approach becomes imperative when tackling the challenges associated with approaching and modeling human behavior. It involves considering the entirety of the context, recognizing the dynamic nature of human actions, and finding comprehensive solutions. Integrating Process Mining techniques with Change Point Detection and Clustering techniques becomes a powerful strategy. These techniques play a pivotal role in identifying recurring patterns in human behavior, highlighting deviations, and pinpointing potential disorders or anomalies. This enriched approach contributes to a more nuanced and comprehensive understanding of the complexities inherent in human processes.

Based on the above, a framework has been created to identify anomalies or disorders in human behavior. The encoding methods employed in human activity event logs play a crucial role in determining the number and position of detected change points. The differentiation in penalty values is based on the type (scalar or vector) and context of encoding techniques. Aggregating multiple activities into a single point helps prevent inaccuracies in assessing changes in human behavior caused by variations in the order of activities within a specific time zone. The experiments reveal that an increase in the number of grouped activities tends to lead to a decrease in the number of identified change points, as the resident's routine remains consistent within a time zone.

The choice of the most suitable Change Point detection algorithm depends on the approach taken to understand human behavior. For example, if the aim is to pinpoint only significant changes in human life, it is advisable to increase the penalty value. When investigating daily fluctuations in human behavior and identifying changes within specific time zones, adjusting the jump parameter becomes essential. Setting the min size parameter is imperative when specifying the minimum distance between changes. In cases where the number of changes in human behavior to be detected is known, Dynamic Programming must be selected. The completion time of Pelt depends more on the number of change points than the chosen encoding method. Although

Dynamic Programming takes considerably more time to detect the same change points compared to Pelt, this detection time experiences a significant reduction when utilizing the jump and min size parameters. Dynamic Programming proves useful in finding the most significant change in human behavior. In contrast to Pelt, Binary Segmentation identifies fewer change points with the respective penalty parameter values.

Clustering, when integrated with Change Point Detection techniques, has the capacity to refine the identified change points, retaining only those that hold genuine significance.

6.2 Limitations

While this study has made significant gains in the field of human behavior analysis, there are also certain limitations.

Firstly, Human activities inherently exhibit considerable variability and noise, presenting a challenge in precisely identifying change points. The presence of fluctuations and inconsistencies in the data introduces difficulty in distinguishing meaningful transitions from inherent randomness. The impact of this noise on the accuracy of change point detection algorithms must be considered when interpreting the results.

Human behavior is deeply influenced by context, posing a challenge for change point detection to adapt to diverse contextual factors. Changes in the environment, individual preferences, or situational factors may influence the accuracy of identifying change points. The framework's ability to robustly handle varying contexts is an important consideration, and potential limitations in this adaptability should be acknowledged.

The wide range of individual variations in how activities are performed introduces a significant limitation in change point detection. Developing generalized models may struggle to effectively capture the nuances of diverse individual behaviors, necessitating the exploration of personalized models for accurate change point detection. The consideration of subject variability is crucial in understanding the limitations of the proposed methodology.

Lastly, acquiring accurately labeled data for training and evaluating change point detection models is a complex and intricate task. Human-labeled datasets, while valuable, may introduce subjective biases or errors that affect the reliability and generalizability of the models. The inherent difficulty in obtaining high-quality labeled data poses a substantial constraint in the development and assessment of change point detection algorithms. Acknowledging these challenges is essential for a comprehensive understanding of the potential limitations associated with the proposed methodology.

According to the limitations and challenges of this study, the following suggestions might be interesting directions for future work.

6.3 Future Work

Future research aims to both tackle existing challenges and explore ways to make improvements in the field of human behavior analysis. The main objective is to contribute to advancing the field by understanding current complexities and finding ways to enhance and evolve the approaches used.

Extending research to multi-residents systems in smart environments is another imperative topic. This is especially challenging in shared environments where overlapping activities and different users coexist. Understanding how applied techniques can adapt to and differentiate between activities performed by various users is a complex but necessary undertaking.

The integration of multi-variate data provides an opportunity to move beyond activity labels and adopt a holistic approach. Exploring the inclusion of attributes related to activities, such as environmental conditions, biometric data, or contextual information found in event logs, can lead to a more comprehensive understanding of human behavior.

Techniques such as deep learning methods, recurrent neural networks (RNNs), or attention mechanisms could be applied for improved feature extraction and representation learning from human activity event logs.

Another critical area is the development of enhanced prediction models for human activities. Forecasting upcoming activities and identifying change points is essential for comprehending shifts in behavior or routine changes, enabling proactive intervention.

Real-time adaptability is a topic that warrants further investigation, particularly in the context of on-line change point detection in human activity. This is especially relevant in dynamic environments where activity patterns may undergo swift transitions.

REFERENCES

[1] Smith, E. A. (2017). Three styles in the evolutionary analysis of human behavior. In Adaptation and human behavior (pp. 27-46). Routledge.

[2] Cooper, J. O., Heron, T. E., & Heward, W. L. (2020). Applied behavior analysis. Pearson UK.

[3] Arshad, M. H., Bilal, M., & Gani, A. (2022). Human activity recognition: Review, taxonomy and open challenges. Sensors, 22(17), 6463.

[4] Bouchabou, D., Nguyen, S. M., Lohr, C., LeDuc, B., & 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.

[5] Soni, V., Yadav, H., Semwal, V. B., Roy, B., Choubey, D. K., & Mallick, D. K. (2023, January). A novel smartphone-based human activity recognition using deep learning in health care. In Machine Learning, Image Processing, Network Security and Data Sciences: Select Proceedings of 3rd International Conference on MIND 2021 (pp. 493-503). Singapore: Springer Nature Singapore.

[6] Pajak, G., Krutz, P., Patalas-Maliszewska, J., Rehm, M., Pajak, I., & Dix, M. (2022). An approach to sport activities recognition based on an inertial sensor and deep learning. Sensors and Actuators A: Physical, 345, 113773.

[7] Babangida, L., Perumal, T., Mustapha, N., & Yaakob, R. (2022). Internet of things (IoT) based activity recognition strategies in smart homes: A review. IEEE Sensors Journal, 22(9), 8327-8336.

[8] Shenoy, A., & Thillaiarasu, N. (2022, March). A survey on different computer vision based human activity recognition for surveillance applications. In 2022 6th International Conference on Computing Methodologies and Communication (ICCMC) (pp. 1372-1376). IEEE.

[9]Kulsoom, F., Narejo, S., Mehmood, Z., Chaudhry, H. N., Butt, A., & Bashir, A. K. (2022). A review of machine learning-based human activity recognition for diverse applications. Neural Computing and Applications, 34(21), 18289-18324.

[10] Bhattacharya, D., Sharma, D., Kim, W., Ijaz, M. F., & Singh, P. K. (2022). Ensem-HAR: An ensemble deep learning model for smartphone sensor-based human activity recognition for measurement of elderly health monitoring. Biosensors, 12(6), 393.

[11] Mekruksavanich, S., & Jitpattanakul, A. (2022). Cnn-based deep learning network for human activity recognition during physical exercise from accelerometer and photoplethysmographic

sensors. In Computer Networks, Big Data and IoT: Proceedings of ICCBI 2021 (pp. 531-542). Singapore: Springer Nature Singapore.

[12] Khatun, M. A., Yousuf, M. A., Ahmed, S., Uddin, M. Z., Alyami, S. A., Al-Ashhab, S., ... & Moni, M. A. (2022). Deep CNN-LSTM with self-attention model for human activity recognition using wearable sensor. IEEE Journal of Translational Engineering in Health and Medicine, 10, 1-16.

[13] Zhang, Y., Doughty, H., Shao, L., & Snoek, C. G. (2022). Audio-adaptive activity recognition across video domains. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 13791-13800).

[14] Lv, S., Hong, H., Yang, L., Ding, J., & Song, R. (2022, July). Solving In-door Human Activity Recognition via RFID based on Unsupervised Domain Adaptation. In 2022 IEEE 4th International Conference on Power, Intelligent Computing and Systems (ICPICS) (pp. 388-392). IEEE.

[15] Nair, R., Ragab, M., Mujallid, O. A., Mohammad, K. A., Mansour, R. F., & Viju, G. K. (2022). Impact of wireless sensor data mining with hybrid deep learning for human activity recognition. Wireless Communications and Mobile Computing, 2022.

[16] Li, Y., Yang, G., Su, Z., Li, S., & Wang, Y. (2023). Human activity recognition based on multienvironment sensor data. Information Fusion, 91, 47-63.

[17] Sherafat, B., Rashidi, A., & Asgari, S. (2022). Sound-based multiple-equipment activity recognition using convolutional neural networks. Automation in Construction, 135, 104104.

[18] Khan, I. U., Afzal, S., & Lee, J. W. (2022). Human activity recognition via hybrid deep learning based model. Sensors, 22(1), 323.

[19] Hayat, A., Morgado-Dias, F., Bhuyan, B. P., & Tomar, R. (2022). Human activity recognition for elderly people using machine and deep learning approaches. Information, 13(6), 275.

[20] Miranda, L., Viterbo, J., & Bernardini, F. (2022). A survey on the use of machine learning methods in context-aware middlewares for human activity recognition. Artificial Intelligence Review, 1-32.

[21] Babangida, L., Perumal, T., Mustapha, N., & Yaakob, R. (2022). Internet of things (IoT) based activity recognition strategies in smart homes: A review. IEEE Sensors Journal, 22(9), 8327-8336.

[22] Yu, Y., Hao, Z., Li, G., Liu, Y., Yang, R., & Liu, H. (2023). Optimal search mapping among sensors in heterogeneous smart homes. Math. Biosci. Eng, 20, 1960-1980.

[23] Bijalwan, V., Semwal, V. B., & Gupta, V. (2022). Wearable sensor-based pattern mining for human activity recognition: Deep learning approach. Industrial Robot: the international journal of robotics research and application, 49(1), 21-33.

[24] A Survey of Human Activity Recognition in Smart Homes Based on IoT Sensors Algorithms: Taxonomies, Challenges, and Opportunities with Deep Learning

[25] van der Aalst, W. M. (2022). Process mining: a 360 degree overview. In Process Mining Handbook (pp. 3-34). Springer, Cham.

[26] Leotta, F., Mecella, M., & Serral, E. (2023, March). A Survey on the Application of Process Mining to Smart Spaces Data. In Process Mining Workshops: ICPM 2022 International Workshops, Bozen-Bolzano, Italy, October 23–28, 2022, Revised Selected Papers (Vol. 468, p. 57). Springer Nature.

[27] Theodoropoulou, G., Bousdekis, A., Miaoulis, G., & Voulodimos, A. (2020, November). Process mining for activities of daily living in smart homecare. In Proceedings of the 24th Pan-Hellenic Conference on Informatics (pp. 197-201).

[28] Aminikhanghahi, S., & Cook, D. J. (2017, March). Using change point detection to automate daily activity segmentation. In 2017 IEEE international conference on pervasive computing and communications workshops (PerCom workshops) (pp. 262-267). IEEE.

[29] Khan, N., McClean, S., Zhang, S., & Nugent, C. (2023). Performance evaluation of multivariate statistical techniques using edge-enabled optimisation for change detection in activity monitoring. Journal of Cloud Computing, 12(1), 91.

[30] Aalst, W. V. D. (2016). Process mining: data science in action. (No Title).

[31] Guzzo, A., Rullo, A., & Vocaturo, E. (2022). Process mining applications in the healthcare domain: A comprehensive review. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 12(2), e1442.

[32] Van der Aalst, W., Weijters, T., & Maruster, L. (2004). Workflow mining: Discovering process models from event logs. IEEE transactions on knowledge and data engineering, 16(9), 1128-1142.

[33] Weijters, A. J., van Der Aalst, W. M., & De Medeiros, A. A. (2006). Process mining with the HeuristicsMiner algorithm.

[34] Van der Aalst, Wil MP, AK Alves De Medeiros, and A. J. M. M. Weijters. "Genetic process mining." International Conference on Application and Theory of Petri Nets. Springer Berlin Heidelberg, 2005.

[35] Van Der Aalst, W. MP, et al. "Process mining: a two-step approach to balance between underfitting and overfitting." Software and Systems Modeling 9.1 (2010): 87-111.

[36] Günther, C. W., & Van Der Aalst, W. M. (2007, September). Fuzzy mining–adaptive process simplification based on multi-perspective metrics. In International conference on business process management (pp. 328-343). Berlin, Heidelberg: Springer Berlin Heidelberg.

[37] Leemans, Maikel, and Wil MP van der Aalst. "Discovery of frequent episodes in event logs." International Symposium on Data-Driven Process Discovery and Analysis. Springer International Publishing, 2014.

[38] Pulsanong, W., Porouhan, P., Tumswadi, S., & Premchaiswadi, W. (2017, November). Using inductive miner to find the most optimized path of workflow process. In 2017 15th International Conference on ICT and Knowledge Engineering (ICT&KE) (pp. 1-5). IEEE.

[39] Rozinat, A., & Van der Aalst, W. M. (2008). Conformance checking of processes based on monitoring real behavior. Information Systems, 33(1), 64-95.

[40] Van der Aalst, W., Adriansyah, A., & 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.

[41] Naderifar, V., Sahran, S., & Shukur, Z. (2019). A review on conformance checking technique for the evaluation of process mining algorithms. TEM Journal, 8(4), 1232.

[42] van der Aalst, W. M. (2022). Process mining: a 360 degree overview. In Process Mining Handbook (pp. 3-34). Springer, Cham.

[43] W. Van Der Aalst, A. Adriansyah, A. K. A. De Medeiros, F. Arcieri, T. Baier, T. Blickle, J. C. Bose, P. Van Den Brand, R. Brandtjen, J. Buijs, and A. Burattin, “Process mining manifesto,”in International Conference on Business Process Management, pp. 169-194, Springer, Berlin, Heidelberg, August 2011.

[44] Ceravolo, P., Junior, S. B., Damiani, E., & van der Aalst, W. (2023). Tailoring Machine Learning for Process Mining. arXiv preprint arXiv:2306.10341.

[45] Verbeek, H. M. W., Buijs, J. C. A. M., Van Dongen, B. F., & van der Aalst, W. M. (2010). Prom 6: The process mining toolkit. Proc. of BPM Demonstration Track, 615, 34-39.

[46] Günther, C. W., & Rozinat, A. (2012). Disco: Discover Your Processes. BPM (Demos), 940(1), 40-44.

[47] Berti, A., Li, C. Y., Schuster, D., & van Zelst, S. J. (2021). The process mining toolkit (pmtk): Enabling advanced process mining in an integrated fashion. Proceedings of the ICPM demo track.

[48] Blickle, T., & Hess, H. (2009). Automatic process discovery with ARIS process performance manager (ARIS PPM). Expert Paper, IDS Scheer.

[49] Badakhshan, P., Geyer-Klingeberg, J., El-Halaby, M., Lutzeyer, T., & Affonseca, G. V. L. (2020, September). Celonis Process Repository: A Bridge between Business Process Management and Process Mining. In BPM (PhD/Demos) (pp. 67-71).

[50] Drakoulogkonas, P., & Apostolou, D. (2021). On the selection of process mining tools. Electronics, 10(4), 451.

[51] Yu, L., Zhou, R., Chen, R., & Lai, K. K. (2022). Missing data preprocessing in credit classification: One-hot encoding or imputation?. Emerging Markets Finance and Trade, 58(2), 472-482.

[52] Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol. 1, pp. 4171–4186 (2019)

[53] Smart Aging System: Uncovering the Hidden Wellness Parameter for Well-Being Monitoring and Anomaly Detection

[54] Truong, C., Oudre, L., & Vayatis, N. (2020). Selective review of offline change point detection methods. Signal Processing, 167, 107299.

[55] Taylor, W. A. (2000). Change-point analysis: a powerful new tool for detecting changes.

[56] Jackson, B., Scargle, J. D., Barnes, D., Arabhi, S., Alt, A., Gioumousis, P., ... & Tsai, T. T. (2005). An algorithm for optimal partitioning of data on an interval. IEEE Signal Processing Letters, 12(2), 105-108.

[57] Killick, R., Fearnhead, P., & Eckley, I. A. (2012). Optimal detection of changepoints with a linear computational cost. Journal of the American Statistical Association, 107(500), 1590-1598.

[58] Fryzlewicz, P. (2014). Wild binary segmentation for multiple change-point detection.

[59] Saxena, A., Prasad, M., Gupta, A., Bharill, N., Patel, O. P., Tiwari, A., ... & Lin, C. T. (2017). A review of clustering techniques and developments. Neurocomputing, 267, 664-681.

[60] Song, M., Günther, C. W., & Van der Aalst, W. M. (2009). Trace clustering in process mining. In Business Process Management Workshops: BPM 2008 International Workshops, Milano, Italy, September 1-4, 2008. Revised Papers 6 (pp. 109-120). Springer Berlin Heidelberg.

[61] Sinaga, K. P., & Yang, M. S. (2020). Unsupervised K-means clustering algorithm. IEEE access, 8, 80716-80727.

[62] Pham, D. T., Dimov, S. S., & Nguyen, C. D. (2005). Selection of K in K-means clustering. Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, 219(1), 103-119.

[63] Schubert, E., Sander, J., Ester, M., Kriegel, H. P., & Xu, X. (2017). DBSCAN revisited, revisited: why and how you should (still) use DBSCAN. ACM Transactions on Database Systems (TODS), 42(3), 1-21.

[64] Hussain, Z., Sheng, M., & Zhang, W. E. (2019). Different approaches for human activity recognition: A survey. arXiv preprint arXiv:1906.05074.

[65] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, “Deep learning for sensor-based activity recognition: A survey,” arXiv preprint arXiv:1707.03502, 2017.

[66] M. Cornacchia, K. Ozcan, Y. Zheng, and S. Velipasalar, “A survey on activity detection and classification using wearable sensors,” IEEE Sensors Journal, vol. 17, no. 2, pp. 386–403, 2017.

[67] S. Herath, M. Harandi, and F. Porikli, “Going deeper into action recognition: A survey,” Image and vision computing, vol. 60, pp. 4–21, 2017.

68] Park, S., & Kang, Y. S. (2016). A study of process mining-based business process innovation. Procedia Computer Science, 91, 734-743.

[69]Van Der Aalst, W. M., Reijers, H. A., Weijters, A. J., van Dongen, B. F., De Medeiros, A. A., Song, M., & Verbeek, H. M. W. (2007). Business process mining: An industrial application. Information systems, 32(5), 713-732.

[70]Burattin, A. (2015). Process mining techniques in business environments. Lecture Notes in Business Information Processing, 207.

[71]Zisimou, A., Kalaitzoglou, I., Theodoropoulou, G., Bousdekis, A., & Miaoulis, G. (2021, July). Evaluation of Public Funding Processes by Mining Event Logs. In 2021 12th International Conference on Information, Intelligence, Systems & Applications (IISA) (pp. 1-7). IEEE.

[72] Bousdekis, A., Kerasiotis, A., Kotsias, S., Theodoropoulou, G., Miaoulis, G., & Ghazanfarpour, D. (2023). Modelling and Predictive Monitoring of Business Processes under Uncertainty with Reinforcement Learning. Sensors, 23(15), 6931.

[73] Cook, Jonathan E., and Alexander L. Wolf. "Automating process discovery through event-data analysis." Software Engineering, 1995. ICSE 1995. 17th International Conference on. IEEE, 1995.

[74] Fernández-Llatas, C., Benedi, J. M., García-Gómez, J. M., & Traver, V. (2013). Process mining for individualized behavior modeling using wireless tracking in nursing homes. Sensors, 13(11), 15434-15451.

[75] Kaouni, A., Theodoropoulou, G., Bousdekis, A., Voulodimos, A., & Miaoulis, G. (2021, October). Visual Analytics in Process Mining for Supporting Business Process Improvement. In NiDS (pp. 166-175).

[76] Ma'arif, M. R. (2017, September). Revealing daily human activity pattern using process mining approach. In 2017 4th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI) (pp. 1-5). IEEE.

[77] Guzzo, A., Rullo, A., & Vocaturo, E. (2022). Process mining applications in the healthcare domain: A comprehensive review. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 12(2), e1442.

[78]Pulkkinen, T.; Sallinen, M.; Son, J.; Park, J.H.; Lee, Y.H. Home network semantic modeling and reasoning—A case study. In Proceedings of the 15th International Conference on information fusion, Singapore, 9–12 July 2012; pp. 338–345.

[79] Lara, O.D.; Pérez, A.J.; Labrador, M.A.; Posada, J.D. Centinela: A human activity recognition system based on acceleration and vital sign data. Pervasive Mob. Comput. 2012, 8, 717–729.

[80] Suryadevara, N.K.; Mukhopadhyay, S.C.; Kelly, S.D.T.; Gill, S.P.S. WSN-based smart sensors and actuator for power management in intelligent buildings. IEEE/ASME Trans. Mechatron. 2015, 20, 564–571.

[81] Han, Y.; Han, M.; Lee, S.; Sarkar, A.M.; Lee, Y.K. A framework for supervising lifestyle diseases using long-term activity monitoring. Sensors 2012, 12, 5363–5379.

[82] Sitova, I., & Pecerska, J. (2020, October). Process Data Analysis Using Visual Analytics and Process Mining Techniques. In 2020 61st International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS) (pp. 1-6). IEEE.

[83] Nguyen, P., Turkay, C., Andrienko, G., Andrienko, N., & Thonnard, O. (2017). A visual analytics approach for user behavior understanding through action sequence analysis.

[74][84] Kubrak, K., Milani, F., & Nolte, A. (2023). A visual approach to support process analysts in working with process improvement opportunities. Business Process Management Journal, 29(8), 101-132.

[85] Bouchabou, D., Nguyen, S. M., Lohr, C., Leduc, B., & Kanellos, I. (2021). Fully convolutional network bootstrapped by word encoding and embedding for activity recognition in smart homes. In Deep Learning for Human Activity Recognition: Second International Workshop, DL-HAR 2020, Held in Conjunction with IJCAI-PRICAI 2020, Kyoto, Japan, January 8, 2021, Proceedings 2 (pp. 111-125). Springer Singapore.

[86] Yuan, S., Bhatia, P., Celikkaya, B., Liu, H., & Choi, K. (2021, January). Towards User Friendly Medication Mapping Using Entity-Boosted Two-Tower Neural Network. In International Workshop on Deep Learning for Human Activity Recognition (pp. 126-138). Singapore: Springer Singapore.

[87] Haynes, K., Eckley, I. A., & Fearnhead, P. (2014). Efficient penalty search for multiple changepoint problems. arXiv preprint arXiv:1412.3617.

[88]Thakur, D., & Biswas, S. (2022). Online change point detection in application with transition-aware activity recognition. IEEE Transactions on Human-Machine Systems, 52(6), 1176-1185.

[89] Preis, A., & Schwaar, S. (2023). Change point detection in text data. Behaviormetrika, 1-20.

[90] Li, J., Fearnhead, P., Fryzlewicz, P., & Wang, T. (2022). Automatic change-point detection in time series via deep learning. arXiv preprint arXiv:2211.03860.

[91] Theodosiadou, O., Pantelidou, K., Bastas, N., Chatzakou, D., Tsikrika, T., Vrochidis, S., & Kompatsiaris, I. (2021). Change point detection in terrorism-related online content using deep learning derived indicators. Information, 12(7), 274.

[92] He, H., Tan, Y., & Zhang, W. (2018). A wavelet tensor fuzzy clustering scheme for multi-sensor human activity recognition. Engineering Applications of Artificial Intelligence, 70, 109-122.

[93] Ma, H., Zhang, Z., Li, W., & Lu, S. (2021). Unsupervised human activity representation learning with multi-task deep clustering. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, 5(1), 1-25.

[94] T. G. Erdogan, and A. Tarhan, "Systematic Mapping of Process Mining Studies in Healthcare," IEEE Access, vol. 6, pp. 24543-24567, 2018.

[95] Sztyler, T., & Carmona, J. (2015). Activities of daily living of several individuals. University of Mannheim, Germany. Dataset.

[96] W. van der Aalst, “Extracting Event Data from Databases to Unleash Process Mining,”in: vom Brocke J., Schmiedel T. (eds) BPM - Driving Innovation in a Digital World. Management for Professionals, Springer, Cham, 2015.

[97]A. Berti, S. J. van Zelst, and W. van der Aalst, “Process mining for python (PM4PY): bridging the gap between process-and data science, arXiv preprint arXiv:1905.06169., 2019

[98] Van Kasteren, T., Noulas, A., Englebienne, G., & Kröse, B. (2008, September). Accurate activity recognition in a home setting. In Proceedings of the 10th international conference on Ubiquitous computing (pp. 1-9).

[99] Subject3. (n.d.). University Mannheim. https://www.uni-mannheim.de/dws/research/projects/activity-recognition/dataset/dataset-daily-log/subject3/

[100] Ordónez, F. J., De Toledo, P., & Sanchis, A. (2013). Activity recognition using hybrid generative/discriminative models on home environments using binary sensors. Sensors, 13(5), 5460-5477.