EdgeWisePersona: A Dataset for On-Device User Profiling from Natural Language Interactions

Anonymous ACL submission

Abstract

This paper introduces a novel dataset and evaluation benchmark designed to assess and improve small language models deployable on edge devices, with a focus on user profiling from multi-session natural language interactions in smart home environments. At the core of the dataset are structured user profiles, each defined by a set of routines - context-triggered, repeatable patterns of behavior that govern how users interact with their home systems. Using these profiles as input, a large language model (LLM) generates corresponding interaction sessions that simulate realistic, diverse, and context-aware dialogues between users and their devices.

The primary task supported by this dataset is profile reconstruction: inferring user routines and preferences solely from interaction history. To assess how well current models can perform this task under realistic conditions, we benchmarked several state-of-the-art compact language models and compared their performance against large foundation models. Our results show that while small models demonstrate some capability in reconstructing profiles, they still fall significantly short of large models in accurately capturing user behavior. This performance gap poses a major challenge - particularly because on-device processing offers critical advantages, such as preserving user privacy, minimizing latency, and enabling personalized experiences without reliance on the cloud. By providing a realistic, structured testbed for developing and evaluating behavioral modeling under these constraints, our dataset represents a key step toward enabling intelligent, privacy-respecting AI systems that learn and adapt directly on user-owned devices.

1 Introduction

In recent years, the field of artificial intelligence (AI) has witnessed a significant shift toward models that can be developed and executed directly on edge devices, particularly smartphones, TVs, and tablets. This trend has been driven by rapid advancements in both AI model optimization techniques (Yu et al., 2024; Sander et al., 2025; Saha and Xu, 2025) and the computational capabilities of mobile hardware (Jayanth et al., 2024; Li et al., 2025; Barker, 2025). Modern mobile devices are now equipped with increasingly powerful processors, dedicated neural processing units (NPUs) (Tan and Cao, 2021; Xu et al., 2024), and high-efficiency memory systems, enabling them to run sophisticated AI workloads locally. Simultaneously, the development of compact, efficient models - especially in the realm of generative AI - has accelerated at an unprecedented pace (Wang et al., 2025). Recent releases such as Qwen (Qwen et al., 2025), Phi (Abdin et al., 2024), Gemma (Gemma et al., 2025), and other small-scale generative models have demonstrated that meaningful on-device inference is not only possible but increasingly practical for real-world applications. Together, these parallel progressions are catalyzing a fast-paced evolution in mobile AI, bringing once cloud-dependent capabilities into users' hands in real time.

The migration of generative AI capabilities from centralized cloud infrastructures to edge devices brings several compelling advantages (Wang and Jia, 2025; Navardi et al., 2025). Chief among them is reduced latency: by performing inference directly on-device, users experience faster response times without the delays associated with network communication (Ye et al., 2025; Tang et al., 2025; Kim et al., 2024). This is particularly beneficial for interactive applications like real-time translation, voice assistants, and on-the-fly content generation. Additionally, enhanced privacy and data security become inherent features of on-device processing, as sensitive user data no longer needs to be transmitted to remote servers (Aminifar et al., 2024). This local-first approach mitigates risks related to data breaches, unauthorized access, and compli-

ance with data protection regulations. Moreover, on-device AI supports greater reliability and autonomy, allowing applications to function seamlessly in offline or low-connectivity environments. Finally, this paradigm shift also promotes energy efficiency at scale, reducing the load on data centers and cutting down the environmental footprint associated with cloud-based AI processing (Mao et al., 2024; Shafique et al., 2021). Together, these benefits make a strong case for the continued push toward edge-native generative intelligence.

These emerging on-device capabilities are particularly promising for the development of intelligent, privacy-preserving smart home systems. With generative AI models running locally, it becomes feasible to create highly personalized environments - homes that adapt dynamically to user preferences, routines, and contexts without relying on cloud infrastructure. A crucial component of such systems is the ability to construct and maintain detailed user profiles that inform device behavior over time (Wu et al., 2024; Suman et al., 2022; Balog and Zhai, 2025; Birkmose et al., 2025). On-device processing offers a unique advantage here: it enables the continuous refinement of these profiles while keeping sensitive data entirely within the user's control. However, despite the growing potential, there is a notable absence of publicly available datasets that could accelerate research and development in this space. Existing datasets rarely include rich, structured user profiles or realistic interaction patterns representative of smart home use cases. To address this gap, our work introduces a comprehensive dataset that includes both well-defined, structured user profiles and simulated sessions of user–device interactions grounded in those profiles. Beyond the dataset itself, we also evaluate the capacity of current generative AI models to reconstruct accurate user profiles based solely on observed interaction histories - providing insights into their potential for personalization, as well as their limitations.

2 Related Work

As outlined, enabling intelligent, privacy-preserving smart home systems requires constructing structured user profiles through continuous, multi-session interactions. While there has been considerable progress in dialogue systems, preference modeling, and smart home control, existing datasets do not fully meet the requirements for progressive, session-spanning user profile reconstruction. Most available resources focus on isolated dialogues, one-off preference statements, or open-domain conversations without persistent user identity, structured numeric profiling, or domain-specific smart home interactions.

PrefEval (Zhao et al., 2025) evaluates models' ability to follow user preferences but consists only of isolated preference-query pairs with no prolonged, multi-session dialogues. Profiles are limited to single preferences rather than structured behavioral representations. In contrast, our dataset supports incremental profile construction across coherent sessions per user.

AllenAI SODA (Kim et al., 2022) provides large-scale synthetic dialogues emphasizing commonsense reasoning in standalone social scenarios. However, it lacks persistent user identities and does not model cumulative preference acquisition or smart home control, which are central to our work.

ParlAI PersonaChat (Miller et al., 2017) introduces simple personas for chit-chat but restricts each to a single conversation without iterative updates. It focuses on general interests rather than device control or numeric profiling, differing from our emphasis on structured, evolving smart home profiles.

LMSYS-Chat-1M (Zheng et al., 2024) compiles a vast set of independent user–assistant dialogues across diverse topics. While extensive, it lacks user continuity across sessions and does not focus on domain-specific profile accumulation, unlike our design for persistent, profile-driven smart home interactions.

Home-Assistant-Requests (Dataset, 2023) addresses smart home commands but only as single-turn queries, without multi-turn dialogues or user profiling. Our dataset extends beyond single commands to simulate realistic, long-term smart home personalization.

OpenAssistant/oasst1 (Köpf et al., 2023) offers rich, open-domain conversational data without structured user profiles, session continuity, or domain-specific focus. Unlike our dataset, it does not target behavior modeling across multiple smart home sessions.

In summary, while existing resources contribute to dialogue research, preference modeling, and smart home command understanding, none provide the combination of persistent user identity, multi-session continuity, structured numeric profiling, and smart home domain focus that our dataset

offers. By supporting the incremental reconstruction of user profiles over extended, realistic interactions, our work bridges a critical gap between dialogue systems, explainable AI, and behavior-based personalization.

3 Dataset Design

The core component of the dataset consists of natural language sessions between users and their smart home systems. These dialogues simulate realistic, free-form interactions in which users express commands, preferences, or queries. The sessions are grounded in underlying formalized behavioral routines. These routines, along with the user profiles they compose, are also included in the dataset as ground truth annotations. The primary goal is to support and evaluate the ability of compact language models, running directly on edge devices, to reconstruct these routines from unstructured conversational history.

The dataset is synthetic, generated using the DeepSeek-V3 (DeepSeek-AI et al., 2025) language model, but has been meticulously designed according to a set of behavioral and contextual rules described below to ensure realism and consistency. DeepSeek-V3 was selected not only for its strong generative quality but also because its license explicitly permits data generation and derivative benchmark creation, making it well-suited for this purpose. The dataset is inherently self-validating: since sessions are derived from known user routines, and the benchmark task is to recover these routines from the generated dialogues, strong performance by large language models confirms internal coherence and task alignment. In addition, each sample was reviewed by human annotators to ensure local consistency and fidelity to the intended user behavior. Together, these properties make the dataset a reliable foundation for developing and evaluating lightweight, privacy-preserving language models for on-device personalization.

3.1 Definitions

Below we define the core concepts and structures that form the foundation of the dataset. The corresponding data format is provided in the Appendix A.

The smart home environment in our dataset is characterized by a set of predefined **device states**, representing the status of household devices at any given time. The modeled devices include a TV (volume, brightness, input source), an air conditioning (AC) unit (temperature, mode, fan speed), a lighting system (brightness, color temperature, operating mode), a speaker (volume, equalizer settings), and a security system (armed status, alarm volume). These baseline states provide the context upon which user interactions operate. Through natural language sessions, users express preferences and commands that result in actions - specific modifications to these device states.

Contextual conditions describe the external and environmental factors that influence user behavior and device interaction patterns. These include temporal, environmental, and situational parameters such as the time of day (morning, afternoon, evening, or night), day of the week, sun phase (before sunrise, daylight, or after sunset), weather conditions (sunny, cloudy, rainy, or snowy), and outdoor temperature. These contextual features serve as inputs to the session generation process and play a central role in determining when routines are activated.

Triggers are specific combinations of contextual conditions that determine when a routine is activated. They represent precise environmental or temporal states that must be met for a routine to apply. For example, a trigger might be configured to activate when the time of day is "evening", the weather is "rainy", and the outdoor temperature is "warm". Triggers thus function as logical match rules that map specific context snapshots to the execution of user behaviors.

Routine refers to a repeatable, structured pattern of user behavior that is activated under specific contextual triggers and results in predefined modifications to smart home device states. Routines encapsulate users' habitual interactions, such as adjusting lighting, setting temperatures, or controlling appliances in response to particular environmental or temporal conditions. Each routine is defined by a pair of components: the **triggers** that specify when it should activate, and the **actions** that describe how devices should respond. A collection of routines associated with a user forms a structured behavioral profile, characterizing their typical preferences and interactions within the smart home environment. Determining these routines and reconstructing user profiles solely from natural language interaction sessions is the central goal of the benchmark task defined on top of our dataset.

3.2 Dataset Construction

The dataset is constructed through a controlled, multi-stage process designed to generate realistic, diverse, and structured natural language interactions grounded in user behavior. Each session is derived from a predefined user profile consisting of routines, with the ultimate goal of enabling evaluation of compact language models for routine and profile reconstruction under edge-device constraints. The general architecture of the dataset generation process is illustrated in Figure 1.

Profile and Routine Specification. Each user is associated with a profile that combines a set of behavioral routines with a brief textual description of their personality and lifestyle. While this description does not influence the logic of the routines, it is used during session generation to guide the language model toward producing more natural and individualized dialogue. In contrast, the routines determine the behavioral content by specifying which actions are likely to be taken under given contextual conditions. Together, these components ensure that each generated session reflects both consistent behavior patterns and varied, user-specific linguistic style.

Contextual Sampling. For each session, a realistic context is sampled, including variables such as time, day, sun phase and weather conditions. This context determines which routines may be active and influences the content of the generated dialogue.

Session Generation. Given a user profile and sampled context, a natural language interaction is generated. The prompt includes structured profile information, context, and a system message instructing the model to generate a coherent dialogue between the user and their smart home assistant. To promote interaction diversity and avoid repetition, the prompt also includes a summary of previous sessions along with an explicit instruction to generate a new, distinctive dialogue. Sessions vary in routine-dependence; most are grounded in predefined behaviors, while others are spontaneous to reflect natural variation in user interaction.

Human Review. All generated sessions were reviewed by a team of six human annotators to ensure local coherence, alignment with the intended user profile, and general fluency. To maintain high data quality, we initially generated more sessions than required; annotators were instructed to discard any example that raised uncertainty or exhibited inconsistencies. Minor corrections were made when needed, but the generative output was otherwise preserved to reflect realistic model behavior.

Dataset Structure. The dataset is organized at the user level, with each user represented across three aligned `.jsonl` files: one containing their personality description (`characters.jsonl`), one listing their routines (`routines.jsonl`), and one storing their interaction sessions (`sessions.jsonl`). This modular structure supports both supervised training and benchmark evaluation.

The three files are strictly line-aligned: the n-th line in each file corresponds to the same user. This design ensures that each user's personality, routines, and session history are consistently linked. This structure makes the dataset easy to parse and integrate into training or evaluation pipelines while maintaining a clean and modular format.

3.3 Dataset Statistics

The dataset includes interactions from 200 users, with 50 natural language sessions collected for each user (10000 sessions in total). Each session reflects dialogue with the smart home system, supporting the extraction and inference of structured routines. Each session is associated with at most one routine, with approximately 75% of sessions reflecting a routine-driven behavior and the remaining 25% representing spontaneous or routine-free interactions. This distribution is intended to mirror realistic user behavior, where not every interaction follows a habitual pattern. A summary of key dataset statistics is provided in Table 1.

Table 1: Overview of key statistics for the dataset.

Key	Value
# Users	200
# Sessions per User	50
# Total Sessions	10,000
Avg. # Messages per Session	9.88
Avg. # Routines per User	3.98
Avg. # Triggers per Routine	4.98
Avg. # Devices per Routine	3.11
Avg. # Actions per Routine	8.24

4 Benchmark Results

Our evaluation compares state-of-the-art large LLMs with smaller, resource-efficient models, focusing on their long-context capabilities and overall

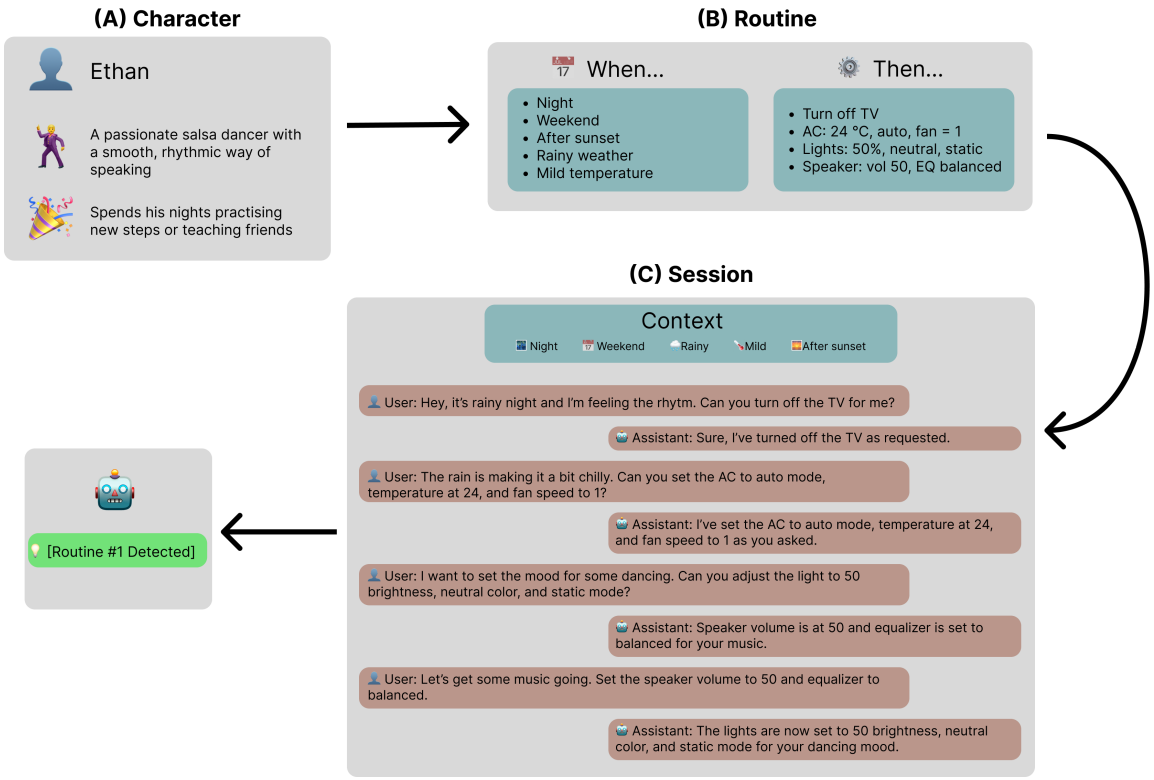

Figure 1: **System architecture for EdgeWisePersona dataset generation and evaluation pipeline.** The framework consists of two interconnected phases: **(A) Character** definition, where Ethan serves as a representative user with defined personality traits and behavioral patterns; **(B) Routine** specification, detailing contextual triggers (night, weekend, after sunset, rainy weather, mild temperature) that activate corresponding smart home actions (TV control, AC settings, lighting adjustments, speaker configuration); and **(C) Session** generation and evaluation, where natural language interactions are produced based on the character's routines and then used to test models' ability to reconstruct the original user profiles.

accuracy. Model selection was guided by performance and context length constraints, using comparative metrics from the LLM-Stats leaderboard (LLM Stats, 2024). As edge-deployable candidates, we selected top-performing models with fewer than 5B parameters that are compatible with mobile and resource-constrained environments, and that support extended input contexts of at least 128k tokens - necessary for handling the long, multi-turn sessions in our dataset. The models used in our evaluation include **Gemma-3-4B** (Gemma et al., 2025), **Qwen-2.5-3B** (Qwen et al., 2025), **Llama3.2-3B** (Grattafiori et al., 2024; Meta AI, 2024), and **Phi-4-mini-4B** (Abdin et al., 2024), all of which offer strong accuracy relative to their size.

As the foundation for our baseline comparisons, we include state-of-the-art large language models, such as **GPT-4o** (OpenAI et al., 2024a,b), **Gemini-2.5-Flash** (Gemini et al., 2025; Google DeepMind, 2025), and **DeepSeek-V3** (DeepSeek-AI et al., 2025). These models represent the upper bound in terms of both scale and capability, and serve as strong reference points for evaluating the performance of smaller, mobile-deployable models.

Compute setup. Large-scale models were accessed via their official hosted APIs. Compact models were benchmarked locally on a single NVIDIA RTX 3090 GPU (24 GB). To validate on-device compatibility, we exported each model to mobile-friendly formats and tested execution across four runtime stacks, confirming that they can be deployed and run on real edge hardware. We used three smartphones for this evaluation: the Qualcomm Snapdragon 675-based TCL 10 5G and TCL T1 Pro, and the MediaTek Dimensity 9300-powered OPPO Find X7. Execution was tested using widely adopted mobile inference frameworks: Alibaba MNN (Jiang et al., 2020), Executorch (Executorch, 2024) with MediaTek NeuroPilot (Lee et al., 2023), ONNX Runtime (developers, 2021) Mobile with NNAPI and GPU acceleration, and llama.cpp (Gerganov, 2023) for CPU-only fallback.

4.1 Evaluation Protocol

We evaluate each model under a uniform prediction-count policy to ensure fair comparison. Specifically, models are prompted to predict exactly the same number of routines as the ground truth contains. If a model generates more routines than the reference, we truncate the extra predictions; if it generates fewer, we retain the given predictions (leaving the missing routines as blank). This alignment guarantees a one-to-one correspondence for scoring each predicted routine against its true counterpart. All accuracy metrics are then computed per routine and averaged over the corpus.

4.2 Routine-Level Accuracy

We report two complementary metrics for overall routine prediction: strict exact-match accuracy and Jaccard similarity. **Exact-match accuracy** measures the proportion of predicted routines that match the ground-truth routines exactly - across all fields, including contextual triggers, actions, and their parameters. This metric provides a stringent assessment of model performance, as even a small deviation results in a non-match. **Jaccard similarity**, by contrast, measures the degree of partial overlap between the predicted and reference sets of routine components, offering a more forgiving view of routine reconstruction quality. We express both metrics as percentages to support clear interpretation and facilitate direct comparison between models.

As shown in Figure 2, large models substantially outperform smaller ones on both metrics. For example, Gemini-2.5-Flash achieves an exact-match accuracy of 44%, followed by GPT-4o at 35% and DeepSeek-V3 at 27%. In comparison, the best-performing small model, Gemma-3-4B, achieves only 2%, while the remaining small models are close to zero - indicating they rarely predict a complete routine correctly.

A similar trend appears in Jaccard similarity scores: Gemini reaches 90%, followed by GPT-4o (85%) and DeepSeek-V3 (82%). Among the smaller models, Gemma-3-4B performs best with a score of 66%, while Qwen-2.5-3B and Llama3.2-3B achieve 55% and 40% respectively. Phi-4-mini-4B lags behind with 23%. These results highlight a clear performance gap: while large models often capture most aspects of a routine (reflected in high Jaccard similarity scores), smaller models struggle to do so - and achieving perfect predictions remains a challenge even for the best systems.

4.3 Trigger vs. Action Prediction

We further break down model performance by separately evaluating predictions of triggers and actions. For each routine predicted by a model, we first identify the most similar ground-truth routine using Jaccard similarity. We then apply exact-match evaluation to the trigger and action components of that matched pair - checking whether each part was reproduced exactly. As shown in Figure 3, this reveals a stark difference in difficulty between the two.

All models perform substantially better on trigger prediction than on action prediction. The large models almost always identify the correct trigger: for example, GPT-4o and Gemini-2.5-Flash each exceed 90% trigger accuracy, and DeepSeek-V3 reaches 58%. Even smaller models like Qwen-2.5-3B and Llama3.2-3B achieve 77%, suggesting that compact models can often guess the correct initiating condition. This is likely because triggers consist of categorical values (e.g., time of day, sun phase), which are easier to recover than scalar outputs.

In contrast, action prediction proves far more challenging. The best-performing large model (Gemini-2.5-Flash) achieves only about 45% accuracy on full action sequences, followed by GPT-4o at 36% and DeepSeek-V3 at 32%. These results indicate that even advanced models struggle to reproduce full sets of actions precisely. For the smaller models, performance is near-zero: Gemma-3-4B reaches just 4%, while the others fail to generate correct actions in most cases. This highlights the structured complexity of action prediction as a major bottleneck in accurate routine reconstruction.

4.4 Scalar Fields Accuracy

Many routine actions involve scalar parameters (numeric values such as brightness levels, volume settings, or target temperatures). We evaluate each model's ability to predict these numeric fields using mean absolute error (MAE). Table 2 reports the results. The large models deliver near-perfect precision: GPT-4o achieves an average MAE of approximately 0.014, and Gemini-2.5-Flash is nearly identical at 0.018. DeepSeek-V3 is slightly higher but still low, at 0.130. In contrast, compact models miss the mark by an order of magnitude. Gemma-3-4B reaches 1.049, while Qwen-2.5-3B and Llama3.2-3B follow with 1.837 and

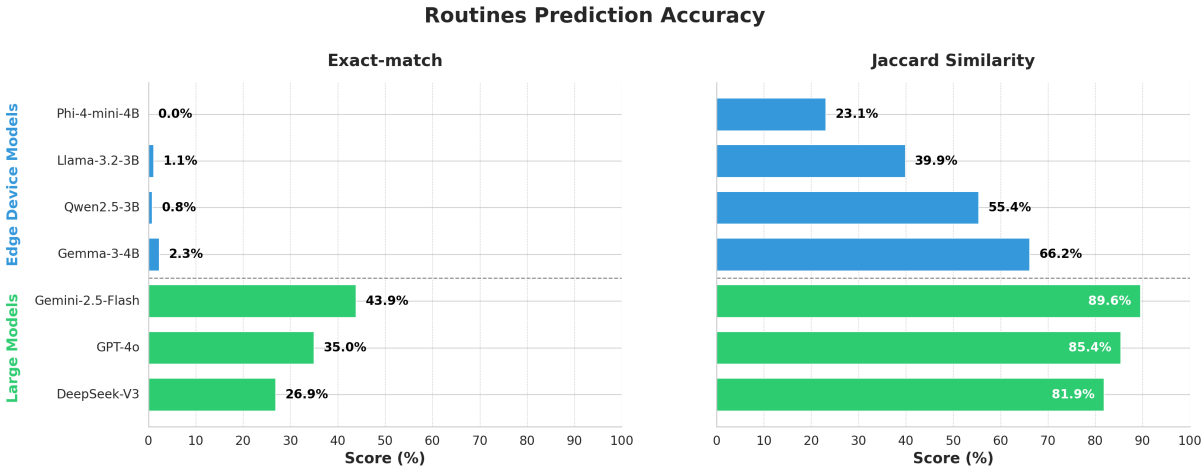

Figure 2: Comparison of model performance on the routine prediction task using exact-match accuracy (left) and Jaccard similarity (right). Results are grouped by model size. Exact-match reflects full structural correctness, while Jaccard similarity captures partial overlap between predicted and reference routines.

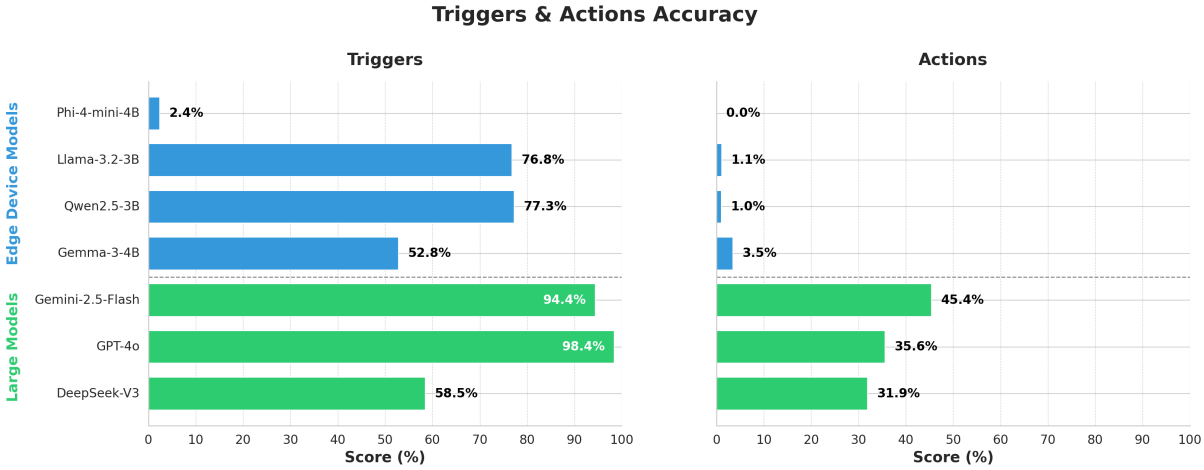

Figure 3: Model accuracy on trigger (left) and action (right) prediction. Scores are computed using exact-match evaluation against the best-matching ground-truth routine.

1.895, respectively. Phi-4-mini-4B performs worst, with an average MAE of 7.348 (peaking at 13.399 on the lights brightness field). These gaps show that large LLMs can infer numeric device settings with sub-unit accuracy, whereas edge-oriented models frequently deviate by several units - well outside a tolerable range for reliable smart-home control.

4.5 Categorical Fields Accuracy

Finally, we assess **categorical fields** in the routines, such as device modes or preset names (e.g. *cool*, *heat* or *auto* as AC mode settings). For these discrete settings, we measure exact-match accuracy - whether the model selected the correct category out of all possible options. Table 3 summarizes results across seven representative fields. The large models get almost every categorical choice right. GPT-4o is the top performer, averaging close to 97% accuracy across all fields, followed by DeepSeek-V3 and Gemini-2.5-Flash at approximately 95% and 92%, respectively. Among compact models, Gemma-3-4B does best, attaining almost 88% accuracy - respectable, yet still below the large models. The other edge-sized models struggle: Qwen2.5-3B is correct a little over 50% of the time, while Phi-4-mini-4B and LLaMA-3.2-3B trail at roughly 32% and 24%, respectively. These disparities underscore how much fine-grained device settings rely on nuanced contextual cues: large LLMs capture those cues consistently, whereas smaller models often default to common or *safe* categories. For reliable prediction of categorical device parameters in real deployments, the larger models therefore retain a clear advantage.

Table 2: Mean Absolute Error (MAE) for predicted scalar device fields. These fields include numeric settings for various devices in the routines. Lower values denote more accurate predictions of user-specific settings. Values are averaged across all users.

Model	Avg. MAE	TV Volume	TV Brightness	AC Temp.	Lights Bright.	Speaker Vol.	Security Alarm
DeepSeek-V3	0.130	0.087	0.138	0.030	0.340	0.064	0.119
GPT-4o	**0.014**	**0.017**	**0.017**	**0.000**	0.052	**0.000**	**0.000**
Gemini-2.5-Flash	0.018	**0.017**	0.034	0.008	**0.026**	0.025	**0.000**
Gemma-3-4B	**1.049**	**1.172**	1.970	**0.158**	**1.680**	**0.266**	**1.050**
Qwen2.5-3B	1.837	1.436	1.702	0.422	4.867	1.355	1.242
LLaMA-3.2-3B	1.895	2.245	**1.556**	0.258	3.773	0.702	2.838
Phi-4-mini-4B	7.348	6.250	5.833	1.327	13.399	5.661	11.616

Table 3: Accuracy for categorical device fields in the routines. Each column corresponds to a device setting that the model must predict correctly. The values indicate the percentage of predictions that exactly matched the true category.

Model	Avg. Acc.	TV Input	AC Mode	AC Fan	Lights Color	Lights Mode	Speaker EQ	Sec. Armed
DeepSeek-V3	95.2	96.0	98.9	98.3	99.2	**94.6**	91.3	88.1
GPT-4o	**96.9**	**98.3**	**99.4**	**99.2**	**100.0**	90.1	**97.9**	**93.6**
Gemini-2.5-Flash	91.8	**98.3**	99.2	98.6	99.2	66.1	93.7	87.8
Gemma-3-4B	**87.7**	**84.7**	**94.7**	**94.1**	**94.6**	**65.2**	**93.7**	**87.2**
Qwen2.5-3B	56.0	27.9	76.2	75.1	78.8	58.6	50.1	25.6
LLaMA-3.2-3B	23.2	14.7	26.7	25.8	25.7	24.8	26.4	19.6
Phi-4-mini-4B	32.2	23.1	27.2	41.9	60.9	13.2	41.0	17.8

4.6 Open Source Benchmarking Tools

The full benchmarking pipeline, including evaluation scripts and data class definitions, is open-source and publicly available. This allows other researchers to reproduce our results, extend the benchmark to new models, or adapt it for related user profiling tasks. The extensibility of the framework supports further exploration into long-context modeling, explainable personalization, and low-latency edge deployment of conversational AI.

5 Future Work

The dataset and evaluations presented in this study establish a foundation for developing privacy-preserving, on-device intelligent agents for smart home environments. However, several avenues remain open for further exploration.

One promising direction is the expansion of user profile complexity to incorporate evolving behaviors, multi-user dynamics within shared spaces, and more nuanced contextual factors such as emotional states or environmental changes. This would allow models to learn and adapt to richer, more lifelike scenarios.

Future iterations of this research would also benefit from real-world deployment and feedback loops, where user interaction data - collected and processed entirely on-device - can inform iterative improvements.

6 Conclusions

This work presents a novel dataset designed to enable the reconstruction of structured user profiles from natural language interaction sessions with smart home devices. By starting from rich, well-defined profiles and generating realistic sessions using a large language model, the dataset provides a diverse environment for developing and benchmarking compact, on-device AI systems.

While the results demonstrate the feasibility of profile reconstruction using small language models, it is important to acknowledge that current state-of-the-art on-device models still lag behind fully-fledged large language models in their ability to accurately infer complex routines and user behaviors. Significant work remains to close this performance gap and optimize inference capabilities under the computational constraints of edge devices. We believe this resource lays an important foundation for future advances in privacy-preserving personalization, behavior-based user modeling, and the development of transparent, adaptive smart environments directly on user-owned devices.

Limitations

While our dataset provides a valuable testbed for evaluating the ability of language models to reconstruct structured user profiles in smart home settings, several limitations of the dataset design

and scope should be noted.

First, the evaluation of model performance is focused primarily on reconstructing structured profiles from predefined interaction sessions. This approach does not yet address challenges related to long-term personalization, profile drift over time, or handling ambiguous or conflicting user signals.

Second, the current study assumes a high degree of user cooperation and clean data input, which may not always be feasible in practical applications. In real settings, data quality can be affected by noise, interruptions, or inconsistent user engagement, potentially impacting model reliability and the quality of generated profiles.

These limitations highlight key areas for future research aimed at refining system robustness, scalability, and real-world applicability.

Ethical Considerations

Potential for Misuse and Surveillance. The techniques and models developed using this dataset could theoretically be applied to analyze real user behavior patterns without explicit consent. While our intended application focuses on improving user experience through voluntary personalization, the same methods could enable unauthorized surveillance or profiling. We emphasize that any real-world deployment of these techniques should require explicit user consent, transparent data practices, and strong technical safeguards to prevent misuse.

Informed Consent in Real Applications. When methods developed with this dataset are deployed in real systems, it is essential that users clearly understand what behavioral data are collected, how they are processed, and what inferences are drawn about their preferences. Because modern AI systems are complex and often opaque, achieving truly informed consent remains difficult and calls for continued research into interpretable and transparent personalization.

Bias and Representation. Our synthetic dataset generation process, while designed to create diverse user profiles, inherently reflects the biases present in the underlying language model (DeepSeek-V3) used for generation. During the human review phase, we applied targeted checks to identify and correct overt demographic, cultural, or behavioral biases, aiming to enhance diversity and balance across user profiles. Nevertheless, complete neutrality cannot be guaranteed, and certain overlooks may still occur. As a result, the dataset may exhibit skewed representations of user behavior patterns, potentially favoring some groups while underrepresenting others. Additionally, our smart home device categories and interaction patterns may not fully capture the diversity of global user behaviors or technological access. Future work should actively address these representational gaps through more inclusive and empirically grounded data generation strategies.

We encourage researchers using this dataset to consider these ethical implications carefully and to develop corresponding safeguards when transitioning from synthetic evaluation environments to real-world applications involving actual user data.

LLM usage statement

This manuscript acknowledges the use of ChatGPT (OpenAI, 2022), powered by the GPT-5 language model developed by OpenAI, to improve language clarity, refine sentence structure, and enhance overall writing precision.

References

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J. Hewett, Mojan Javaheripi, Piero Kauffmann, James R. Lee, Yin Tat Lee, Yuanzhi Li, Weishung Liu, Caio C. T. Mendes, Anh Nguyen, Eric Price, Gustavo de Rosa, Olli Saarikivi, and 8 others. 2024. Phi-4 technical report. *Preprint*, arXiv:2412.08905.

Amin Aminifar, Matin Shokri, and Amir Aminifar. 2024. Privacy-preserving edge federated learning for intelligent mobile-health systems. *Future Generation Computer Systems*, 161:625–637.

Krisztian Balog and ChengXiang Zhai. 2025. User simulation in the era of generative ai: User modeling, synthetic data generation, and system evaluation. *Preprint*, arXiv:2501.04410.

Ryan Barker. 2025. Advancements in mobile edge computing and open ran: Leveraging artificial intelligence and machine learning for wireless systems. *Preprint*, arXiv:2502.02886.

Rune Birkmose, Nathan Mørkeberg Reece, Esben Hofstedt Norvin, Johannes Bjerva, and Mike Zhang. 2025. On-device llms for home assistant: Dual role in intent detection and response generation. *Preprint*, arXiv:2502.12923.

Home Assistant Requests Dataset. 2023. Accessed: 2025-05-14. [link].

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, and 181 others. 2025. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

ONNX Runtime developers. 2021. Onnx runtime. https://onnxruntime.ai/. Version: x.y.z.

Executorch. 2024. Executorch. GitHub repository. https://github.com/pytorch/executorch.

Team Gemini, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Melvin Johnson, Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy Lillicrap, Angeliki Lazaridou, and 1332 others. 2025. Gemini: A family of highly capable multimodal models. *Preprint*, arXiv:2312.11805.

Team Gemma, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, and 197 others. 2025. Gemma 3 technical report. *Preprint*, arXiv:2503.19786.

Georgi Gerganov. 2023. llama.cpp: Efficient inference of llms on cpus. GitHub repository. https://github.com/ggerganov/llama.cpp.

Google DeepMind. 2025. Gemini 2.5 flash preview: Model card. Technical report, Google. Accessed: 2025-05-15.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Rakshith Jayanth, Neelesh Gupta, and Viktor Prasanna. 2024. Benchmarking edge ai platforms for high-performance ml inference. *Preprint*, arXiv:2409.14803.

Xiaotang Jiang, Huan Wang, Yiliu Chen, Ziqi Wu, Lichuan Wang, Bin Zou, Yafeng Yang, Zongyang Cui, Yu Cai, Tianhang Yu, Chengfei Lv, and Zhihua Wu. 2020. Mnn: A universal and efficient inference engine. In *MLSys*.

Hyunwoo Kim, Jack Hessel, Liwei Jiang, Peter West, Ximing Lu, Youngjae Yu, Pei Zhou, Ronan Le Bras, Malihe Alikhani, Gunhee Kim, Maarten Sap, and Yejin Choi. 2022. Soda: Million-scale dialogue distillation with social commonsense contextualization. *ArXiv*, abs/2212.10465.

Minseo Kim, Wei Shu, and Mohsen Amini Salehi. 2024. He2c: A holistic approach for allocating latency-sensitive ai tasks across edge-cloud. *Preprint*, arXiv:2411.19487.

Andreas Köpf, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi-Rui Tam, Keith Stevens, Abdullah Barhoum, Nguyen Minh Duc, Oliver Stanley, Richárd Nagyfi, Shahul ES, Sameer Suri, David Glushkov, Arnav Dantuluri, Andrew Maguire, Christoph Schuhmann, Huu Nguyen, and Alexander Mattick. 2023. Openassistant conversations – democratizing large language model alignment. *Preprint*, arXiv:2304.07327.

Chao-Lin Lee, Chun-Ping Chung, Sheng-Yuan Cheng, Jenq-Kuen Lee, and Robert Lai and. 2023. Accelerating ai performance with the incorporation of tvm and mediatek neuropilot. *Connection Science*, 35(1):2272586.

Jinhao Li, Jiaming Xu, Shan Huang, Yonghua Chen, Wen Li, Jun Liu, Yaoxiu Lian, Jiayi Pan, Li Ding, Hao Zhou, Yu Wang, and Guohao Dai. 2025. Large language model inference acceleration: A comprehensive hardware perspective. *Preprint*, arXiv:2410.04466.

LLM Stats. 2024. Llm stats – leaderboard and metrics for large language models. Project homepage: https://llm-stats.com, Accessed: 2025-05-14.

Yuyi Mao, Xianghao Yu, Kaibin Huang, Ying-Jun Angela Zhang, and Jun Zhang. 2024. Green edge ai: A contemporary survey. *Preprint*, arXiv:2312.00333.

Meta AI. 2024. Llama 3. https://www.llama.com/models/llama-3/. Accessed: 2025-09-27.

A. H. Miller, W. Feng, A. Fisch, J. Lu, D. Batra, A. Bordes, D. Parikh, and J. Weston. 2017. Parlai: A dialog research software platform. *arXiv preprint arXiv:1705.06476*.

Mozhgan Navardi, Romina Aalishah, Yuzhe Fu, Yueqian Lin, Hai Li, Yiran Chen, and Tinoosh Mohsenin. 2025. Genai at the edge: Comprehensive survey on empowering edge devices. *Preprint*, arXiv:2502.15816.

OpenAI. 2022. Chatgpt. https://chat.openai.com/. Large language model-based conversational system.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024a. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

OpenAI, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec

Radford, Aleksander Mądry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, Alex Nichol, and 400 others. 2024b. Gpt-4o system card. *Preprint*, arXiv:2410.21276.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, and 24 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Shaibal Saha and Lanyu Xu. 2025. Vision transformers on the edge: A comprehensive survey of model compression and acceleration strategies. *Preprint*, arXiv:2503.02891.

Jacob Sander, Achraf Cohen, Venkat R. Dasari, Brent Venable, and Brian Jalaian. 2025. On accelerating edge ai: Optimizing resource-constrained environments. *Preprint*, arXiv:2501.15014.

Muhammad Shafique, Alberto Marchisio, Rachmad Vidya Wicaksana Putra, and Muhammad Abdullah Hanif. 2021. Towards energy-efficient and secure edge ai: A cross-layer framework iccad special session paper. In *2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, page 1–9. IEEE.

Shashi Suman, Francois Rivest, and Ali Etemad. 2022. Towards personalization of user preferences in partially observable smart home environments. *Preprint*, arXiv:2112.00971.

Tianxiang Tan and Guohong Cao. 2021. Deep learning on mobile devices through neural processing units and edge computing. *Preprint*, arXiv:2112.02439.

Jiuqiang Tang, Raman Sarokin, Ekaterina Ignasheva, Grant Jensen, Lin Chen, Juhyun Lee, Andrei Kulik, and Matthias Grundmann. 2025. Scaling on-device gpu inference for large generative models. *Preprint*, arXiv:2505.00232.

Xubin Wang and Weijia Jia. 2025. Optimizing edge ai: A comprehensive survey on data, model, and system strategies. *Preprint*, arXiv:2501.03265.

Xubin Wang, Zhiqing Tang, Jianxiong Guo, Tianhui Meng, Chenhao Wang, Tian Wang, and Weijia Jia. 2025. Empowering edge intelligence: A comprehensive survey on on-device ai models. *ACM Computing Surveys*, 57(9):1–39.

Bin Wu, Zhengyan Shi, Hossein A. Rahmani, Varsha Ramineni, and Emine Yilmaz. 2024. Understanding the role of user profile in the personalization of large language models. *Preprint*, arXiv:2406.17803.

Daliang Xu, Hao Zhang, Liming Yang, Ruiqi Liu, Gang Huang, Mengwei Xu, and Xuanzhe Liu. 2024. Fast on-device llm inference with npus. *Preprint*, arXiv:2407.05858.

Shengyuan Ye, Bei Ouyang, Liekang Zeng, Tianyi Qian, Xiaowen Chu, Jian Tang, and Xu Chen. 2025. Jupiter: Fast and resource-efficient collaborative inference of generative llms on edge devices. *Preprint*, arXiv:2504.08242.

Zhongzhi Yu, Zheng Wang, Yuhan Li, Haoran You, Ruijie Gao, Xiaoya Zhou, Sreenidhi Reedy Bommu, Yang Katie Zhao, and Yingyan Celine Lin. 2024. Edge-llm: Enabling efficient large language model adaptation on edge devices via layerwise unified compression and adaptive layer tuning and voting. *Preprint*, arXiv:2406.15758.

Siyan Zhao, Mingyi Hong, Yang Liu, Devamanyu Hazarika, and Kaixiang Lin. 2025. Do llms recognize your preferences? evaluating personalized preference following in llms. *Preprint*, arXiv:2502.09597.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric P. Xing, Joseph E. Gonzalez, Ion Stoica, and Hao Zhang. 2024. Lmsys-chat-1m: A large-scale real-world llm conversation dataset. *Preprint*, arXiv:2309.11998.

A Appendix: Dataset Schemas

To support transparency, usability, and reproducibility, we include in this appendix detailed schemas representing the internal structure of two key components of our dataset: `routines` and `sessions`. These schemas serve as precise technical documentation, defining the data format used for each synthetic user's behavior and interaction history.

The `routines` schema provides a full description of how each user's behavioral logic is encoded. Routines are central to both the data generation and evaluation processes, and their schema captures the nested structure of contextual triggers, associated device actions, and routine-level metadata such as priority. Each routine is composed of clearly defined fields for environmental and temporal conditions (e.g., time of day, weather, sun phase), as well as structured action blocks targeting specific devices (e.g., lighting, AC, speaker). This format supports interpretable comparisons between ground-truth behavior and reconstructed routines during benchmarking.

Routine Model Structure

```
Routine:
  triggers:
    time_of_day: Optional[
      "morning" |
      "afternoon" |
      "evening" |
      "night"
    ]
    day_of_week: Optional[
```

```
      "weekday" |
      "weekend"
    ]
    sun_phase: Optional[
      "before_sunrise" |
      "daylight" |
      "after_sunset"
    ]
    weather: Optional[
      "sunny" |
      "cloudy" |
      "rainy" |
      "snowy"
    ]
    outdoor_temp: Optional[
      "very cold" |
      "cold" |
      "mild" |
      "warm" |
      "hot"
    ]

  actions:
    tv:
      volume: Optional[int]
      brightness: Optional[int]
      input_source: Optional[
        "HDMI1" |
        "HDMI2" |
        "AV" |
        "Netflix" |
        "YouTube"
      ]

    ac:
      temperature: Optional[int] # Range: 16-30
      mode: Optional[
        "cool" |
        "heat" |
        "auto"
      ]
      fan_speed: Optional[int] # Range: 0-3

    lights:
      brightness: Optional[int]
      color: Optional[
        "warm" |
        "cool" |
        "neutral"
      ]
      mode: Optional[
        "static" |
        "dynamic"
      ]

    speaker:
      volume: Optional[int]
      equalizer: Optional[
        "bass boost" |
        "balanced" |
        "treble boost"
      ]

    security:
      armed: Optional[bool]
      alarm_volume: Optional[int]
```

The sessions schema defines the structure of the generated natural language dialogues. Each session entry encapsulates a complete multi-turn interaction between a user and their smart home system, enriched with metadata about the contextual state in which the session takes place. The schema captures the contextual snapshot and optional markers for routine activation or model-generated reasoning. This structured format allows for controlled evaluation scenarios and enables models to learn associations between language, context, and user behavior.

Session Model Structure

```
Session:
  session_id: int

  meta:
    time_of_day:
      "morning" |
      "afternoon" |
      "evening" |
      "night"
    day_of_week:
      "weekday" |
      "weekend"
    sun_phase:
      "before_sunrise" |
      "daylight" |
      "after_sunset"
    weather:
      "sunny" |
      "cloudy" |
      "rainy" |
      "snowy"
    outdoor_temp:
      "very cold" |
      "cold" |
      "mild" |
      "warm" |
      "hot"

  messages:
    - role:
        "user" |
        "assistant"
      text: str

  applied_routines:
    List[int]
```

Together, these schemas provide a clear and comprehensive preview of the dataset's internal design. They are intended to guide parsing and integration into training pipelines and we hope these specifications will facilitate robust adoption and adaptation of the dataset across a wide range of downstream tasks and evaluation settings.