Using Process Mining and LLMs for Activity Recognition in a Smart Home Environment

Student: Mattia Curri

Formal Methods in Computer Science
Master Degree in Computer Science

Introduction

Data-driven technique that analyzes event logs from information systems to discover, monitor, and improve real business processe

Extracts process models from event data.
Compares actual processes with predefined models.
Identifies inefficiencies and suggests optimizations.

Business process optimization
Compliance auditing
Customer journey analysis

Dataset

The MIT smart home dataset is a dataset in the field of Activity Recognition, collected in a real-world smart home environment.

Data was collected in a smart home and features various binary sensors.

Timestamp, CaseID, Activity

```
Timestamp   CaseID  Activity
2003-03-27 06:43:40  Cabinet_67  Cabinet_67 ON Toileting
2003-03-27 06:44:06  Toilet_Flush_100    Toilet_Flush_100 ON Toileting
2003-03-27 06:44:20  Light_switch_101    Light_switch_101 ON Toileting
2003-03-27 06:44:35  Medicine_cabinet_57     Medicine_cabinet_57 ON Toileting
2003-03-27 06:44:36  Medicine_cabinet_58     Medicine_cabinet_58 ON Toileting
2003-03-27 06:44:49  Cabinet_67  Cabinet_67 ON Toileting
2003-03-27 06:45:45  Drawer_82   Drawer_82 ON Toileting
2003-03-27 06:46:12  Drawer_71   Drawer_71 ON None
2003-03-27 06:51:43  Cabinet_80  Cabinet_80 ON Preparing_breakfast
2003-03-27 06:54:09  Microwave_143   Microwave_143 ON Preparing_breakfast
2003-03-27 06:54:16  Cabinet_55  Cabinet_55 ON Preparing_breakfast
2003-03-27 06:57:05  Door_54     Door_54 ON Preparing_breakfast
2003-03-27 07:04:55  Door_54     Door_54 ON Bathing
2003-03-27 07:05:22  Shower_faucet_93    Shower_faucet_93 ON Bathing
2003-03-27 07:05:39  Cabinet_72  Cabinet_72 ON Bathing
2003-03-27 07:11:23  Cabinet_67  Cabinet_67 ON Bathing
2003-03-27 07:34:05  Medicine_cabinet_57     Medicine_cabinet_57 ON None
2003-03-27 07:34:47  Toilet_Flush_100    Toilet_Flush_100 ON None
2003-03-27 07:35:27  Drawer_82   Drawer_82 ON Dressing
2003-03-27 07:35:50  Drawer_75   Drawer_75 ON Dressing
2003-03-27 07:36:23  Drawer_82   Drawer_82 ON Dressing
2003-03-27 07:37:18  Drawer_75   Drawer_75 ON Dressing
2003-03-27 07:38:48  Drawer_84   Drawer_84 ON Preparing_breakfast
2003-03-27 07:38:48  Drawer_84   Drawer_84 ON Preparing_breakfast
2003-03-27 07:39:07  Cabinet_72  Cabinet_72 ON Preparing_breakfast
2003-03-27 07:39:14  Cabinet_73  Cabinet_73 ON Preparing_breakfast
2003-03-27 07:40:32  Dishwasher_70   Dishwasher_70 ON Preparing_breakfast
2003-03-27 07:40:46  Drawer_135  Drawer_135 ON Preparing_breakfast
2003-03-27 07:41:08  Refrigerator_91     Refrigerator_91 ON Preparing_breakfast
2003-03-27 07:41:36  Cabinet_73  Cabinet_73 ON Preparing_breakfast
2003-03-27 07:41:36  Cabinet_73  Cabinet_73 ON Preparing_breakfast
2003-03-27 07:43:22  Sink_faucet_-_cold_88   Sink_faucet_-_cold_88 ON Grooming
2003-03-27 07:43:23  Sink_faucet_-_hot_68    Sink_faucet_-_hot_68 ON Grooming
2003-03-27 07:44:10  Medicine_cabinet_57     Medicine_cabinet_57 ON Grooming
2003-03-27 07:48:49  Door_140    Door_140 ON Going_out_to_work
2003-03-27 11:35:24  Door_140    Door_140 ON Preparing_lunch
2003-03-27 11:36:15  Freezer_137     Freezer_137 ON Preparing_lunch
```

A sample of the dataset

Algorithms

The Alpha algorithm constructs a Petri net representation of a process by analyzing event logs.

The Heuristic Miner algorithm is an extension of the Alpha algorithm that deals with noise and incompleteness in event logs. It uses frequency-based heuristics to construct a more robust process model.

The Inductive Miner algorithm aims to produce sound process models that are guaranteed to be free of deadlocks and other anomalies.
The noise threshold parameter is used to filter out infrequent and potentially noisy behavior from the event logs.

Evaluation

Fitness: Measures how well the discovered model can reproduce the behavior observed in the event log.

Precision: Measures how much behavior allowed by the model is observed in the event log.

Simplicity: Measures the complexity of the discovered model.

Generalization: Measures how well the model can generalize to unseen behavior.

Experiments

No processing vs Processing

Base model vs Tuned model (Noise Threshold)

Trimming of little traces vs No Trimming

Processing

CSV Formatting

Removing Suffix Numbers
e.g. combining Sink_faucet_-_hot and Sink_faucet_-_cold into Sink_faucet

Activity Windows: segment the activities into windows based on changes in activity or case ID. This function ensured that each activity window had a start and end timestamp, providing a clear view of the activity duration.

🧠 LLMs Integration

Dataset Augmented with data generated by large language models.

GPT-4o, Claude 3.5 Sonnet, Qwen 2.5 Plus, Deepseek-R1, Mistral were used to generate data.

Limited context window -> Activities less represented

System Prompt - 1

You are a synthetic data generator specialized in process mining. Your task is to expand the user-provided dataset (with columns [Timestamp, CaseID, Activity]) by creating new realistic cases that strictly adhere to the existing activities in the original dataset.
Binding Instructions:
Mandatory Input: The user will always provide an existing dataset as reference.
Activity Preservation: Use only the activities present in the original dataset (no new invented activities).
Output Structure: Maintain the exact 3-column order: Timestamp, CaseID, Activity.

Generation Guidelines:
Temporal Patterns:
Create plausible chronological sequences based on observed logic in the original data.

CaseID Generation:
CaseIDs represent sensors from which the data was obtained. Do not invent new CaseIDs.
Follow the original CaseID format (e.g., C_1001, S_205).

System Prompt - 2

Realistic Timestamps:
Ensure intra-case chronological consistency (no time travel within a case).

Required Output Example:
Timestamp,CaseID,Activity
2023-11-05T08:02:34Z,C_1001,Start
2023-11-05T08:03:11Z,C_1001,Material_Loading
2023-11-05T08:07:22Z,C_1001,Quality_Check
2023-11-05T08:09:45Z,C_1001,End

Absolute Constraints:
Do not invent new activities or CaseIDs.
Do not alter column order or add/remove fields.
Use Cases:
Enrich training data for process discovery algorithms.

Base Model Experiments

Miner Configuration	Fitness	Precision	Simplicity	Generalization
Inductive Miner (0.0)	**1.000**	0.276	0.640	0.304
Inductive Miner (0.1)	0.999	0.356	0.643	**0.308**
Inductive Miner (0.2)	0.994	0.445	0.645	0.300
Inductive Miner (0.3)	0.989	0.511	0.648	0.294
Inductive Miner (0.5)	0.978	0.636	0.660	0.282
Inductive Miner (0.6)	0.976	0.629	0.661	0.278
Inductive Miner (0.8)	0.970	0.624	0.664	0.265
Heuristic Miner	0.955	**0.698**	0.588	0.232
Alpha Miner	0.554	0.025	**1.000**	0.297

Performance metrics comparison. Inductive Miner shown with different noise thresholds, compared to baseline Heuristic and Alpha miners.

Processed Dataset Experiments

Miner Configuration	Fitness	Precision	Simplicity	Generalization
Inductive Miner (0.0)	**0.999**	0.111	0.633	0.407
Inductive Miner (0.1)	**0.999**	0.222	0.634	0.410
Inductive Miner (0.2)	0.991	0.176	0.636	0.407
Inductive Miner (0.3)	0.985	0.337	0.640	0.405
Inductive Miner (0.5)	0.969	0.389	0.647	0.344
Inductive Miner (0.6)	0.969	0.368	0.646	0.349
Inductive Miner (0.8)	0.975	0.382	0.650	0.318
Heuristic Miner	0.973	0.450	0.558	0.294
Alpha Miner	0.425	0.015	1.000	0.412

Performance metrics comparison. Inductive Miner shown with different noise thresholds, compared to baseline Heuristic and Alpha miners.

Trimmed Dataset Experiments

Miner Configuration	Fitness	Precision	Simplicity	Generalization
Inductive Miner (0.0)	**1.000**	0.262	0.641	0.306
Inductive Miner (0.1)	0.999	0.338	0.644	**0.310**
Inductive Miner (0.2)	0.994	0.424	0.646	0.302
Inductive Miner (0.3)	0.988	0.489	0.649	0.297
Inductive Miner (0.5)	0.976	0.617	0.662	0.284
Inductive Miner (0.6)	0.974	0.610	0.663	0.279
Inductive Miner (0.8)	0.968	0.605	0.665	0.266
Heuristic Miner	0.982	**0.677**	0.588	0.233
Alpha Miner	0.533	0.022	**1.000**	0.299

Performance metrics comparison. Inductive Miner shown with different noise thresholds, compared to baseline Heuristic and Alpha miners. Best results per column are highlighted.

GPT-4o

GPT-4o (Original Dataset)				
Miner Configuration	**Fitness**	**Precision**	**Simplicity**	**Generalization**
Inductive Miner (0.0)	**1.000**	0.271	0.640	**0.299**
Inductive Miner (0.2)	0.994	0.428	0.644	0.293
Inductive Miner (0.5)	0.978	0.599	0.659	0.273
Inductive Miner (0.6)	0.976	0.593	0.660	0.269
Inductive Miner (0.8)	0.970	0.589	0.662	0.256
Heuristic Miner	0.955	**0.664**	0.589	0.227
Alpha Miner	0.554	0.025	**1.000**	0.291

Performance metrics comparison **on the original dataset**. Inductive Miner shown with different noise thresholds, compared to baseline Heuristic and Alpha miners.

GPT-4o (Augmented Dataset)				
Miner	**Fitness**	**Precision**	**Simplicity**	**Generalization**
Inductive Miner (0.0)	**1.000**	0.298	0.640	**0.305**
Inductive Miner (0.2)	0.995	0.474	0.644	0.301
Inductive Miner (0.5)	0.979	0.657	0.659	0.278
Inductive Miner (0.6)	0.978	0.656	0.660	0.278
Inductive Miner (0.8)	0.973	0.650	0.662	0.267
Heuristic Miner	0.947	**0.730**	0.589	0.234
Alpha Miner	0.533	0.029	**1.000**	0.299

Performance metrics comparison **on the augmented dataset**. Inductive Miner shown with different noise thresholds, compared to baseline Heuristic and Alpha miners.

Claude 3.5 Sonnet

Claude 3.5 Sonnet (Original Dataset)				
Miner Configuration	**Fitness**	**Precision**	**Simplicity**	**Generalization**
Inductive Miner (0.0)	**0.987**	0.261	0.635	0.279
Inductive Miner (0.2)	0.980	0.413	0.645	0.273
Inductive Miner (0.5)	0.967	0.563	0.650	0.257
Inductive Miner (0.6)	0.966	0.567	0.652	0.258
Inductive Miner (0.8)	0.955	0.577	0.654	0.246
Heuristic Miner	0.958	**0.694**	0.589	0.212
Alpha Miner	0.589	0.024	**1.000**	**0.291**

Performance metrics comparison **on the original dataset**. Inductive Miner shown with different noise thresholds, compared to baseline Heuristic and Alpha miners.

Claude 3.5 Sonnet (Augmented Dataset)				
Miner Configuration	**Fitness**	**Precision**	**Simplicity**	**Generalization**
Inductive Miner (0.0)	**1.000**	0.266	0.635	0.302
Inductive Miner (0.2)	0.990	0.425	0.645	0.297
Inductive Miner (0.5)	0.972	0.579	0.650	0.282
Inductive Miner (0.6)	0.972	0.583	0.652	0.281
Inductive Miner (0.8)	0.962	0.593	0.654	0.273
Heuristic Miner	0.951	**0.697**	0.589	0.231
Alpha Miner	0.536	0.025	**1.000**	**0.325**

Performance metrics comparison **on the augmented dataset**. Inductive Miner shown with different noise thresholds, compared to baseline Heuristic and Alpha miners.

Qwen 2.5 Plus

Qwen 2.5 Plus (Original Dataset)				
Miner	**Fitness**	**Precision**	**Simplicity**	**Generalization**
Inductive Miner (0.0)	**1.000**	0.239	0.639	**0.281**
Inductive Miner (0.2)	0.994	0.338	0.643	0.270
Inductive Miner (0.5)	0.978	0.419	0.655	0.246
Inductive Miner (0.6)	0.976	0.416	0.656	0.243
Inductive Miner (0.8)	0.970	0.416	0.658	0.230
Heuristic Miner	0.955	**0.535**	0.597	0.213
Alpha Miner	0.554	0.025	**1.000**	0.237

Performance metrics comparison **on the original dataset**. Inductive Miner shown with different noise thresholds, compared to baseline Heuristic and Alpha miners.

Qwen 2.5 Plus (Augmented Dataset)				
Miner	**Fitness**	**Precision**	**Simplicity**	**Generalization**
Inductive Miner (0.0)	**1.000**	0.434	0.639	**0.281**
Inductive Miner (0.2)	0.997	0.621	0.643	0.270
Inductive Miner (0.5)	0.988	0.772	0.655	0.242
Inductive Miner (0.6)	0.987	0.770	0.657	0.240
Inductive Miner (0.8)	0.984	0.764	0.658	0.229
Heuristic Miner	0.800	**0.852**	0.597	0.213
Alpha Miner	0.754	0.064	**1.000**	0.237

Performance metrics comparison **on the augmented dataset**. Inductive Miner shown with different noise thresholds, compared to baseline Heuristic and Alpha miners.

Deepseek-R1

Deepseek-R1 (Original Dataset)				
Miner	**Fitness**	**Precision**	**Simplicity**	**Generalization**
Inductive Miner (0.0)	**1.000**	0.245	0.639	**0.276**
Inductive Miner (0.2)	0.994	0.353	0.642	0.265
Inductive Miner (0.5)	0.978	0.447	0.654	0.240
Inductive Miner (0.6)	0.976	0.444	0.655	0.237
Inductive Miner (0.8)	0.970	0.443	0.657	0.224
Heuristic Miner	0.956	**0.509**	0.592	0.206
Alpha Miner	0.554	0.024	**1.000**	0.246

Performance metrics comparison **on the original dataset**. Inductive Miner shown with different noise thresholds, compared to baseline Heuristic and Alpha miners.

Deepseek-R1 (Augmented Dataset)				
Miner	**Fitness**	**Precision**	**Simplicity**	**Generalization**
Inductive Miner (0.0)	**1.000**	0.408	0.639	**0.282**
Inductive Miner (0.2)	0.997	0.596	0.642	0.272
Inductive Miner (0.5)	0.986	0.754	0.654	0.245
Inductive Miner (0.6)	0.985	0.753	0.655	0.245
Inductive Miner (0.8)	0.982	0.747	0.657	0.234
Heuristic Miner	0.940	**0.850**	0.592	0.213
Alpha Miner	0.690	0.052	**1.000**	0.259

Performance metrics comparison **on the augmented dataset**. Inductive Miner shown with different noise thresholds, compared to baseline Heuristic and Alpha miners.

Mistral

Mistral (Original Dataset)				
Miner	**Fitness**	**Precision**	**Simplicity**	**Generalization**
Inductive Miner (0.0)	**1.000**	0.258	0.640	**0.282**
Inductive Miner (0.2)	0.994	0.388	0.644	0.272
Inductive Miner (0.5)	0.978	0.514	0.657	0.247
Inductive Miner (0.6)	0.976	0.510	0.658	0.244
Inductive Miner (0.8)	0.970	0.508	0.660	0.232
Heuristic Miner	0.955	**0.593**	0.591	0.215
Alpha Miner	0.554	0.025	**1.000**	0.274

Performance metrics comparison **on the original dataset**. Inductive Miner shown with different noise thresholds, compared to baseline Heuristic and Alpha miners.

Mistral (Augmented Dataset)				
Miner	**Fitness**	**Precision**	**Simplicity**	**Generalization**
Inductive Miner (0.0)	**1.000**	0.353	0.640	0.296
Inductive Miner (0.2)	0.996	0.540	0.644	0.290
Inductive Miner (0.5)	0.984	0.712	0.657	0.266
Inductive Miner (0.6)	0.982	0.710	0.659	0.262
Inductive Miner (0.8)	0.978	0.705	0.660	0.255
Heuristic Miner	0.907	**0.795**	0.591	0.228
Alpha Miner	0.550	0.040	**1.000**	**0.302**

Performance metrics comparison **on the augmented dataset**. Inductive Miner shown with different noise thresholds, compared to baseline Heuristic and Alpha miners.

🤖 LLM Analysis

Large Language Models (LLMs) can be used to directly analyze datasets by understanding and interpreting the event logs.

LLMs can help in identifying patterns and anomalies in event logs, providing insights that may not be immediately apparent through traditional analysis methods.

LLMs can assist in preprocessing event logs by normalizing and structuring data, making it easier to apply process mining algorithms and improving the overall quality of the analysis.

🤖 Models Used

GPT-4o

DeepSeek-R1 (🧠!)

Qwen2.5-Max

Llama3.2:3B (SLM in local)

🤖 Prompt

Here is an activity recognition dataset. Do a full analysis. Find any anomalies, bottlenecks, and special things.

Thanks for the attention! 🚀