LLM-based event abstraction and integration for IoT-sourced logs

Mohsen Shirali[1,2], Mohammadreza Fani Sani[3], Zahra Ahmadi[1], and Estefanía Serral[1]

[1]Research Centre for Information Systems Engineering (LIRIS), KU Leuven, Brussels 1000, Belgium
(zahra.ahmadi,estefania.serralasensio}@kuleuven.be)
[2]ICTeam, UCLouvain, Louvain-la-Neuve, Belgium mohsen.shirali@uclouvain.be
[3]Microsoft Development Center Copenhagen, Copenhagen, Denmark
(*mfanisani@microsoft.com*

Corresponding author: Mohsen Shirali (mohsen.shirali@uclouvain.be)

Abstract

The continuous flow of data collected by Internet of Things (IoT) devices, has revolutionised our ability to understand and interact with the world across various applications. However, this data must be prepared and transformed into event data before analysis can begin. In this paper, we shed light on the potential of leveraging Large Language Models (LLMs) in event abstraction and integration. Our approach aims to create event records from raw sensor readings and merge the logs from multiple IoT sources into a single event log suitable for further Process Mining applications. We demonstrate the capabilities of LLMs in event abstraction considering a case study for IoT application in elderly care and longitudinal health monitoring. The results, showing on average an accuracy of 90% in detecting high-level activities . These results highlight LLMs' promising potential in addressing event abstraction and integration challenges, effectively bridging the existing gap.

Keywords. Internet of Things Large Language Models Event Abstraction Log Integration Multi-modality.

1 Introduction

The IoT technology has the potential to create a new "cyber-physical" world, where "things" can directly operate, act, and influence the physical world [14]. Smart devices have seamlessly integrated into everyday life, and low-cost sensors are embedded in various applications, from smart homes to smart cities and industrial IoT in smart factories. To extract meaningful insights from raw IoT data, advanced data mining techniques are required. However, raw sensor-level data is unstructured and non-informative, making it unsuitable for data mining. Raw data must be transformed into event logs to make the information discoverable and comprehensible for analysis [25]. Without proper pre-processing, the analysis foundation is compromised, creating an abstraction

gap [22, 5].

Looking in particular into Process Mining (PM), as one of the data analysis disciplines, it requires event logs, where each event represents a single activity at a particular time [12]. However, in complex settings, systems often lack process-centric data, rendering the data unsuitable for immediate analysis. Likewise, in smart homes, events are often logged at the sensor trigger level, which is too granular for meaningful pattern mining, complicating the application of PM techniques [5]. Additionally, the required data often resides in various databases and/or generated by multiple information systems or sources. Data from multiple sources must be integrated to provide a comprehensive view, facilitating the understanding of system behaviour and enhancing analysis capabilities from various angles.

Preparing event logs is typically manual, requiring domain knowledge to group correlated tasks into meaningful activities. This manual effort can be inconsistent and error-prone, especially for complex processes involving multiple data sources [9]. Moreover, integrating heterogeneous logs introduces challenges like format inconsistencies, timestamp misalignment, and data quality issues [13]. Besides, event data may have mixed granularity and contextual information, complicating the application of data analysis techniques [29].

In this way, pre-processing techniques are vital to build meaningful event logs before any actual analysis, like PM, can begin [5] and resources have to be allocated for this purpose. A study in [28] revealed that the laborious effort for data preparation or the necessity for proper expertise can terminate PM projects and cause them to be called off. If event logs are not generated properly, the potential benefits of using recorded logs for data analysis are significantly limited.

Log abstraction and integration are two essential data preparation or pre-processing tasks often used in data-driven analytics, to enhance result interpretability [32]. Furthermore, log abstraction involves summarising or grouping low-level data elements into higher-level event-based representations, aligning with the executed activities in a process. This simplifies the analysis and assigns meaning to sets of raw data records, allowing them to be interpreted as a single event [21, 5]. Log integration also merges combining data from multiple sources/systems to create a unified view for analysis [17].

Existing event abstraction techniques are either unsupervised or supervised [15, 32]. Unsupervised methods rely on control-flow similarities between low-level event types without requiring input on targeted high-level activities [21]. Conversely, supervised methods require extensive input on high-level activities, demanding significant manual effort and domain knowledge [21]. These two extremes necessitate a balance to support users in abstraction, addressing the challenges of manual effort, domain knowledge, and consistency.

Motivated by these challenges, we propose using Large Language Models (LLMs) to automate raw data pre-processing and event log generation, minimising the need for user background knowledge and effort. LLMs, trained on extensive data, are ideal for this task, capable of processing textual information and performing natural language understanding and generation tasks [2]. Our approach aims to utilise LLMs to analyse IoT sensor logs, generate meaningful event records, and merge logs from multiple sources, facilitating Process Mining applications.

We explore how an LLM, given a prompt with event label samples and minimal user input, can abstract sensor readings into event logs. In the prompt, we outline the task, clarify the role of the LLM, describe the inputs it will process, and specify the expected outputs. Hence, the contributions of our work are as follows; first, We use LLMs to automate the detection and labelling

of low-level sensor data to create abstracted events. Second, we develop a method for abstracting and generating events based on data streams, making the proposed approach suitable for online applications. Third, information from multiple sources will be merged into a unified event log to enhance analytics by providing a comprehensive view of the processes. Our work aims to streamline the process of event log generation, making it more efficient and accessible, and ultimately improving the utility of IoT-sourced sensor data in data-driven analytical applications.

The rest of this article is organised as follows: Section 2 reviews the existing event abstraction and integration methods and describes the multi-source IoT dataset which is used for evaluation. Section 3 describes the proposed LLM-based event log abstraction and integration approach. Then, in Section 4, the results and performance of the proposed approach are evaluated and a discussion on the usefulness of utilising LLMs besides IoT systems is provided in Section 5. Finally, Section 6 concludes the article's findings.

2 Background knowledge

2.1 Existing works for event abstraction and integration

The journey from raw data to event logs suitable for PM has been extensively studied in works like [5] and [32]. For example, van Zelst *et al.* [32] and Fahland *et al.* [7] have used techniques like log abstraction to group events into higher-level activities for simplified visualisation and analysis Additionally, a hierarchical framework for event abstraction based on notion of activity instances was proposed in [16]. Similarly, Senderovich *et al.* [25] suggested a knowledge-driven approach to transform raw sensor data into standardised event logs. The integration of Complex Event Processing (CEP) and BPM, as highlighted by Soffer *et al.* has gained attention for creating more efficient systems, especially for IoT applications. Mangler *et al.* [19] also address the challenges of connecting process data with IoT data and propose a semi-automatic framework for transforming low-level IoT sensor data into higher-level process events. Furthermore, Di Federico and Burattin [4] introduce CvAMoS, a method to identify recurring sequences of activities and consider their context of execution for event abstraction. Additionally, [24] introduces the concept of activity signatures for deriving knowledge about activity executions and training supervised learning models to detect higher-level process activity executions for larger datasets.

Moreover, it is particularly useful (or even crucial in many use cases) to build a ground truth based on domain knowledge and user expertise in the early stages of IoT data analysis, which can later facilitate automated activity detection [24]. For instance, in some IoT logs, human expertise can help in interpreting sensor-level data and mapping them to activity-level events using samples, without predefined descriptions.This approach, used in activity recognition, maps sensor readings such as opening the door of a refrigerator to activities like "preparing meal" [29]. In addition, system designers can translate sensor readings into events using predefined rules. For example, [27] and [18] used senor locations to create event logs for discovering behaviour models and mobility patterns by PM techniques. Also, studies like [8, 3] consider sensor types and attached objects to create event logs for performed activities of daily living (ADLs).

To overcome the required domain knowledge, [21] applies a method to learn event representations and automatically suggest related event groups for abstraction. This approach allows users to select meaningful event groups without initial input. Additionally, an unsupervised method in [20] continuously transforms user interactions into event streams for downstream analysis. Another unsupervised learning technique is also proposed in [11] to discretize sensor data into identifiable activities, and refine their sequences by deducing sub-models to identify concurrency and inter-

leaving within the data in smart home scenario. Furthermore, recent studies have explored using LLMs for event abstraction like [9] that grouped tasks into high-level activities and assigned labels to them based on their similarity.

In terms of log integration methods to address data heterogeneity, in [6], logs from different organizational departments were combined for end-to-end process analysis. Also, in [23], a time-based heuristics miner is used to discover high-level and low-level process models in parallel from multi-source logs, integrating them with PetriNet refinement.

2.2 Multi-source IoT Dataset

Our study aimed to assess the effectiveness and accuracy of using LLMs on IoT-sourced datasets to create event logs suitable for Process Mining. We applied our proposed LLM to sensor logs from an IoT dataset collected in an Ambient Assisted Living scenario. This dataset contains 146 days of data, capturing the daily activities of a 60-year-old woman living independently in an apartment [8]. The data was collected using ambient sensors installed in the house, a wristband, and a smartphone.

The ambient sensors included PIR sensors placed on furniture and appliances, a power usage sensor indicating TV usage, contact sensors for detecting door openings and closings, and a gas detection sensor for identifying cooking activity. These sensors were positioned in six different areas of the house, as shown in Figure 1 and recorded event data with timestamps, sensor names, and sensor values (e.g., on/off states). The participant also wore a Xiaomi Mi Band-3 wristband to collect sleep-related information such as sleep time, duration, and quality. Additionally, smartphone usage data, including timestamps and the names of used applications, were collected using an application.

The ambient sensors, such as the TV sensor and kitchen appliance sensors, are triggered when the participant performs specific activities. As a result, their measurements are at a low level of abstraction representing the presence of the subject near the appliances, the open or closed state of doors, and the use of the stove. By knowing the timestamp of their triggered states, we can discover the corresponding activities to express the exact start and end times and the duration of the conducted activity. These activities are mainly known as instrumented Activity Daily Livings (iADL), the activities performed using specific instruments and are investigated in several behaviour modelling and analysis studies and projects, like [3, 26].

With the entire sequence of captured records for a specific duration, we can infer events related to the person's presence in different areas of the house, the performed activities, and the sequence of their movements. For this purpose, an event abstraction step is needed to elevate the abstraction level of sensor raw readings into recognisable activity events in an event log. The events related to smartphone usage should also be converted and inserted into the log, while the names of applica-

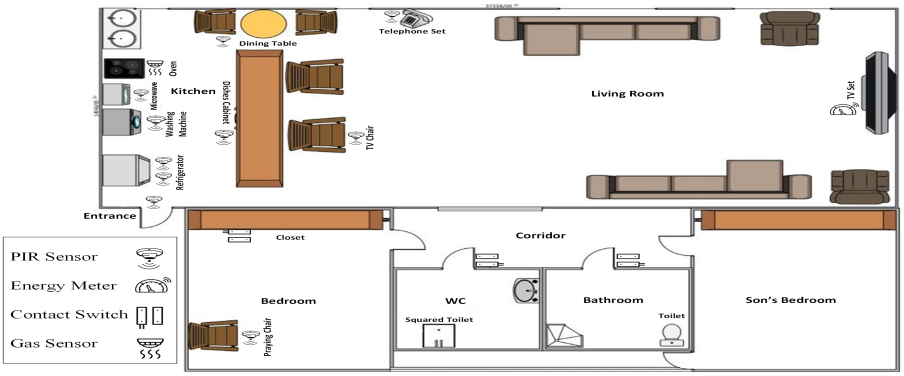

Figure 1: The house floor plan, location and types of ambient sensors

Date	TV	Telephone	TV_Chair	Dining_Chair	Bathroom	WC	Closet	Entrance	PrayingChair	Refrigrator	Fridge	Washing Machine	Microwave	Stove	Cupboard	Kitchen	Smartphone	Sleep	Wristband	SensorStates	Ambient Label	Smartphone Label	Smartphone AppType	Wristband Label	Wristband Type
1/9/2020 14:14:52	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10000000000000000					
1/9/2020 14:15:32	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10010000000000000	meal begin				
1/9/2020 14:34:01	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10000000000000000	meal end				
1/9/2020 14:37:35	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10001000000000000	toilet begin				
1/9/2020 14:39:02	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10000000000000000	toilet end				
1/9/2020 14:39:43	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	10000000000000100					
1/9/2020 14:39:46	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10000000000000000					
01/09/20 14:42:23	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	10000000000000010		Start Using Smartphone	Call		
01/09/20 14:43:04	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10000000000000000		End Using Smartphone			
01/09/20 14:43:36	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	10000000000000010		Start Using Smartphone	Messaging		
01/09/20 14:43:37	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10000000000000000		End Using Smartphone			
01/09/20 14:43:44	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	10000000000000010		Start Using Smartphone	Call		
01/09/20 14:43:44	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10000000000000000		End Using Smartphone			
1/9/2020 14:51:42	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	10000000000000100					
1/9/2020 14:51:45	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10000000000000000					
1/9/2020 14:51:54	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	10000000010000000					
1/9/2020 14:52:03	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10000000000000000					
1/9/2020 15:03:22	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	10000000010000000					
1/9/2020 15:03:52	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10000000000000000					
1/9/2020 15:06:03	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10100000000000000	watching tv begin				
1/9/2020 15:06:48	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	10100000010000000					
1/9/2020 15:06:51	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10100000000000000					
01/09/20 15:07:00	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	10100000000000001				Start Sleeping	W-Day Sleep
1/9/2020 15:07:38	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	10000000000000001	watching tv end			Sleeping	W-Day Sleep
1/9/2020 15:07:56	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	10000010000000001	relax begin			Sleeping	W-Day Sleep
1/9/2020 15:08:05	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	10000000000000001				Sleeping	W-Day Sleep

(a) We have 17 sensors and three categories of labels.

Source	Location/Events	Date	StartTime	EndTime	Duration
Location	Kitchen	1/9/2020	1/9/2020 14:11:00	1/9/2020 14:36:59	0:25:59
Ambient	eating and watching TV	1/9/2020	1/9/2020 14:15:32	1/9/2020 14:34:01	0:18:29
Location	LivingRoom	1/9/2020	1/9/2020 14:37:00	1/9/2020 14:37:01	0:00:01
Location	Bathroom	1/9/2020	1/9/2020 14:37:04	1/9/2020 14:38:59	0:01:55
Location	LivingRoom	1/9/2020	1/9/2020 14:39:02	1/9/2020 14:39:03	0:00:01
Location	Kitchen	1/9/2020	1/9/2020 14:39:04	1/9/2020 15:05:59	0:26:55
Smartphone	Call	1/9/2020	1/9/2020 14:42:23	1/9/2020 14:43:04	0:00:41
Smartphone	Messaging	1/9/2020	1/9/2020 14:43:36	1/9/2020 14:43:37	0:00:01
Smartphone	Call	1/9/2020	1/9/2020 14:43:44	1/9/2020 14:43:44	0:00:00
Location	LivingRoom	1/9/2020	1/9/2020 15:06:00	1/9/2020 15:06:01	0:00:01
Location	Kitchen	1/9/2020	1/9/2020 15:06:02	1/9/2020 15:06:59	0:00:57
Location	LivingRoom	1/9/2020	1/9/2020 15:07:00	1/9/2020 15:07:01	0:00:01
Wristband	day sleeping	1/9/2020	1/9/2020 15:07:00	1/9/2020 16:30:00	1:23:00

(b) Activities and presence in locations can be concurrent and be happened on different days.

Figure 2: A view of (a) sensor logs and (b) resulted event log (the ground truth version) for the experimented IoT dataset.

tions can be replaced with their type or with a general label such as 'Using Smartphone' to preserve the participants' privacy. Furthermore, the sleep-related information collected by the wristband will complete the preferred event log.

In the current case study, the event log includes location events, such as *Bedroom*, *Bathroom*, *Kitchen*, and the activity events like *sleeping*, *meal preparation*, *praying*, watching tv. A snapshot of sensor logs and the required event log, including the information of all modalities (the ground truth version), is depicted in Figure 2. We fed these sensor logs obtained from devices' reports in our multi-modal IoT dataset into the proposed LLM model, asking it to create a PM-friendly event log. The resulting event log was then compared with a ground truth event log that was generated through a pre-processing step by applying multiple event detection rules determined by domain experts and then inspecting manually to correct any falsely detected events, as described and used in [26].

3 Event log abstraction and integration by LLMs

As previously stated, PM techniques rely on having an event log as an input, and without it, gaining insights is not possible. While many scientific works assume the availability of the event log, the procedure of pre-processing raw data, detecting events, abstracting them into activities, and generating an event log can be time-consuming in real scenarios. Abstracting events from

sensor data, particularly when dealing with streams of input data and potential concurrent events and activities recorded by a combination of sensors, can be quite challenging [19]. In this section, we explain how we can use Generative AI and LLMs to (1) abstract the events from sensor data to activities, and (2) generate an event log from different resources.

3.1 Abstraction of events into activities

We propose to use LLMs to detect activities from sensor data. In this regard, we give some basic explanations about the sensors, possible activity labels, and a few examples to LLM to provide us with a label for the detected change in the status of sensors. In other words, we call an LLM whenever we have a change in sensor values. The changes in the binary values of the sensors have the highest significance regarding process-level events, as it is stated in the [19]. Thus, each sensor log entry should denote a change in the status of sensors. Therefore, we can consider this task as a classification task in that LLM will be the classifier and activity labels are the classes. The number of classes is the number of possible desired activities plus one as we may encounter sensor changes that do not correspond to any specific activity. However, there are instances where sensor changes occur without any corresponding activities due to noises, irrelevant sensors, etc. In such cases, a blank class is defined to address these situations.

For this task, we utilised GPT-4 as the model and applied few-shot learning [31] and chain of thought [30] techniques. In the prompt template, we incorporated placeholders for the label of activities, the sensor explanation, and the output format[1].

In general, the prompt can be modified if we have more information based on the rules that we have in business. For instance, here the output of all sensors in our dataset are binary values (i.e., 0 and 1), and it is important to note that we take this advantage to save tokens and reduce the cost by just passing the aggregated sensor information in a binary format to LLM (the number of consumed tokens in a generated a prompt is one of the limitations in using LLMs). Hence, the LLM will need to recognise which value corresponds to which sensor based on the order of binary digits in the sensor state. Some examples of SensorStates are given in Figure 2a. We provide the SensorStates for the current and previous state in our prompt to simplify the task for the LLM. The LLM will first identify what has changed in these two values and then provide the activity name.

3.2 Integration and event log generation

We can have several information sources for one event log. Note that for each activity we have two events; the first one corresponds to the start of the activity and the second one indicates that the activity has ended. In real world, we may have more events for each activity considering the complete life cycle of activities [1]. We propose to use LLM to gather information from different resources and combine them into one event log. However, we have some limitations:

- We cannot give the whole information to LLM as we have a limitation on the number of tokens,
- We do not have a caseID[2] as all the information related in our dataset is related to one process instance.

[1]The sources and examples for our implemented LLM with more detailed information are publicly available at https://github.com/mfanisani/LLM4IoT

[2]caseID is one of the main columns in any event log that represents the process instance.

To overcome the above limitations, we considered date as caseID. This will allow us to analyse the process of activities that have been done each day. Therefore, our data will include the date as the case ID, the activity, start time, and end time. However, it is possible that an activity start on one day and end on the next day, e.g. sleeping from 2022-11-23 22:00 to 2022-11-24 07:00. In this case, we will provide LLM with the information for activities that span two days and ask it to trim the event at midnight, and adjust the event log for each day accordingly. To handle this, we introduce another column with binary values indicating whether the activity is completed on the same date or not. If the activity crosses over to the next day, we will not include the end time for that next day. While we assumed activities span at most two consecutive days to mitigate the constraint on the number of tokens per LLM call, it's important to recognise that real-world scenarios may not adhere to this assumption and may have longer execution times.

Our prompt template includes possible activities, two-shot examples, the information for the date of interest and the following day, and the format of the desired output. In this way, we can increase the performance of the event log generation task, as we will call LLM once per day. Note that the output does not contain the header of event log columns. We use a basic script to collect the output information generated by LLM for different dates and compile it into a single event log.

The proposed framework for event log abstraction and integration can be used on different sensors or data logs with different prompts and on various LLMs. The samples for the appropriate labels are given to the model and it returns the expected output for the stream of data inputs immediately. Therefore, the framework is also able to deal with streams of data sources and the output of this layer is sufficient for the PM tools without any further involvement of the users and interventions.

4 Evaluation

It is important to assess the effectiveness of LLMs in event abstraction by analysing the results and evaluating the model's accuracy in generating event records and assigning the appropriate labels. This section presents the results of proposed LLM-based event abstraction on the real-world IoT-sourced dataset for our case study. We have conducted the evaluation in two steps. First, the abstracted events for each data source are examined and compared with the intended labels to measure the accuracy of the LLM model in identifying and abstracting the raw sensor records for each data source. This comparison will highlight the LLM model's potential in dealing with different IoT sources. In the next step, the outcome of merging three logs from ambient, smartphone and wristband into the final integrated PM-friendly event log is cross-checked with the ground truth event log.

4.1 Event abstraction and integration results

The accuracy results of the LLM in the abstraction of events foreach of the three modalities in our dataset are provided in Table 1. For ambient sensors, we just consider activities with an occurrence of at least 50 times. The 21 labels in this source correspond to 10 activities with a start and end plus a label that corresponds to no activity. For smartphones and wristbands, we have respectively 13 and 3 labels indicating the type of used application or day/night sleep. For each activity, we have start, continue and end labels. Similar to ambient sensors, we have a label for no activity.

The accuracy for ambient sensor data is much lower than the other datasets as the number of related sensors and the number of labels are higher. For all three datasets, the lowest accuracy is for *None* label that corresponds to no activity. In several cases, we predicted an activity (e.g.,

Watching TV start/end) instead of *None*. We randomly checked some of the wrongly assigned labels and in almost all of them, the reasoning part produced by LLM was wrong.

4.2 Evaluation of LLM-generated event log

The evaluation of the LLM-based event log in terms of the accuracy of the recognised events is crucial to assess the success of LLMs in the abstraction and integration task. This evaluation involves comparing two event logs that contain various events with their respective start and end times. A snapshot of the event log generated using LLM is presented in Table 2. Therefore, the matching of events labels, events orders, and the duration and time precedence of events are the important factors to be considered for evaluation. We have utilised the number of correctly generated events, the ratio of perfectly aligned date and Edit Distance Alignment (EDA) [10] metrics to measure the success rate of our proposed LLM-based approach.

In our experiment, we provided the GPT4-0613 model with sensor states and their original labels, tasking it to generate an event log. Notably, we used the ground truth labels, not those predicted by LLM. The ground truth dataset contains 17,165 events, while the LLM-generated event log has 15,420 events. Among these generated events, 12,741 were correctly detected, with 11,365 having matching start and end activities. In addition, LLM is able to generate all the unique activities that are defined for it.

In our analysis, we observed that self-loops—consecutive events with the same label—play a significant role. After removing these self-loops, we found that the generated event log contains 11,248 events, while the ground truth dataset has 11,431 events. Interestingly, LLM sometimes aggregates self-loops, particularly when the user interacts with her smartphone, where the interactivity times between self-loops are negligible.

To show how the generated event log is aligned with the ground truth event log, we applied EDA between similar dates in both event logs using the technique proposed in [10]. This metric measures how similar the sequences of events are, where a value closer to *one* indicates higher similarity. The results are presented in Table 3 and indicate that the event log generated by LLM captures similar behaviour compared to the ground truth event log but it aggregated the self-loops.

5 Discussion

The results of this case study indicate that utilising LLMs can significantly alleviate the challenges associated with event log abstraction and integration for IoT-sourced logs. LLMs can bridge the gap between data collection and the demand for event logs at appropriate abstraction levels for data analysis techniques, including Process Mining. Although, traditional methods for event abstraction and integration are often manual, error-prone, and time-consuming, but, LLMs can automate and speed up these tasks while reducing human errors. They can swiftly pre-process large volumes of raw sensor data, identify and abstract events with high accuracy, and uniformly label them, thus freeing up human resources for more strategic tasks. However, it's important to acknowledge that the output of our proposed solution may not always be highly reliable. Therefore, we recommend incorporating human oversight in applications that require a high level of reliability.

Table 1: The accuracy of our activity detection method for different sensors

	Ambient sensors	**Smartphone**	**wristband**
Number of labels	21	13	3
Accuracy	0.81	0.97	0.92

Table 2: The snapshot of the LLM-generated event log from three sources (the values of Date column are caseIDs).

Date	Type	Activity	Start Time	End Time	Next date
1/8/2020	Ambient	sleeping	23:04:33	08:28:59	True
1/8/2020	Ambient	toilet	23:05:39	23:07:21	False
1/8/2020	Smartphone	messaging	23:17:28	23:17:49	False
1/8/2020	Smartphone	messaging	23:17:52	23:18:31	False
1/8/2020	Smartphone	messaging	23:18:32	23:20:28	False
1/8/2020	Smartphone	call	23:20:32	23:20:45	False
...	...	...	...	...	...

Table 3: Behavioral comparison of event logs with/without considering self loops.

Edit Distance Alignment		**Perfectly Aligned Dates(%)**	
All events	**Without Self-loops**	**All events**	**Without Self-loops**
0.83	0.94	0	33

Key advantages of using LLMs in event log generation and integration for IoT systems include:

- **Automation and efficiency.** LLMs can automate the pre-processing of raw data and make the process more efficient by reducing manual effort and errors.

- **Reduced need for domain knowledge.** Expertise is crucial in supervising the abstraction process and interpreting IoT logs requires substantial domain knowledge to map low-level sensor readings to high-level activities. LLMs, trained on diverse datasets, can understand and abstract sensor data into meaningful events without deep domain-specific knowledge. This allows more users with limited background knowledge to generate valuable insights and make IoT data analytics more feasible.

- **Handling multi-modality.** IoT systems often collect data from multiple sources, resulting in mixed granularity and inconsistencies in data formats and characteristics. LLM-based models can understand and integrate heterogeneous data inputs, by handling this complexity. These models can align data from various sensors, ensuring a cohesive and consistent event log, particularly for scenarios involving time misalignment and varied data structures.

- **Real-time processing.** LLMs are capable of handling streams of data inputs, making them suitable for online applications. Initially, in our study, we triggered LLM calls whenever there was a change in sensor values. However, this approach can be computationally expensive and slow in certain applications. To address this limitation, we propose using batching—calling the LLM fewer times by grouping input data— similar to our approach for event log generation. This means that LLMs can process incoming data in real-time, handling sensor records one by one without relying on maintaining extensive change histories, and continuously updating the event logs with new information. This ensures that the event logs are always current, facilitating timely analysis and decision-making.

- **Performance optimisation with prompt engineering and user feedback.** Designing prompts that clearly describe the task, input data, and expected output, can significantly enhance the quality of the LLM's output. Moreover, users' and domain experts involvement in the process remains crucial for fine-tuning the LLM's outputs and improving accuracy over time. A continuous feedback loop between LLMs and users helps in refining the model, adapting to evolving data patterns, and ensuring that the system meets the specific needs of the application. This collaboration between humans and intelligent systems ensures that the benefits of LLMs are fully realised while maintaining high standards of accuracy and reliability.

In the end, LLMs offer a powerful solution to the challenges of event log abstraction and integration in IoT systems. By automating the manual, repetitive, and error-prone data preparation tasks, LLMs streamline the process, reduce the need for extensive domain knowledge, and handle the complexities of multi-modality and mixed granularity in data. Additionally, with proper prompt engineering and user feedback mechanisms, LLMs can provide accurate and up-to-date event logs, enhancing the overall efficiency and effectiveness of IoT data analytics. Hence, integrating LLMs with IoT systems can revolutionise the way we process and analyse sensor data, unlocking new potential for real-time insights and decision-making.

6 Conclusion and future works

The rapid advancement of IoT technologies promises valuable data-driven insights and enhanced operational efficiency. However, a gap exists between the raw IoT data and the processed data needed for analysis, hindering the full potential of IoT technologies. In this study, LLMs are utilised to aid users in pre-processing raw data, converting it into event logs containing all the essential fields for event data analytics techniques, like Process Mining. The results underscore LLMs' potential in addressing event log abstraction and integration for IoT-generated logs. Despite these promising outcomes, the use of LLMs in this field is in its early stages, with room for improvement. Future research could focus on applying LLMs to text-based raw data, improving prompt engineering techniques, and fine-tuning LLMs to further enhance their performance and usefulness in data preparation.

Acknowledgement.

Mohsen Shirali, is now affiliated at UCLouvain in Belgium under grant Win4Collective number 2310088. The work of Zahra Ahmadi and Estefanía Serral was supported by the Flemish Fund for Scientific Research (FWO) with grant number G0B6922N.

References

[1] Bernardi, M.L., Cimitile, M., Di Francescomarino, C., Maggi, F.M.: Do activity lifecycles affect the validity of a business rule in a business process? Information Systems **62**, 42–59 (2016)

[2] Berti, A., Schuster, D., van der Aalst, W.M.: Abstractions, scenarios, and prompt definitions for process mining with LLMs: A case study. In: International Conference on Business Process Management. pp. 427–439. Springer (2023)

[3] Cook, D.J., Crandall, A.S., Thomas, B.L., Krishnan, N.C.: Casas: A smart home in a box. Computer **46**(7), 62–69 (2013)

[4] Di Federico, G., Burattin, A.: CvAMoS—event abstraction using contextual information. Future Internet **15**(3) (2023). https://doi.org/10.3390/fi15030113

[5] Diba, K., Batoulis, K., Weidlich, M., Weske, M.: Extraction, correlation, and abstraction of event data for process mining. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **10**(3), e1346 (2020)

[6] van Eck, M.L., Lu, X., Leemans, S.J.J., van der Aalst, W.M.P.: PM^2: A process mining project methodology. In: Advanced Information Systems Engineering. pp. 297–313. Springer International Publishing, Cham (2015)

[7] Fahland, D.: Extracting and pre-processing event logs. CoRR **abs/2211.04338** (2022). https://doi.org/10.48550/arXiv.2211.04338

[8] Falah Rad, M., Shakeri, M., Khoshhal Roudposhti, K., Shakerinia, I.: Probabilistic elderly person's mood analysis based on its activities of daily living using smart facilities. Pattern Analysis and Applications pp. 1–14 (2022)

[9] Fani Sani, M., Sroka, M., Burattin, A.: LLMs and process mining: Challenges in RPA. In: Process Mining Workshops. pp. 379–391. Springer Nature (2024)

[10] Fani Sani, M., van Zelst, S.J., van der Aalst, W.: Conformance checking approximation using subset selection and edit distance. In: 32nd International Conference on Advanced Information Systems Engineering (CAiSE). pp. 234–251. Springer (2020)

[11] Janssen, D., Mannhardt, F., Koschmider, A., van Zelst, S.: Process model discovery from sensor event data. In: Process Mining Workshops. pp. 69–81. Springer (2021)

[12] Jessen, U., Sroka, M., Fahland, D.: Chit-chat or deep talk: Prompt engineering for process mining. arXiv preprint arXiv:2307.09909 (2023)

[13] Jia, Z., Shen, C., Yi, X., Chen, Y., Yu, T., Guan, X.: Big-data analysis of multi-source logs for anomaly detection on network-based system. In: 13th IEEE conference on automation science and engineering (CASE). pp. 1136–1141 (2017)

[14] Karkouch, A., Mousannif, H., Al Moatassime, H., Noel, T.: Data quality in Internet of Things: A state-of-the-art survey. Journal of Network and Computer Applications **73**, 57–81 (2016)

[15] de Leoni, M., Dündar, S.: Event-log abstraction using batch session identification and clustering. In: Proceedings of the 35th Annual ACM Symposium on Applied Computing. pp. 36–44 (2020)

[16] Li, C.Y., van Zelst, S., van der Aalst, W.: An activity instance based hierarchical framework for event abstraction. In: 3rd International Conference on Process Mining (ICPM). pp. 160–167. IEEE (2021)

[17] Liu, Y., Stein Dani, V., Beerepoot, I., Lu, X.: Turning logs into lumber: Preprocessing tasks in process mining. In: Process Mining Workshops. pp. 98–109. Springer (2024)

[18] Lull, J.J., Bayo-Monton, J.L., Shirali, M., Ghassemian, M., Fernandez-Llatas, C.: Interactive Process Mining in IoT and Human Behaviour Modelling, pp. 217–231. Springer International Publishing, Cham (2021)

[19] Mangler, J., Seiger, R., Benzin, J.V., Grüger, J., Kirikkayis, Y., Gallik, F., Malburg, L., Ehrendorfer, M., Bertrand, Y., Franceschetti, M., Weber, B., Rinderle-Ma, S., Bergmann, R., Asensio, E.S., Reichert, M.: From Internet of Things data to business processes: Challenges and a framework (2024), `https://arxiv.org/abs/2405.08528`

[20] Rebmann, A., van der Aa, H.: Unsupervised task recognition from user interaction streams. In: International Conference on Advanced Information Systems Engineering. pp. 141–157. Springer (2023)

[21] Rebmann, A., Pfeiffer, P., Fettke, P., Aa, H.v.d.: Multi-perspective identification of event groups for event abstraction. In: International Conference on Process Mining. pp. 31–43. Springer (2022)

[22] Sani, M.F.: Preprocessing event data in process mining. In: CAiSE (Doctoral Consortium). pp. 1–10 (2020)

[23] Sarno, R., Effendi, Y.A.: Hierarchy process mining from multi-source logs. Telecommunication Computing Electronics and Control **15**(4), 1955–1970 (2017)

[24] Seiger, R., Franceschetti, M., Weber, B.: An interactive method for detection of process activity executions from iot data. Future Internet **15**(2) (2023)

[25] Senderovich, A., Rogge-Solti, A., Gal, A., Mendling, J., Mandelbaum, A.: The road from sensor data to process instances via interaction mining. In: 28th Conference on Advanced Information Systems Engineering. pp. 257–273. Springer (2016)

[26] Shirali, M., Ahmadi, Z., Fernández-Llatas, C., Bayo-Monton, J.L.: Synergy of information in multimodal IoT systems–discovering the impact of daily behaviour routines on physical activity level. arXiv preprint arXiv:2403.14707 (2024)

[27] Shirali, M., Bayo-Monton, J.L., Fernandez-Llatas, C., Ghassemian, M., Traver Salcedo, V.: Design and evaluation of a solo-resident smart home testbed for mobility pattern monitoring and behavioural assessment. Sensors **20**(24) (2020)

[28] Stein Dani, V., Leopold, H., van der Werf, Jan Martijn, M., Beerepoot, I., Reijers, H.: From loss of interest to denial: A study on the terminators of process mining initiatives. In: 36th Conference on Advanced Information Systems Engineering. pp. 371–386. Springer (2024)

[29] Tax, N., Sidorova, N., Haakma, R., van der Aalst, W.: Mining process model descriptions of daily life through event abstraction. In: Intelligent Systems and Applications (IntelliSys). pp. 83–104. Springer (2016)

[30] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q.V., Zhou, D., et al.: Chain-of-thought prompting elicits reasoning in large language models. Advances in neural information processing systems **35**, 24824–24837 (2022)

[31] Yong, G., Jeon, K., Gil, D., Lee, G.: Prompt engineering for zero-shot and few-shot defect detection and classification using a visual-language pretrained model. Computer-Aided Civil and Infrastructure Engineering **38**(11), 1536–1554 (2023)

[32] van Zelst, S., Mannhardt, F., de Leoni, M., Koschmider, A.: Event abstraction in process mining: literature review and taxonomy. Granular Computing **6**, 719–736 (2021)