Bootstrapping Contextual Understanding in Privacy-Preserving Smart Environments

Prasoon Patidar

CMU-S3D-26-110

June 2026

Software and Societal Systems Department
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Dr. Yuvraj Agarwal, Chair
Dr. Mayank Goel
Dr. Andrew Begel
Dr. Ranveer Chandra (Microsoft)

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Societal Computing.

Copyright © 2026 **Prasoon Patidar**

The research reported here was supported, in whole or in part, by the National Science Foundation under its Secure and Trustworthy Cyberspace (SaTC) program (Grants No. 1564009 and 1801472) and its Cyberlearning program (Grant No. 1822813); by the Wilton E. Scott Institute for Energy Innovation and a gift from Trane Technologies; and through the Center for Machine Learning and Health (CMLH) Translational Fellowship and the CyLab Security and Privacy Institute Presidential Fellowship. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government, or any other entity.

Keywords: Human Activity Recognition, Privacy-Preserving Sensing, Context-Aware Computing, Smart Environments, Ubiquitous Computing

To my mentors, my family, and my friends,
your faith in me gave me the courage to follow my curiosity.

Abstract

Physical spaces instrumented with sensors that infer the activities of their occupants and offer them insights, automation, and assistance have been a longstanding vision of the research community and companies. Today, the technology that has made such understanding possible at scale typically relies on machine learning models built for high fidelity sensors like cameras and microphones due to the fact that there is ample training data available. This reliance on high fidelity sensors creates a tradeoff: to make a smart space understand you, you have to give up some privacy. In recent years, the research community has worked to shift this tradeoff by turning to "privacy-preserving" sensors like mmWave radars, thermal arrays, and motion sensors. These sensors observe enough of human behavior to be useful while revealing far less about the identity, appearance, or conversations of those being observed.

The machine learning models behind cameras and microphones scale because they are global: trained once on large labeled datasets, they transfer across environments with little adaptation. Privacy-preserving sensors do not have this property yet. Their signals are shaped by the physical environment, *e.g.*, the sequence in which a collection of motion sensors activates depends on furniture placement, sensor position, and room layout. The same person, performing the same routine, produces a different signal pattern in a different home. Without global models, the understanding that these systems need is manually reconstructed in each new environment. Deploying a privacy-preserving sensing system today burdens users with hands-on involvement at every stage of the pipeline: labeling training data, defining which activities to detect, authoring rules that synthesize higher-level context, and navigating purpose-built interfaces to access insights. It is this per-environment human effort that prevents privacy-preserving sensing from moving beyond controlled research settings.

The thesis behind this dissertation is that the user burden required to set up and use context-aware sensing systems, from providing training labels to navigating purpose-built interfaces, can be lowered by opportunistically engaging a more capable resource with appropriate scaffolding. At each stage where a conventional pipeline demands human involvement, a more capable resource, whether a temporarily deployed camera, a vision-language model, a formal ontology, or a large language model, supplies the missing understanding, and scaffolding constrains how that resource operates, keeping its contributions predictable and bounded. In some cases the resource is temporary, assisting during an initial setup phase and then departing. In others it remains but in a bounded role, reasoning over processed outputs rather than raw sensor data. This shifts the user's role from constructing the system to confirming or correcting what it produces.

I demonstrate this principle at four successive levels of abstraction in the sensing pipeline. At the raw sensor signal level, deploying privacy-preserving sensors con-

ventionally requires a user to perform and label activities so the sensors can learn to recognize them using supervised ML models. **VAX** [154] reduces this burden through temporary privileged observation: a camera and microphone are paired with the privacy-preserving sensors during an initial setup phase, and their output, combined with off-the-shelf audio and video ML models, generates training labels automatically; once the sensor models are trained, the privileged sensors are removed. At the vocabulary level, any sensing system deployment requires someone to specify which activities the system should detect, but the right vocabulary for a given space depends on what the occupant does and what activities the sensors can actually distinguish. Defining this vocabulary upfront is difficult as occupants may not anticipate which of their routines matter, and developers cannot predict what a particular sensor configuration will be able to capture. **OrganicHAR** [155] addresses this by discovering activities directly from "privacy-preserving" sensor data, identifying recurring patterns in privacy-preserving sensor streams and using a Vision Language Model (VLM) during brief key moments to understand what those patterns represent. At the context level, the burden shifts from detecting activities to interpreting what they mean together: someone typing, talking, and sitting could be in a meeting, on a phone call, or doing focused work, and distinguishing these conventionally requires hand-authored rules specific to each environment. **TAO** [36] replaces these rules with a combination of a formal ontology that models how activities compose into contexts and an unsupervised deep temporal clustering pipeline that discovers context patterns directly from activity streams. The ontology provides structured, transferable domain knowledge that does not need to be rewritten for each environment, and the temporal pipeline learns recurring patterns without supervision. At the interaction level, a system that understands sensor data, activities and context is only useful if occupants can access what it knows. **BuildingChat** replaces purpose-built interfaces and bespoke apps with a conversational layer in which a large language model assembles analytical pipelines from pre-verified operators on the fly. Here the language model operates in a bounded role, orchestrating verified operators rather than using sensor data directly.

Individually, these four contributions each lower the user burden at one level of the sensing pipeline. Together, they sketch a path toward an end-to-end system that bootstraps much of its own understanding when deployed in a new environment, rather than requiring an occupant to reconstruct that understanding by hand at every stage. Combining them into a single self-configuring deployment is the next step this work makes possible. What the four systems establish is that the per-environment effort at each stage, from labeling training data to navigating purpose-built interfaces, is not inherent to privacy-preserving sensing but can be carried by a more capable resource under appropriate scaffolding. This is a step toward the longstanding vision of physical spaces that understand the people in them without asking those people to give up their privacy.

Acknowledgments

The work in this thesis carries my name, but it was shaped, supported, and sustained by far more people than a single author line can capture. Yuvraj, my advisor, your help, support, and guidance have been pivotal at many points in my journey. Thank you for nurturing my curiosity, shaping a systems person out of a data scientist, and reinforcing a discipline I carried through every project: build things that work in the real world. I also learned, or rather internalized, the quote "Do not let perfect be the enemy of good" from you, and it has been a belief I have carried with me since. Mayank, you have been a co-advisor to me in every sense but title. I always valued our difficult conversations, the ones that pushed me to think beyond the immediate problem and be a little more forward-looking, both in research and in life. You taught me to believe a little more in myself, take a stand, and find the right attitude to parse failures. Perhaps the most liberating lesson I took from you was that in research, results earn their own legitimacy. If your way works, nobody asks how you got there.

To Amy, Tricia, Chris, and John, working with you on Edusense was where I first engaged seriously with the human side of research, and it shaped how I thought about every project that followed. My committee members, Andrew and Ranveer, thank you for the kind of feedback that is hard to hear in the moment and impossible to ignore afterward. Ranveer, you were also my team leader during my internship at Microsoft Research, which makes your perspective on the final product particularly meaningful. Kevin and Alex from the same team, thank you for staying engaged with my research well after the internship ended; Kevin, your depth in systems thinking pushed me to broaden my own, and Alex, you shaped the direction of the internship in ways that carried forward into my later work. Parag Singla, my advisor at IIT Delhi, thank you for encouraging me to pursue this path. Your advice on how to approach difficult choices and face challenges with the right attitude stayed with me well beyond undergrad.

Monika, my wife, you have been my biggest champion for almost a decade now. You have stood by this journey with a conviction that often exceeded my own. A PhD tests every relationship around it, and yours is the one that never wavered. You always had a fresh perspective when I was too deep in the work to see clearly, and during the hardest stretches, you quietly made space for me, keeping everything else moving so I could focus. To my father, you are the reason I think the way I do. Your explanations always came as stories, and I say this only half-jokingly: my personality feels like the culmination of every story you have told me. I am still learning something new from you every day. To my mother, you were the one who worried about me, not the work, and sometimes that quiet concern was exactly the anchor I needed. My brother Shubhendu, it is rare to find someone who shares your curiosity about both life and work, and rarer still when that person is your younger brother. You have always been candid with me, always kept me honest, and always had my back. Nitish, my closest friend, you have been a constant through this entire journey. There are people you

turn to when you do not know who to turn to, and you have been that person for me. We grew up together over the last decade, and somewhere along the way we became each other's counterweight. You brought discipline where I had none, steadiness when I lost mine, and showed up without being asked during the moments I could not think straight. The same goes for my friends from back home, who have been championing me from thousands of miles away without ever losing touch.

My seniors in the lab shaped me in ways that no coursework could have. Han, you helped me build research primitives I still carry through all my work, and your composure taught me to stay calm when things got difficult. Sudershan, you opened my eyes to the depth of embedded systems and helped lay the groundwork for my early years in the PhD. Haojian, your positivity and grit modeled the attitude a researcher should carry, and I tried to pass that forward to my own juniors. My peers and fellow travelers in the lab deserve their own paragraphs, but I will settle for a sentence each. Riku, you had a way of taking my scattered ideas and pointing them somewhere useful. Vimal, you and I shared a curiosity for the parts of research that never make it into a paper. Your projects always had this quality of just making cool things that work seamlessly, which is exactly why we clicked. Haozhe, we shared enough deadline nights and large-scale data collection dreads to last a lifetime, and I have yet to meet someone who outworks you. Anu, you have the rare ability to be curious about literally everything and make everyone around you feel like they should be too. Ben, you have the kind of security and privacy instinct I wish came to me as naturally. Every system we built got cleaner because you were in the room. I would like to thank the many others across SynergyLabs, SmashLabs, and FigLab for the diversity of opinions, for making this journey something worth showing up for, and for building a community I feel lucky to have been part of.

Six years is a long time, and Pittsburgh gave me friendships I did not expect and cannot imagine doing without. Aigerim, you are one of my closest friends. Our curiosity about how the world works turned every long walk and every wild idea into something worth remembering. Aditya and Sujeet, you were my first real crew at CMU. For two and a half years, if there was a bad idea worth trying, we tried it. I would not trade that stretch for anything. Sonali, I met you almost by accident, and you became the anchor for everything fun in my third and fourth years. Parv and Keerti, you showed up at the right time. The final stretch of a PhD is its own kind of endurance test, and you were the people I turned to for everything, whether that meant a serious conversation, a trip, or a party that went too late.

To the Autonomous team, especially Ricardo, Adriano, and Ana, thank you for the close collaboration and for managing deployments across three countries. Working with you reminded me why this research matters, and deploying with people living with dementia was the most grounding experience of my PhD. Finally, Connie, Alisha, and the staff and faculty at S3D, thank you for making the administrative side of a PhD feel manageable rather than overwhelming. You were always prompt, always patient, and always helpful, even when I was not the easiest person to deal with.

To anyone reading this work, if it reads like it was written by someone who enjoyed and cherished the process, it is because of the people listed here.

Contents

1 Introduction **1**
- 1.1 The User Burden in Context-Aware Sensing . . . 2
- 1.2 Towards Self-Configuring Systems for New Environments . . . 4
- 1.3 Thesis Contributions and Organization . . . 6

2 Background and Related Work **9**
- 2.1 Sensing Infrastructure for Smart Environments . . . 9
  - 2.1.1 Building-Scale Sensing Deployments . . . 9
  - 2.1.2 Privacy-Preserving Sensing Modalities . . . 10
  - 2.1.3 Environment Specificity and the Data Challenge . . . 12
- 2.2 Human Activity Recognition . . . 13
  - 2.2.1 Audio and Video-Based Activity Recognition . . . 13
  - 2.2.2 Supervised HAR with Privacy-Preserving Sensors . . . 14
  - 2.2.3 Approaches to Reducing the Data Burden . . . 14
- 2.3 Foundation Models . . . 16
  - 2.3.1 Vision-Language Models . . . 16
  - 2.3.2 Large Language Models . . . 17
- 2.4 Context-Aware Computing . . . 17
  - 2.4.1 Knowledge-Driven Approaches . . . 18
  - 2.4.2 Data-Driven Approaches . . . 18
  - 2.4.3 Hybrid Approaches and Remaining Gaps . . . 19
- 2.5 Conversational Interfaces for Smart Environments . . . 19
  - 2.5.1 The Fragmentation Problem . . . 20
  - 2.5.2 Conversational Interaction as a Unifying Paradigm . . . 20
  - 2.5.3 Challenges Specific to Building Sensor Data . . . 21

3 Bootstrapping Activity Recognition from Audio and Video Models **23**
- 3.1 Introduction . . . 23
- 3.2 VAX Sensing Platform . . . 25
- 3.3 System Design . . . 26
  - 3.3.1 Study Design and Data Collection . . . 27
  - 3.3.2 Activity Recognition Using Off-the-shelf A/V Models . . . 28
  - 3.3.3 Improving A/V Labels with Privacy-Preserving Sensor Data . . . 30
  - 3.3.4 Training Models for Privacy-Preserving Sensors . . . 31

3.3.5 End-to-End VAX Pipeline . . . 33
3.4 Evaluation and Results . . . 34
3.4.1 Performance of the A/V Pipeline . . . 34
3.4.2 Performance of the VAX Pipeline . . . 35
3.4.3 Comparison with Baseline Approaches . . . 37
3.5 Discussion . . . 38
3.6 Summary and Looking Ahead . . . 40

4 Discovering Activities from Sensor Patterns 41
4.1 Introduction . . . 41
4.2 Sensing Infrastructure . . . 43
4.3 System Design . . . 45
4.3.1 Featurization . . . 46
4.3.2 Key Moment Identification . . . 46
4.3.3 Label Discovery . . . 48
4.3.4 Semantic Granularity Control . . . 50
4.3.5 Training Activity Recognition Models . . . 51
4.4 Data Collection . . . 52
4.5 Evaluation . . . 52
4.5.1 Performance of the Label Discovery Pipeline . . . 53
4.5.2 Performance of Activity Recognition Models . . . 55
4.5.3 Real-World Deployment . . . 60
4.6 Discussion . . . 62
4.6.1 From Discovery to Customization . . . 62
4.6.2 Applications in Healthcare and Assistive Technologies . . . 63
4.6.3 Privacy Considerations During Training . . . 64
4.6.4 Limitations and Future Directions . . . 64
4.7 Summary . . . 65

5 Synthesizing Context from Activity Patterns 67
5.1 Introduction . . . 67
5.2 Positioning Against Prior Work . . . 69
5.3 System Architecture . . . 70
5.3.1 Ontological Pipeline . . . 71
5.3.2 Temporal Clustering Pipeline . . . 73
5.3.3 Combining the Pipelines . . . 78
5.4 Implementation . . . 78
5.5 Evaluation . . . 79
5.5.1 Evaluation Setup . . . 79
5.5.2 RQ1: Evaluation of the Ontological, Temporal, and Hybrid Pipelines . . . 80
5.5.3 RQ2: Comparison with Prior Approaches . . . 83
5.5.4 RQ3: Evaluation in a Real-World Deployment . . . 84
5.6 Application: Context-Driven Wellness Metrics . . . 86
5.6.1 Wellness Metrics for Productivity . . . 87

5.6.2 Wellness Metrics for Stress . . . 87
5.6.3 Significance of Context for Wellness Applications . . . 88
5.7 Discussion and Limitations . . . 89
5.8 Summary . . . 90

6 Natural-Language Interaction with Building Sensor Infrastructure 93
6.1 Introduction . . . 94
6.2 Positioning Against Prior Work . . . 96
6.2.1 Querying Building and IoT Sensor Data . . . 97
6.2.2 Access Control for Shared Sensor Data . . . 98
6.2.3 Research Gap . . . 98
6.3 Soliciting User Needs from Smart Buildings . . . 99
6.3.1 Method . . . 99
6.3.2 Findings . . . 100
6.3.3 Design Implications . . . 101
6.4 BuildingChat System Design . . . 102
6.4.1 Smart Building Infrastructure . . . 103
6.4.2 Query Understanding and Reformulation . . . 104
6.4.3 Query Execution through Composable Operations . . . 108
6.4.4 Response Generation . . . 110
6.4.5 Interface Design . . . 111
6.4.6 Implementation . . . 112
6.5 Design Iterations from Initial User Study . . . 112
6.6 Technical Evaluation . . . 113
6.6.1 Evaluation Setup . . . 113
6.6.2 Comparing BuildingChat with Baseline Approaches . . . 115
6.7 Usability Evaluation . . . 119
6.7.1 Method . . . 119
6.7.2 Overall User Experience . . . 119
6.7.3 User Experience of Privacy Boundaries . . . 121
6.8 Discussion . . . 122
6.8.1 Lessons from the Usability Study . . . 123
6.8.2 Design Implications for Privacy-Aware Building Interfaces . . . 123
6.8.3 Limitations and Future Work . . . 124
6.9 Summary . . . 124

7 Ongoing Work, Discussion, and Future Directions 127
7.1 Ongoing Efforts and Emerging Applications . . . 127
7.1.1 The Autonomous Project: Supporting People Living with Dementia . . . 127
7.1.2 BuildingChat in TCS Hall . . . 129
7.1.3 Toward an Integrated Home Deployment . . . 130
7.2 Discussion . . . 130
7.2.1 What the Bootstrapping Principle Achieved . . . 130
7.2.2 The Pipeline as a Whole . . . 131

7.2.3 Longitudinal Robustness . . . 132
7.2.4 The Limits of Reactive Interaction . . . 132
7.2.5 Sensor Placement . . . 133
7.3 Future Directions . . . 133
7.3.1 Edge-Deployed Vision-Language Models . . . 133
7.3.2 System-Guided Sensor Placement . . . 134
7.3.3 Continuous Adaptation and Concept Drift . . . 134
7.3.4 LLMs as Dynamic Context Reasoners . . . 134
7.3.5 The Operator-Composition Paradigm Beyond Buildings . . . 135
7.3.6 Healthcare Deployments as a Proving Ground . . . 135
7.3.7 The System-Initiated Conversation . . . 136
7.4 Societal Impact . . . 136

A Supplementary Material for OrganicHAR **139**
A.1 Featurization Pipelines for Various Sensing Modalities . . . 139
A.2 Prompts for LLMs and VLMs . . . 140
A.3 VLM Variability and Configuration Analysis . . . 141
A.4 Activity Confusion Analysis (Other Sensor Configurations) . . . 143

B Supplementary Material for TAO **145**
B.1 Ontology Details . . . 145
B.2 Dataset Characteristics . . . 146
B.3 Detailed Scenarios for Real-World Study . . . 147
B.4 Details on Ground Truth Annotations . . . 147

C Supplementary Material for BuildingChat **151**
C.1 LLM Prompts . . . 151
C.2 Formative Study Interview Script . . . 151
C.3 Template Library Details . . . 153
C.4 Test Query Generation Details . . . 154
C.4.1 Generation Procedure . . . 154
C.4.2 Representative Prompt . . . 155
C.4.3 Example Test Cases . . . 155
C.5 Design Iteration Study Interview Script . . . 155
C.5.1 Introduction and Consent . . . 156
C.5.2 Establishing Persona . . . 157
C.5.3 Set up BuildingChat . . . 157
C.5.4 Walking Through Imagined Scenarios . . . 157
C.5.5 Overall Reactions and Conclusion . . . 158
C.6 Usability Evaluation Interview Script . . . 158
C.6.1 Introduction and Consent . . . 158
C.6.2 Establishing Persona . . . 159
C.6.3 Questions about Overall Experience and Insights . . . 159
C.6.4 Using BuildingChat for Specific Scenarios . . . 159

C.6.5 Using BuildingChat for User-Generated Questions 159
C.6.6 Questions about Usability 160
C.6.7 Questions about Scenarios Where BuildingChat Cannot Answer Questions 160
C.6.8 SUS Survey 161
C.6.9 Wrap-Up 161
C.7 Usability Evaluation Hypothetical Scenarios 161

Bibliography **165**

List of Figures

2.1 Visualizing information across various activities from various privacy-preserving sensors. (a) shows the movement range and radial velocity of moving objects using a Doppler sensor and object distance and angle using a Lidar sensor in the sensor plane across a sequence of activities happening in the living room. (b) shows snapshots of Thermal and Sound Localization at a given timestamp for various kitchen activities. 11

3.1 VAX hardware apparatus. The Doppler radar, thermal sensor, and Mites board are mounted vertically, while the Lidar and Micarray are oriented horizontally. The setup sits on a box housing a combined USB and Ethernet switch that routes data to an Intel NUC compute node. 27
3.2 High-level architecture of the VAX training and inference pipeline. During the training phase, VAX uses a set of off-the-shelf audio and video models to recognize activities and generates labels that are used to train privacy-preserving sensor models. Once trained, the A/V sensors are removed and the trained models on the privacy-preserving sensors are deployed for activity recognition. 27
3.3 Building the VAX A/V model from data collected in reference homes. A/V data (X^h) from reference homes are passed through off-the-shelf models deployed locally, and the resulting raw labels (L_k) together with ground-truth VAX activity labels ($y^h \in A$) are used to train the global VAX A/V model. 29
3.4 Generating activity labels in new homes using the trained A/V model and support models from privacy-preserving sensors. 29
3.5 2-D t-SNE embedding of clusters from Lidar data. Dark colored points represent clustered data; shaded regions highlight dense areas for different activities. Activities like Walking and Drinking/Eating are clearly separable, while Coughing and Vacuum overlap within a single cluster. 30
3.6 2-D tSNE embedding of clusters from Thermal data. Thermal signatures show good separation for activities like Coughing and Vacuum, which overlap in the Lidar feature space, illustrating the complementary nature of different modalities. 30
3.7 Improvement in activity detection rate using support models from individual privacy-preserving sensors (Lidar, Thermal, Doppler) and from all three combined (Lidar+Thermal+Doppler). Activities like Drinking/Eating and WashingDishes benefit substantially from sensor-based support. Activities like Exercising and HairDryer show no improvement, as the A/V pipeline already detects all instances with high confidence for these activities. 32

3.8 A/V pipeline: Accuracy, Precision, and Recall for activity recognition across all participants on detected activities (75% of all instances). The A/V pipeline performs consistently well across homes, with an average Accuracy of 90.5%, Precision of 89.5%, and Recall of 90.5%. The A/V model for each participant is trained in a leave-one-home-out fashion. 34

3.9 Privacy-preserving sensor pipeline: Accuracy, Precision, and Recall for activity recognition across all 17 activities for all participants. We observe an average Accuracy of 84%, Precision of 84%, and Recall of 84%, with consistent performance across homes. Variation across participants reflects differing amounts of noise in the A/V-generated labels. 34

3.10 A/V pipeline: Accuracy and Detection Rate as the number of reference homes increases. Activity detection accuracy (correct predictions out of detected activities) remains high even with few reference homes, while the detection rate (number of activities detected with high confidence) increases monotonically with more reference homes. 35

3.11 F1-Score for activity detection across participants for three conditions: (a) the A/V pipeline alone, (b) the VAX privacy-preserving sensor pipeline trained on A/V labels only (no user input), and (c) the VAX pipeline with one user-provided label for each undetected activity. In the A/V pipeline, most activities are detected across all homes, but activities like Baking, FridgeOpen, and CookingOnStove are missed in most homes due to high inter-activity confusion leading to low confidence. The VAX pipeline with A/V labels only (Figure 3.11(b)) performs well on detected activities but shows no detection for activities absent from the A/V labels. With a single user input per undetected activity (Figure 3.11(c)), VAX performs well across all activities, showing that the privacy-preserving sensors' classification capability is limited only by the availability of at least one labeled instance per activity. 36

3.12 Activity confusion matrices averaged across all participants for (a) the A/V pipeline and (b) the privacy-preserving sensor pipeline. The A/V pipeline performs well for activities with descriptive sound signatures (WashingDishes, Blender, ToiletFlushing) but confuses activities with similar pose patterns, such as FridgeOpen and Baking (both involve opening and closing doors). The privacy-preserving sensor pipeline detects most activities well due to differentiating signatures in one or more sensors, with the weakest performance on bathroom activities like HairBrush, HairDryer, and ShaverInUse, where limited body movement leaves the system reliant on small Doppler variations. 37

3.13 Privacy-preserving sensor pipeline: F1-Score at the activity level from individual sensors. Doppler, Thermal, and Lidar show the highest F1-Scores across most activities. The Micarray performs well for activities with distinctive sound signatures. Environmental sensors and PIR motion contribute minimally across all activities. 38

3.14 Comparing VAX with two baseline approaches. Without any user input (left), VAX substantially outperforms the baseline on both activities detected by the A/V pipeline (84% *vs.* 41%) and across all activities (74% *vs.* 38%). With user input (right), VAX achieves slightly better performance with considerably less labeling effort (∼2 labels/home *vs.* 17 labels/home) compared to baselines trained directly on privacy-preserving sensor data. 39

4.1 OrganicHAR's sensor-first workflow for activity discovery. Privacy-preserving sensors identify recurring signal patterns and temporal anomalies (Step 1), which are used to select *key moments* for targeted VLM analysis that generates natural language activity descriptions (Step 2). These descriptions are then organized into hierarchical activity labels at adjustable granularity levels (Step 3). Unlike prior approaches that require predefined activity sets, fixed granularity, and continuous video, OrganicHAR discovers activities from sensor patterns, adapts granularity to sensor capabilities, and processes only key moments. 42

4.2 Privacy-preserving sensor data from the OrganicHAR sensing platform. (a) shows snapshots of the low-resolution thermal array and depth array at a given timestamp for various kitchen activities. For depth visualization, pixels grow darker when the user is closer to the sensor. The Doppler and Lidar visualizations for this platform follow the same patterns described in Chapter 2 (Figure 2.1). (b) shows our hardware rig, which integrates a 2D Lidar, Doppler radar, OAK-D Lite stereo camera, and thermal sensor into a compact form factor. 44

4.3 Overall architecture of OrganicHAR. Raw sensor signals from different hardware configurations (Section 4.2) are featurized for each sensor separately (Section 4.3.1), and used to identify *key moments* (Section 4.3.2) for targeted video processing using VLMs. The raw descriptions are then converted to discrete activity labels using LLMs (Section 4.3.3), and further refined using a relaxation parameter λ (Section 4.3.4). The final activity labels are used to train activity recognition models (Section 4.3.5) for deployment. 45

4.4 Kitchen environments used in our study: (left) Kitchen 1 with compact galley layout, (middle) Kitchen 2 with island counter, and (right) Kitchen 3 with integrated appliances. 51

4.5 Percentage of video data requiring VLM analysis across sensor configurations. Our approach processes only 9–11% of total video data, demonstrating efficiency compared to continuous monitoring. 53

4.6 Discovered activity labels across three semantic granularity settings: Conservative ($\lambda = 0.4$, left), Balanced ($\lambda = 0.3$, middle), and Relaxed ($\lambda = 0.2$, right). The flow width represents the proportion of data with each label, showing how broader categories branch into more fine-grained activities as granularity increases. For example, the "Counter Activity" broad category (left) differentiates into "Food Preparation," "Cleaning Counter Area," and other activities in the Balanced setting. 54

4.7 Per-participant accuracy across three sensing configurations. Kitchen 1 participants (P1–P4) show consistently high accuracy even with basic ambient sensing, Kitchen 2 participants (P5–P8) show greater degradation with finer granularity, and Kitchen 3 participants (P9–P12) benefit most from wearable IMU. Horizontal dashed lines indicate average performance per granularity setting. 57
4.8 Per-participant F1 scores across sensing configurations, revealing sharper performance drops than accuracy metrics with increasing granularity. F1 scores for the Relaxed setting show particularly large variance between participants and kitchens, illustrating the impact of environmental layout on the precision-recall balance in fine-grained activity recognition. 57
4.9 Confusion matrices showing HAR performance using basic ambient sensors across three granularity settings. As granularity increases from Conservative (a) to Relaxed (c), more fine-grained activities emerge, revealing specific confusion patterns between semantically or spatially related activities. 58
4.10 Incremental training analysis of OrganicHAR: (a) count of VLM queries after incorporating each new training session, and (b) recognition accuracy measured on all future sessions. Error bars represent 95% confidence intervals across users. 59
4.11 Kitchen environments used in real-world home deployments: (Home-1) compact galley layout from overhead diagonal perspective, (Home-2) medium-sized kitchen with distinct functional zones, (Home-3) narrow galley configuration from elevated angle, (Home-4) spacious kitchen with multiple countertops from horizontal perspective, and (Home-5) linear wall-mounted kitchen with integrated appliances from horizontal viewpoint. 60
4.12 Overall recognition accuracy in real-world home deployments using leave-one-session-out cross-validation with Balanced granularity and *Ambient(Advanced)+Wearable(IMU)* configuration. The horizontal dashed line indicates average accuracy (74.2%) across all participants. 61
4.13 Incremental training trajectories showing forward-looking evaluation, where the system trains on sessions 1 through N and tests on all remaining future sessions ($N + 1$ onwards). Performance patterns differ from Figure 4.12 due to temporal dependencies and chronological activity pattern evolution. 61
4.14 Prototype interface for customizing sensor configurations and activity labels. The percentage values show how well each deployed sensing configuration can predict activities when compared with discovered activity labels. Users can toggle sensors on or off, edit discovered labels, and add new ones. 62

5.1 Daily activities (*e.g.*, *A: typing*, *A: running*, *A: chewing*) can be translated into semantically richer contexts (*e.g.*, *C: Office work*, *C: Exercise*, *C: Having meal*). Real-world activity patterns are complex, with activities happening sequentially, in parallel, and interleaved, requiring an approach that can handle all three. The TAO system combines ontological and temporal clustering components to convert these activity patterns into rich contexts, which can then enable applications such as wellness monitoring. 68

5.2 Overview of the TAO system architecture. TAO leverages OWL-based ontologies and temporal clustering approaches to identify context from the stream of activities obtained from Human Activity Recognition (HAR) systems. The contexts detected by TAO are then sent to wellness applications which infer productivity and stress. 71

5.3 Overview of the TAO ontological pipeline. 72

5.4 SPARQL query to infer *C: Office work* from parallel activities *A: typing* and *A: talking*. 73

5.5 SPARQL query to infer *C: On a phone call* from sequential activities with time constraints. 73

5.6 An example of context representation (five activities only) for a 2-minute stacking window (t) and a 10-minute lag window (T). 74

5.7 Architecture of the Temporal AutoEncoder (TAE). 75

5.8 Overview of the TAO temporal pipeline. The Representation Trainer (a) pretrains a TAE encoder-decoder to learn dense context embeddings from sparse input representations. The Cluster Trainer (b) uses the TAE encoder and learned embeddings to discover the optimal context cluster count and their sparse representations. The Context Labeler (c) uses the ontology to provide labels to the generated context clusters. 75

5.9 A representative figure illustrating the evaluation of context prediction accuracy using the Jaccard Coefficient (JC). Predicted contexts ($Context_{Det}$) are compared against ground truth ($Context_{GT}$). The JC ranges from 0 to 1: 0 when there is no overlap between ground truth and detected contexts, and 1 when they are identical. JC penalizes methods that detect contexts not present in the ground truth (increasing the union) and that miss contexts present in the ground truth (decreasing the intersection). 80

5.10 Comparing the accuracy (JC), precision, and recall of context detection for TAO's temporal pipeline across different *lag window* durations on the *Casas* and *ExtraSensory* datasets. The approach achieves high accuracy (65% to 75%) for short lag windows (5 and 10 minutes) compared to longer windows (30 and 60 minutes). Recall of context detection is consistently higher than precision across both datasets. 81

5.11 Comparing the accuracy (JC), precision, and recall of context detection for TAO's temporal pipeline as the training data (in days) increases. For the *Casas* dataset, the approach shows a consistent increase in accuracy, precision, and recall as training data grows. For the *ExtraSensory* dataset, the approach achieves high accuracy within two days of training data. 82

5.12 Comparing the distribution of predicted contexts learned over 45 days (1/7/14/30/45 days) for *Casas* and six days (1/2/4/6 days) for *ExtraSensory*. In *Casas*, 30 days of training data enables the pipeline to detect a wide range of contexts at 72% JC. For *ExtraSensory*, several contexts are detected within two days of training data. . 83

5.13 Comparing the accuracy (JC), precision, and recall of the TAO pipeline. Results are shown for the ontological pipeline alone, the temporal pipeline alone, and TAO's hybrid combined approach. The temporal pipeline achieves approximately 75% accuracy across both datasets. The ontological pipeline achieves higher accuracy (87%) on *Casas* but lower accuracy (57%) on *ExtraSensory*. The combined approach achieves high accuracy (80% and 87%) and high precision (81% and 91%) across both datasets. 84

5.14 Comparing the accuracy (JC), precision, and recall of TAO's context detection in the real-world user study. Activity inferences are obtained either from manually annotated video data or from the Mites platform [139]. TAO detects contexts at higher accuracy (JC: 72%, precision: 83%, recall: 69%) using video-annotated activities. Accuracy drops (JC: 22%, precision: 36%, recall: 24%) when using Mites activity predictions, reflecting inaccuracies in the underlying activity detection pipeline. 86

5.15 Productivity score for six users from the *ExtraSensory* dataset over six days. . . . 87

5.16 Stress score for six users from the *Casas* dataset over five weeks. 87

5.17 Comparison of productivity and stress scores generated using context predictions from TAO, ground-truth context labels, and activity information alone, for a single user (user 3) in the *ExtraSensory* dataset. Both the productivity scores (avg/week = 16.57) and stress scores (avg/week = 30.12) from TAO's predictions are similar to the ground truth (productivity avg/week = 14.41, stress avg/week = 30.17), while scores generated from activity information alone diverge substantially (productivity avg/week = 29.65, stress avg/week = 46.25). 89

6.1 Overview of BuildingChat. (a) The current state of occupant access to building data requires consulting separate applications for each subsystem. (b, c) BuildingChat provides a single conversational entry point that decomposes natural language queries into composable analytical operations, enforces access control, and produces contextual recommendations. 95

6.2 System overview for BuildingChat. **Stage 1 (Query Understanding and Reformulation):** User queries are parsed to extract intent, spatial tags, and temporal scope (Section 6.4.2.1). The system derives access requirements and validates them against policies (Section 6.4.2.2). If the query is underspecified or unauthorized, the reformulation module (Section 6.4.2.3) proposes default interpretations or accessible alternatives, prompting the user for confirmation or clarification. **Stage 2 (Query Execution):** Once validated, the template composition module selects and arranges templates (Section 6.4.3.1) from the library into a Directed Acyclic Graph (DAG) (Section 6.4.3.2). The system retrieves aggregated sensor statistics based on the approved policy, executes template operators (Section 6.4.3.3), and generates a natural language response (Section 6.4.4). Yellow badges indicate LLM-powered components; other components execute deterministic logic. 102

6.3 Query processing traces illustrating four scenarios: (1) a well-specified query proceeds directly to execution, (2) an underspecified query is resolved through profile preferences or a proposed default, (3) an unauthorized query prompts the system to propose an accessible alternative requiring user consent, and (4) a query beyond the building's sensing capabilities ends with an explanation of available alternatives. 105
6.4 DAG compositions for different query types. Each panel shows how natural language queries decompose into template operations connected by typed data packets. (a) Simple measurements use a linear pipeline. (b) Comparisons create parallel branches that merge at a `spatial_compare` node. (c) Multi-criteria queries over N candidate locations produce parallel branches per modality, each processing N locations (indicated by stacked boxes). (d) Temporal queries produce dual outputs for time series and summarization. 110
6.5 The BuildingChat interface. (a) The empty-state screen guides query formulation through categorized suggestions. (b) Each response includes an expandable panel exposing the template pipeline and confidence metrics. (c, d) Inline visualizations complement text responses, ranging from comparative time-series charts to spatial heatmaps on floor plans. 111
6.6 Query accuracy across six language models. BuildingChat achieves the highest accuracy on every model (76–93%). Code Generation ranges from 8% to 79%, In-Context Data from 49% to 80%, and ReAct from 65% to 80%. 116
6.7 Accuracy by query category. BuildingChat maintains high performance across all query types. The advantage over baselines is most pronounced for trend queries, which require multi-step temporal reasoning. 116
6.8 Error type distribution across 1800 queries (300 $\times$ 6 models). Code Gen. is dominated by execution failures (30%), In-Context by context limits (14%), and ReAct by analysis errors (19%) and execution failures (16%). 116
6.9 Query latency breakdown. ReAct is the slowest (14s) due to iterative tool calls. BuildingChat and Code Generation achieve comparable latency (11s and 13s). In-Context is fastest (5s) but at the cost of higher token usage and context overflow errors. 117
6.10 Token usage per query. ReAct consumes 93K tokens on average (16$\times$ BuildingChat), driven by accumulating conversation history across iterations. BuildingChat, Code Generation, and In-Context Data remain in the 5–7K range. 117
6.11 Correctness of unauthorized query rejection for policy-aware baselines. All three achieve reasonable precision (76–89%) but low recall, with ReAct correctly rejecting only 26% of unauthorized queries. 118
6.12 Data leak rates by user profile for policy-aware baselines. ReAct leaks 46% of restricted queries for the more constrained user profile. BuildingChat has zero leaks by construction. 118

A.1 Confusion matrices showing activity recognition performance for *Ambient(Basic)+Wearable(IMU)* configuration across three granularity settings. 143

A.2 Confusion matrices showing activity recognition performance for *Ambient(Advanced)+Wearable(IMU)* configuration across three granularity settings. 143

B.1 A real-world timeline of different activities an individual performs from the ExtraSensory dataset and the corresponding ground truth and context predictions from the ontology. The annotations in the graph show different activity patterns, such as (a) sequential and (b) parallel activity patterns. 146
B.2 Examples of sequential and parallel activity pattern definitions in the ontology. . 147

List of Tables

4.1 Average accuracy and F1 scores (mean±std) of discovered activity labels compared to ground truth across three semantic granularity settings. Conservative ($\lambda = 0.4$) represents coarse-grained activities, Balanced ($\lambda = 0.3$) shows medium granularity, and Relaxed ($\lambda = 0.2$) captures fine-grained activities. 53

4.2 Average accuracy and F1 scores (mean±std) of the HAR models across sensor configurations and granularity settings. These models perform effectively despite being trained on organically discovered labels rather than predetermined categories. Relaxed setting results (shown in gray) represent exploratory findings illustrating current technical boundaries rather than recommended configurations. 56

5.1 Comparing the accuracy (JC) and total contexts detected by TAO's ontological pipeline with prior work on two real-world HAR datasets. Our approach detects different types of contexts (sequential and parallel) at significantly higher accuracy (72.8% on *Casas* and 53.9% on *ExtraSensory*). 80

5.2 Accuracy comparison (F1 score) of TAO's combined pipeline with prior approaches by Meditskos *et al.* [137] and Riboni *et al.* [177] for context detection on the *Casas* dataset [55]. Overall, TAO performs significantly better across a broader set of contexts. 85

5.3 Context-level performance for the real-world study, showing precision and recall for each context when using video-annotated activities and Mites activity predictions. 85

6.1 Comparison across capabilities required for conversational sensor data access in multi-stakeholder buildings. Rows 1–4 are building data systems; rows 5–6 are access control frameworks for conversational systems, for which sensor analytics and query composition are outside their design scope (—). ✗ indicates the capability is absent. 99

6.2 Query distribution from our formative study by role and analytical complexity. Point measurements retrieve a single sensor reading; cross-location queries compare or filter across multiple sensor groups; temporal queries analyze patterns over time; outside scope queries require capabilities beyond sensing (*e.g.*, wayfinding). 101

6.3 An illustration of tag types and values for sensor organization. 103

6.4 Query structure components extracted during parsing. 104

6.5 Data access policies used in our deployment. We included policies for personal data access, querying information about public spaces, and querying aggregates across larger sections of the building (which may include private offices). The `location` tag refers to the room number, `floor` and `cardinality` to approximate location in the building, and `visibility` and `locationType` to the type of space (*e.g.*, public corridor). 107
6.6 Time resolution constraints for non-personal data. We allow high-granularity access to more-recent data, and lower-granularity access to historical data. 107
6.7 Packet types defining data contracts between templates. 108
6.8 Template library derived from our formative study query patterns. 108

A.1 Label discovery performance across VLM models and semantic granularity settings. Detection rate represents the percentage of key moments that generate useful activity descriptions. 141
A.2 Impact of frame rate on label discovery performance using GPT-4.1. Detection rates remain consistent across frame rate settings. 142

B.1 Summary of the HAR datasets used for evaluation and their characteristics. . . . 147
B.2 Overview of the scenarios used in the real-world user study. We identify several common activity patterns in daily life from which context can be inferred and categorize them into different scenarios. 148
B.3 Example activities from the *ExtraSensory* dataset [207]. We manually annotate the context for these activities during our ground truth labeling process. 149
B.4 Amount of data annotated for ground truth of 14 different contexts across the *ExtraSensory* and *Casas* datasets. 149

C.1 Example test cases from the evaluation set, showing sampled parameters, computed ground truth, and generated natural language query for each category. . . . 156
C.2 Scenarios discussed in the design iteration study and usability evaluation interviews. Each scenario was presented to participants, who then formulated their own queries to BuildingChat. The "Expected Response" column indicates whether the scenario was designed to be answerable (Allowed), to involve private data the user should not access (Private data), or to request information not available in the system (No data). 162

Chapter 1

Introduction

We spend most of our waking hours inside buildings, and these spaces are increasingly instrumented with sensors [216]. A modern office building may have thousands of sensors monitoring temperature, humidity, occupancy, and air quality [34]. A home equipped with even a few consumer IoT products generates a continuous stream of data about its environment and the occupants [41]. Yet for all this instrumentation, the spaces we inhabit remain largely unaware of what is actually happening inside them. A motion sensor detects someone in the kitchen but not whether they are preparing a meal or just passing through. An office building tracks occupancy but cannot tell that someone has been in focused work for three hours and could benefit from a break. And when an occupant wants to know which available room is quietest, there is no straightforward way to ask. Bridging this gap between raw sensor data and actionable contextual understanding has been a central goal of ubiquitous computing research for over two decades [171].

The technology that has come closest to making spaces aware of their occupants' behavior relies on cameras and microphones. Given the availability of several large annotated datasets such as Kinetics [47], ActivityNet [84], and AudioSet [74], researchers have trained deep learning models for video-based action recognition [140], body pose estimation [135, 141], and sound-based activity detection [65, 117]. These models share a property that is central to their practical value: they are *global*. Trained once on diverse data, they generalize to new environments with minimal adaptation. A pose estimation model trained on internet videos works in a kitchen it has not been trained on. A sound classifier trained on AudioSet recognizes a running blender in an apartment it has never encountered. This transferability is what makes audio- and video-based activity understanding deployable at scale.

Consumers, however, are increasingly uncomfortable with always-on cameras and microphones in their homes and workplaces [5, 43, 68, 69, 99]. These concerns are not merely matters of preference, as high-profile incidents of compromised IoT devices have shown that sensor data collected in private spaces can be exposed or misused [19, 78, 184]. The result is a tradeoff at the foundation of the field: to make a smart space understand you, you have to let it watch you, thereby compromising privacy. In response, researchers have explored new sensing modalities that have enough signals and patterns to be useful for observing human behavior, while revealing far less about who is being observed. These privacy-preserving sensors, including mmWave Doppler radars, low-resolution thermal arrays, 2D lidars, motion sensors *etc.*, capture signals that are not personally identifiable yet still carry information about the activities of nearby

occupants [11, 27, 114, 115, 191].[1]

The privacy benefits are real, but they come at a cost. Unlike audio and video models, models built on these sensors do not generalize across environments. Their signals are shaped by the physical environment: the sequence in which a collection of sensors activates during a morning routine depends on furniture placement, sensor position, and room layout. The same person performing the same routine produces a different sequence in a different home. Without global models, the contextual understanding that these systems aspire to provide must be manually reconstructed in each new deployment. This reconstruction burdens users with hands-on involvement at every stage of the pipeline: labeling training data, defining which activities to detect, authoring rules to interpret those activities, and navigating purpose-built interfaces to access insights. It is this per-environment effort that has kept privacy-preserving context-aware sensing largely confined to controlled research settings.

The thesis behind this dissertation is that the user burden required to set up and use context-aware sensing systems, from providing training labels to navigating purpose-built interfaces, can be lowered by opportunistically engaging a more capable resource with appropriate scaffolding. The user's role shifts from constructing the system to confirming or correcting what it produces. The remainder of this chapter develops this claim. Section 1.1 describes how conventional sensing pipelines work and identifies the user effort each stage demands. Section 1.2 presents the broader scientific vision this dissertation pursues. Section 1.3 summarizes the four research contributions, and provides a roadmap of the dissertation.

1.1 The User Burden in Context-Aware Sensing

Building a context-aware sensing system with privacy-preserving sensors requires a series of design decisions, each of which demands human input specific to the target environment. These decisions determine what the system can perceive, what it can reason about, and what it can communicate to its users. To understand why it has remained confined to controlled studies [87] , it helps to trace what a deployment actually requires and where current solutions fall short.

The first requirement is training data. Privacy-preserving sensors such as mmWave radars, thermal arrays, motion sensors, and lidars produce time-series signals whose characteristics depend on the geometry of the space, the placement of the sensor, and the physical properties of the environment [115, 154]. A radar's Doppler signature for someone cooking in one kitchen will differ from the signature in another kitchen, because sensor distance, countertop layout, and reflective surfaces all shape the received signal. Models trained in one environment therefore do not transfer to another, and each new deployment requires its own labeled dataset, typically produced by an occupant or researcher performing and annotating activities over hours or days. Researchers have pursued several strategies to reduce this burden. Domain adaptation techniques synthesize training data for one modality from another: Ahuja *et al.* converted video into synthetic Doppler radar signals [11], and similar approaches have mapped video or audio to synthetic IMU data [113, 125]. Self-supervised learning methods learn representations from unlabeled sensor

[1]We adopt the term *privacy-preserving sensors* to distinguish these modalities from cameras and microphones. The term reflects their reduced information capture, not a formal privacy guarantee, and this dissertation does not evaluate their privacy properties.

data and fine-tune with a small number of labeled examples [59, 95]. These approaches are limited to a single target modality and a narrow set of activities [11], while self-supervised methods still require predefined activity sets and per-environment labeled samples for fine-tuning [95, 167].

The second requirement is an activity vocabulary, and with it, the granularity at which activities should be recognized. Before any data can be labeled, someone must decide not only what activities the system should detect, but how finely it should distinguish between them. In practice, both choices are made by system designers before deployment [171, 212]. But the right vocabulary and level of detail depend on the specific environment, the sensors deployed in it, and what the user wants. Consider a kitchen where one person uses a single countertop appliance to prepare tea, coffee, and hot chocolate: a radar placed across the room sees nearly identical motion patterns for all three, making "preparing a hot beverage" the finest granularity it can reliably support. In a different kitchen, where tea is made on the stove and coffee with a machine on a separate counter, the same radar can distinguish the two because they involve movement to different locations. The appropriate vocabulary is not just a function of what the occupant does, but of what the available sensors can perceive in that particular physical arrangement. Too fine-grained a vocabulary leads to poor accuracy when sensor signals are not distinguishable, and too coarse a vocabulary discards the detail that would make the system useful for applications. Researchers have explored unsupervised clustering methods that group similar sensor patterns without predefined labels [17, 54], and interactive learning strategies that refine models through user feedback [106]. Clustering can identify recurring patterns, but the resulting groups still require manual interpretation to assign meaningful names [88, 221], and offer no mechanism to select the granularity that matches sensor capabilities. What is missing is a way to discover, from the sensor data itself, both what activities occur in an environment and at what level of detail they can be reliably told apart.

The third requirement is a mapping from detected activities to meaningful context. Fine-grained activity detection produces a stream of low-level observations: typing, talking, sitting, walking, using the microwave. These observations are not directly useful because the same activity can participate in entirely different contexts. Talking while sitting at a desk likely indicates a meeting or a phone call; talking while eating in the evening likely indicates a social meal; talking while walking outdoors may be a phone call or a conversation during a commute. The activity is identical, but the context differs in each case. What determines context is the temporal structure of activities: which ones co-occur, in what sequence, for how long, and how they interleave. Cutting, chopping, and washing vegetables in sequence indicates preparation of a fresh meal; using the microwave and opening the refrigerator indicates reheating a pre-cooked one. Both take place in the same kitchen, but the contexts they represent are fundamentally different. Moreover, real-world behavior rarely unfolds as a clean sequence. People routinely overlap activities: answering a phone call while exercising, browsing email during a meal, or alternating between typing and conversation in a way that constitutes neither focused work nor a meeting [90]. Researchers have tackled this through ontology-based frameworks that define logical relationships between activities and contexts [49, 138, 209], temporal clustering methods [173, 230], and hybrid approaches combining both [108, 149, 175]. However, ontological rules are static and do not distinguish whether activities happen sequentially or in parallel, which is the very distinction that determines context. Hybrid methods improve coverage but remain bound to the patterns their authors anticipated, and the space of possible activity combinations is too vast and too

person-specific for any fixed rule set to cover.

The fourth requirement is an interface for a person to interact with smart infrastructure to derive insights. A system that detects activities and synthesizes context is only useful if occupants can ask questions of it and receive answers they understand. Today, this means either navigating application-specific dashboards, one for temperature, another for occupancy, a third for energy, each embedding assumptions about what questions users will ask [70, 81], or relying on developers to build bespoke applications against low-level sensor APIs, an undertaking that still answers only the queries it was written for. Any question that falls outside the designer's anticipation simply cannot be asked. What occupants want is closer to how they would interact with a knowledgeable colleague: describe a need in their own words and receive an answer drawn from whatever data is relevant, without knowing what sensors exist or how data is organized. Recent systems have begun applying language models to this problem [62, 119], but they route each query to a single analytical function and cannot compose multi-step analyses that many practical questions require. The challenge deepens in shared spaces like office buildings, where the same sensors serve multiple people. Here, privacy extends beyond what the sensors observe to what one person can learn about another through the system. A facility manager asking for average floor occupancy is making a reasonable operational query; an occupant asking whether anyone was in a specific room last night may be tracking a colleague's movements. The data is the same, but the privacy implications depend on who is asking, and should be handled on per-query basis.

What connects these four requirements is that each demands effort local to a particular environment and its occupants. The research community has made progress on each front, yet existing solutions address one stage while leaving others unchanged. Labels must still be collected in situ, even if fewer are needed. Vocabularies must still be chosen before deployment, even if clustering helps organize them. Context rules must still be authored for anticipated patterns. Interfaces must still be designed for expected queries. When any of these choices fail to match the reality of a particular environment, the system becomes unreliable, and correcting the mismatch requires further human effort. This compounding cost is why privacy-preserving context-aware systems, despite two decades of research demonstrating their technical feasibility, have not achieved broad, sustained deployment.

1.2 Towards Self-Configuring Systems for New Environments

Consider an elderly person living alone whose family worries about whether meals are being prepared regularly, or whether the stove is being left on [52]. Consider a patient recovering from surgery at home who needs to follow a structured routine of wound care and medication, and whose care team wants to know whether those routines are being maintained without installing cameras in the home [80]. Consider an office occupant who simply wants to find a quiet, comfortable room to work in, drawing on data from sensors they did not install and whose existence they may not even be aware of [81]. They represent real needs that have motivated this research and that we are actively pursuing in ongoing deployments: supporting people living with dementia in their homes through privacy-preserving activity monitoring [21], and enabling occupants of a university building instrumented with over 300 multi-modal sensors to query their environment through natural conversation [34]. What connects these applications is a shared requirement: the

system must understand what is happening in a specific environment, for specific people, using sensors that preserve their privacy, and it must do so without expecting those people to train or configure the system themselves.

With this dissertation, I put forward the vision that deploying a context-aware system in a new environment should require nothing more than choosing a set of sensors, giving the system a few days of silent observation in which it labels its own data and trains high-quality models, and then simply conversing with it to gain insights from what it can detect. The choice of sensors should be the occupant's: what they can afford, what they are comfortable having in their home or workplace, and what level of observation they are willing to accept. *Occupants should not have to surrender their privacy to benefit from context-aware sensing. And those who choose privacy-preserving alternatives should not need technical expertise to make them work.* The people who stand to benefit most from context-aware environments, elderly individuals living alone, patients managing recovery at home, occupants of complex shared buildings, are precisely the people for whom configuring a sensing system is least realistic [41]. After a brief observation period, the system should be able to initiate a conversation with the occupant: here is what I can detect in your space, here is what I cannot, what would be useful for you to know? Once this exchange establishes what the system should provide, it begins generating insights, periodically checking with the occupant whether its interpretations are correct, and refining its models when they are not. The system acts less like a piece of infrastructure that must be programmed and more like a curious partner that learns through dialogue, one that can be configured with nothing more than natural language.

What makes this vision technically tractable, rather than aspirational, is a convergence of capabilities that did not exist a decade ago. Cameras and microphones, temporarily deployed alongside privacy-preserving sensors, can observe a broad range of human activities that no radar or thermal array can distinguish on its own. Vision-language models can describe scenes in natural language with enough fidelity to serve as temporary annotators. Formal ontologies can encode how activities compose into higher-level contexts in a form that transfers across environments without per-deployment authoring. And large language models can reason about analytical operations over structured data, composing queries and interpreting results in ways that previously required a human analyst. Each of these is a more capable resource that can supply understanding a conventional pipeline would demand from the user.

But capability alone is not enough; each resource also requires scaffolding that constrains how it operates. An ensemble pipeline reconciles the noisy outputs of audio and video models into reliable training labels. A key-moment detection mechanism limits what a vision-language model sees to brief, informative segments rather than continuous surveillance. A temporal clustering pipeline grounds an ontology's abstract knowledge in patterns actually observed in sensor data. And a set of pre-verified operators ensures that a language model orchestrates analyses without ever touching raw sensor readings. In some cases, the resource is temporary: a camera assists during an initial training phase and is then removed, or a vision-language model processes brief video segments during setup and does not run afterward. In others, the resource remains but operates in a bounded role, reasoning over processed outputs rather than raw data. In each case, the sensitive data stays within the privacy-preserving confines of the system.

This pattern, opportunistically engaging a more capable resource and bounding it with appropriate scaffolding so that it either departs after bootstrapping or remains in a constrained role, is

the recurring design principle of my dissertation. It repositions the human from the role of system constructor to that of a participant who confirms, corrects, or refines what the system produces. The four contributions presented in this dissertation each demonstrate that this principle is viable at one stage of the sensing-to-interaction pipeline, and that the resulting automated approach can reduce the effort required to deploy context-aware systems in new environments.

1.3 Thesis Contributions and Organization

This dissertation presents four systems, each reducing the user effort for one or more aspects of context-aware sensing system discussed in Section 1.1.

Training labels: In a conventional sensor-based deployment, someone must perform and annotate activities in each new environment so that privacy-preserving sensors can learn to recognize them. VAX (Chapter 3) [154] eliminates most of this burden by temporarily pairing a camera and microphone with the privacy-preserving sensors. We then use off-the-shelf audio and video ML models to generate training labels automatically, and once the sensor models are trained, the camera and microphone are removed. Across ten homes and 17 activities, VAX matches a fully supervised baseline while requiring roughly one-eighth of the manual labeling effort.

Activity vocabulary: VAX assumes a predefined set of activities, but deciding what to detect in a given space is itself a specification problem: the right vocabulary depends on what the occupant does and what the sensors can distinguish. OrganicHAR (Chapter 4) [155] addresses this by discovering activities directly from sensor data. It identifies recurring patterns in privacy-preserving sensor streams, uses a vision-language model during brief key moments to understand what those patterns represent, and produces activity labels at a granularity matched to the sensors' actual capabilities. In a study with 12 participants, OrganicHAR discovers 4 to 8 meaningful activity categories per environment with 87% average accuracy, while requiring 10% of the video to be processed by VLMs for significant cost savings and improved privacy.

Context: Detecting individual activities is not the same as understanding what they mean together. Someone typing, talking, and sitting could be in a meeting, on a phone call, or doing focused work, depending on how these activities unfold over time. TAO (Chapter 5) replaces hand-authored rules with a formal ontology that models how activities compose into contexts and an unsupervised temporal clustering pipeline that discovers context patterns directly from activity streams, handling sequential, parallel, and interleaved activities that rule-based approaches cannot anticipate. Our evaluation with eight participants shows that TAO [36] can produce context-based wellness assessments within 1.1% of ground truth, compared to 30% for a baseline.

Interaction: A system that detects activities and synthesizes context is only useful if occupants can easily access what it knows. BuildingChat (Chapter 6) provides a conversational interface through which occupants can query their building's sensor infrastructure. It assembles analytical pipelines from pre-verified operators on the fly, ensuring that responses are predictable and access control is enforced by construction rather than by prompting. BuildingChat achieves 88% query accuracy while eliminating the unauthorized data leakage that code-generation and agent baselines exhibit, and a usability study with 10 participants in a 300-sensor building shows that occupants rate it highly usable (SUS 80 ± 8) and use it to surface insights about their own spaces.

Together, these four systems progressively replace per-environment user effort with automated alternatives at each stage. No single system realizes the full vision of a self-configuring environment described in Section 1.2, but taken together, they demonstrate that the bootstrapping principle is viable across the pipeline and that the effort required to deploy context-aware systems in new environments can be significantly reduced. Chapter 2 provides shared background on the sensing modalities and computational foundations common to these systems, and Chapter 7 reflects on the limitations of this work and the research directions it opens.

Chapter 2

Background and Related Work

The previous chapter identified four stages at which deploying conventional context-aware sensing pipelines burdens users with hands-on effort, and outlined how each can be addressed by a more capable resource with appropriate scaffolding. Before presenting those alternatives in detail, it helps to establish the technical foundations they build on. This chapter covers the sensing infrastructure that modern buildings employ (Section 2.1), the computational methods used to recognize human activities from sensor data (Section 2.2). It also introduces the foundation models that have recently made new forms of automation possible and approaches researchers have developed for reasoning about higher-level context from activity streams (Section 2.3 and 2.4). Finally, we present the state of conversational interfaces for smart environments (Section 2.5). The aim of this chapter is to lay the common groundwork that the four contributions build on, and comparisons to specific related systems appear in each chapter that follows.

2.1 Sensing Infrastructure for Smart Environments

Buildings today are instrumented at a scale that would have been difficult to imagine two decades ago. A modern commercial building may contain hundreds of sensors monitoring temperature, humidity, air quality, occupancy, lighting, *etc.*. Research buildings push further, incorporating specialized sensing platforms that capture motion, vibration, sound levels, and thermal signatures at high temporal resolution. Understanding what these sensors are, what they measure, and what they cannot tell us is essential context for the activity recognition, context reasoning, and conversational access systems described in later chapters.

2.1.1 Building-Scale Sensing Deployments

Building Management Systems (BMS) form the traditional backbone of building instrumentation, managing subsystems for heating, cooling, lighting, and energy consumption [216]. In practice, however, BMS deployments are fragmented and narrowly utilized: Hahn *et al.* found that these systems serve primarily energy management, occupant comfort, and fault detection, with sensor data flowing to facility management dashboards rather than to the people who occupy the building [81]. Standardized metadata schemas such as Brick [24] have been developed to address

interoperability across vendors, enabling applications to be written against a common vocabulary of sensors, equipment, and spatial relationships.

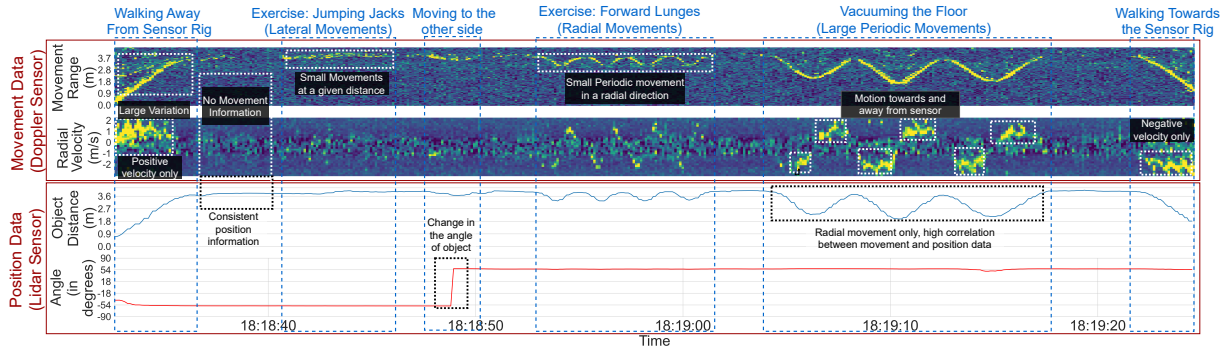
The research community has pursued sensing deployments that go well beyond what commercial building management requires. SmartSantander deployed thousands of IoT devices across a city to explore urban sensing applications [182]. MakeSense demonstrated the feasibility of deploying and managing dense sensor networks across multiple buildings [98]. The Mites platform [34] represents a more recent effort toward general-purpose sensing infrastructure for buildings, integrating twelve sensing dimensions, including temperature, humidity, pressure, light intensity, light color, motion, vibration, and low-resolution infrared, into a single compact device. Mites has been deployed at scale in our university building (TCS Hall, CMU) and provides the sensing infrastructure for several of the evaluations reported in this work. These platforms have surfaced important second-order challenges in networking, device management, and data transport, but their focus has remained on the infrastructure layer, leaving open the question of how to make sensed data interpretable and accessible to the systems that sit above them.

2.1.2 Privacy-Preserving Sensing Modalities

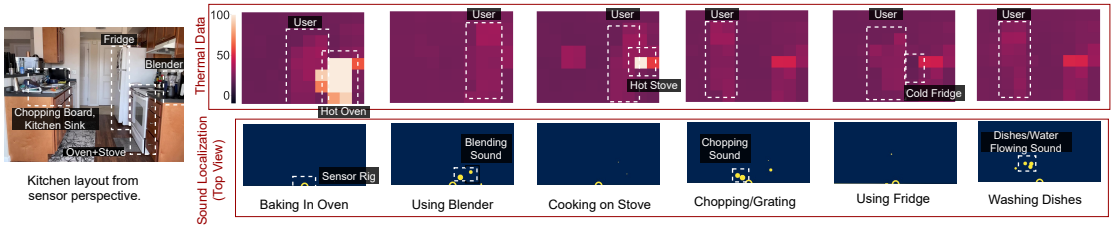
The tension between sensing fidelity and privacy was introduced in Chapter 1: cameras and microphones produce signals rich enough for accurate activity recognition but too privacy-invasive for continuous deployment in private spaces, while alternative modalities preserve privacy at the cost of richness in their signals. This subsection provides a technical grounding in the privacy-preserving sensor types that appear throughout this work, describing what each physically measures, what information it captures about human activity, and what it fundamentally cannot observe. Figure 2.1 illustrates the signals these sensors produce across a range of everyday activities.

Doppler radars: Frequency-Modulated Continuous Wave (FMCW) mmWave radars emit a reference radio-frequency signal and measure the signal reflected from objects in their vicinity. When a person moves, the reflected signal undergoes a Doppler shift proportional to their radial velocity relative to the sensor (Figure 2.1(a)). With frequency modulation, the radar can simultaneously estimate the range of moving objects, producing a two-dimensional representation of velocity and distance over time [191]. These range-Doppler signatures carry substantial information about gross body movements: walking produces a characteristic periodic pattern, exercises like jumping jacks produce rapid oscillations at a fixed range, and fine-grained activities like cooking produce more localized, irregular signatures. mmWave radars do not capture any visual information and operate through walls and in complete darkness, making them one of the most privacy-preserving modalities available for activity sensing [169]. The systems reported here use 77GHz FMCW evaluation boards operating at 5 to 15Hz.

Lidar: Light Detection and Ranging (Lidar) sensors use the time of flight of a laser beam to measure distance to objects. A two-dimensional lidar, spinning on a single axis, measures range across a 360-degree horizontal plane, producing a profile of object distances around the sensor that changes as people move through the space. Lidar is widely used in robotics for obstacle detection and mapping, but it can be repurposed to detect activities with positional relevance [114]: sitting on a sofa places a person at a different range and angle from the sensor than moving to vacuum,



(a) Movement and position information using Doppler and Lidar sensors over time.



(b) Thermal data (10x8) and Sound Localization information at a single timestamp (snapshot).

Figure 2.1: Visualizing information across various activities from various privacy-preserving sensors. (a) shows the movement range and radial velocity of moving objects using a Doppler sensor and object distance and angle using a Lidar sensor in the sensor plane across a sequence of activities happening in the living room. (b) shows snapshots of Thermal and Sound Localization at a given timestamp for various kitchen activities.

and the transition between the two is visible in the distance and angle (Figure 2.1(a)).

Low-resolution thermal arrays: Thermal sensors detect infrared radiation emitted by objects, producing a map of surface temperatures in their field of view. At high resolution, thermal cameras can reveal identifiable physical features such as body shape and facial contours [180]. At sufficiently low resolution, however, they become privacy-preserving: a 10×8 pixel thermal array can detect human presence and distinguish between interactions with heat-emitting appliances, a kettle, a stove, or a refrigerator, without resolving any personally identifiable information (Figure 2.1(b)). The thermal modality is particularly valuable for activities that involve temperature changes in the environment, such as cooking or using a space heater, which produce no distinctive motion or positional signature but alter the thermal profile of the scene.

Inertial Measurement Units (IMUs): IMUs typically combine an accelerometer, a gyroscope, and a magnetometer to capture motion and orientation data. When worn on the body, as in a smartwatch, they provide detailed information about limb movements and hand gestures at high temporal resolution [125, 142]. When mounted in the ambient environment, such as on a wall or surface, they capture structural vibrations coupled through walls and furniture during activities like exercising, operating appliances, or walking [115]. IMU data is inherently non-visual and non-auditory, though wearable configurations do require the user to wear a device consistently.

Sound localization: Unlike full-fidelity audio, which captures intelligible speech and is therefore

highly privacy-invasive, sound localization extracts only the direction from which sounds originate and their intensity level. This is accomplished using an array of microphones spatially separated on a printed circuit board; the difference in signal arrival times across the array elements allows the system to estimate the angular position of sound sources without recording the acoustic content itself [227]. At low sampling rates (on the order of 1Hz), a sound localization sensor captures a coarse spatial profile of acoustic activity in a room, which can help distinguish between activities that produce sound in different directions relative to the sensor, such as using a blender on one countertop versus running water at a sink across the kitchen (Figure 2.1(b)). The privacy properties of this modality are more nuanced than those of radars or lidars: even a microphone array operating in localization-only mode contains hardware capable of recording full-fidelity audio, and a compromised device could in principle be repurposed to do so. Emerging work on ML sensors, hardware designed to be physically incapable of outputting more than a specified representation, may eventually resolve this concern [217], but for now, sound localization occupies an intermediate position in the privacy spectrum.

Environmental sensors: A range of additional sensing dimensions, including temperature, humidity, barometric pressure, ambient light intensity, light color, and passive infrared (PIR) motion detection, capture coarse indicators of human activity through their effects on the surroundings [60, 115]. A humidity spike may indicate activity in a bathroom. A light level change may indicate someone turning on a lamp or a television. A PIR trigger indicates gross movement within a room. Individually, each of these signals is highly ambiguous; combined, they provide a low-fidelity but broad environmental fingerprint that complements more informative modalities. The Mites platform [34] integrates several of these environmental dimensions into a single device.

No single modality among those described above captures enough information to recognize activities reliably on its own. A Doppler radar sees motion but not position. A lidar sees position but not thermal state. A thermal array sees heat but not movement direction. In practice, multimodal configurations that combine several of these sensors produce richer and more discriminative feature spaces than any one sensor alone [10, 154]. The sensor rigs used in the studies reported here combine between three and six of these modalities, depending on the deployment setting and the privacy preferences of the participants.

2.1.3 Environment Specificity and the Data Challenge

A property shared by all of the modalities described above is that their signals are shaped as much by the physical environment as by the activity being performed. A Doppler radar's signature for someone cooking depends on the distance between the sensor and the countertop, the layout of the kitchen, and the reflective properties of nearby surfaces. A lidar's distance profile for "sitting on the sofa" is entirely determined by where the sofa happens to be relative to the sensor. A thermal array's reading during cooking depends on the placement and type of appliances in the kitchen. The same person performing the same activity produces a different sensor signature in a different room.

This environment specificity means that models trained on privacy-preserving sensor data from one space do not transfer to another. Each new deployment is, in effect, a new learning problem. Unlike the audio and video domains, where large annotated datasets such as Kinetics [47] and

AudioSet [74] have enabled models that generalize across environments with minimal adaptation, no comparable datasets exist for privacy-preserving modalities, and it is unclear how such datasets could be constructed given the inherent dependence on local geometry. The result is a cold-start problem: deploying an activity recognition system with privacy-preserving sensors in a new home or office has traditionally required collecting and labeling training data in that specific space, a process that demands hours of effort from either the occupant or a researcher. How the research community has attempted to address this data scarcity, through transfer learning, self-supervised methods, and in-situ bootstrapping, is the subject of the next section.

2.2 Human Activity Recognition

Human Activity Recognition is the computational field responsible for translating raw sensor signals into semantically meaningful labels that describe what a person is doing. It sits at the foundation of the sensing pipeline introduced in Chapter 1. The field has developed along two largely separate tracks, one built on cameras and microphones and the other on privacy-preserving modalities, and the gap between them in data availability and model maturity is what makes the bootstrapping problem described in the previous section so persistent.

2.2.1 Audio and Video-Based Activity Recognition

The most capable activity recognition systems today are built on video and audio data. Their strength derives from a combination of expressive sensing modalities, large annotated datasets, and model architectures that learn representations abstract enough to transfer across environments.

For video, datasets such as Kinetics [47], ActivityNet [84], and UCF101 [192] provide hundreds of thousands of labeled video clips spanning hundreds of activity classes, drawn from diverse real-world footage. These datasets have enabled a range of deep learning architectures for action recognition. Skeleton-based approaches first extract the human pose from video frames using models such as HRNet [195] or OpenPose [135], then classify the resulting pose sequences using graph convolutional networks (STGCN [225]) or 3D convolutional architectures (PoseC3D [67]). Because their input is an abstract skeleton rather than raw pixels, these models are largely invariant to background, lighting, and camera angle, which is what makes them generalizable across environments. Appearance-based approaches operate directly on pixel-level features using architectures like I3D [127] and two-stream networks, capturing both spatial appearance and temporal motion. Open-source toolboxes such as MMAction2 [140] and MMPose [141] have made these models readily accessible as off-the-shelf components.

For audio, sound-based activity recognition exploits the fact that many household activities produce distinctive acoustic signatures: a blender, a vacuum cleaner, a running faucet, and an electric shaver each generate characteristic sound patterns. Audio-based approaches typically convert sound signals into two-dimensional spectrograms using short-time Fourier transforms, then apply convolutional neural network architectures to classify them. Models like YAMNet [65], trained on AudioSet [74], a corpus of over two million labeled audio segments spanning 521 event classes, can recognize a broad range of environmental sounds without environment-specific

training. Ubicoustics [117] demonstrated that such models can be deployed for real-time activity recognition in homes and offices using a single microphone.

A critical property of both video and audio models is that they are *global*: trained on diverse data drawn from many environments, they generalize to new spaces with little or no adaptation. A skeleton-based action recognition model trained on internet videos works in a kitchen it has never seen, because the underlying representation, human joint positions over time, is largely independent of the physical setting. An audio classifier trained on AudioSet recognizes a running blender regardless of the room's acoustics or layout. No individual model is perfect across all activities, and combining multiple models with complementary strengths can improve coverage, but the key point is that the generalization problem for audio and video is largely solved at the model level.

2.2.2 Supervised HAR with Privacy-Preserving Sensors

Activity recognition with the privacy-preserving modalities described in Section 2.1.2 follows a different regime. Because models for these sensors do not generalize across environments, the standard approach is fully supervised: a researcher or occupant collects labeled data in the target space, extracts features from each sensor modality, and trains a classifier, typically a support vector machine, random forest, or shallow neural network, on the resulting feature-label pairs [152]. Multi-modal sensor fusion, combining features from multiple privacy-preserving sensors before classification, has been shown to improve accuracy over any single modality [10, 146, 215], but the underlying data requirement remains: someone must collect and annotate training examples in each new space.

This supervised paradigm carries two further assumptions that limit its flexibility. First, the set of activities to be recognized must be defined before data collection begins [212]. A researcher decides that the system should detect "cooking," "exercising," and "watching TV," then collects examples of each. Activities not anticipated at design time cannot be recognized, and adapting the system to a new activity requires a new round of data collection. Second, the granularity at which activities are defined is fixed at design time. The appropriate granularity depends on what the sensors can actually distinguish in a given physical arrangement, and this is not something a designer can know in advance. Too fine a vocabulary leads to poor classification accuracy when sensor signals are not discriminative enough; too coarse a vocabulary discards detail that would make the system useful. Most existing HAR systems treat both the vocabulary and its granularity as design-time choices rather than quantities to be discovered from data [171]. Typically, the research papers only report test results from these pre-defined set of activities.

2.2.3 Approaches to Reducing the Data Burden

The cost of per-environment labeled data collection has motivated a range of techniques aimed at reducing or eliminating the human effort involved. These approaches fall into three broad categories: cross-modal transfer, self-supervised learning, and in-situ bootstrapping.

Cross-modal transfer learning: The most direct approach to the data scarcity problem is to leverage the abundant labeled data available for video and audio to generate training signal

for privacy-preserving sensors. Researchers have developed methods to synthesize sensor data in one modality from recordings in another: Vid2Doppler [11] generates synthetic Doppler radar signatures from video of human activities, IMUtube [113] and AudioIMU [125] map video and audio to synthetic IMU signals, and IMU2Doppler [32] transfers from wearable IMU to Doppler radar. More recent work has strengthened this paradigm through advances in generative models [61, 121, 123, 232]. These synthesis-based approaches require a physical or statistical relationship between the source and target signal spaces: one can generate a plausible Doppler signature from a video of someone walking, because walking produces predictable radar reflections, but generating a plausible thermal array reading from the same video is far harder, because the thermal signal depends on appliance placement and ambient temperature rather than body motion. As a result, cross-modal synthesis has been largely limited to motion-related activities and single target modalities [11]. Approaches that target a broader range of activities, covering both motion-based and appliance-based activities across multiple sensor types, require a different strategy, one that does not depend on synthesizing the target signal at all.

Self-supervised learning: Self-supervised learning (SSL) takes a complementary approach: rather than transferring labeled data across modalities, it learns useful representations from the unlabeled sensor data itself. Models are pre-trained using pretext tasks, such as predicting masked segments of a signal, distinguishing temporally adjacent windows from distant ones, or enforcing consistency across augmented views of the same input, and the resulting representations are then fine-tuned on a small number of labeled examples for the downstream recognition task. Deldari *et al.* provide a comprehensive survey of SSL methods for temporal and multi-modal data [59]. In the HAR domain, ColloSSL [95] applies contrastive learning across time-synchronized multi-device settings, and FedCLR [167] combines federated learning with contrastive representations to reduce per-user labeling. SSL methods reduce the number of labeled examples required, but they do not eliminate the need for labels entirely: fine-tuning still requires some annotated data from the target environment. They also operate within a predefined activity vocabulary, offering no mechanism for discovering activities that the designer did not anticipate.

In-situ bootstrapping and activity discovery: A third line of work aims to discover activity patterns directly from sensor data in a new environment, without predefined labels. Clustering-based methods group similar sensor segments into clusters that ideally correspond to distinct activities [17, 54]. The resulting clusters, however, are unlabeled: they reveal that "something recurrent happens between 7:00 and 7:30 each morning" but not what that something is. Assigning meaningful names to discovered clusters has required manual interpretation by a researcher or occupant [88, 221]. Interactive learning approaches refine models through user feedback over time [106], but require sustained user involvement and still depend on the user to provide the semantic grounding that connects patterns to meanings.

What remains missing across these three categories is an approach that can simultaneously train models across multiple privacy-preserving modalities without requiring a physical signal-synthesis relationship, and a method that can discover what activities occur in a new environment and name them at a granularity matched to what the deployed sensors can actually distinguish. Recent advances in foundation models, particularly vision-language models and large language models, have opened new paths to both.

2.3 Foundation Models

The previous section described three families of techniques for reducing the data burden in privacy-preserving HAR, and noted that each leaves significant gaps: cross-modal transfer requires physical signal relationships, self-supervised learning still needs predefined vocabularies and some labeled data, and unsupervised discovery produces clusters that lack semantic meaning. What has changed in recent years is the emergence of foundation models, large neural networks trained on internet-scale data, that possess capabilities well suited to filling these gaps. Vision Language Models (VLMs) can describe a scene in natural language from a few video frames, providing the semantic grounding that clustering alone cannot. Large Language Models (LLMs) can reason over structured information, decompose complex queries into sequences of operations, and generate natural language responses. Together, these models can serve as the "richer sources" that enable the automated alternatives presented in later chapters.

2.3.1 Vision-Language Models

Vision-Language Models (VLMs) are neural networks trained jointly on visual and textual data, enabling them to process images or video and produce natural language descriptions, answer questions about visual content, or classify scenes without task-specific training [37]. Models such as GPT-4V, Gemini, and Claude have demonstrated broad visual understanding: given a photograph of a kitchen, a VLM can describe the scene layout, identify objects and appliances, and infer what a person in the image appears to be doing.

For activity recognition, VLMs offer a qualitatively different capability from the specialized video models described in Section 2.2.1. Skeleton-based models like PoseC3D classify motion into one of a fixed set of action categories learned during training. A VLM, by contrast, can generate open-ended natural language descriptions of what it observes, describing activities it was never explicitly trained to recognize. Recent work has explored this capability for HAR directly, using VLMs as zero-shot or few-shot activity recognizers [18, 122, 132, 179]. The results are promising: VLMs can identify a wide range of household activities from short video clips without any environment-specific training data. This open-vocabulary capability is what makes VLMs particularly valuable for the activity discovery problem. A clustering algorithm applied to privacy-preserving sensor data can identify that a recurring pattern exists, but it cannot say what that pattern represents. A VLM, shown a brief video clip from the moment when that pattern occurs, can describe the scene: "a person is standing at the kitchen counter, using a knife to chop vegetables on a board." The description provides the semantic label that the sensor cluster lacked.

As we show in Chapter 4, VLMs by themselves have a number of limitations. Their descriptions can be inconsistent across similar scenes, using different vocabulary for the same activity observed minutes apart. They can hallucinate details not present in the image, particularly when prompted for specifics. Their performance depends on frame rate, resolution, and prompting strategy, and generating structured, machine-parseable output requires careful prompt engineering. These practical challenges are addressed in the system designs presented in Chapter 4.

2.3.2 Large Language Models

Large Language Models (LLMs) are autoregressive neural networks trained on massive text corpora that exhibit emergent capabilities in reasoning, planning, and task decomposition [101]. Beyond generating fluent natural language, LLMs can parse ambiguous instructions into structured representations, maintain coherent multi-step reasoning chains, and produce output in specified formats such as JSON or code. These capabilities have made them central to a growing range of systems that bridge natural language and computational operations.

Several paradigms have emerged for using LLMs to perform analytical tasks over structured data. Program synthesis approaches prompt the model to generate executable code, typically Python or SQL, that retrieves and processes data [50, 73, 163]. This is flexible but unpredictable: the generated code may contain errors, access data it should not, or produce results that are difficult to audit. Tool-use approaches equip the model with a set of predefined functions it can invoke, with the model deciding which tools to call and in what order [156, 168]. Agent frameworks like ReAct [228] extend this further, allowing the model to interleave reasoning steps with tool calls in an iterative loop, observing intermediate results and adapting its strategy. At the other extreme, in-context reasoning approaches provide the data directly in the prompt and ask the model to reason over it without external computation [193]. Each paradigm trades off flexibility against predictability, and the choice among them depends on the domain's requirements for accuracy, auditability, and safety.

In the smart environment domain, LLMs have been explored as central coordinators that interpret occupant intent and orchestrate device actions. Systems like Sasha [109] and SAGE [178] use LLM-based agents to respond to underspecified smart home commands, decomposing goals like "make it cozy" into sequences of device actions. For building sensor data, systems like BuildingGPT [124] and OntoSage [62] have applied Retrieval-Augmented Generation (RAG) and query routing to enable natural language access to metadata and sensor readings. These systems demonstrate that LLMs can bridge the gap between how occupants think about their environment and how building systems organize data, but they face open challenges around multi-step analytical composition, access control in multi-stakeholder settings, and the reliability of LLM-generated responses. The specific ways in which these challenges have been addressed are discussed in Chapter 6.

Beyond conversational interaction, LLMs can also serve as semantic reasoning engines for tasks that do not involve end-user dialogue. Matching inconsistent natural language descriptions to a consistent set of labels, computing semantic similarity between activity descriptions, and structuring unstructured observations into machine-parseable formats are all tasks where LLM capabilities have proven useful. In this capacity, the LLM acts less as a conversational partner and more as a computational tool for processing language-encoded information, a role it plays in the activity discovery pipeline described in Chapter 4.

2.4 Context-Aware Computing

The previous sections described how sensors capture data about physical environments and how activity recognition systems translate that data into labels for what occupants are doing. But

knowing that someone is typing, talking, and sitting is not the same as knowing they are in a meeting. Knowing that someone used the microwave and opened the refrigerator is not the same as knowing they are reheating a pre-cooked meal rather than preparing a fresh one. The distinction between detecting activities and understanding what they mean together is the domain of context-aware computing, a field that has evolved from simple location-based services to systems that reason about complex human situations [31, 160]. This section describes the two principal approaches to context reasoning and the structural difficulty that has limited both.

2.4.1 Knowledge-Driven Approaches

Knowledge-driven approaches model context through formal representations of the relationships between activities, locations, and higher-level situations. The most common framework uses the Web Ontology Language (OWL) [77] to define a vocabulary of activity classes and context classes, linked by properties that encode how activities compose into contexts. In a smart home ontology, for instance, the context "office work" might be defined as requiring the co-occurrence of activities like "typing" and "writing," while "on a phone call" might require "phone ring" followed sequentially by "talking" [49, 209]. Once the ontology is defined, SPARQL queries can be used to match observed activity patterns against these definitions and infer which contexts are active at a given time [137, 138].

The strength of ontological approaches is their interpretability: the reasoning chain from activities to context is explicit and auditable, and the context definitions can be inspected and modified by domain experts. Their limitation is that the definitions must be authored in advance. Every context the system can recognize must be anticipated by whoever wrote the ontology, and the activity patterns that indicate each context must be specified as formal rules. This makes ontological systems brittle in practice: they work well for contexts whose definitions are stable and well understood, but cannot recognize patterns that their authors did not foresee [9, 49]. The space of possible activity combinations in real life is vast and person-specific, and no fixed rule set can cover it comprehensively.

A further limitation is that most ontological systems do not distinguish between different temporal structures of the same set of activities. "Typing" and "talking" happening in parallel likely indicates a meeting, while "typing" followed by a long stretch of "talking" may indicate a transition from focused work to a phone call. Standard OWL reasoning operates over sets of facts rather than over sequences, which means that temporal ordering, duration, and interleaving, the very features that often determine context, are difficult to express within the ontological framework alone [138].

2.4.2 Data-Driven Approaches

Data-driven approaches take the opposite strategy: rather than defining context rules in advance, they discover recurring patterns in activity streams through temporal analysis and clustering. The input is typically a time series of activity labels, and the goal is to identify groups of activity sequences that recur across time and likely correspond to distinct contexts.

Temporal clustering methods have been applied at multiple levels of granularity. Prior work focuses on identifying behavior routines from mobile and ambient sensor data [104, 172], while

other approaches target complex activity sequences such as multi-step procedures [76, 112]. Probabilistic and statistical methods have been used to model temporal activity patterns and identify abnormal behaviors [20, 176]. More recently, deep learning architectures have been applied to this problem: temporal autoencoders that combine convolutional layers for capturing parallel activity patterns with bidirectional LSTMs for modeling sequential dependencies can produce dense embeddings that encode the temporal structure of activity streams [134].

Data-driven methods can discover patterns that ontological rules would miss, including interleaved and recurring combinations that no designer anticipated. Their limitation is the inverse of the ontological approach: the discovered clusters are unlabeled. A clustering algorithm can identify that a particular interleaving of typing, drinking, and talking recurs every weekday morning, but it cannot determine that this pattern constitutes "office work." Providing semantic labels for discovered clusters has traditionally required either manual interpretation or mapping back to a predefined vocabulary, reintroducing the manual effort that data-driven methods were meant to avoid.

2.4.3 Hybrid Approaches and Remaining Gaps

Recognizing the complementary strengths of knowledge-driven and data-driven methods, several hybrid approaches have been proposed [108, 149, 175]. The general strategy is to use data-driven techniques to discover temporal patterns in activity streams and then apply ontological reasoning to assign semantic labels to the discovered patterns. This combines the pattern-discovery capability of clustering with the interpretability and semantic grounding of ontologies.

In practice, however, hybrid systems have struggled with two issues. First, the ontological component can only label patterns that fall within its predefined vocabulary; novel contexts that emerge from the data but were not anticipated in the ontology remain unlabeled. Second, most hybrid approaches have been evaluated on narrow activity sets and specific application domains, and it remains unclear how well they generalize to the diversity of real-world behavior, where a person's daily routine may span contexts as varied as exercising, commuting, focused work, socializing, and meal preparation, often interleaved within the same hour [164]. Existing surveys provide comprehensive coverage of the landscape [31, 160, 164]; a detailed comparison of specific systems with the hybrid approach presented in Chapter 5 is provided in that chapter's related work section.

2.5 Conversational Interfaces for Smart Environments

The preceding sections traced a path from raw sensor data through activity recognition to context understanding. The final link in the pipeline is the interface through which occupants access what the system knows. This section describes the current state of that interface and why it remains a bottleneck even when the backend capabilities are mature.

2.5.1 The Fragmentation Problem

Modern smart buildings suffer from a severe fragmentation of user-facing interfaces. Each building subsystem, HVAC control, room booking, lighting management, occupancy dashboards, energy monitoring, typically comes with its own dedicated application, each presenting a different interface, requiring separate credentials, and embedding assumptions about what questions occupants will ask [70, 81]. The result is that an occupant who wants to find a quiet, well-lit room with comfortable temperature must consult three separate tools, mentally integrate their outputs, and make a decision based on information that was never designed to be combined. For questions that span subsystems, such as "which rooms on my floor have been consistently comfortable this week," there is often no way to ask at all.

One response to this fragmentation is to address it at the platform layer rather than the interface layer. Building operating systems such as HomeOS [64], BOSS [57], BuildingDepot [218], and Home Assistant [4] expose programmatic APIs that abstract over heterogeneous subsystems, and standardized metadata schemas such as Brick [24] let an application be written once against a common vocabulary of sensors, equipment, and spatial relationships rather than against each vendor's proprietary interface. These platforms make the underlying data accessible to software, and they are the foundation that systems like BuildingChat build upon. They do not, however, resolve the fragmentation that occupants experience. Turning sensor data into an answer to a specific question still requires someone to write an application for that question, which reduces to building an app for every need: space booking, comfort monitoring, occupancy analytics, and so on. This model carries a development cost for each new use case, leaves occupants who do not program unable to ask anything the available apps were not built to answer, and still requires that every useful question be anticipated when the application is written. The integration problem is solved at the data layer, while the access problem for occupants is relocated rather than removed.

This fragmentation persists despite decades of research in building sensing infrastructure. The gap between the data that buildings collect and the insights that occupants can extract from it has been documented repeatedly [81, 216]. Facility managers, who have training and dedicated tools, can access some of this data through management dashboards, but everyday occupants are largely excluded. The sensing infrastructure described in Section 2.1 generates a continuous, rich data stream; the interface layer discards most of its potential value by forcing occupants into narrow, pre-designed interaction paths.

2.5.2 Conversational Interaction as a Unifying Paradigm

Conversational interfaces offer a natural alternative to the app-per-subsystem model. Instead of requiring occupants to identify the right application, learn its interface, and formulate a query in its terms, a conversational system allows them to express their needs in natural language and receive answers drawn from whatever data sources are relevant. The interaction shifts from command-based to intent-based: the occupant states what they want to know, and the system determines how to best answer it.

This paradigm is well suited to building environments for several reasons. Occupants' questions are naturally expressed in terms of subjective experience and spatial relationships ("is my office warmer than usual today," "which conference room is quietest right now") rather than in

terms of sensor identifiers and database schemas. The same underlying data can answer a wide range of questions, from point measurements to cross-location comparisons to temporal trends, but only if the interface is flexible enough to accept questions that the designer did not anticipate. And in shared buildings, where occupants range from technically sophisticated researchers to administrative staff to visitors, a natural language interface imposes the least expertise burden on the user.

2.5.3 Challenges Specific to Building Sensor Data

Applying conversational interfaces to building sensor infrastructure introduces requirements that go beyond what smart home assistants or general-purpose chatbots face. Three are particularly salient.

First, answering occupant questions often requires multi-step analytical composition rather than single lookups. A question like "find me a comfortable place to work on the third floor" requires identifying what "comfortable" means (perhaps quiet and well-lit), determining which sensors measure noise and light on the third floor, retrieving and aggregating their readings, filtering candidate rooms, and ranking them. Each of these is a distinct analytical operation, and the answer requires composing them in the right order. Systems that route each query to a single analytical function cannot express such compositions [62, 119].

Second, shared buildings introduce a multi-stakeholder privacy problem that does not arise in single-user smart home settings. The same sensor data serves multiple people with different roles and different legitimate needs. A facility manager asking for average floor occupancy is making a reasonable operational query; an occupant asking whether anyone was in a specific colleague's office last night may be tracking someone's movements. The data is identical, but the privacy implications depend on who is asking, what spatial and temporal granularity they are requesting, and whether the answer could reveal patterns about identifiable individuals [241]. Access control in this setting must operate at the level of individual queries, not just roles, and must handle the case where a query is partially answerable: some aspects are permitted while others are not.

Third, occupants routinely underspecify their queries in ways that require clarification rather than guessing. "Is it noisy on my floor" does not specify which rooms, what time period, or what threshold constitutes "noisy." A system that guesses wrong wastes the occupant's time; a system that asks too many clarifying questions becomes tedious. The interface must negotiate with the user to resolve ambiguity while minimizing friction, a design challenge that interacts with both analytical composition and access control. The specific mechanisms through which these challenges have been addressed are described in Chapter 6.

Together, the five areas covered in this chapter, sensing infrastructure, activity recognition, foundation models, context-aware computing, and conversational interfaces, form the technical substrate on which the contributions of this work are built. The next four chapters present those contributions in sequence, each addressing one stage of the sensing-to-interaction pipeline: training labels (Chapter 3), activity vocabulary (Chapter 4), context synthesis (Chapter 5), and occupant interaction (Chapter 6).

Chapter 3

Bootstrapping Activity Recognition from Audio and Video Models

This chapter presents VAX[1], which addresses the first stage of the sensing pipeline described in Chapter 1: obtaining training labels for privacy-preserving sensors in a new environment without requiring the occupant to perform and annotate activities.

3.1 Introduction

As described in Chapter 2, the three principal approaches to reducing the labeled data burden for privacy-preserving sensors, cross-modal transfer learning, self-supervised learning, and unsupervised clustering, each address part of the problem but leave a common gap. Cross-modal synthesis approaches like Vid2Doppler [11] and IMUtube [113] require a physical or statistical relationship between the source and target signal spaces, restricting them to motion-related activities and single target modalities. Self-supervised methods such as ColloSSL [95] reduce the number of labeled examples needed but still require a predefined activity vocabulary and some annotated data from each new environment. Neither family of approaches can train models across a heterogeneous set of privacy-preserving sensors, covering both motion-based and appliance-based activities, without substantial per-environment effort.

This chapter presents VAX (Video/Audio to 'X'), a system that addresses this gap by taking a different approach to the cross-modality problem. Rather than synthesizing the target sensor's signal from video or audio recordings, VAX uses the *output labels* of off-the-shelf audio and video ML models to train activity recognition models for a diverse set of privacy-preserving 'X' sensors. The approach is built on four observations: (1) current-generation pre-built A/V models are reasonably good at identifying a variety of activity patterns [65, 66]; (2) these models generalize across home environments without retraining; (3) combining multiple privacy-preserving sensors produces a feature space rich enough to distinguish activities that no single sensor can separate on its own; and (4) the small number of activities that A/V models cannot reliably detect can be covered with minimal user input.

[1] `github.com/synergylabs/vax`

The design embodies the bootstrapping principle introduced in Section 1.2: a camera and microphone are *temporarily* deployed alongside the privacy-preserving sensors, off-the-shelf A/V models generate training labels during this period, and once the models for the privacy-preserving sensors are trained, the camera and microphone are physically removed. The A/V sensors serve as a richer, more capable source that departs after bootstrapping, leaving behind trained models that require no audio or video data to operate. Because VAX operates on labels rather than on synthesized signals, it is agnostic to the choice of privacy-preserving sensors. Adding a new sensor modality requires only a featurization function for that modality, not a signal-synthesis model relating it to the A/V domain.

Realizing this approach required solving three technical challenges. First, off-the-shelf A/V models are trained with diverse and non-exhaustive label sets [66, 127], and labels like "jumping jacks" from one video model and "exercising" from another must be reconciled into a single consistent vocabulary. Second, even the best-performing A/V models achieve overall accuracy below 75–80% on their native benchmarks [140], and incorrect labels propagated to the privacy-preserving sensor training pipeline directly degrade its accuracy. Third, data collected from different privacy-preserving modalities is heterogeneous in format, sampling rate, and signal characteristics, requiring modality-specific featurization and model training rather than a monolithic approach.

To address these challenges, VAX constructs an ensemble of off-the-shelf A/V models, learns a non-linear mapping from their raw predictions to a consistent set of activity labels using data from a set of reference homes, and deploys this ensemble to predict activities in new homes. To mitigate the impact of noisy and missing labels, VAX opportunistically uses data from the privacy-preserving sensors themselves: unsupervised clustering on sensor data identifies activities that are separable in a given modality's feature space, allowing the system to increase detection rates for low-confidence A/V predictions. For the small number of activities that neither the A/V pipeline nor the support models can detect, VAX solicits a single labeled instance from the user. Finally, VAX applies a confidence-learning approach to identify and remove noisy labels before training the final activity recognition models for each sensor modality.

We built and deployed VAX in ten participants' homes while they performed 17 pre-selected activities of daily living across three locations (kitchen, living room, and bathroom). Our A/V pipeline detects approximately 15 out of 17 activities per home with an average accuracy of 90% on detected instances. Privacy-preserving sensor models trained with labels from the A/V pipeline achieve an accuracy of 84% on detected activities without any user input. When augmented with a single labeled instance for each undetected activity (approximately 2 out of 17 per home on average), VAX achieves an overall accuracy of 84% across all 17 activities. This is comparable to a fully supervised baseline that requires one labeled instance per activity per home (79% accuracy), but with roughly one-eighth of the labeling effort ($\sim$2 labels *vs.* 17 labels per home).

In summary, this chapter makes the following contributions:

- We present VAX, an end-to-end system for training activity recognition models for a set of 'X' privacy-preserving sensing modalities using labels from off-the-shelf A/V models. Once the privacy-preserving sensor models are trained, the A/V sensors can be removed entirely.
- We present a method to bootstrap and combine off-the-shelf A/V models, reconciling their heterogeneous label spaces, and to augment their output using unsupervised clustering on

privacy-preserving sensor data to improve detection rates.

- We evaluate VAX with 10 participants performing 17 activities in their own homes. We show that VAX achieves an accuracy of 84% using labels generated by its A/V pipeline and only a single user-provided label for each undetected activity ($\sim$2 labels *vs.* 17 labels per home). Without any user input, VAX achieves 74% accuracy across all activities, considerably outperforming a baseline that trains directly on privacy-preserving sensor data from reference homes (38%).

The remainder of this chapter is organized as follows. Section 3.2 describes the VAX sensing platform and the specific hardware components integrated into our prototype. Section 3.3 presents the system design, covering the A/V pipeline, the support model approach for improving detection rates, and the training pipeline for privacy-preserving sensors. Section 3.4 reports the evaluation results across three dimensions: A/V pipeline performance, end-to-end VAX pipeline performance, and comparison with baseline approaches. Section 3.5 discusses the limitations of the current prototype and opportunities for future work, and Section 3.6 summarizes the chapter's findings and introduces the question that motivates the next chapter.

3.2 VAX Sensing Platform

The VAX prototype integrates a camera and microphone for temporary A/V-based label generation with a suite of privacy-preserving sensors selected to capture a broad range of activity signatures. Chapter 2 described the sensing principles behind each of these modalities and the types of information they capture about human activity (Section 2.1.2); this section focuses on the specific hardware components we selected for our prototype and the design decisions involved in assembling them into a deployable sensing rig. We limited ourselves to ambient sensors to avoid burdening participants with wearable devices, though incorporating wearable modalities such as smartwatch IMUs could present interesting opportunities in future work.

For audio and video sensing, we used a Logitech USB Webcam (C910, 720p resolution) and a Yeti USB microphone, with an iPhone serving as a backup data source for both modalities. In selecting the privacy-preserving sensors, we reviewed prior work in HAR [100, 118, 170] and considered aspects such as sensing range, breadth of signals captured, and signal fidelity. The following paragraphs describe the specific hardware we used for each sensing modality.

Doppler radar: We used the AWR1642BOOST-ODS [92], a 77GHz mmWave FMCW evaluation board from Texas Instruments with an integrated DSP and ARM Cortex-R4 processor. Its patch-antenna structure provides a wide field of view in both azimuth and elevation planes. The integrated microcontroller provides processed data at only 5Hz, which we supplemented with a DCA1000 FPGA board [93] connected directly to the radar's ADCs. This board extracts raw Doppler samples in binary format and transmits them via UDP to our compute node, where we process them into range-Doppler heatmaps at a sampling rate of 15Hz.

Thermal sensing: We used a FLIR Lepton 3.5 thermal camera with radiometry support [7], interfaced with a Purethermal Mini development board for USB communication. This sensor captures thermal data at 160×120 pixels at 8Hz. At this native resolution, however, we observed that the thermal data could visibly distinguish between individuals based on physical features such

as height, body shape, and facial contours. To address this privacy concern, we downsampled the resolution by 16× to just 10×8 pixels, at which point no personally identifiable information (PII) is discernible while the sensor still captures meaningful thermal signatures from appliances and human presence. We initially attempted to use a Panasonic Grid-Eye sensor (8×8 resolution), but its data quality and sensing range were insufficient, so we used the downsampled FLIR sensor as a proxy for a very low-resolution thermal array.

Lidar: We used the Slamtec RPLIDAR A1M8 [105], a mechanical 2D lidar with a 360° horizontal scanning range, a maximum range of 6m, and a scanning frequency of 5.5Hz sampling 360 points per revolution at 1° angular resolution. We also explored a 3D lidar sensor but chose not to use it due to significant latency in providing stable readings and intermittent data gaps during periods of high movement.

Sound localization: We used a ReSpeaker 4-Mic Array with a Raspberry Pi [227], a quad-microphone expansion board designed for voice and AI applications. We capture only the direction of arrival and intensity of sound at a very low resolution (1Hz) from this sensor, not full-fidelity audio content.

Environmental and vibration sensing: We used the Mites multi-modal sensor board [35, 139] to sense ambient vibrations through its onboard IMU (accelerometer, gyroscope, and magnetometer) and a range of environmental signals including temperature, humidity, barometric pressure, light intensity, light color, movement via PIR, and low-resolution infrared. Although the Mites board includes an onboard microphone capable of producing featurized audio output, we did not use that data in VAX.

Figure 3.1 shows our assembled VAX hardware prototype. While the selected privacy-preserving sensors could in principle be integrated onto a single custom board, we built our prototype from off-the-shelf hardware modules and sensor evaluation boards to facilitate rapid iteration. All sensors connect via USB or Ethernet to a small form-factor PC (Intel NUC, 8-core, 16GB RAM) that collects and stores processed data from each modality. The entire rig is wall-powered and can be mounted on a tripod, making it portable between locations within a home and across different homes. Figure 2.1 in Chapter 2 illustrates the types of signals these privacy-preserving sensors produce across a range of everyday activities.

3.3 System Design

Figure 3.2 shows the high-level architecture of VAX, which separates the system into a training phase and a deployment phase. During the training phase, we deploy a camera and microphone alongside the privacy-preserving sensors in the participant's home. Our Audio-Video (A/V) pipeline runs a set of off-the-shelf pre-trained ML models on the captured audio and video data to detect activities and generate training labels. These labels, potentially noisy and incomplete, are then passed to the privacy-preserving sensor pipeline, which uses them to train per-home activity recognition models for each sensor modality. During the deployment phase, the camera and microphone are physically removed, and the trained models on the privacy-preserving sensors handle activity detection on their own, with no A/V data involved. We now describe each component of this pipeline in detail, beginning with how we collected our evaluation data.

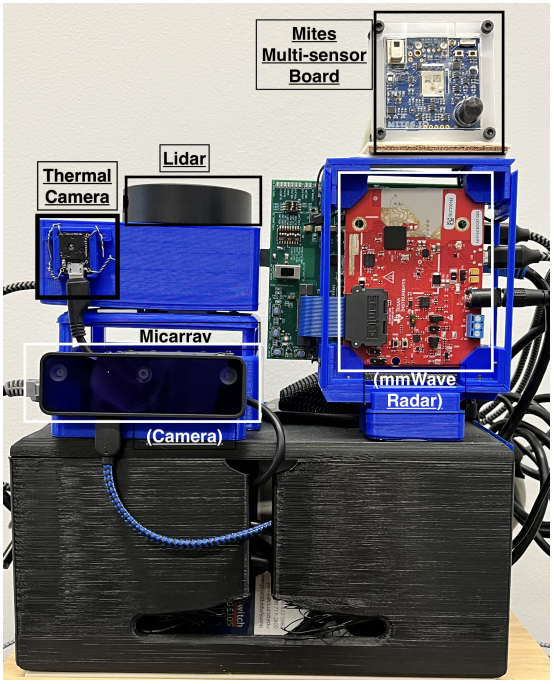


Figure 3.1: VAX hardware apparatus. The Doppler radar, thermal sensor, and Mites board are mounted vertically, while the Lidar and Micarray are oriented horizontally. The setup sits on a box housing a combined USB and Ethernet switch that routes data to an Intel NUC compute node.

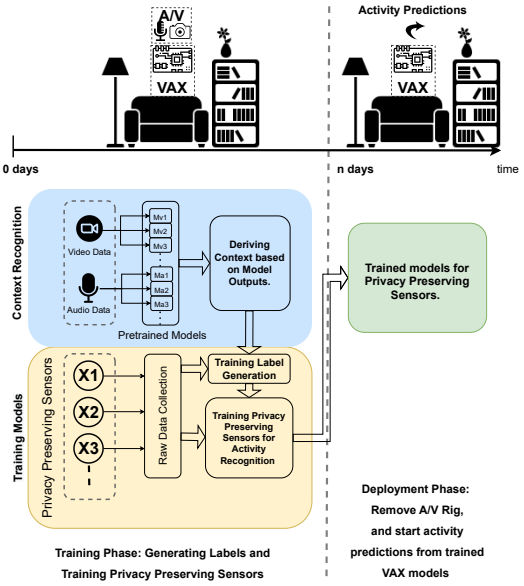


Figure 3.2: High-level architecture of the VAX training and inference pipeline. During the training phase, VAX uses a set of off-the-shelf audio and video models to recognize activities and generates labels that are used to train privacy-preserving sensor models. Once trained, the A/V sensors are removed and the trained models on the privacy-preserving sensors are deployed for activity recognition.

3.3.1 Study Design and Data Collection

We deployed our VAX prototype and collected data from participants in their own homes. For each home, we set up the prototype in three locations (kitchen, bathroom, and living room) to capture the diversity of activities across different domestic spaces. We started with a union set of over 30 activities drawn from recently published activity recognition literature [115, 117, 142] and filtered this set based on two criteria: we removed activities that do not occur in the three location contexts we considered (*e.g.*, activities like drilling or using screwdrivers), and we removed activities that do not produce any observable movement or audio signature (*e.g.*, sitting or sleeping). We intentionally retained a range of high-level activities such as cooking and exercising to allow for diverse movement and sound patterns across participants, thus training a more robust set of activity recognition models.

We finalized a list of 17 activities distributed across the three locations. For the kitchen, we collected samples for using the oven (Baking), using a blender or mixer (Blender), heating food in the microwave (Microwave), using a fridge (FridgeOpen), chopping and grating vegetables or fruits (Chopping+Grating), cooking a meal on the stove (CookingOnStove), and washing dishes in the sink (WashingDishes). For the bathroom, we collected data on using a hair brush (HairBrush),

using an electric hair dryer (HairDryer), washing hands (HandWash), using an electric shaver (ShaverInUse), and flushing the toilet (ToiletFlushing). Finally, we collected samples for activities that can happen anywhere in a home, such as eating or drinking (Eating/Drinking), exercising (Exercising), coughing (Coughing), operating a vacuum cleaner (Vacuum), and walking (Walking), and asked participants to perform these in their living room. All activities were performed by a single participant in their own home. We collected data from 10 participants; the sample size is consistent with similar participant studies in HAR literature [32, 51]. Each study session took around 4 to 5 hours, including the time to transport the VAX rig to the participant's home and set it up at multiple locations, resulting in approximately 10 hours of data collection per participant across all three locations. Our study design was approved by our Institutional Review Board (IRB).

3.3.2 Activity Recognition Using Off-the-shelf A/V Models

A straightforward approach to generating activity labels from A/V data would be to select the best-performing off-the-shelf model for each modality, or to run a confidence vote across multiple models. We shortlisted twenty popular off-the-shelf A/V models [30, 65, 67, 71, 72, 75, 79, 126, 127, 130, 142, 188, 190, 194, 201, 204, 214, 220, 225, 226, 244] based on open-source availability of the models and pre-trained weights, as well as their performance benchmarks against publicly available datasets [47, 84, 192]. However, our empirical evaluation of these models, used as-is on the data we collected for our 17 activities of daily living, did not produce accurate results. The reasons for this inaccuracy are twofold: the models are trained to provide labels for activities that can have overlapping signatures (*e.g.*, a blender in a kitchen *vs.* a power drill in a workshop), and they are trained for HAR in diverse settings that do not map cleanly to common activities of daily living (*e.g.*, atomic actions like "sitting" can be attributed to multiple higher-level activities). Merely combining the outputs of all models is also non-trivial, since different models are better suited to different categories of activities.

Through our experiments with these twenty models, several patterns emerged that guided our pipeline design. Skeleton-based action recognition models performed most consistently across different home settings, since their input, the human pose, is largely agnostic to the visual environment. However, no single model provided consistent labels across all 17 activities; interestingly, the same skeleton-based architecture trained on different large-scale datasets learned different activity signatures and performed well on different subsets of activities. Audio-based models performed consistently well on activities with distinctive sound signatures (*e.g.*, blender, electric shaver, microwave), but showed sufficient redundancy that including more than one audio model in the ensemble provided no additional benefit. Video-based models showed greater variability, with their detection capability and confidence depending on the camera's vantage point and occlusions in each home. Most importantly, the labels produced by different models often lacked one-to-one correspondence with each other or with our chosen activities: one model might label an activity as "Exercising" while another calls the same action "Doing Jumping Jacks," requiring reconciliation into a single consistent vocabulary.

Based on these observations, we finalized a set of five off-the-shelf models for our A/V pipeline. For video-based activity recognition using human pose estimation, we selected *PoseC3D* [67] and *STGCN* [225], both trained on the *NTU60* dataset [187] with 60 activity classes. We further

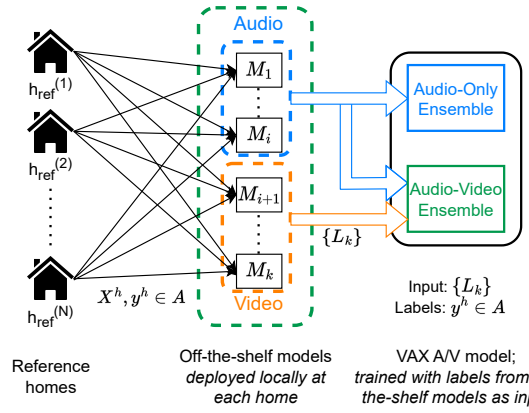


Figure 3.3: Building the VAX A/V model from data collected in reference homes. A/V data (X^h) from reference homes are passed through off-the-shelf models deployed locally, and the resulting raw labels (L_k) together with ground-truth VAX activity labels ($y^h \in A$) are used to train the global VAX A/V model.

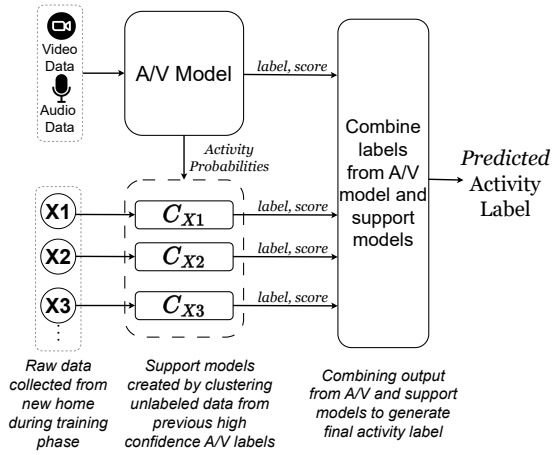


Figure 3.4: Generating activity labels in new homes using the trained A/V model and support models from privacy-preserving sensors.

included three additional copies of *PoseC3D*, each trained on a different dataset: *NTU120* [128] (120 classes), *UCF101* [192] (101 classes), and *HMDB51* [111] (51 classes). For audio-based activity recognition, we selected *YAMNet* [65], a deep network architecture that predicts 521 audio event classes based on the AudioSet-YouTube corpus [74].

Building an A/V model from reference homes: The raw labels produced by these five models do not map directly to our target activity vocabulary, but we observed that the top-k predictions from each model are *weakly consistent* across different home environments. Audio models are inherently agnostic to the visual environment, and the skeleton-based video models are trained on large-scale datasets like YouTube-8M [6] that contain observations for the same activity across a wide variety of home settings. To exploit this consistency, we collect the top-k predictions from each off-the-shelf model for a set of activities across a set of reference homes with known ground-truth labels. For each model, we then train a shallow neural network classifier that learns a non-linear mapping from the model's raw A/V label distribution to VAX activity labels (Figure 3.3).

We observed that audio models produce a more consistent set of top labels for a given VAX activity across environments than video models do. This led us to design two separate ensembles. The first, an *audio-only ensemble*, consists of classifiers built from the audio model and works well for activities with distinct sound signatures (*e.g.*, Blender, HairDryer, Vacuum). The second, an *audio-video ensemble*, combines classifiers from both audio and video models to detect the full range of activities. We found empirically that combining these two ensembles outperforms using a single audio-video ensemble or maintaining two separate ensembles for audio-only and video-only models. The two ensembles are combined to form the final VAX A/V model.

To predict activity labels in a new home (Figure 3.4), we run all five off-the-shelf models on the captured A/V data and collect their top-k label predictions. These predictions are passed

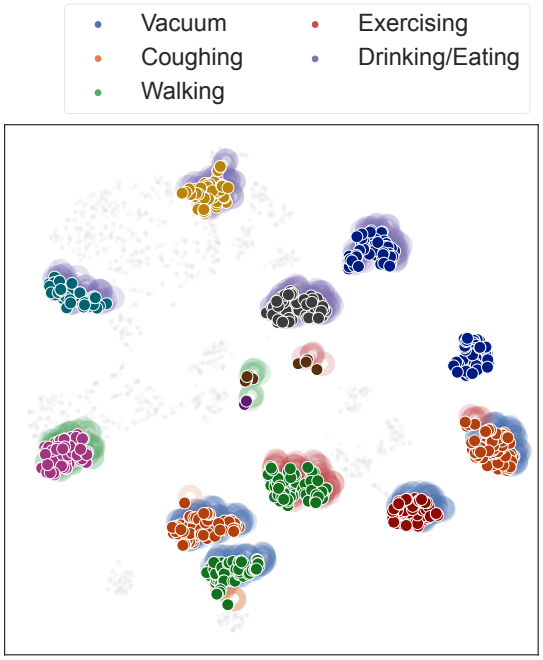


Figure 3.5: 2-D t-SNE embedding of clusters from Lidar data. Dark colored points represent clustered data; shaded regions highlight dense areas for different activities. Activities like Walking and Drinking/Eating are clearly separable, while Coughing and Vacuum overlap within a single cluster.

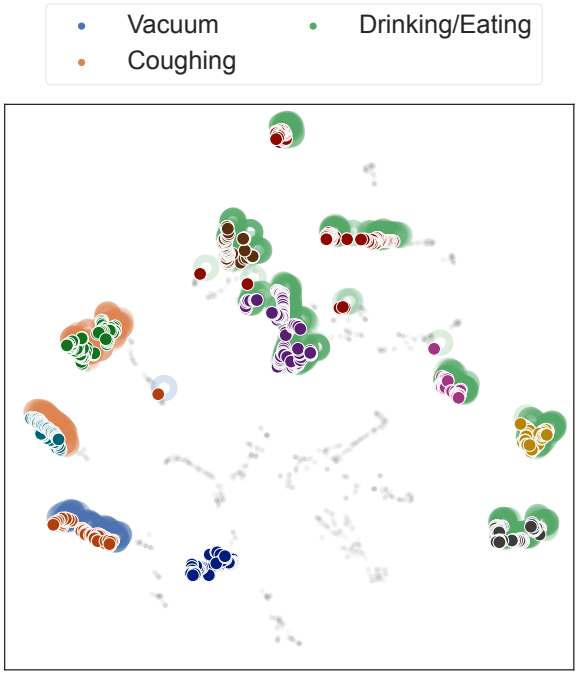


Figure 3.6: 2-D tSNE embedding of clusters from Thermal data. Thermal signatures show good separation for activities like Coughing and Vacuum, which overlap in the Lidar feature space, illustrating the complementary nature of different modalities.

through both the audio-only and audio-video ensembles to generate activity predictions, and the activity with the highest confidence value across both ensembles is selected. To control the error rate, we define confidence thresholds for each ensemble; an activity instance is marked as *undetected* if the predictions from both ensembles fall below their respective confidence thresholds (60% for the audio-only ensemble and 40% for the audio-video ensemble). These thresholds were determined empirically to work well in our training homes.

3.3.3 Improving A/V Labels with Privacy-Preserving Sensor Data

A central challenge in our A/V pipeline is setting the confidence cutoff thresholds that balance accuracy against detection rate, defined as the fraction of activity instances whose confidence score exceeds the threshold. Optimizing for high accuracy with a low detection rate risks missing all instances of a particular activity, leaving no training labels for that activity in the privacy-preserving sensor pipeline. Optimizing for high detection rates at the cost of accuracy introduces more noisy labels, which degrades the downstream models. We need a way to increase the detection rate for the A/V pipeline without sacrificing accuracy.

To address this, we opportunistically use information from the raw data collected by the

privacy-preserving 'X' sensors to boost the detection rate. The key insight is that certain pairs or groups of activities are naturally separable in a given sensor modality's feature space, even without any activity labels. For example, activities like FridgeOpen and Baking have clearly distinct thermal signatures (Figure 2.1 in Chapter 2), while activities like Walking and Drinking/Eating produce clearly distinct Doppler signatures. If the A/V pipeline labels one instance of an activity with high confidence but leaves another instance of the same activity below the confidence threshold, and that unlabeled instance falls within the same sensor-data cluster as the labeled one, we can increase the confidence for the unlabeled instance. This approach, however, only works for activities for which the A/V pipeline has already classified at least one instance with high confidence; it cannot bootstrap activities that the A/V pipeline has never detected.

For each privacy-preserving modality, we apply unsupervised clustering (OPTICS) on the raw sensor features and identify activity pairs that are separable in that modality's feature space, determined by whether two or more high-confidence A/V labels from different activities fall into distinct clusters. We then use these clustering models to enhance confidence scores for activity instances that fall below the A/V pipeline's threshold but belong to a cluster associated with a specific activity.

Figure 3.5 and Figure 3.6 illustrate this process with 2-D tSNE embeddings of clusters from Lidar and Thermal data, respectively. In the Lidar embedding, activities like Walking (green region) and Drinking/Eating (purple region) form well-separated clusters, meaning Lidar data can help the A/V pipeline assign labels to low-confidence instances of these activities. Activities like Coughing (red region) and Vacuum (blue region), however, overlap within a single cluster in the Lidar space. The Thermal embedding shows a complementary pattern: Coughing and Vacuum are well separated in the Thermal feature space, while other activity pairs that are separable with Lidar may not be with Thermal data. This complementarity across modalities is what makes the support model approach effective.

Figure 3.7 summarizes the impact of support models on detection rates across all activities. Activities like Drinking/Eating and WashingDishes see substantial improvement in detection rate when support from all three sensors is combined ("AV+All"). Some activities like Exercising and HairDryer show no improvement from any privacy-preserving sensor, because the A/V pipeline already detects all of their instances with high confidence, leaving no undetected instances for the support models to rescue.

3.3.4 Training Models for Privacy-Preserving Sensors

Once we have activity labels from the A/V pipeline (augmented by the support models described above), the next step is to use these labels to train activity recognition models for each privacy-preserving sensor modality. This subsection describes the four components of our training pipeline: obtaining labels for activities that the A/V pipeline could not detect, extracting features from each sensor modality, handling class imbalances in the generated labels, and reducing label noise before final model training.

Labels for undetected activities: Even after applying the support models, a subset of activities may remain undetected by the A/V pipeline, and these undetected activities may vary from one home to another. Without at least one labeled instance for each activity class, we cannot train

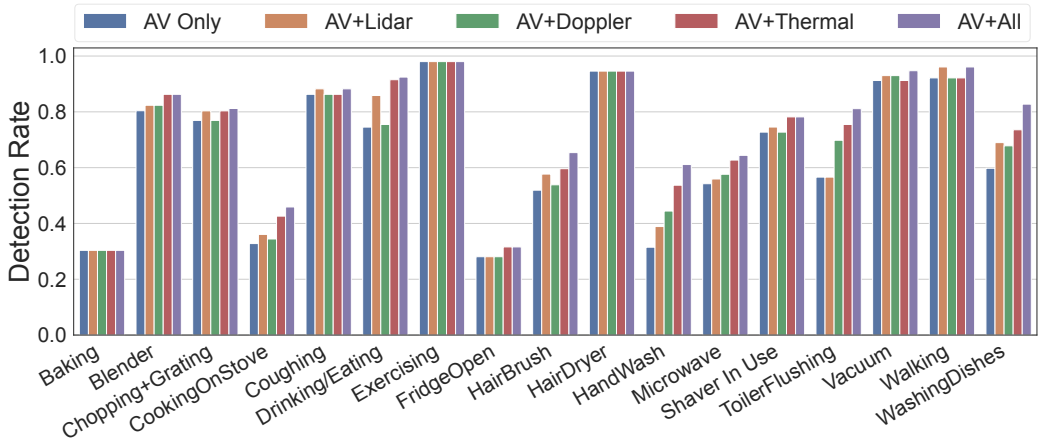


Figure 3.7: Improvement in activity detection rate using support models from individual privacy-preserving sensors (Lidar, Thermal, Doppler) and from all three combined (Lidar+Thermal+Doppler). Activities like Drinking/Eating and WashingDishes benefit substantially from sensor-based support. Activities like Exercising and HairDryer show no improvement, as the A/V pipeline already detects all instances with high confidence for these activities.

the privacy-preserving sensor models to recognize them. To address this, we ask the user for limited input: specifically, we ask them to perform *a single instance* of each undetected activity. As we show in Section 3.4, the number of undetected activities is small (approximately 2 out of 17 per home on average), so this places minimal burden on the user. With this single labeled instance for each undetected activity, VAX achieves an accuracy of 84%, which is better than a baseline approach where users provide one label for each of all 17 activities (accuracy of 79%), with significantly less effort.

Featurization: We train an ensemble of per-sensor models rather than a monolithic model that fuses all sensing modalities into a single feature space. While a fused approach could in principle capture cross-modal correlations, we prioritized reliability for long-term deployment: a per-sensor ensemble continues to function if one sensor fails or is removed. This design also allows incorporating existing modality-specific models (*e.g.*, Vid2Doppler for Doppler radar [11]) and takes advantage of well-studied featurization techniques for each modality [11, 114, 115]. We used the following featurization for each sensor. For Thermal data, we take a rolling window of 5 contiguous frames (10×8 pixels each) and compute the maximum temperature value in °C for each pixel. For Doppler data, we take a rolling window of 20 contiguous range-Doppler heatmaps (256×32) and decompose each into two components: the range bins of primary motion activity over time, and the average and standard deviation for different velocity bins over time; these two components are concatenated into a single feature vector. For Lidar data, we first remove the static boundary of the room, then record the distance of the closest object for each angle (0–360°) and take the maximum value over a rolling window of 10 contiguous samples. For the Micarray, we use a featurization approach similar to Lidar, with the distance from a sound source weighted by its amplitude; because Micarray data is sampled at only 1Hz, we do not aggregate across multiple samples. For the remaining sensors (PIR, IMU, environmental sensors), we directly use

the featurized data produced by the Mites sensor board at 100ms intervals [139].

Handling class imbalances: The labels generated by the A/V pipeline are not uniformly distributed across activities: some activities occur more frequently or are detected more reliably, leading to class imbalance in the training set. We use SMOTE (Synthetic Minority Oversampling Technique) [48], a minority oversampling approach that generates synthetic training examples by interpolating between existing examples in the feature space. In general, oversampling risks reducing generalizable accuracy by introducing synthetic data points. However, our setting offers a natural advantage: activity classes with higher anomaly scores (and thus more instances removed as outliers) also tend to have greater separability in the feature space due to denser data distributions for normal samples. As a result, the classes that benefit most from oversampling are precisely those whose feature distributions make synthetic interpolation most reliable.

Reducing label noise: Since the A/V pipeline generates labels that are inherently noisy, training directly on these labels degrades the accuracy of the privacy-preserving sensor models. We apply a confidence learning approach adapted from Northcutt *et al.* [148], which identifies noisy labels in settings where no annotation information is available beyond the observed (potentially incorrect) labels. For each privacy-preserving sensor x, we first train a naive classifier $\theta^{(x)}$ on the featurized sensor data using the noisy A/V-generated labels, and compute out-of-sample predicted probabilities $P_{ik}^{(x)}$ for all data points. We then calculate a confidence joint matrix $C_{ij}^{(x)}$ that partitions and counts labels across activity classes: diagonal entries count correct labels, and off-diagonal entries capture asymmetric label error counts (*e.g.*, $C_{i=3,j=1}^{(x)} = 10$ reads as "ten examples are labeled as activity 3 but should be labeled activity 1, according to sensor x"). This matrix is normalized to $Q_{ij}^{(x)}$ such that each row sums to the marginal probability of the corresponding activity under $\theta^{(x)}$, and the total probability across all activities sums to 1. We then compute per-activity confidence thresholds $\delta_i^{(x)}$, and any sample predicted as activity i is classified as noise if $P_{ii}^{(x)} < \delta_i^{(x)}$. Samples classified as noise by the majority of sensors are removed from the training set. A critical hyperparameter in this approach is the choice of the naive classifier $\theta^{(x)}$. We selected an ensemble of K-nearest neighbors (KNN) and SVM with a Gaussian kernel, which prior literature has shown to be among the most robust classifiers for training with imperfect labels [45], and which we empirically confirmed works best on our training data. After removing the noisy samples, we retrain the same classifier on the cleaned data to produce the final per-sensor model. For final activity predictions at inference time, we return the activity label with the highest predicted probability aggregated across all sensors.

3.3.5 End-to-End VAX Pipeline

The complete VAX pipeline consists of three stages: (i) training the A/V model from reference homes, (ii) generating activity labels using the trained A/V model and support models from the privacy-preserving sensors, and (iii) training per-sensor activity recognition models using the generated labels. Our implementation comprises over 10,000 lines of Python code. For machine learning methods across all components (KNN, SVM, OPTICS clustering, and SMOTE oversampling), we use implementations from the scikit-learn and imbalanced-learn libraries. Our noise reduction algorithm is built on top of Cleanlab [148], an open-source package for identifying and reducing label noise in datasets. We have open-sourced the entire VAX system [165]; the

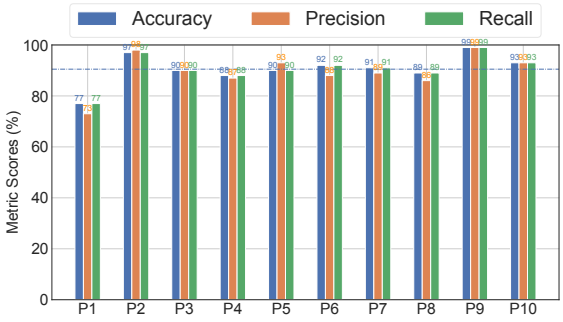


Figure 3.8: A/V pipeline: Accuracy, Precision, and Recall for activity recognition across all participants on detected activities (75% of all instances). The A/V pipeline performs consistently well across homes, with an average Accuracy of 90.5%, Precision of 89.5%, and Recall of 90.5%. The A/V model for each participant is trained in a leave-one-home-out fashion.

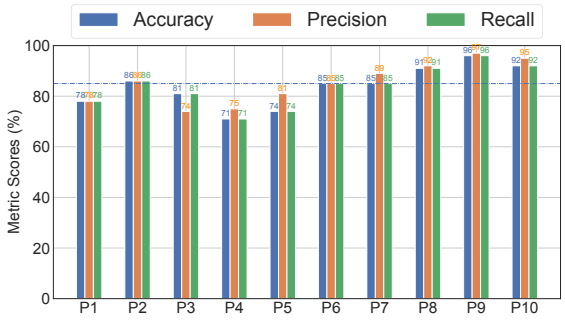


Figure 3.9: Privacy-preserving sensor pipeline: Accuracy, Precision, and Recall for activity recognition across all 17 activities for all participants. We observe an average Accuracy of 84%, Precision of 84%, and Recall of 84%, with consistent performance across homes. Variation across participants reflects differing amounts of noise in the A/V-generated labels.

pipeline can be used out of the box with label predictions from any set of off-the-shelf A/V models and can be extended to incorporate additional privacy-preserving sensor modalities with no changes to the core implementation.

3.4 Evaluation and Results

We evaluate VAX along three dimensions: how accurately our A/V pipeline detects activities using models trained on reference homes, how well the end-to-end VAX pipeline trains privacy-preserving sensor models from these A/V-generated labels, and how VAX compares with baseline approaches that train privacy-preserving sensors directly.

3.4.1 Performance of the A/V Pipeline

Figure 3.8 shows the performance of our A/V pipeline, trained in a leave-one-home-out fashion, across all participants for activities whose confidence score exceeds the cutoff thresholds (60% for the audio-only ensemble and 40% for the audio-video ensemble). We observed that 75% of all activity instances meet these criteria and are thus labeled by the A/V pipeline. Across these detected activities, our A/V pipeline achieves high accuracy, precision, and recall, and the performance is consistent across homes. This consistency indicates that the information captured from A/V models trained on one set of reference homes transfers reliably to new homes.

Figure 3.12(a) shows the confusion matrix for activity predictions from the A/V pipeline, averaged across all participants. The pipeline reliably identifies most activities with observable sound signatures, such as WashingDishes, Blender, and ToiletFlushing. It also performs well for activities with unique motion signatures, like Exercising and HairBrush. The highest confusion

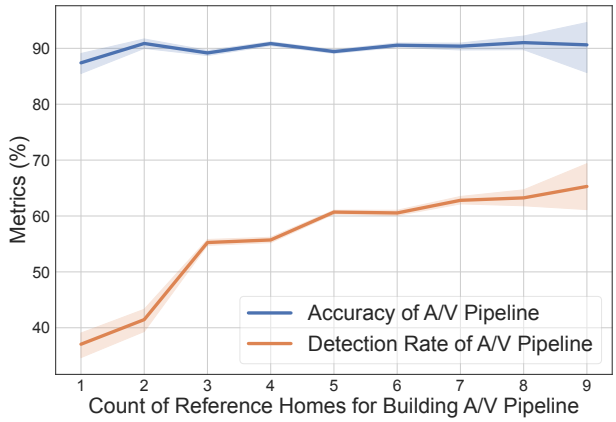


Figure 3.10: A/V pipeline: Accuracy and Detection Rate as the number of reference homes increases. Activity detection accuracy (correct predictions out of detected activities) remains high even with few reference homes, while the detection rate (number of activities detected with high confidence) increases monotonically with more reference homes.

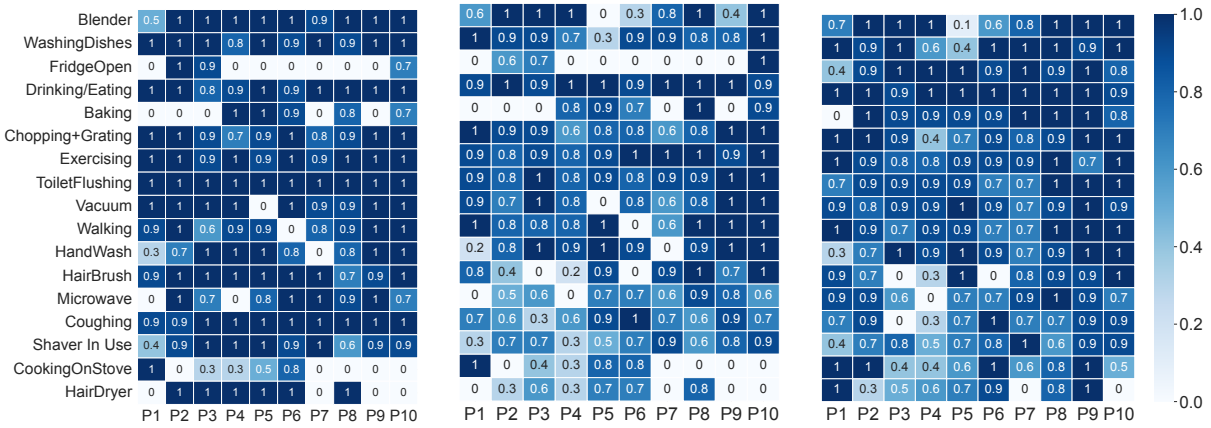
occurs between activities that involve similar body poses: Baking and FridgeOpen are frequently confused because both involve opening and closing doors, and Chopping+Grating is sometimes classified as CookingOnStove because both involve similar hand motions in close proximity to a kitchen surface. Activities with very little motion and insufficient sound, such as CookingOnStove, are the hardest to detect using audio or video models alone.

Figure 3.10 evaluates how the number of reference homes used to train the A/V model affects accuracy and detection rate. As the number of reference homes increases, the count of detected activities increases monotonically. Importantly, even with few reference homes and a low detection rate, the A/V pipeline maintains high accuracy on the activities it does detect. This means that when the A/V labels are used to train privacy-preserving sensor models, the training may proceed slowly (not all activities will have labels early on), but the labels that are generated will be accurate, which is the more important property for downstream model quality.

Figure 3.11(a) shows the per-activity, per-participant F1-Score for the A/V pipeline. Our pipeline can classify several activities (Blender, WashingDishes, Exercising, *etc.*) reliably across almost all homes. However, across all homes, we miss true detection for at least one activity (*e.g.*, P3) and at most five activities (*e.g.*, P7), with approximately two undetected activities per home on average.

3.4.2 Performance of the VAX Pipeline

Once we obtain labels from the A/V pipeline, we train our privacy-preserving sensors using leave-one-instance-out cross-validation for each participant. Figure 3.11(b) shows the F1-Score for VAX when trained exclusively on labels from the A/V pipeline with no user input. As expected, activities that the A/V pipeline could not detect receive no training labels and therefore cannot be recognized by the privacy-preserving sensor models either. In a real deployment, VAX would identify these undetected activities by examining which activity classes are absent from the A/V



(a) A/V pipeline.

(b) VAX pipeline (A/V labels only, no user input).

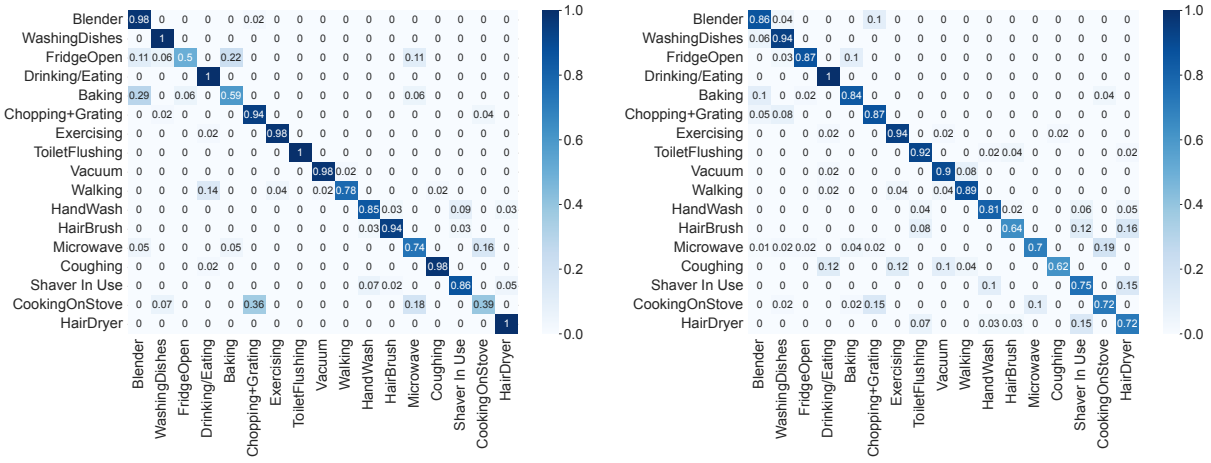
(c) VAX pipeline (single user input on undetected activities).

Figure 3.11: F1-Score for activity detection across participants for three conditions: (a) the A/V pipeline alone, (b) the VAX privacy-preserving sensor pipeline trained on A/V labels only (no user input), and (c) the VAX pipeline with one user-provided label for each undetected activity. In the A/V pipeline, most activities are detected across all homes, but activities like Baking, FridgeOpen, and CookingOnStove are missed in most homes due to high inter-activity confusion leading to low confidence. The VAX pipeline with A/V labels only (Figure 3.11(b)) performs well on detected activities but shows no detection for activities absent from the A/V labels. With a single user input per undetected activity (Figure 3.11(c)), VAX performs well across all activities, showing that the privacy-preserving sensors' classification capability is limited only by the availability of at least one labeled instance per activity.

predictions and ask the user to provide one labeled instance for each.

Figure 3.11(c) shows the F1-Score across all activities and participants once we include one labeled instance per activity (either from the A/V pipeline or a simulated user input for undetected activities). We observe that the VAX pipeline can reliably classify activities with only a single sample for previously undetected classes. Notably, the F1-Score for activities that *were* included in the A/V pipeline also improves in this condition. This improvement occurs because the additional ground-truth labels reduce overall noise in the training data, benefiting the models across all activity classes.

Figure 3.9 shows the aggregate performance of VAX with one label for each activity class (either generated automatically by the A/V pipeline or provided by the user): an average accuracy of 84%, precision of 84%, and recall of 84% across all participants. This result demonstrates the effectiveness of fusing multiple privacy-preserving sensors for activity classification. Figure 3.12(b) shows the corresponding confusion matrix. Detection is good across most activities due to clean differentiating signatures in one or more sensors. The weakest performance occurs for bathroom activities such as HairBrush, HairDryer, and ShaverInUse, which are all performed at roughly the same location with limited body movement. For these activities, the system relies primarily on small variations in hand and torso movement captured by the Doppler sensor. With more training samples, the Doppler models should be able to better separate these activities,



(a) A/V pipeline.

(b) Privacy-preserving sensor pipeline.

Figure 3.12: Activity confusion matrices averaged across all participants for (a) the A/V pipeline and (b) the privacy-preserving sensor pipeline. The A/V pipeline performs well for activities with descriptive sound signatures (WashingDishes, Blender, ToiletFlushing) but confuses activities with similar pose patterns, such as FridgeOpen and Baking (both involve opening and closing doors). The privacy-preserving sensor pipeline detects most activities well due to differentiating signatures in one or more sensors, with the weakest performance on bathroom activities like HairBrush, HairDryer, and ShaverInUse, where limited body movement leaves the system reliant on small Doppler variations.

though confirming this hypothesis would require a longer-term deployment.

Figure 3.13 breaks down the F1-Score by individual sensor across activities. For most activities, Thermal, Doppler, and Lidar sensors show the highest F1-Scores, suggesting that in a practical deployment, even a subset of these three sensors might provide sufficient accuracy while reducing cost. The Micarray shows good performance for activities with distinctive sound signatures, such as HairBrush, Coughing, Blender, WashingDishes, Drinking/Eating, and Vacuum. Other low-fidelity sensors, including environmental sensors and PIR motion, contribute minimally across all activities, suggesting they could be removed or replaced in future iterations of the VAX hardware without significant impact on accuracy.

3.4.3 Comparison with Baseline Approaches

Figure 3.14 shows a head-to-head comparison of VAX with two baseline approaches. The first baseline, *X-Only (pre-trained)*, trains models on privacy-preserving sensor data from the reference homes and runs prediction on a new home in a leave-one-home-out fashion. The second baseline, *X-Only (no pre-training)*, trains a model from scratch in each test home using privacy-preserving sensor data with labels collected directly from the user. We compare these baselines with VAX in two settings.

When no user input is provided (Figure 3.14(a)), the X-Only (no pre-training) baseline has no labels to train on, so it cannot operate. Compared with X-Only (pre-trained), VAX performs

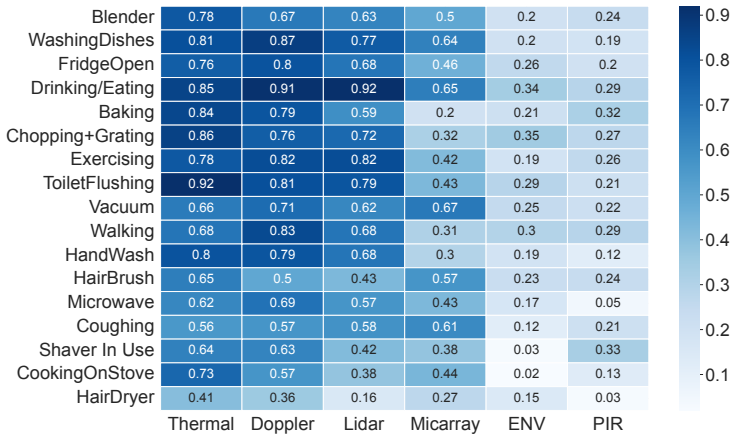


Figure 3.13: Privacy-preserving sensor pipeline: F1-Score at the activity level from individual sensors. Doppler, Thermal, and Lidar show the highest F1-Scores across most activities. The Micarray performs well for activities with distinctive sound signatures. Environmental sensors and PIR motion contribute minimally across all activities.

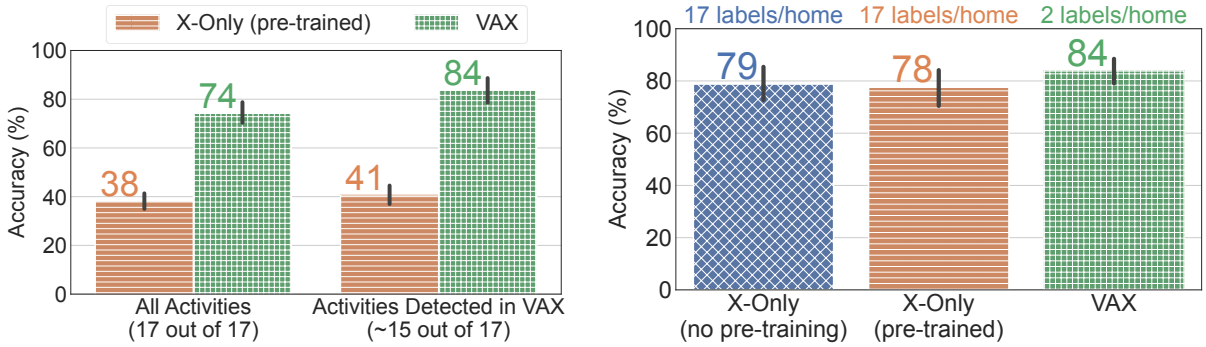
considerably better across all activities (74% *vs.* 38%). When we restrict the comparison to activities that the A/V pipeline detects (15 out of 17 on average), VAX's accuracy increases from 74% to 84%, whereas X-Only (pre-trained) improves only marginally from 38% to 41%. This result shows that VAX's performance is primarily limited by the A/V pipeline's ability to detect at least one instance of each activity, rather than by the capacity of the privacy-preserving sensors themselves.

When user input is provided (Figure 3.14(b)), we give the X-Only (no pre-training) baseline one label per activity (17 labels per home) and fine-tune the X-Only (pre-trained) baseline with the same 17 labels. For VAX, we provide only one label for each activity undetected by the A/V pipeline (approximately 2 labels per home on average). Even with considerably less labeling effort (2 labels *vs.* 17 labels per home), VAX achieves better performance than both baselines (84% *vs.* 79% and 78%). We also observe that VAX's accuracy across all activities with user input on undetected activities (84%) closely matches its accuracy without any user input on activities detected by the A/V pipeline (84%), confirming that VAX's performance ceiling is determined by the ability of the off-the-shelf A/V models to separate the chosen activities, and that filling in the undetected activities with a single user-provided label is sufficient to reach that ceiling.

3.5 Discussion

This section discusses the limitations of the current VAX prototype and identifies opportunities for future improvement.

Modular hardware design: Our results point to a limitation of the monolithic rig design we used for deployment. By integrating all sensors, including the camera, microphone, and privacy-preserving sensors, along with the Intel NUC compute node into a single apparatus mounted on a tripod, we ensured ease of deployment and portability across homes and locations. However, this



(a) No User Input.

(b) With User Input.

Figure 3.14: Comparing VAX with two baseline approaches. Without any user input (left), VAX substantially outperforms the baseline on both activities detected by the A/V pipeline (84% *vs.* 41%) and across all activities (74% *vs.* 38%). With user input (right), VAX achieves slightly better performance with considerably less labeling effort (∼2 labels/home *vs.* 17 labels/home) compared to baselines trained directly on privacy-preserving sensor data.

design also meant a single vantage point for all sensors in each room. In several cases, we could not obtain A/V labels for certain activities due to occlusion (*e.g.*, the participant's back was to the camera, or furniture blocked the line of sight) or poor viewing angles. This is a significant limitation because the A/V models are otherwise largely agnostic to vantage point; if they cannot observe an activity, no labels are generated, and the privacy-preserving sensors cannot be trained for that activity at all. In a future iteration, making VAX modular so that the A/V sensors can be placed at different locations from the privacy-preserving sensors could improve detection coverage. Similarly, some privacy-preserving sensors are directional (*e.g.*, the Thermal sensor, PIR) with a limited field of view, while others are more sensitive depending on their placement (*e.g.*, the IMU for measuring structural vibrations). Having more than one instance of these sensors at different locations could boost accuracy. Furthermore, a portable A/V sensing solution, such as using a smartphone to capture A/V labels, could be deployed opportunistically across multiple rooms, training privacy-preserving sensors in different environments with a single VAX system.

In-the-wild deployment: While we evaluated VAX in a diverse set of 10 homes, our data collection for each home lasted only a few hours, during which we asked participants to perform specific activities while we recorded A/V data and ground-truth labels. This setup, though a significant endeavor for our team, is not what an in-the-wild deployment of VAX would look like. We envision a scenario where the VAX rig is deployed in a person's home for a day or two while the occupant goes about their routine without any constraints. During this period, the A/V pipeline would generate labels as activities naturally occur, and the privacy-preserving sensor models would be trained and refined progressively. After this bootstrapping period, the system would begin predicting activities using only the privacy-preserving sensors, and the camera and microphone could be unplugged. Exploring this unconstrained, longitudinal deployment remains future work.

Sensor selection and cost: Our evaluation (Figure 3.13) shows that Doppler, Thermal, and Lidar

sensors capture the widest range of activities with the highest accuracy, while environmental sensors and PIR contribute minimally. However, all three high-performing modalities require a direct field of view, and installing multiple instances across an entire home could become costly. One practical approach is to select sensors opportunistically based on the types of activities expected in a given room: a Thermal and Doppler sensor may suffice for a kitchen, where most activities have distinctive thermal signatures or movement patterns, while a Doppler sensor alone may be sufficient in a bathroom, where activities are spatially constrained and distinguished primarily by motion. Beyond the modalities we explored, incorporating sensors like downsampled audio [142] or custom ML sensor hardware [217] with VAX could expand the range of detectable activities while keeping costs manageable.

3.6 Summary and Looking Ahead

This chapter presented VAX, an end-to-end system that uses off-the-shelf audio and video ML models to generate activity labels for in-situ training of privacy-preserving sensors. The A/V pipeline combines the output of multiple models through a learned non-linear mapping to a consistent activity vocabulary, and augments its detection rate using unsupervised clustering on privacy-preserving sensor data. Once the privacy-preserving sensor models are trained, the camera and microphone are removed. Across ten homes and 17 activities, our A/V pipeline detects approximately 15 out of 17 activities with no human supervision, and with just one user-provided label for each undetected activity (approximately 2 out of 17 per home), the privacy-preserving sensors achieve 84% accuracy, matching a fully supervised baseline while requiring roughly one-eighth of the labeling effort.

VAX demonstrates that the bootstrapping principle, temporarily deploying a richer sensing source to train a more privacy-preserving one, is viable for the training label stage of the sensing pipeline. However, VAX operates under an assumption that the preceding discussion did not address: the set of 17 activities was chosen before deployment, curated from the literature, and fixed across all homes. In practice, deciding *what* activities to detect in a new environment is itself a specification problem. The right vocabulary depends on what the occupant actually does, what the deployed sensors can distinguish in that particular physical arrangement, and how finely activities should be differentiated. A predefined taxonomy cannot account for this interaction. The next chapter presents OrganicHAR, which addresses this activity vocabulary problem by discovering activities directly from sensor data, at a granularity matched to what the sensors can actually resolve.

Chapter 4

Discovering Activities from Sensor Patterns

The previous chapter showed that privacy-preserving sensors can be trained to recognize activities without requiring occupants to label data themselves. By temporarily pairing a camera and microphone with the target sensors, VAX uses off-the-shelf audio and video models to generate training labels automatically, and once the sensor models are trained, the privileged modalities are removed. This eliminates the first bottleneck in the sensing pipeline described in Chapter 1, the need for per-environment labeled data. But VAX, like most activity recognition systems, assumes that the set of activities to detect has been decided in advance. Someone must specify a vocabulary before data collection begins, and the system can only recognize what that vocabulary contains. Everything else is relegated to an uninformative "Other" category.

This assumption is limiting, as the right set of activities for a given space depends not only on what the occupant does, but on what the available sensors can distinguish in that particular physical arrangement. Consider a kitchen where tea, coffee, and hot chocolate are all prepared at the same countertop appliance. A radar placed across the room sees nearly identical motion patterns for all three, making "preparing a hot beverage" the finest granularity it can reliably support. In a different kitchen, where tea is made on the stove and coffee with a machine on a separate counter, the same radar can distinguish the two because they involve movement to different locations. A predefined vocabulary cannot account for this interaction between environment, sensor placement, and occupant behavior. Too fine-grained a vocabulary leads to poor accuracy when sensor signals are genuinely indistinguishable; too coarse a vocabulary discards the detail that would make the system useful.

4.1 Introduction

This chapter presents OrganicHAR, a framework that removes the need for a predefined activity vocabulary by discovering what activities occur in an environment directly from the sensor data. The core idea is a *sensor-first* inversion of the conventional paradigm. Rather than defining activities and then training sensors to detect them, OrganicHAR lets the sensors reveal what patterns exist, and then uses a vision-language model (VLM) to understand what those patterns represent. Privacy-preserving sensors run continuously, identifying recurring signal patterns and moments of significant change. Only at these *key moments*, is a camera temporarily activated

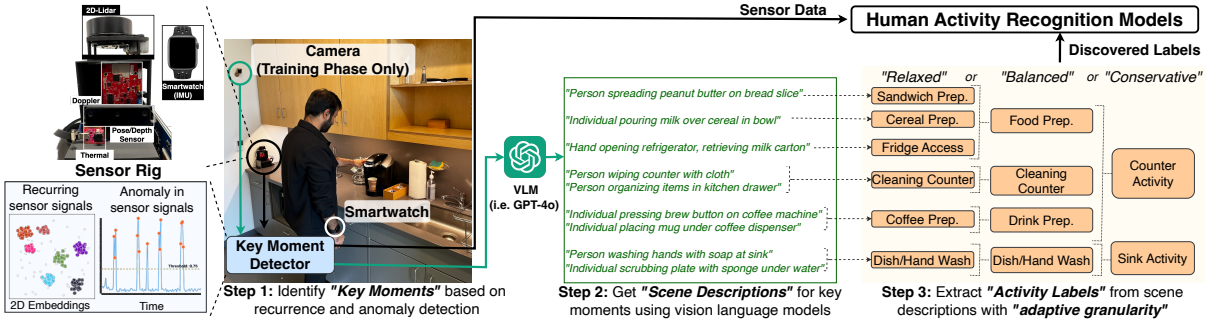


Figure 4.1: OrganicHAR's sensor-first workflow for activity discovery. Privacy-preserving sensors identify recurring signal patterns and temporal anomalies (Step 1), which are used to select *key moments* for targeted VLM analysis that generates natural language activity descriptions (Step 2). These descriptions are then organized into hierarchical activity labels at adjustable granularity levels (Step 3). Unlike prior approaches that require predefined activity sets, fixed granularity, and continuous video, OrganicHAR discovers activities from sensor patterns, adapts granularity to sensor capabilities, and processes only key moments.

and a VLM consulted to generate natural language descriptions of what is happening. A large language model (LLM) then consolidates these descriptions into a coherent set of activity labels, organized hierarchically so that the granularity can be adjusted through a single parameter to match what the sensors can reliably tell apart. Once the labels are established, activity recognition models are trained on the privacy-preserving sensors. The result is an activity vocabulary that is tailored to both the occupant's behavior and the sensor configuration's capabilities, discovered without manual annotation or designer anticipation.

OrganicHAR is motivated by three requirements that emerged from our prior work on privacy-preserving activity recognition. First, we need to discover what activities *actually occur* in an environment, rather than assuming a fixed set. The diversity of real-world behavior, the way people improvise, combine tasks, and develop personal routines, means that any predefined taxonomy will miss activities that matter and include activities that never happen [13, 14, 15, 16]. Second, we need to determine what activities can be *sensed* by the system, since the answer depends on which sensors are deployed and how the physical environment shapes their signals. Third, we need to identify which activities are *beneficial to track* for the occupant, since not every detectable pattern is worth reporting. These three requirements, what occurs, what can be sensed, and what is useful, are inherently local to each deployment and cannot be resolved at design time.

Prior approaches to reducing the annotation burden in new environments have not addressed this vocabulary problem directly. Clustering-based methods can group similar sensor patterns without predefined labels [17, 54], but the resulting clusters require manual interpretation to assign meaningful names [88, 221], and they offer no mechanism to select a granularity that matches sensor capabilities. Interactive learning strategies gradually refine activity models based on user feedback [106], but they require significant user involvement and struggle to capture the full diversity of activities in a space. Systems that temporarily deploy cameras to capture ground truth labels, including our own VAX, bootstrap models for privacy-preserving sensors but remain limited to a predefined activity set and struggle with complex or unknown activities [140, 154].

Cross-modal transfer learning approaches, reviewed in Chapter 2, convert training data from richer modalities into synthetic signals for privacy-preserving sensors [11, 113, 123], but they transfer knowledge *into* predefined activity sets rather than discovering what the set should be. What distinguishes OrganicHAR from all of these is that it uses the privacy-preserving sensors themselves to determine when and where to look, minimizing both the volume of video data processed and the need for human involvement, while producing an activity vocabulary grounded in what the sensors can actually detect.

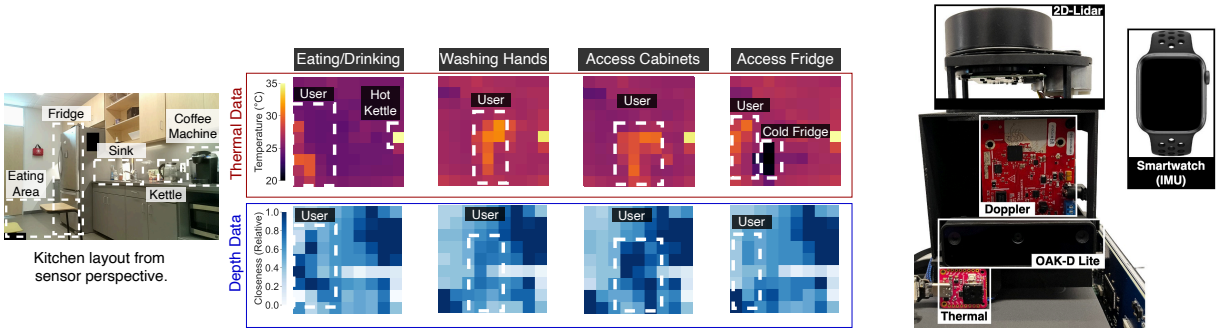
To build and evaluate OrganicHAR, we collected a multimodal dataset from 12 participants who performed four breakfast preparation tasks (preparing sandwiches, tea, coffee, and cereal) across three distinct kitchen environments. Participants received no instructions on how to perform these tasks, allowing us to capture organic activity patterns as they would occur in real homes. Our evaluation demonstrates that OrganicHAR can reliably discover 4 to 8 activity labels per user (15 unique labels across all participants) with 87% accuracy when compared to human-annotated ground truth. Activity recognition models trained on these discovered labels achieve 70–88% accuracy depending on sensor configuration and semantic granularity, with performance comparable to what predefined-label systems achieve under similar sensing constraints. The framework processes only approximately 10% of the total video data during the training phase, and we show that this fraction decreases over time as the system accumulates more training data. We further validated OrganicHAR through deployment in five homes for seven days each, where the system adapted to natural daily routines and diverse kitchen environments, achieving 74.2% accuracy while demonstrating effective learning over time.

The remainder of this chapter is organized as follows. Section 4.2 describes the sensing infrastructure, building on the platform introduced in Chapter 3 with modifications tailored to OrganicHAR's requirements. Section 4.3 presents the system design, tracing the pipeline from raw sensor signals through key moment identification, label discovery, and granularity control to trained activity recognition models. Section 4.4 describes our data collection protocol, and Section 4.5 reports the evaluation across controlled and real-world settings. Section 4.6 discusses the implications of our findings, and Section 4.7 summarizes the chapter's contributions and their connection to the next stage of the pipeline.

4.2 Sensing Infrastructure

Our sensing platform for OrganicHAR builds on the hardware infrastructure introduced in Chapter 3, retaining the core privacy-preserving modalities that proved effective for activity recognition while making two kinds of changes: removing sensors that did not contribute meaningfully in our target domain, and adding new modalities that keep data local while giving users a richer set of privacy-capability tradeoffs to choose from.

From the VAX platform, we retained the three ambient sensors that provided the strongest activity signatures across our deployments: the 77GHz FMCW Doppler radar (AWR1642BOOST-ODS) operating at 5Hz for movement sensing, the 2D Lidar (Slamtec RPLIDAR A1M8) providing 360-degree distance measurements at 6–8Hz for spatial occupancy, and the low-resolution thermal array (FLIR Lepton 3.5, downsampled to 10×10 pixels) capturing thermal signatures at 8Hz. These modalities are described in detail in Section 2.1.2. We removed the sound localization



(a) Thermal data (10×10) and Depth Array information at a single timestamp (snapshot).

(b) Hardware setup used for data collection.

Figure 4.2: Privacy-preserving sensor data from the OrganicHAR sensing platform. (a) shows snapshots of the low-resolution thermal array and depth array at a given timestamp for various kitchen activities. For depth visualization, pixels grow darker when the user is closer to the sensor. The Doppler and Lidar visualizations for this platform follow the same patterns described in Chapter 2 (Figure 2.1). (b) shows our hardware rig, which integrates a 2D Lidar, Doppler radar, OAK-D Lite stereo camera, and thermal sensor into a compact form factor.

array and the ambient environmental sensors (the Mites platform [139] providing temperature, humidity, light, and vibration data) that were part of the VAX rig. In our kitchen deployments, these sensors contributed limited discriminative information for the activities we aimed to discover: the sound localization array has low signal to noise ratio (SNR) and struggled with precision due to factors like capturing reflected sound from different angles, and the environmental sensor readings changed too slowly to capture the temporal dynamics of kitchen tasks. Removing them simplified the hardware without measurably reducing recognition capability.

We introduced two new sensing modalities designed to provide richer activity information while keeping all data processing local to the device. The first is a *wearable IMU* captured through a custom iOS application running on an Apple Watch, providing accelerometer, gyroscope, and magnetometer data at 50Hz. The wearable captures personalized hand and arm movements that complement the ambient sensors' view of the scene, enabling the system to distinguish between activities that look similar from a fixed vantage point but involve different hand movements, such as stirring versus pouring at the same countertop location. The second addition is the *OAK-D Lite* stereo camera platform [189], which runs pose estimation and depth computation entirely on-device using its integrated vision processing unit. The OAK-D outputs only processed representations: normalized pose coordinates at 640×480 resolution and low-resolution depth maps (Figure 4.2(a)). No raw images leave the device, making this an important middle ground between the privacy guarantees of non-optical sensors and the spatial richness of camera-based systems. Together, pose and depth information provide the system with awareness of the user's body configuration and spatial relationship to objects, which proves particularly valuable for disambiguating fine-grained activities that involve similar ambient sensor signatures but different postures or locations within the kitchen.

These sensing modalities are organized into three configuration tiers, each representing

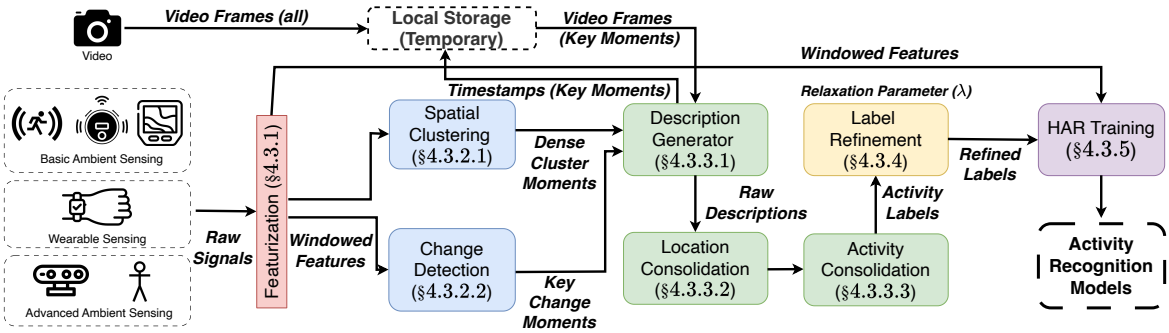


Figure 4.3: Overall architecture of OrganicHAR. Raw sensor signals from different hardware configurations (Section 4.2) are featurized for each sensor separately (Section 4.3.1), and used to identify *key moments* (Section 4.3.2) for targeted video processing using VLMs. The raw descriptions are then converted to discrete activity labels using LLMs (Section 4.3.3), and further refined using a relaxation parameter λ (Section 4.3.4). The final activity labels are used to train activity recognition models (Section 4.3.5) for deployment.

a different point along the privacy-capability tradeoff. The first tier, *Ambient (Basic) Only*, includes only the Doppler radar, 2D Lidar, and low-resolution thermal array. These non-optical sensors cannot capture any personally identifiable information, requiring no user interaction after installation. The second tier, *Ambient (Basic) + Wearable (IMU)*, adds the smartwatch-based motion data, enabling personalized motion tracking at the cost of requiring the user to wear the device consistently. The third tier, *Ambient (Advanced) + Wearable (IMU)*, adds the OAK-D Lite's on-device pose and depth processing. While the OAK-D contains optical hardware, its on-device processing ensures that no raw images are transmitted or stored, though the presence of an optical sensor may raise concerns in some settings, making this configuration better suited for common areas where the user is comfortable with its presence. Emerging work on ML sensors that constrain functionality at the hardware level [63, 143, 217] could address these concerns in the future. By offering multiple configuration tiers, we allow each deployment to balance privacy preferences with recognition capabilities based on individual needs and spatial contexts, and as we demonstrate in our evaluation (Section 4.5), our framework adapts its activity discovery to whatever sensing tier is available.

Our current prototype integrates these sensing modalities (Figure 4.2(b)) through a small form-factor PC (Intel NUC, 8-core, 16GB RAM) that handles all sensor data acquisition and processing locally. For the initial training phase only, we use an iPhone for video capture to enable VLM-based scene understanding. This video stream is processed during key moments identified by the privacy-preserving sensors, as described in the next section, and is not retained after the training phase is complete.

4.3 System Design

Figure 4.3 presents the overall architecture of OrganicHAR. The framework operates through a sensor-first pipeline where privacy-preserving sensors drive activity discovery rather than

responding to a predefined vocabulary. Raw sensor signals from the hardware configurations described in Section 4.2 are first featurized independently for each modality (Section 4.3.1). These features feed into two complementary mechanisms for identifying *key moments* that warrant targeted video analysis (Section 4.3.2): spatial clustering in the feature space to find recurring activity patterns, and temporal change detection to capture transitions between activities. Only at these key moments is a VLM consulted, generating natural language descriptions of the scene from brief video segments. An LLM-based pipeline then consolidates these descriptions into discrete activity labels (Section 4.3.3), and a semantic granularity control mechanism allows the label set to be tuned from coarse to fine-grained through a single relaxation parameter λ (Section 4.3.4). Finally, these discovered labels serve as training data for activity recognition models that operate solely on privacy-preserving sensors during deployment (Section 4.3.5), completing the bootstrapping process.

4.3.1 Featurization

Each sensor modality produces signals with distinct characteristics and limitations, and we designed tailored featurization pipelines to maximize their individual strengths before combining them. We employ a sliding window approach with fixed-length 5-second segments and a 0.5-second stride for both training and inference. For feature extraction at timestamp t_s, we compute features using sensor data from the interval $[t_s - 5, t_s]$ seconds. This window size is consistent with recent sensor fusion approaches [25, 199] and provides sufficient temporal context to capture the structure of short-duration activities while maintaining computational efficiency. During activity recognition, we aggregate consecutive predictions from overlapping windows to determine activity boundaries and durations, accommodating both brief interactions and extended tasks. This multimodal featurization achieves complementary fusion across sensor modalities, enabling our system to identify scene-relevant moments and activity patterns more robustly than any single modality could alone. We provide detailed per-sensor featurization specifications in Appendix A.1.

4.3.2 Key Moment Identification

Once we have featurized data from each sensor modality, we identify *key moments*: segments of sensor data that warrant further analysis by the VLM. This is a critical step in our pipeline, as it determines how much video data needs to be processed and whether the selected moments capture the diversity of activities occurring in the environment. We process the featurized sensor streams through two complementary approaches: a spatial clustering module that identifies recurring patterns representing consistent activities in the feature space, and a temporal change detection module that identifies segments with notable signal shifts indicating activity transitions. Together, these two mechanisms ensure that we capture both the steady-state activities that define a person's routine and the transitional moments that mark boundaries between them.

4.3.2.1 Spatial Clustering

We employ clustering in the feature space of each sensor modality to identify recurring patterns representing distinct activity signatures. While we refer to this as "spatial clustering," it operates

not in the physical space but in the high-dimensional feature space specific to each sensor, where signals from similar activities form natural clusters regardless of their physical location in the environment.

We use HDBSCAN (Hierarchical Density-Based Spatial Clustering of Applications with Noise) [136], selected for its ability to discover clusters of varying densities without requiring a predefined cluster count, a property that is important for unsupervised activity discovery in unfamiliar environments where we do not know in advance how many distinct activities will emerge. After applying standard preprocessing (robust scaling and dimensionality reduction using principal component analysis), we optimize the clustering process through careful hyperparameter tuning. The key hyperparameters governing our feature space clustering include min_cluster_size, which determines the minimum number of points required to form a cluster and directly influences the granularity of detected activity patterns; min_samples, which controls how conservative the clustering is, with higher values producing more stringent cluster formation; and n_components, which defines the dimensionality of the feature space after reduction.

Central to our approach is a scoring function that evaluates clustering quality across all sensor modalities for a given combination of hyperparameters:

$$\text{score} = w_1 \cdot S_{count} + w_2 \cdot S_{noise} + w_3 \cdot S_{prob} \tag{4.1}$$

This function balances three aspects of effective activity clustering. S_{count} evaluates how well the number of discovered clusters aligns with expected activity diversity (between C_{min} and C_{max} clusters), penalizing both overly granular and overly coarse solutions. S_{noise} assesses the proportion of data points assigned to meaningful clusters rather than noise, preferring solutions with noise ratios below N_{max}, which helps separate random sensor activations from genuine activity patterns. S_{prob} captures the average cluster membership probability, reflecting confidence in cluster assignments and favoring well-defined, distinct activity signatures. The weighted combination balances cluster count appropriateness (w_1) with noise handling and assignment confidence (w_2 and w_3).

From each cluster, we select representative windows with high membership confidence as key moments for VLM analysis, significantly reducing the required video processing while capturing the essential activity patterns. Our approach excels at activity discovery by identifying natural signal groupings that correspond to distinct behaviors: cooking activities might cluster together in Doppler feature space despite occurring at different kitchen locations, while refrigerator interactions might form clusters in thermal feature space despite varying user movements. Through empirical evaluation across sensors, we selected hyperparameter ranges that provide effective clustering: min_cluster_size $\in \{3\text{–}8\}$, min_samples $\in \{2\text{–}5\}$, and n_components $\in \{8\text{–}80\}$, with modality-specific adjustments (*e.g.*, thermal sensing uses cluster_selection_epsilon $\in \{0.01\text{–}0.03\}$ for tighter clusters). The scoring weights ($w_1 = 0.3\text{–}0.4$, $w_2 = 0.15\text{–}0.3$, $w_3 = 0.1\text{–}0.3$) and target parameters (desired clusters between 20–40, maximum noise ratio of 0.8) are consistent across modalities.

4.3.2.2 Temporal Change Detection

Complementing the spatial clustering, we also leverage temporal changes in the signals of each modality, expecting that these transitional moments capture useful activity boundaries. We use an

online, unsupervised outlier detection algorithm based on a Gaussian Mixture Model (GMM), specifically the approach proposed by Yamanishi *et al.* [224]. This algorithm computes an anomaly score for each incoming data sample based on its likelihood under the current GMM and simultaneously updates the model parameters, enabling it to capture changes in time-series signals without prior knowledge of what those changes might represent.

Specifically, let $\mathbf{y}_t \in \mathbb{R}^M$ denote the input sample at time t, and let $\pi_{i,t}$, $\mu_{i,t}$, and $\Sigma_{i,t}$ represent the weight, mean vector, and covariance matrix of the ith component (for $i = 1, \ldots, K$) of the GMM at time t. The anomaly score for $\mathbf{y}_t$ is defined as

$$s_t = -\ln\left(\sum_{i=1}^{K} \pi_{i,t-1}\,\mathcal{N}\left(\mathbf{y}_t \mid \mu_{i,t-1}, \Sigma_{i,t-1}\right)\right). \tag{4.2}$$

Following this, the GMM parameters are updated according to

$$\begin{aligned}
\lambda_{i,t} &= \frac{\pi_{i,t-1}\,\mathcal{N}\left(\mathbf{y}_t \mid \mu_{i,t-1}, \Sigma_{i,t-1}\right)}{\sum_{j=1}^{K} \pi_{j,t-1}\,\mathcal{N}\left(\mathbf{y}_t \mid \mu_{j,t-1}, \Sigma_{j,t-1}\right)}, \\
\pi_{i,t} &= (1-\alpha)\,\pi_{i,t-1} + \alpha\,\lambda_{i,t}, \\
\tilde{\mu}_{i,t} &= (1-\alpha)\,\tilde{\mu}_{i,t-1} + \alpha\,\lambda_{i,t}\,\mathbf{y}_t, \\
\mu_{i,t} &= \frac{\tilde{\mu}_{i,t}}{\pi_{i,t}}, \\
\tilde{\Sigma}_{i,t} &= (1-\alpha)\,\tilde{\Sigma}_{i,t-1} + \alpha\,\lambda_{i,t}\,\mathbf{y}_t\mathbf{y}_t{}^T, \\
\Sigma_{i,t} &= \frac{\tilde{\Sigma}_{i,t}}{\pi_{i,t}} - \mu_{i,t}\mu_{i,t}^T,
\end{aligned} \tag{4.3}$$

where α is a forgetting factor that controls the influence of past observations. After computing the anomaly score for each t using Equation 4.2, we select the top N points as detected moments. The parameters K, M, and α are hyperparameters of this component. We empirically set $K = 2$, $M = 10$, and $\alpha = 0.1$, which works well on average across all modalities.

4.3.3 Label Discovery

After identifying key moments through spatial clustering and temporal change detection, our framework translates these sensor-identified moments into meaningful activity labels through a multi-stage process. This is where the sensor-first approach connects to semantic understanding: the sensors have told us *when* something interesting is happening, and now we need to determine *what* it is.

4.3.3.1 Semantic Description Generation

For each key moment, we prompt a VLM to generate semantic descriptions from video frames. This is the only stage in the pipeline where video data is processed, and it occurs exclusively during the initial training phase. We iteratively tested multiple configurations to determine optimal video input parameters, and found that higher frame rates and resolutions frequently led to

hallucinations and inconsistent descriptions. Our empirical testing revealed that low-resolution frames (640×480) sampled at 1 FPS for 5-second clips provided the best balance between detail and consistency for current VLM capabilities (as of July 2025), though this balance may shift as vision-language models improve.

A critical challenge in prompt design was achieving dual objectives: controlling VLM speculation while balancing specificity and consistency. VLMs tend to infer intentions beyond what is directly observable and struggle with appropriate granularity, either overgeneralizing activities (*e.g.*, labeling everything as "kitchen activity") or focusing on irrelevant details. We addressed these tensions by explicitly instructing the model to *"focus only on what you can clearly see"* while providing structured categories guiding appropriate detail levels. We directed the VLM to analyze four aspects: (1) *actions*, including movements between locations; (2) *objects* the person interacts with; (3) *location* where the activity occurs; and (4) *activity structure*, including initial conditions, main actions, and results. For each category, we provided examples at appropriate granularity levels to establish consistent reference points. By requiring JSON-structured output with self-assessed confidence scores, we enabled systematic filtering of observations ($\theta_{conf} = 0.8$), rejecting both speculative and overly generic descriptions. While accurate, the resulting descriptions exhibited inconsistent terminology and granularity across similar activities, a challenge addressed by our label consolidation process. Our complete VLM prompt is provided in Appendix A.2.

4.3.3.2 Location Consolidation

We implemented a two-step label consolidation process, beginning with location consolidation followed by activity description consolidation. The location-based approach works naturally with privacy-preserving ambient sensors, which often provide clean signal separation for spatial differentiation. Additionally, the VLM outputs were more consistent for location references than for action descriptions, providing a stable foundation for subsequent activity labeling.

We designed an LLM prompt to consolidate various location references from VLM outputs (*e.g.*, "at sink," "by sink," "near sink basin") into consistent *functional zones* like "sink area," "counter area," and "coffee machine area." The prompt instructs the LLM to group locations based on the activities they support, merge functionally similar spaces, and maintain separation between distinct activity zones. This spatial organization helps disambiguate semantically similar actions through their context (*e.g.*, distinguishing "washing hands" from "washing dishes" based on precise sink location) and creates a foundation for more accurate activity recognition that leverages the spatial detection capabilities of our sensors.

4.3.3.3 Activity Consolidation

With established functional zones, we transform the unstructured action descriptions from the VLM into discrete activity labels. For each functional zone, we use an LLM prompt to create high-level clusters based on the action descriptions from that zone. The prompt differentiates between actions (physical movements like "pouring" or "washing") and purpose or context (the goal or situation, such as "breakfast preparation" or "dishwashing"). This distinction helps maintain important separations: for example, keeping "pouring cereal" and "pouring coffee" as different

activities despite sharing the same physical action. In the sink area, "washing dish with sponge" and "scrubbing plate with sponge" merge into "washing dishes with sponge," while remaining distinct from "washing hands with soap."

We then use LLM-assisted matching to assign each activity description to existing clusters based on interaction patterns. For each description, we extract action and object information, then use an LLM to calculate weighted similarity scores based on action alignment ($w_a = 60$), object consistency ($w_o = 25$), and location match ($w_l = 15$). These empirically determined weights reflect the relative importance of each factor in determining activity similarity. The mechanism effectively consolidates variations like "filling cup using coffee machine" and "using coffee machine with mug" into "preparing coffee drink" while distinguishing them from functionally different activities occurring at the same location.

4.3.4 Semantic Granularity Control

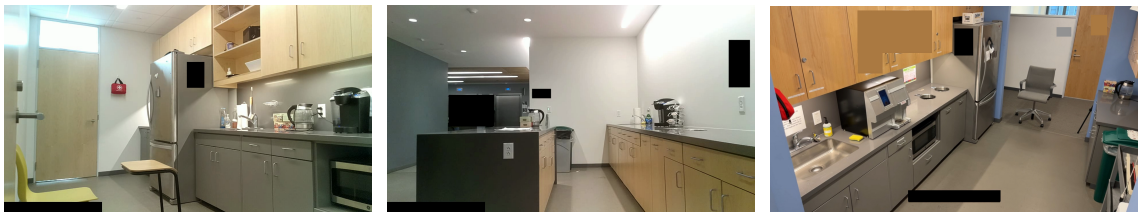
While our two-step consolidation produces well-defined activity labels, we also need to provide control over how aggressively labels are merged at different granularities. From a human perspective, certain distinctions are more meaningful than others: "rinsing dishes" and "washing dishes with sponge" should merge before "washing hands with soap" and "washing dishes" do, despite textual similarity. To capture these intuitions, we analyze semantic relationships across six dimensions. We use an LLM to expand each activity label along: (1) action type (*e.g.*, "cleaning," "preparing"), (2) objects involved (*e.g.*, "dishes," "sponge," "water"), (3) sub-location (*e.g.*, "sink basin," "counter edge"), (4) purpose or goal (*e.g.*, "remove food residue," "prepare beverage"), (5) access patterns (*e.g.*, "reaching for scrubber," "turning faucet"), and (6) related activities (*e.g.*, "rinsing dishes" relates to "drying dishes"). This multi-dimensional characterization provides a rich semantic representation that goes beyond surface-level textual similarity.

We then compute a pairwise similarity matrix S where the score S_{ij} between activity labels a_i and a_j combines weighted cosine similarities across these dimensions:

$$\begin{aligned} S_{ij} = w_{\text{action}} \cdot \text{sim}_{\text{action}}(a_i, a_j) + w_{\text{object}} \cdot \text{sim}_{\text{object}}(a_i, a_j) + w_{\text{location}} \cdot \text{sim}_{\text{location}}(a_i, a_j) + \\ w_{\text{purpose}} \cdot \text{sim}_{\text{purpose}}(a_i, a_j) + w_{\text{access}} \cdot \text{sim}_{\text{access}}(a_i, a_j) + w_{\text{relation}} \cdot \text{sim}_{\text{relation}}(a_i, a_j) \end{aligned} \tag{4.4}$$

For each dimension, we generate textual embeddings using an embedding model and compute cosine similarities between them. The weights reflect each dimension's contribution to human-relevant distinction: action type ($w_{\text{action}} = 0.20$), object involvement ($w_{\text{object}} = 0.25$), sub-location ($w_{\text{location}} = 0.15$), purpose or goal ($w_{\text{purpose}} = 0.15$), interaction patterns ($w_{\text{access}} = 0.15$), and explicit relationships ($w_{\text{relation}} = 0.10$).

Finally, we apply hierarchical clustering with a relaxation parameter λ to group activity labels into clusters C such that $\forall a_i, a_j \in c_k,\ S_{ij} \geq 1 - \lambda$. The values of λ typically range from 0 (one cluster per activity, preserving all distinctions) to 1 (merging all activities into a single cluster). This gives users control over recognition granularity through a single parameter: smaller λ values preserve fine distinctions between activity labels, while larger values merge semantically related activities. In our evaluation, we examine three settings: Conservative ($\lambda = 0.4$), Balanced ($\lambda = 0.3$), and Relaxed ($\lambda = 0.2$), spanning a range from coarse-grained categories to fine-grained activity distinctions.



(a) Kitchen 1 (b) Kitchen 2 (c) Kitchen 3

Figure 4.4: Kitchen environments used in our study: (left) Kitchen 1 with compact galley layout, (middle) Kitchen 2 with island counter, and (right) Kitchen 3 with integrated appliances.

4.3.5 Training Activity Recognition Models

The final phase of our pipeline trains HAR models on the privacy-preserving sensors using the discovered activity labels, after which no video data is required. For each sensor modality, we evaluate multiple classifiers that are robust to imbalanced datasets, implementing leave-one-session-out cross-validation to assess generalizability. This addresses practical sensor deployment challenges including missing values and class imbalance across different environments. After training individual classifiers, we employ grid search to identify suitable sensor-classifier combinations for each functional zone, evaluating two ensemble methods: soft voting (combining probability distributions from multiple classifiers) and hard voting (weighted voting using each classifier's highest-confidence prediction) [22]. Our framework computes balanced accuracy (macro-recall) for each configuration to determine effective deployment combinations.

At inference time, OrganicHAR implements a hierarchical approach: it first identifies the user's functional zone, then applies the appropriate zone-specific activity recognition model using the selected sensor ensemble. This context-aware structure recognizes that activities present different sensor signatures in different zones, and that the most discriminative sensor combination may vary across zones. By activating only the necessary sensors and models for the detected zone, the system balances privacy preservation with recognition accuracy while operating entirely on local sensors after the training phase.

OrganicHAR is implemented in Python with approximately 8,000 lines of code. Sensor data processing uses NumPy [82] for array manipulation, Pandas [151] for time-series analysis, and OpenCV [39] for video processing. The key moment identification implements Gaussian Mixture Models from scikit-learn [159] for temporal change detection and HDBSCAN [136] for spatial clustering. During the training phase, we leverage OpenAI's API [150] with the GPT-4o model to generate semantic descriptions, which are organized into hierarchical activity clusters using SciPy's [211] squareform and linkage functions. For model training, we employ multiple classifier implementations from scikit-learn and imbalanced-learn [120], including RUSBoost [186], Balanced Random Forest [229], EasyEnsemble [129], KNN [45], and SVM [45]. We have open-sourced the OrganicHAR implementation [166].

4.4 Data Collection

We collected data from 12 participants across three distinct kitchen environments (Figure 4.4). Participants performed four breakfast preparation tasks: *preparing tea*, *making coffee from a coffee machine*, *preparing cereal*, and *making sandwiches* with various spreads. Critically, participants received no explicit instructions on how to execute these tasks, and researchers did not intervene or provide prompts during the activities. Each participant approached the tasks in their own way, using natural movements and sequences that reflected genuine everyday behavior rather than scripted demonstrations. The data collection for each participant lasted approximately 1.5 hours, with each task performed 2–3 times with natural variations. To ensure clear segmentation while maintaining ecological validity, participants washed their hands before each task and used a hand clap to mark segment boundaries.

The three kitchenettes represent meaningfully different spatial configurations. Kitchen 1 (participants P1–P4) featured a compact galley-style layout with a single countertop, a refrigerator at one end, a sink in the middle, and a coffee machine at the opposite end. Kitchen 2 (participants P5–P8) had an open layout with an island counter separate from the main preparation area, featuring a sink and coffee machine. Kitchen 3 (participants P9–P12) was a medium-sized kitchenette with a microwave, coffee machine, and sink aligned along a single wall, with a refrigerator perpendicular to the main counter. This diversity of layouts is important because the same activity can produce different sensor signatures depending on the spatial arrangement of appliances and work surfaces, and our evaluation examines how OrganicHAR adapts to these differences.

Our dataset comprises approximately 4,300 five-second segments (roughly 6 hours) of breakfast preparation activities across six sensor modalities: Doppler radar, 2D Lidar, low-resolution thermal array, wearable IMU, on-device pose sensing, and depth processing. Unlike existing HAR datasets featuring scripted activities, our dataset captures the natural variations in how different people perform the same everyday tasks. Ground truth annotation was performed by an unbiased observer who labeled each segment without prior knowledge of the system-generated labels. We have open-sourced the complete dataset [166], including all sensor data, VLM annotations, and ground truth labels.

4.5 Evaluation

Our evaluation of OrganicHAR addresses three research questions. First, how effective and efficient is our activity discovery pipeline in identifying meaningful activity labels compared to traditional approaches (Section 4.5.1)? Second, how do the HAR models trained on these discovered labels perform across different sensor configurations and semantic granularity settings (Section 4.5.2)? Third, how does OrganicHAR perform in real-world deployment scenarios, and how does its performance evolve over time (Section 4.5.3)? For each participant and sensor configuration, we evaluated OrganicHAR across three semantic granularity settings: Conservative ($\lambda = 0.4$), Balanced ($\lambda = 0.3$), and Relaxed ($\lambda = 0.2$), corresponding to increasingly granularity of semantically similar activities as described in Section 4.3.4.

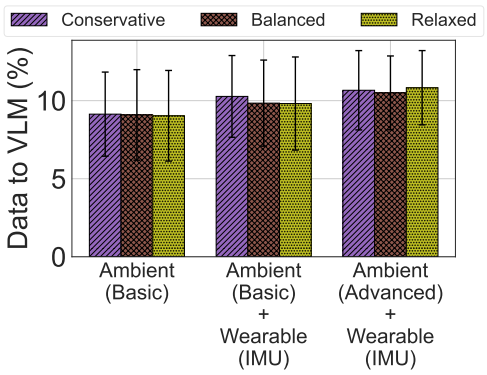


Figure 4.5: Percentage of video data requiring VLM analysis across sensor configurations. Our approach processes only 9–11% of total video data, demonstrating efficiency compared to continuous monitoring.

Granularity	**Metrics**	**Sensor Config**		
		Ambient (Basic) Only	**Ambient (Basic)+ Wearable (IMU)**	**Ambient (Advanced)+ Wearable (IMU)**
Conservative	**Accuracy**	90.4%±8.9%	91.1%±6.7%	91.7%±6.9%
	F1 Score	89.4%±9.8%	90.6%±6.4%	90.3%±7.4%
Balanced	**Accuracy**	89.9%±6.3%	85.5%±7.1%	87.1%±7.9%
	F1 Score	84.1%±8.7%	78.0%±9.9%	81.1%±11.8%
Relaxed	**Accuracy**	86.7%±4.9%	83.8%±6.5%	85.9%±6.7%
	F1 Score	73.2%±10.9%	72.4%±8.7%	75.6%±9.9%

Table 4.1: Average accuracy and F1 scores (mean±std) of discovered activity labels compared to ground truth across three semantic granularity settings. Conservative ($\lambda = 0.4$) represents coarse-grained activities, Balanced ($\lambda = 0.3$) shows medium granularity, and Relaxed ($\lambda = 0.2$) captures fine-grained activities.

4.5.1 Performance of the Label Discovery Pipeline

4.5.1.1 Efficiency of Key Moment Identification

One of the primary advantages of our sensor-first approach is minimizing the amount of video data that requires processing during the training phase. Figure 4.5 shows the percentage of video data selected for VLM analysis across different sensor configurations and granularity settings, with error bars showing variability (standard deviation) across participants. Our key moment identification approach requires analysis of only approximately 9–11% of the total collected video data across all configurations. The *Ambient(Basic) Only* configuration required the least video processing (around 9%), while the advanced configuration with *Ambient(Advanced)+Wearable(IMU)* utilized slightly more (around 10.5%). This modest increase is expected, as richer sensing capabilities can identify more nuanced activity transitions that warrant VLM analysis. In practical

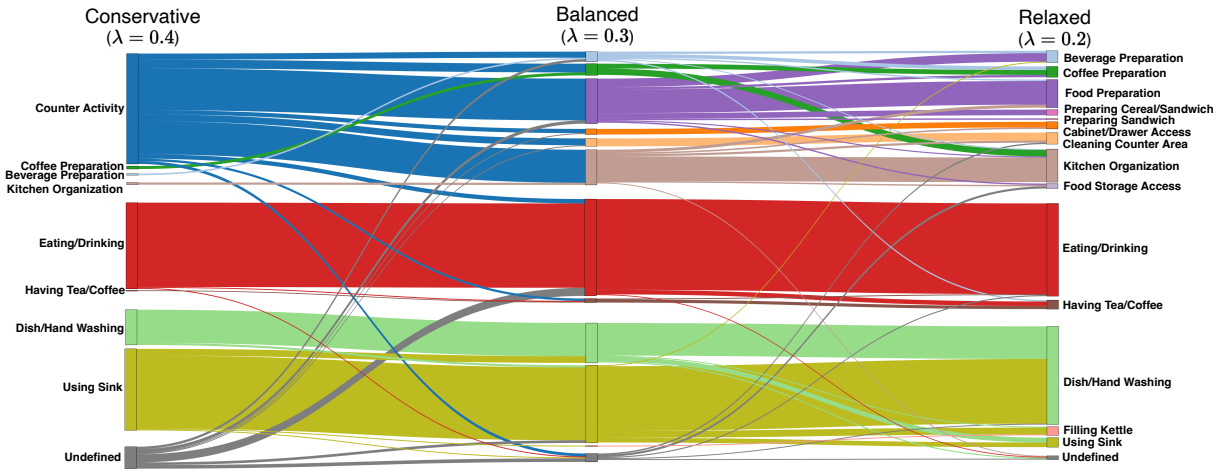


Figure 4.6: Discovered activity labels across three semantic granularity settings: Conservative ($\lambda = 0.4$, left), Balanced ($\lambda = 0.3$, middle), and Relaxed ($\lambda = 0.2$, right). The flow width represents the proportion of data with each label, showing how broader categories branch into more fine-grained activities as granularity increases. For example, the "Counter Activity" broad category (left) differentiates into "Food Preparation," "Cleaning Counter Area," and other activities in the Balanced setting.

terms, this means that for a 1.5-hour data collection session, only about 9–10 minutes of video are actually sent to the VLM for processing, a substantial reduction in both computational cost and the amount of visual data that leaves the local device.

4.5.1.2 Discovered Activities with Different Granularity

Figure 4.6 illustrates the discovered activity labels across granularity settings, revealing how broader categories in the Conservative setting branch into fine-grained distinctions in the Balanced and Relaxed settings. The Conservative setting identified four primary activities: "Counter Activity," "Using Sink," "Dish/Hand Washing," and "Eating/Drinking," capturing kitchen zones and high-level behaviors. Some participants showed more nuanced counter activities at this level, though with low discovery rates. In the Balanced setting, "Counter Activity" branches into specific categories like "Food Preparation," "Cleaning Counter Area," and "Cabinet/Drawer Access," discovering activity hierarchies that match the underlying task groupings. The Relaxed setting continues this differentiation, with "Food Preparation" splitting into "Preparing Cereal/Sandwich" and other specialized tasks.

Several patterns emerge through this granularity analysis. Activities with distinctive sensor signatures, such as "Eating/Drinking," remain stable across all settings, consistently recognized regardless of granularity. "Using Sink" appears broadly in the Conservative setting but becomes less prominent in the Relaxed setting as the system refines it to "Dish/Hand Washing" specifically, demonstrating how our approach clarifies ambiguous activities as the similarity threshold changes. The flow patterns follow logical parent-child relationships rather than arbitrary recombination: "Cabinet/Drawer Access" emerges specifically from "Counter Activity," not from unrelated cat-

egories. Overall, our system discovered 15 distinct activity labels, with individual participants typically having 4–8 relevant activities depending on the granularity setting and sensor configuration. These results suggest that OrganicHAR can adapt to what each sensor configuration reliably detects while maintaining meaningful relationships that match human understanding of kitchen activities, offering clear advantages over predefined activity sets.

4.5.1.3 Accuracy of Discovered Activities

We evaluated our discovered activity labels against ground truth annotations, as shown in Table 4.1. Since discovered labels often use different terminology than human-annotated ground truth, we performed manual mapping to convert annotations into the discovered activity label space for comparison. The Conservative setting demonstrates high accuracy above 90% and strong F1 scores across all sensor configurations. Performance moderately declines with increasing granularity, yet remains robust with accuracy above 83% even in the most challenging scenarios. The *Ambient(Basic) Only* configuration sometimes outperforms *Ambient(Basic)+Wearable(IMU)* in finer-grained settings, suggesting that wearable data occasionally introduces variability when distinguishing semantically similar activities. The *Ambient(Advanced)+Wearable(IMU)* configuration delivers the strongest Relaxed setting performance (85.9% accuracy), highlighting the value of depth and pose information for fine-grained recognition.

The more pronounced F1 score decline in the Relaxed setting (72–75%) reflects inherent challenges in precisely identifying fine-grained activities with VLMs processing low-frame-rate data (1 FPS), camera angle limitations, and partial occlusions. Given these results, we consider the Balanced setting ($\lambda = 0.3$) the most practical compromise for most deployments, offering meaningful activity differentiation while maintaining strong accuracy (85–89%) and F1 scores (78–84%). The consistently high performance across granularity settings demonstrates that OrganicHAR can automatically identify meaningful activity labels that align well with human observations.

To validate robustness across VLM architectures, we also tested three state-of-the-art models (GPT-4.1, Gemini 2.5 Flash, and Claude Sonnet 4). Our 1 FPS, 640×480 configuration remains optimal across all VLMs, though detection rates vary significantly (50.2–92%). Consistent fine-grained recognition limitations across all models (F1 scores of 35.6–55.4%) suggest these challenges reflect current VLM capabilities broadly rather than weaknesses specific to our approach. Detailed analysis is presented in Appendix A.3.

4.5.2 Performance of Activity Recognition Models

The label discovery evaluation above revealed lower F1 scores (72–75%) in the Relaxed setting, indicating imperfect alignment between some discovered fine-grained labels and ground truth. This misalignment would propagate to HAR models trained on these labels, particularly when the system attempts to distinguish between highly similar activities. This observation informs our interpretation of the results below: the Conservative and Balanced settings ($\lambda = 0.4$ and $\lambda = 0.3$) represent more realistic deployment scenarios for OrganicHAR with current sensing capabilities, while the Relaxed setting ($\lambda = 0.2$) should be interpreted as an exploration of current technical boundaries rather than a recommended configuration.

Table 4.2: Average accuracy and F1 scores (mean±std) of the HAR models across sensor configurations and granularity settings. These models perform effectively despite being trained on organically discovered labels rather than predetermined categories. Relaxed setting results (shown in gray) represent exploratory findings illustrating current technical boundaries rather than recommended configurations.

Granularity	**Metrics**	**Sensor Config**		
		Ambient(Basic) Only	**Ambient(Basic)+ Wearable(IMU)**	**Ambient(Advanced)+ Wearable(IMU)**
Conservative	**Accuracy**	83.9%±10.1%	85.1%±6.5%	87.8%±8.4%
	F1 Score	82.9%±10.6%	82.5%±9.3%	86.8%±9.3%
Balanced	**Accuracy**	75.2%±9.2%	72.5%±6.6%	78.8%±8.9%
	F1 Score	65.5%±13.3%	63.7%±9.7%	70.6%±11.1%
Relaxed	**Accuracy**	70.4%±11.5%	69.0%±9.0%	73.2%±10.4%
	F1 Score	51.4%±10.3%	47.8%±10.5%	55.6%±14.8%

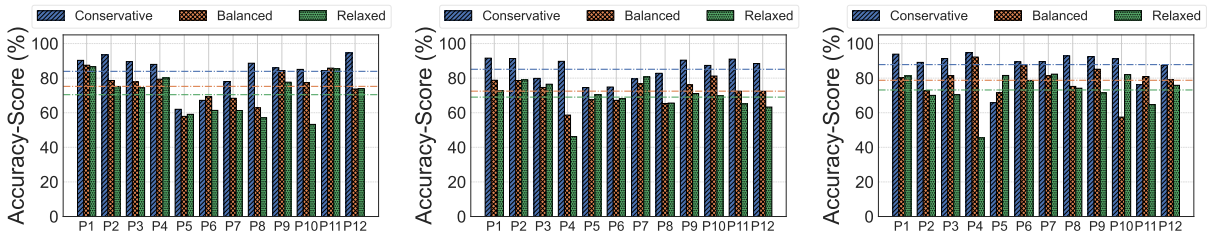
4.5.2.1 Overall Performance

Table 4.2 shows the performance of our trained HAR models across sensor configurations and granularity settings. The models demonstrate robust performance with 70–88% accuracy and 48–87% F1 scores, varying by sensor configuration and semantic granularity. Performance decreases with increasing granularity: the Conservative setting ($\lambda = 0.4$) achieves 84–88% accuracy and 83–87% F1 scores across all configurations, the Balanced setting ($\lambda = 0.3$) reaches 72–79% accuracy and 64–71% F1 scores, and the Relaxed setting ($\lambda = 0.2$) attains 69–73% accuracy and 48–56% F1 scores. The *Ambient(Advanced)+Wearable(IMU)* configuration consistently outperforms others, particularly for fine-grained activities, achieving 73% accuracy and 56% F1 score even in the challenging Relaxed setting. The additional spatial awareness from pose estimation and depth sensing helps resolve ambiguities between semantically similar activities. Interestingly, *Ambient(Basic) Only* occasionally outperforms *Ambient(Basic)+Wearable(IMU)*, indicating that wearable data can introduce noise when movements are not consistently captured.

4.5.2.2 Per-Participant Performance Variability

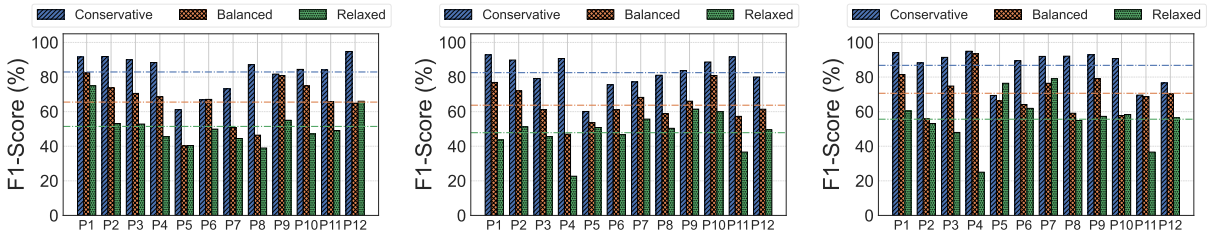
Figure 4.7 and Figure 4.8 show accuracy and F1 scores across participants for the three sensor configurations and granularity levels. Participants P1–P4 (Kitchen 1) maintain high performance across granularity settings with basic ambient sensing, while P5–P8 (Kitchen 2) show significant degradation with increased granularity. This suggests that environment layout has a meaningful impact on recognition: Kitchen 1's constrained galley layout enables more reliable activity differentiation than Kitchen 2's open layout with its island counter, likely because the compact arrangement produces more spatially distinct sensor signatures for different activities. For P9–P12 (Kitchen 3), wearable IMU data notably improves performance, providing valuable complementary motion information in that particular spatial configuration.

F1 scores prove more sensitive to sensor configuration and granularity than accuracy due



(a) Ambient(Basic) Only (b) Ambient(Basic)+Wearable(IMU) (c) Ambient(Adv.)+Wearable(IMU)

Figure 4.7: Per-participant accuracy across three sensing configurations. Kitchen 1 participants (P1–P4) show consistently high accuracy even with basic ambient sensing, Kitchen 2 participants (P5–P8) show greater degradation with finer granularity, and Kitchen 3 participants (P9–P12) benefit most from wearable IMU. Horizontal dashed lines indicate average performance per granularity setting.



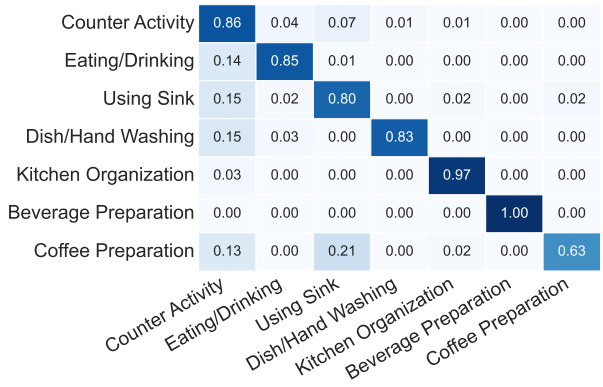
(a) Ambient(Basic) Only (b) Ambient(Basic)+Wearable(IMU) (c) Ambient(Adv.)+Wearable(IMU)

Figure 4.8: Per-participant F1 scores across sensing configurations, revealing sharper performance drops than accuracy metrics with increasing granularity. F1 scores for the Relaxed setting show particularly large variance between participants and kitchens, illustrating the impact of environmental layout on the precision-recall balance in fine-grained activity recognition.

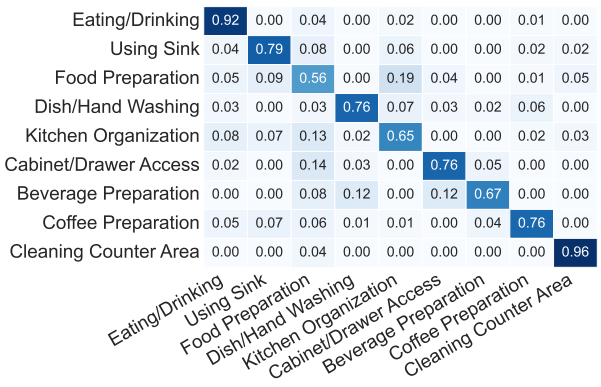
to label imbalance, dropping 31–35 percentage points from Conservative to Relaxed settings versus 13–16 points for accuracy. This steeper decline highlights the challenges in maintaining both precision and recall when distinguishing semantically similar activities with limited sensing. The *Ambient(Advanced)+Wearable(IMU)* configuration shows a smaller Conservative-to-Relaxed performance gap (15% accuracy difference) than the other configurations, indicating that richer sensing enables finer-grained recognition without sacrificing coarse-grained performance. These results illustrate that OrganicHAR achieves robust performance across various hardware configurations by adapting recognition granularity to available sensing capabilities rather than forcing predetermined labels.

4.5.2.3 Activity Confusion Analysis

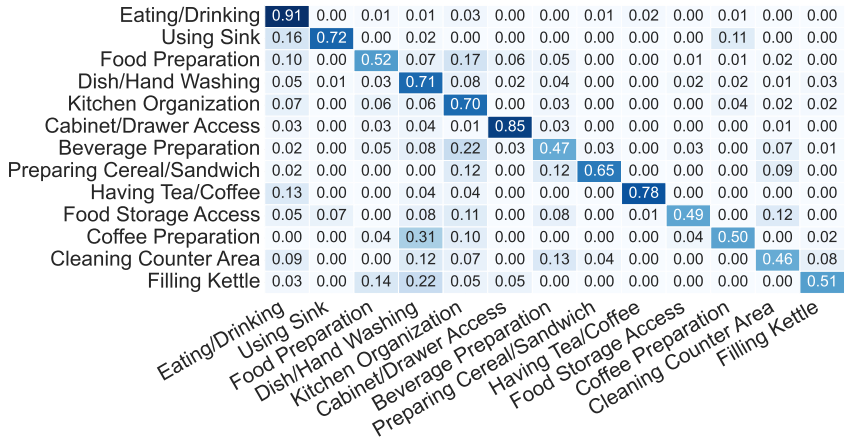
Figure 4.9 shows confusion matrices for the *Ambient(Basic) Only* configuration across the three granularity settings. The Conservative setting (Figure 4.9a) achieves high accuracy with minimal cross-category confusion. “Counter Activity” shows some confusion with other categories because



(a) Conservative ($\lambda = 0.4$)



(b) Balanced ($\lambda = 0.3$)

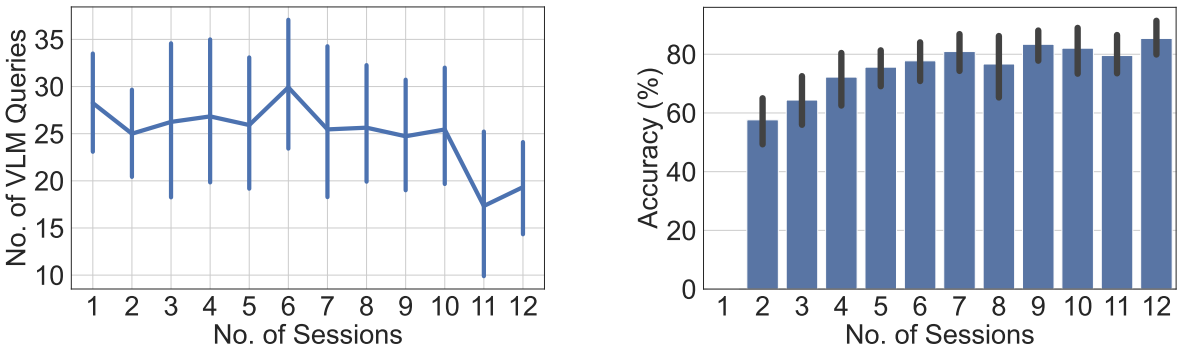


(c) Relaxed ($\lambda = 0.2$)

Figure 4.9: Confusion matrices showing HAR performance using basic ambient sensors across three granularity settings. As granularity increases from Conservative (a) to Relaxed (c), more fine-grained activities emerge, revealing specific confusion patterns between semantically or spatially related activities.

it encompasses multiple fine-grained activities at this coarse level, and "Coffee Preparation" confuses with "Using Sink" due to spatial proximity in most kitchen layouts. The Balanced setting (Figure 4.9b) shows increased confusion with finer distinctions. "Food Preparation" has the lowest accuracy, confusing with "Kitchen Organization" and "Using Sink" due to similar ambient sensor signatures. "Beverage Preparation" significantly confuses with cabinet access and washing activities, since the preparation routine often incorporates these subsidiary actions. In the Relaxed setting (Figure 4.9c), "Beverage Preparation" degrades further, confusing primarily with "Kitchen Organization" as ambient sensors cannot distinguish fine manipulations occurring in similar locations. Kettle-related and washing activities show substantial confusion, illustrating the difficulty of differentiating tasks with similar postures and locations using only ambient sensors.

Despite increasing confusion at finer granularities, certain activities maintain excellent performance across all settings. "Eating/Drinking" achieves consistently high recognition thanks to its



(a) VLM queries required across sessions

(b) Recognition accuracy across sessions

Figure 4.10: Incremental training analysis of OrganicHAR: (a) count of VLM queries after incorporating each new training session, and (b) recognition accuracy measured on all future sessions. Error bars represent 95% confidence intervals across users.

distinctive ambient signatures, and "Cabinet/Drawer Access" similarly maintains high accuracy through its characteristic movement patterns captured by the basic ambient sensors. These patterns validate our sensor-first approach: OrganicHAR adapts activity granularity based on what the sensors can reliably detect rather than forcing potentially indistinguishable predetermined categories. This analysis also identifies which sensor configurations are suited to specific recognition goals, enabling informed privacy-capability tradeoffs. Confusion matrices for the other sensor configurations are presented in Appendix A.4.

4.5.2.4 Incremental Training Analysis

To understand how VLM query requirements and recognition accuracy evolve with increasing training data, we conducted an incremental training analysis simulating sequential session processing. We used the *Ambient(Advanced)+Wearable(IMU)* configuration as it has the highest VLM usage among our configurations. With each new session, OrganicHAR identifies interesting segments across all accumulated data, applies the current model, invokes VLM analysis for low-confidence predictions or novel patterns, and retrains with the expanded dataset.

Figure 4.10a shows total VLM queries required after incorporating each session, including queries for new data and previously-seen data that becomes significant due to pattern repetition across sessions. Query counts remain stable (25–30 per session) for the first 9 sessions despite the growing dataset, as the system continuously reassesses historical data alongside new observations, sometimes identifying patterns in earlier sessions that were not initially flagged when the dataset was smaller. The downward trend in sessions 10–11 (17–19 queries) is notable, though extended data collection would be needed to confirm this pattern. Figure 4.10b shows recognition accuracy (measured on sessions not included in training) improving from 57.6% after the first session to 85.4% after 11 sessions, with most gains concentrated in the early sessions. These results demonstrate promising trends: steady accuracy improvement coupled with potential VLM query reduction in later sessions, suggesting that OrganicHAR could become both more effective and more computationally efficient over time.



(a) Home-1 (b) Home-2 (c) Home-3 (d) Home-4 (e) Home-5

Figure 4.11: Kitchen environments used in real-world home deployments: (Home-1) compact galley layout from overhead diagonal perspective, (Home-2) medium-sized kitchen with distinct functional zones, (Home-3) narrow galley configuration from elevated angle, (Home-4) spacious kitchen with multiple countertops from horizontal perspective, and (Home-5) linear wall-mounted kitchen with integrated appliances from horizontal viewpoint.

4.5.3 Real-World Deployment

We extended our evaluation to examine OrganicHAR's performance in actual home environments, where participants engage in daily routines without experimental constraints. We deployed OrganicHAR in 5 homes for 7 days each, with participants collecting 6–8 sessions based on their kitchen usage frequency. Participants used their kitchens normally, cooking, cleaning, snacking, and socializing, without task restrictions or researcher guidance, capturing the complexity of real environments including activity interruptions and varying lighting conditions (Figure 4.11).

The deployment generated approximately 11 hours of sensor data across diverse home layouts and activity patterns. For accuracy evaluation, we sampled 250 five-second segments per participant using temporal stratified random sampling: 60 from early sessions (1–2), 60 from middle sessions (3–5), and 130 from late sessions (6–8), enabling analysis of both learning progression and mature system performance. We annotated ground truth using the same methodology described in Section 4.4, mapping labels to discovered activities. Based on our controlled evaluation results, we used the Balanced granularity setting ($\lambda = 0.3$) and the *Ambient(Advanced)+Wearable(IMU)* sensor configuration for all real-world deployment analyses.

4.5.3.1 Overall Performance

Figure 4.12 presents the overall recognition accuracy across homes using leave-one-session-out cross-validation. OrganicHAR achieves an average accuracy of 74.2%$\pm$14.8% across the five homes, with performance ranging from 60% to 95%. This represents a moderate decrease compared to the controlled evaluation results (78.8%$\pm$8.9% for the same sensor configuration and granularity setting), reflecting the additional complexity of unstructured home environments. Performance varied significantly across individual homes. Home-1 achieved the highest accuracy at 95%, followed by Home-2 at 85%. Home-3, Home-4, and Home-5 achieved lower performance at 69%, 63%, and 60% respectively. This variability appears partially related to both kitchen layout constraints and camera positioning challenges. Home-4 and Home-5 utilized more lateral camera placements due to their linear kitchen configurations, which limited the VLM's ability to capture comprehensive activity context compared to the elevated diagonal perspectives used in Home-1 and Home-2. Additionally, Home-4's consistently dim lighting throughout most sessions

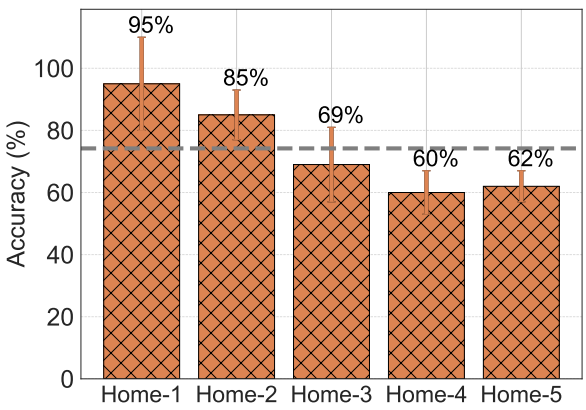


Figure 4.12: Overall recognition accuracy in real-world home deployments using leave-one-session-out cross-validation with Balanced granularity and *Ambient(Advanced)+Wearable(IMU)* configuration. The horizontal dashed line indicates average accuracy (74.2%) across all participants.

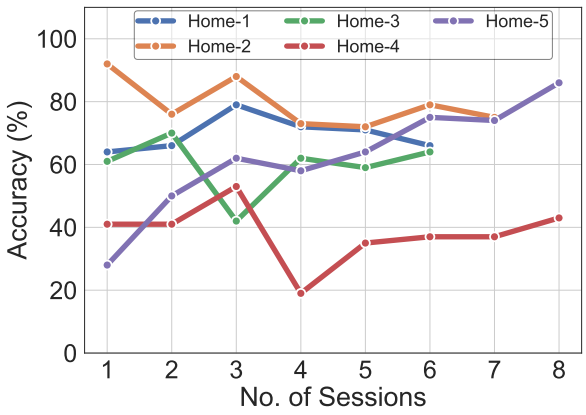


Figure 4.13: Incremental training trajectories showing forward-looking evaluation, where the system trains on sessions 1 through N and tests on all remaining future sessions ($N+1$ onwards). Performance patterns differ from Figure 4.12 due to temporal dependencies and chronological activity pattern evolution.

further degraded VLM performance during key moment analysis. These real-world constraints, which are difficult to replicate in controlled settings, highlight the importance of strategic camera positioning and adequate lighting for effective VLM-based activity discovery.

4.5.3.2 Discovered Activity Patterns in the Wild

Across all five homes, OrganicHAR identified four common kitchen activities: "stovetop cooking," "refrigerator interaction," "sink area tasks," and "countertop food preparation." These represent behaviors that occur reliably across diverse kitchen environments and could serve as a baseline vocabulary for new deployments. Beyond these commonly occurring behaviors, we discovered activities specific to each home. Home-4 had "coffee preparation" routines with distinct patterns for coffee station tasks and coffee machine operation. Home-5 showed "rice cooker usage" with separate patterns for preparation and serving phases, along with "dishwasher interaction" patterns. Home-3 was the only home where "waste disposal" was detected as a standalone recurring activity. Home-2 demonstrated particularly granular activity detection, with "refrigerator door interactions" distinguished from "general refrigerator access." This dual-layer discovery, common activities supplemented by environment-specific patterns, illustrates how users can expect baseline functionality while the system adapts to their specific routines.

4.5.3.3 Incremental Learning in Deployment

Figure 4.13 shows learning trajectories using forward-looking evaluation in chronological session order, where accuracy represents the system's performance on all future sessions after training exclusively on sessions 1 through N. The performance patterns may differ from Figure 4.12 due

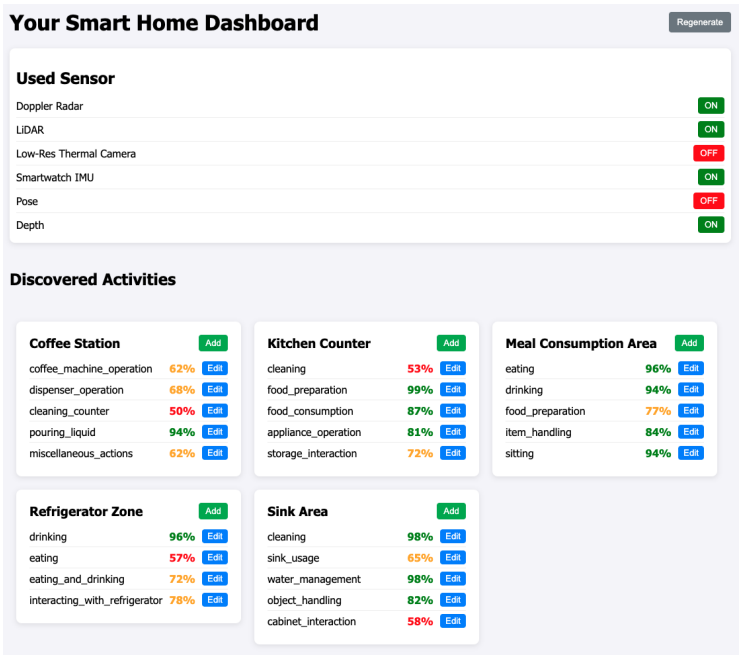


Figure 4.14: Prototype interface for customizing sensor configurations and activity labels. The percentage values show how well each deployed sensing configuration can predict activities when compared with discovered activity labels. Users can toggle sensors on or off, edit discovered labels, and add new ones.

to several temporal factors: the amount of training data increases with each session while test data decreases, and activity patterns evolve over the deployment period, making early sessions better or worse predictors of later behavior depending on the participant's routine consistency.

The results demonstrate varied adaptation across participants. Home-5 showed the most substantial improvement (28% to 86% accuracy), indicating successful learning of evolving activity patterns over time. Home-2 maintained consistently high performance (92% to 72%), reflecting stable routines that enable reliable prediction from the start. Home-1 achieved stable moderate performance in the 64–79% range. Home-3 and Home-4 exhibited more variable patterns, suggesting that the system requires additional training data or user assistance when participants have irregular routines. These trajectories indicate that OrganicHAR adapts effectively to individual activity patterns, with performance stability correlating with routine consistency: regular kitchen usage enables more stable recognition, while variable patterns present ongoing adaptation challenges.

4.6 Discussion

4.6.1 From Discovery to Customization

OrganicHAR is designed to incrementally adapt to users and their environments while preserving privacy. By continuing data collection and analysis over time, we can surface representative

examples of detectable activities for each environment, allowing users to form realistic expectations before deciding to deploy the system. The set of activities that we can typically detect in a given location, such as the four common kitchen activities identified across all five homes in our deployment (Section 4.5.3.2), can be communicated to the user to manage their expectations. Upon installation, the system begins by identifying activities that are reliably detectable in the user's home using only privacy-preserving sensors. As demonstrated in our deployment study, even with a few days of data, OrganicHAR can detect 4–5 coarse activity categories using ambient sensors alone, and up to 8–9 fine-grained categories with additional modalities like IMU, depth, and pose.

After this initial discovery phase, users can interact with our prototype customization interface (Figure 4.14), which allows them to merge similar activity labels (*e.g.*, "washing dishes" and "rinsing hands" into a broader "sink-related tasks" category), or define entirely new labels they want OrganicHAR to detect. While this interface has not yet been formally evaluated, it is technically feasible because it reuses components from our label clustering and refinement pipeline. Our contribution is not the interface itself, but the underlying paradigm shift: OrganicHAR surfaces candidate activity labels that are grounded in what the sensors can actually detect in each environment, which can then be reconciled with each user's preferences and goals. Investigating how users make sense of, reconfigure, or reject such sensor-driven candidates represents a promising direction for future HCI work.

This customization step also illustrates a theme that recurs throughout this dissertation. The user's role shifts from constructing the system (defining activities, labeling data, configuring sensors) to confirming or correcting what the system produces. The system proposes; the user disposes. This is precisely the interaction model envisioned in Section 1.2, and while OrganicHAR realizes it for activity vocabulary, subsequent chapters extend the same principle to context reasoning (Chapter 5) and occupant interaction (Chapter 6).

4.6.2 Applications in Healthcare and Assistive Technologies

OrganicHAR is particularly promising for healthcare applications, where continuous monitoring can support individuals with special needs while respecting privacy. For people living with dementia and their caregivers, the system can notify household members when critical actions are forgotten (*e.g.*, a stove left on or a refrigerator left open) or summarize high-level activities for clinicians without invasive video monitoring [21, 52]. We are currently deploying this technology with people living with dementia through the Autonomous project [21], where the system learns each person's unique patterns, monitors for safety concerns, and provides assistance tailored to their specific routines and cognitive abilities.

Procedural task assistance represents another application where we are applying OrganicHAR, for instance to support wound care procedures for post-operative skin cancer patients [16, 206]. Unlike existing approaches constrained by manually predefined step sequences, OrganicHAR could make procedural tracking more flexible by automatically identifying meaningful steps based on sensor capabilities rather than forcing predetermined sequences. The hierarchical nature of our discovered activities (from coarse to fine-grained) enables adaptive assistance based on user needs: general awareness of functional zone usage for those requiring minimal support, and detailed step-by-step guidance for those needing more comprehensive assistance.

4.6.3 Privacy Considerations During Training

While OrganicHAR uses only privacy-preserving sensors during deployment, its reliance on VLMs during the training phase introduces privacy considerations that deserve careful discussion. During key moments, video segments are transmitted to cloud-based VLM services for scene understanding. Although OrganicHAR processes only 9–11% of video data rather than requiring continuous monitoring, users may reasonably worry about data misuse, unintended inferences about their private lives, and potential access by service providers or authorities [102, 236, 240].

Several techniques can mitigate these concerns without compromising labeling effectiveness. Since our framework focuses on actions, locations, and object interactions rather than personal identification, automatic face blurring, background anonymization, and removal of personally identifiable information can be applied before any data leaves the device. User agency can be further enhanced by providing transparent previews of what video segments would be transmitted and enabling users to approve or decline each training session before any data is sent. Researchers have also shown that targeted fine-tuning of VLMs can improve handling of sensitive content [181].

These mitigations are consistent with a design principle that runs through this dissertation: the richer, more privacy-invasive source serves a bounded role and then departs. In VAX (Chapter 3), the camera and microphone are physically removed after training. In OrganicHAR, the camera is similarly removed, but the VLM cloud interaction introduces a transient privacy exposure during the training phase that VAX's fully local audio/video pipeline did not. As capabilities of edge-deployed VLMs grow and hardware capacity improves, we believe doing this on the edge will be feasible. Future work on edge-deployed VLMs and federated learning approaches could reduce or eliminate this cloud dependency while maintaining the semantic understanding capabilities that make our approach possible.

4.6.4 Limitations and Future Directions

Our real-world deployment revealed several practical challenges that point toward important future work. Camera positioning constraints in linear kitchen layouts limited VLM effectiveness, as lateral perspectives produced less informative scene descriptions than elevated diagonal views. Varying lighting conditions affected recognition consistency, and participants' evolving routines created temporal adaptation challenges requiring ongoing refinement. These deployment-level challenges suggest that practical guidelines for sensor and camera placement, potentially informed by automatic assessment of VLM description quality, would significantly improve out-of-the-box performance.

Our clustering approach also faces inherent tradeoffs in the early stages of deployment. Users must wait for sufficient data collection before meaningful clusters emerge, and initial recognition performance remains lower until adequate training data accumulates. These limitations suggest promising hybrid approaches where supervised and discovery methods play complementary roles. Our real-world deployment revealed that certain activities (stovetop cooking, refrigerator interaction, sink tasks, countertop preparation) occur consistently across diverse home environments, while others remain highly environment-specific. Supervised models could provide robust baseline recognition for these common activities that generalize across homes, while unsupervised discovery captures the personalized routines that make each home unique.

Perhaps the most fundamental limitation, and the one that motivates the next chapter of this dissertation, is that OrganicHAR produces discrete activity labels but does not reason about what those low-level activities mean in the context of higher-level human behavior. Knowing that someone used the stove, opened the refrigerator, and spent time at the counter tells us *what* happened, but not whether these activities constitute meal preparation, reheating leftovers, or three unrelated actions that happened to occur in sequence. The temporal structure of activities, which ones co-occur, in what order, and for how long, is precisely what determines higher-level context. Chapter 5 addresses this challenge by synthesizing higher-level contexts from temporal patterns in the activity streams that systems like OrganicHAR produce.

4.7 Summary

This chapter presented OrganicHAR, a framework that removes the need for predefined activity vocabularies in privacy-preserving human activity recognition. By inverting the conventional paradigm, letting sensor patterns drive activity discovery rather than requiring designers to anticipate what activities will occur, OrganicHAR addresses the second stage of the sensing pipeline identified in Chapter 1: determining what activities to detect in a new environment.

Our evaluation demonstrated that the sensor-first approach is both effective and efficient. OrganicHAR discovers 4 to 8 meaningful activity categories per environment with discovery accuracy of 87% on average, while processing only 9–11% of the video data that continuous monitoring would require. Activity recognition models trained on these discovered labels achieve 70–88% accuracy across three sensor configurations and three granularity settings, with the semantic granularity control mechanism allowing each deployment to match its vocabulary to what its sensors can reliably distinguish. Our real-world deployment in five homes confirmed that these results extend beyond controlled settings, achieving 74.2% average accuracy while adapting to the natural routines and diverse kitchen layouts of actual households.

Two contributions from this chapter carry forward into the remainder of the dissertation. First, OrganicHAR establishes that a vision-language model can serve as a temporary, bounded annotator during a training phase, generating the semantic understanding that privacy-preserving sensors need without requiring continuous access to video data. This extends the bootstrapping principle from VAX, where audio and video models served a similar temporary role, to a setting where even the activity vocabulary itself is unknown. Second, by producing activity labels at adjustable granularity, OrganicHAR provides the raw input that downstream reasoning systems need: a stream of detected activities whose meaning must be interpreted in temporal context. The next chapter presents TAO, which takes this stream as input and synthesizes the higher-level contexts that make activity data actionable for occupants and their caregivers.

Chapter 5

Synthesizing Context from Activity Patterns

The previous two chapters addressed how a context-aware system acquires the ability to detect human activities from privacy-preserving sensors. VAX showed that training labels can be generated automatically by temporarily pairing a camera with the target sensors, and OrganicHAR showed that the set of activities worth detecting can be discovered from sensor data itself, at a granularity matched to what the sensors can actually distinguish. Together, these systems produce a continuous stream of timestamped activity labels for a given environment. These labels are low-level and short-lived: each captures what a person is doing in a window of a few seconds, such as typing, talking, or walking. What applications need sits at a higher level of abstraction and a longer timescale. Whether someone is in a meeting, doing focused work, or sharing a meal is a property of tens of minutes of behavior rather than of any single instant, and a context of this kind is a pattern of activities over time rather than one activity in isolation. The same momentary label can belong to different contexts depending on what surrounds it: typing in a meeting, typing while working alone, and typing during a call are identical at the activity level but distinct at the context level. This chapter turns to the stage of the pipeline that bridges this gap, translating a stream of seconds-scale activities into the higher-level contexts, spanning tens of minutes, that downstream applications require. We present TAO (Temporal Analysis and Ontology), a hybrid system that combines ontological reasoning with temporal clustering to detect semantically meaningful contexts from complex real-world activity patterns. Section 5.1 motivates the problem and summarizes the contributions. The system architecture, evaluation, and applications follow in subsequent sections.

5.1 Introduction

A stream of fine-grained activities is not, by itself, useful to most applications. Consider an occupant who has been *typing*, *talking*, and *sitting* over the past hour. Are they in a meeting? On a phone call? Doing focused work with occasional conversation? The activities are the same in each case; what distinguishes these situations is how the activities relate to one another in time. *Typing* and *talking* happening simultaneously, sustained over thirty minutes, likely indicates a meeting. *A phone ring* followed by *talking* for five minutes indicates a phone call. *Typing* for a long stretch with a brief interruption of *talking* is more consistent with focused work. These

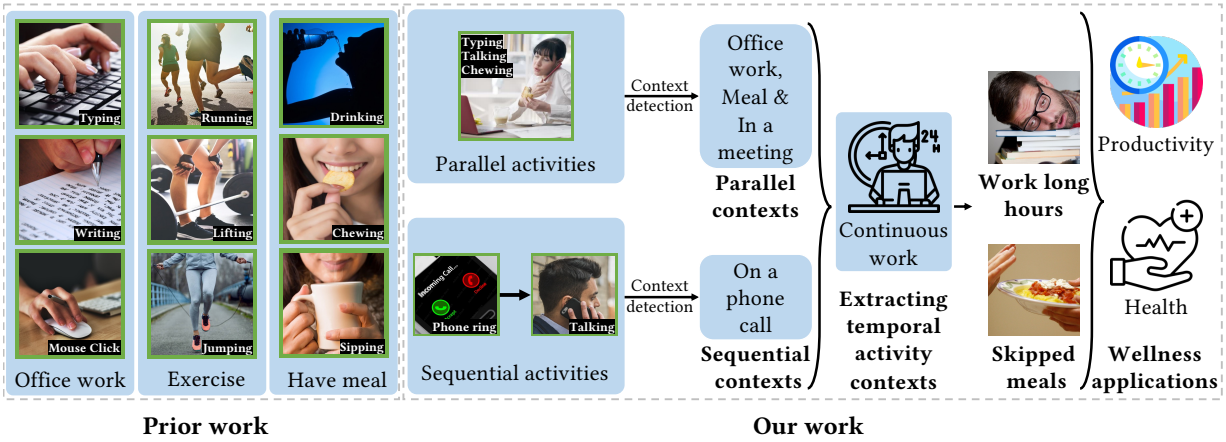


Figure 5.1: Daily activities (*e.g.*, *A: typing*, *A: running*, *A: chewing*) can be translated into semantically richer contexts (*e.g.*, *C: Office work*, *C: Exercise*, *C: Having meal*). Real-world activity patterns are complex, with activities happening sequentially, in parallel, and interleaved, requiring an approach that can handle all three. The TAO system combines ontological and temporal clustering components to convert these activity patterns into rich contexts, which can then enable applications such as wellness monitoring.

higher-level interpretations, which we refer to as *contexts*, are what downstream applications actually need [38, 46]. A wellness application that tracks productivity needs to know whether someone has been in focused work for three hours without a break, not that they have been typing. A smart home system that manages mood lighting automatically needs to know that the occupant is relaxing, not merely that they are sitting. As introduced in Chapter 1, the mapping from activities to contexts constitutes the third stage of the sensing pipeline, and it is the stage this chapter addresses.[1]

The challenge in reliably detecting context from activity streams is that it depends on temporal structure, and temporal structure in real life is rarely clean. As illustrated in Figure 5.1, activities in daily life unfold in three broad patterns. Some are *sequential*: a phone ring followed by talking indicates a phone call. Some are *parallel*: typing and talking happening at the same time may indicate a meeting. And many are *interleaved*: a person alternates between typing, eating, and talking over a thirty-minute window, moving fluidly between office work, having a meal, and perhaps a brief phone call. Most prior approaches toward detecting contexts, as discussed in Chapter 2 (Section 2.4), handle only a subset of these patterns. Ontology-based systems model context through hand-authored rules that map activity combinations to context labels, but these rules are static and do not easily distinguish between sequential, parallel, and interleaved occurrences of the same set of activities [138, 209]. Temporal clustering methods can discover recurring patterns from data without predefined rules, but the clusters they produce are unlabeled, and mapping them back to semantically meaningful contexts reintroduces manual effort [134]. Hybrid approaches that combine the two have shown promise but have been limited

[1]The TAO system was developed in close collaboration with Sudershan Boovaraghavan, who led the project and designed the ontological pipeline. I was primarily responsible for the temporal clustering pipeline, the dataset-based evaluation, and the stress wellness application [36].

to narrow activity sets and specific application domains [108, 173, 230].

We present TAO as a hybrid context detection system that combines an OWL-based ontological pipeline with a temporal autoencoder-based clustering pipeline to detect rich contexts from complex activity patterns. The ontological pipeline provides a structured vocabulary for defining how activities compose into contexts, supporting sequential, parallel, and duration-based relationships through SPARQL queries over a custom ontology. The temporal pipeline learns dense representations of activity patterns using a specialized autoencoder architecture, clusters these representations to discover recurring temporal patterns, and then uses the ontological pipeline to assign semantic labels to the discovered clusters. By combining both pipelines, TAO can handle the deterministic cases where a well-defined activity sequence maps cleanly to a context (the ontology's strength) as well as the messier cases where interleaved and recurring patterns require learning from data (the temporal pipeline's strength). Importantly, TAO is agnostic to how activities are detected: it takes as input a timestamped stream of activity labels from any recognition pipeline, which means it can operate on top of systems like VAX or OrganicHAR without modification.

We evaluate TAO on two large-scale public activity recognition datasets, *ExtraSensory* [207] and *Casas* [55], and show that it achieves 80% and 87% context detection accuracy (measured by Jaccard Similarity Coefficient), respectively, compared to 11% to 20% for prior approaches [137, 173]. We further evaluate TAO in a real-world deployment with eight participants performing activities across four scenarios, demonstrating that the system can detect contexts from real sensor data with 69% to 76% accuracy. Finally, to illustrate the value of rich contextual information for downstream applications, we also prototype two wellness applications that use TAO's context predictions to assess occupant productivity and stress. The wellness metrics computed from TAO's predicted contexts fall within 1.1% of ground truth on average, compared to 30% when using raw activity information alone.

The remainder of this chapter is organized as follows. Section 5.2 positions TAO against the most closely related hybrid approaches. Section 5.3 presents the system architecture, covering the ontological pipeline, the temporal clustering pipeline, and the strategy for combining their outputs. Section 5.4 describes the implementation. Section 5.5 presents the evaluation across public datasets and the real-world study. Section 5.6 demonstrates the wellness applications, and Section 5.7 discusses limitations and implications for the broader pipeline. Section 5.8 summarizes the chapter and connects to the next stage of the pipeline.

5.2 Positioning Against Prior Work

The landscape of context-aware computing, spanning knowledge-driven, data-driven, and hybrid approaches, was introduced in Section 2.4. Here we focus on the specific hybrid systems that are most closely related to TAO and on the gaps that motivated its design.

Several recent efforts have combined knowledge-driven and data-driven techniques to derive semantic relationships between activities for context inference [85, 108, 149, 173, 175, 177, 230]. COSAR [173] uses a statistical classifier to recognize activities, which are then passed to an ontological reasoner built on the ActivO ontology models [174] to obtain context information. Other approaches [85, 149, 230] segment sensor data based on activity relationships inferred

from an ontology and train clustering models to capture temporal patterns related to a context. These hybrid systems demonstrate the promise of combining data-driven preprocessing with ontological reasoning, but they remain vulnerable to the noisy and ambiguous nature of real-world sensor streams, which leads to inaccurate context recognition. Critically, they do not differentiate contexts arising from sequential activity patterns (activity A happening after activity B) from those arising from parallel patterns (A and B happening simultaneously), a distinction that is central to accurate context detection in practice. A more recent approach combines ontologies, Computational Causal Behavior Models (CCBMs), and Hidden Markov Models to identify goals for cooking activity, including distinguishing healthy from unhealthy meals [233]. However, this approach is constrained to specific sensing modalities and a single application domain (kitchen activities), making it difficult to extend to the diversity of contexts that arise in daily life.

A separate line of work has proposed ML-based methods that use fine-grained sensor data to measure wellness indicators such as mood instability [144], productivity, and stress [12, 107] without requiring context detection as an intermediate step. These approaches rely on data from multiple input sources, including cameras, wearables, and smartphones, and typically require manual user input to self-report productivity or stress at regular intervals. In comparison, TAO proposes an ontology-based approach to predict contexts automatically, which can then be used to calculate wellness metrics without continuous self-reporting.

In summary, none of the prior hybrid approaches consider the full range of daily life activity patterns, including sequential, parallel, and interleaved structures, in order to detect a rich and semantically meaningful set of contexts. None are agnostic to the underlying activity recognition system, depending only on timestamped activity predictions rather than raw sensor data. And none have used the contexts extracted from daily activities to build end-to-end applications that provide metrics such as productivity or stress. TAO addresses each of these gaps.

5.3 System Architecture

Figure 5.2 shows the overall architecture of TAO, which consists of two key components: an ontological pipeline (Section 5.3.1) and a temporal clustering pipeline (Section 5.3.2). The ontological pipeline provides a standard vocabulary for modeling activity-related information and formally describes the complex relationships between activities as contexts. It derives context from observed activity patterns using an OWL 2 reasoner combined with SPARQL queries. The temporal pipeline learns context representations from complex activity patterns that unfold over time, using a novel featurization technique and a deep-learning-based unsupervised clustering approach. The unlabeled clusters produced by the temporal pipeline are then passed to the ontological pipeline for semantic labeling. Activity inferences from any human activity recognition pipeline can be fed into TAO, and the two pipelines assess the activity patterns independently before their outputs are combined (Section 5.3.3) to produce a final set of context predictions. We describe each component in detail.

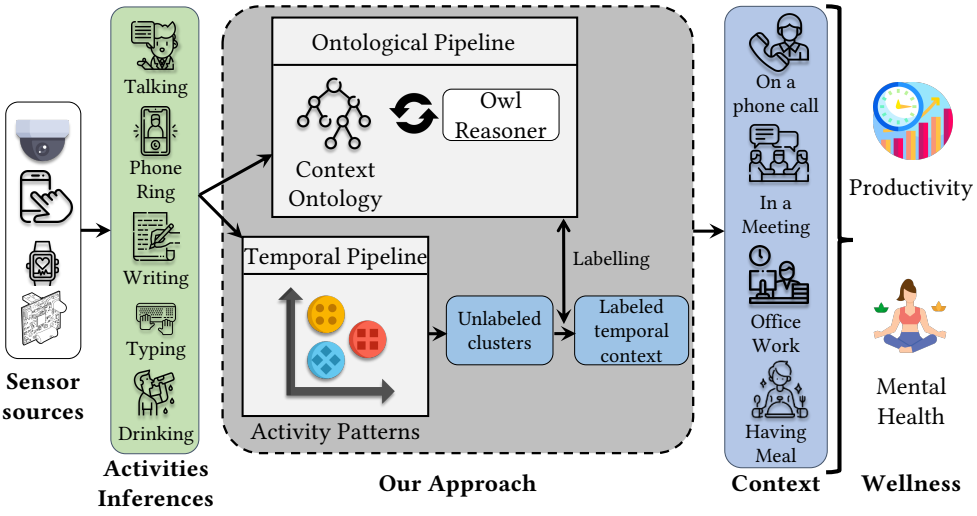


Figure 5.2: Overview of the TAO system architecture. TAO leverages OWL-based ontologies and temporal clustering approaches to identify context from the stream of activities obtained from Human Activity Recognition (HAR) systems. The contexts detected by TAO are then sent to wellness applications which infer productivity and stress.

5.3.1 Ontological Pipeline

The ontological pipeline, shown in Figure 5.3, comprises a Representation Layer and a Reasoning Layer. The representation layer defines the ontology vocabulary, and the reasoning layer uses it to derive contexts from observed activity patterns.

5.3.1.1 Representation Layer

The goal of the representation layer is to provide a standard definition of vocabularies that model contexts as relationships between complex activity patterns in daily life. These contexts include those based on sequential activity patterns, those based on parallel activity patterns, and those based on variation in the duration of an activity. For example, the context *C: On a phone call* requires the activity *A: phone ring* to be sequentially followed by *A: talking*. Multiple activities may occur simultaneously, such as *A: typing*, *A: talking*, and *A: chewing*, indicating that multiple contexts like *C: Office work*, *C: Having meal*, and *C: In a meeting* are active at the same time. Context inference also varies with duration: both *C: On a Zoom call* and *C: On a phone call* require a sequential pattern of *A: phone ring* followed by *A: talking*, but the duration of *A: talking* (5 minutes *vs.* 30 minutes) differentiates them.

To model these patterns, we designed the representation layer with a custom ontology that incorporates concepts from the Meeting Minds Ontology [209] and the lightweight Domain Activity Ontology [138]. We leverage the vocabularies defined by Meeting Minds behavioral scientists for contexts as activity combinations, and the definition concepts from the Domain Activity Ontology to capture time duration information. For example, while Meeting Minds only provides a context *C: House work* as a relationship between *A: standing* and *A: sweeping*, the Domain Activity Ontology concepts allow us to define temporal constraints such as T(A:

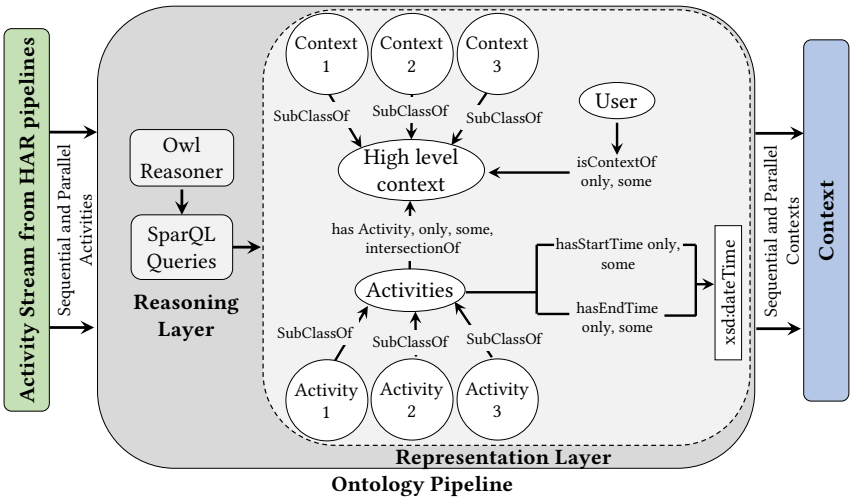


Figure 5.3: Overview of the TAO ontological pipeline.

standing)$_{end}$ → T(A: sweep)$_{start}$. Using this combined approach and including activities from the evaluation datasets (*ExtraSensory* and *Casas*) that were not part of the original ontologies, we expanded the ontology from 16 activities in [209] to 88 activities, with approximately 900 rule definitions that capture relationships between activities across sequential, parallel, and duration-based context scenarios. To bootstrap the context labeling for the ontology, researchers were shown patterns of activity combinations in a timeline view, and multiple researchers jointly and independently labeled the activity-to-context mappings, following a validation process similar to prior approaches [138, 209].

As shown in Figure 5.3, activities of daily life are represented as instances of the *Activity* class, linked to time ranges through *hasStartTime* and *hasEndTime* datatype properties. Instances of the *Context* class model relationships between activity instances using the object property *hasActivity* along with *intersectionOf*, *some*, *only*, and *union* object definitions. For example, the context *C: Office work* is an instance of the class *Context* and models the relationship between instances such as *A: typing* and *A: writing* using *intersectionOf* object properties.

5.3.1.2 Reasoning Layer

The reasoning layer derives context by interpreting the relationships between activities defined in the ontology, using a combination of standard OWL reasoners and SPARQL queries. We use this combined approach because an OWL reasoner by itself cannot support the query-like operations required to interpret contexts. The OWL 2 reasoner handles ontology semantics such as property restrictions, sub-properties, and inverse properties, while domain-specific SPARQL queries realize the context recognition through graph pattern matching.

The SPARQL queries in TAO are defined in terms of a CONSTRUCT clause that specifies what information should be returned upon successful pattern matching in the WHERE clause. Figure 5.4 shows how the composition semantics of the *C: Office work* context are queried using *A: typing* and *A: talking* as sub-activities. Figure 5.5 shows how the *C: On a phone call* context is modeled using *A: phone ring* and *A: talking* as sub-activities with time duration constraints

```
CONSTRUCT {
?x a :OfficeWork .
    :hasActivity ?x;
    :hasActivity ?y;
}
WHERE{

?x a typing;
?y a talking;
BIND((URI(?x, ?y) as ?new) .
NOT EXISTS (?new a [] .)

}
```

Figure 5.4: SPARQL query to infer *C: Office work* from parallel activities *A: typing* and *A: talking*.

```
CONSTRUCT {
?x a :OnAPhoneCall .
}
WHERE{
    ?x a phoneRing;
        :hasStartTime ?st;
        :hasEndTime ?et.
    ?t a talking;
        :hasStartTime ?st1;
        :hasEndTime ?et1.
    FILTER (contains(?st, ?et))
    FILTER (before(?st1, ?et1))
}
```

Figure 5.5: SPARQL query to infer *C: On a phone call* from sequential activities with time constraints.

specifying the sequential relationship. These queries can be updated to infer context based on arbitrarily complex dependencies between activity relationships.

5.3.2 Temporal Clustering Pipeline

Ontological approaches work well for detecting contexts based on short sequential patterns or parallel occurrences of activities. However, in real life, we regularly switch between contexts, and at any given time we can be in multiple contexts simultaneously: reading and having a meal, or listening to music while vacuuming. These settings with multiple, interleaved contexts provide an opportunity to derive richer context information by capturing repetition across interleaved sequential and parallel activity patterns. We designed a temporal clustering-based approach to detect such interleaved and recurring patterns.

5.3.2.1 Featurization

We define an activity detection pipeline as a set of one or more models that take raw sensor data and output instantaneous activity inferences for a given timestamp (*e.g.*, typing, talking, drinking). These outputs are combined into a single activity stream of *timestamp-activity* pairs, which serves as the input to our temporal pipeline. We parameterize the input activity stream with two parameters: a *stacking window* and a *lag window* (Figure 5.6).

The *stacking window* is the time interval over which incoming activity inferences are accumulated. Typical durations range from a few seconds to a few minutes, depending on the output rate of the activity detection pipeline. Activities received within one stacking window are considered to happen in parallel. As shown in Figure 5.6, we might receive one activity inference between 0 and 2 minutes (*A: eating*), no inferences between 4 and 6 minutes, and multiple inferences between 8 and 10 minutes (*A: drinking* and *A: writing*). To capture patterns across sequences of activities, we observe the incoming stream over a longer interval called the *lag window*, typically

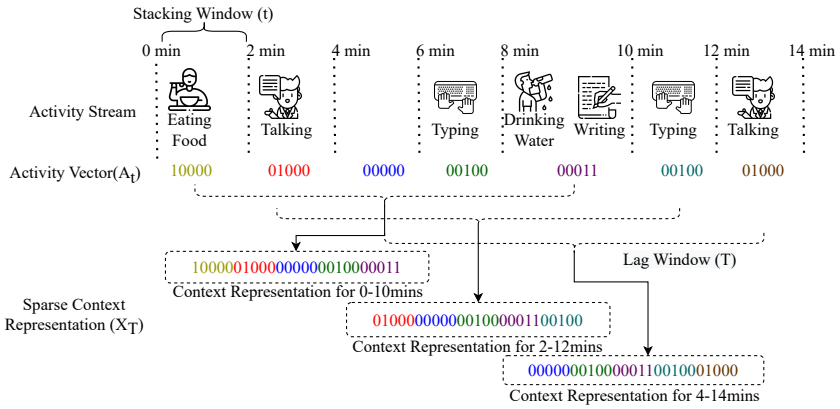


Figure 5.6: An example of context representation (five activities only) for a 2-minute stacking window (t) and a 10-minute lag window (T).

set to a multiplier (5x, 10x, 30x) of the stacking window.

The sequence of parallel activity sets within a lag window is used for context prediction at the end of that window. We encode activities within each stacking window t as a binary vector A_t of size N, where N is the total number of possible activities, with each position set to 1 or 0 depending on whether that activity was detected. We then stack these binary vectors horizontally across the lag window T to create a sparse representation X_T of the activity patterns observed in that window. This sparse representation is the input to the temporal clustering model (Figure 5.8), which consists of three components: a *Representation Trainer* that learns meaningful embeddings from context representations X_T, a *Cluster Trainer* that groups these embeddings into distinct context clusters, and a *Context Labeler* that derives semantic labels for the clusters using the ontological reasoning layer.

5.3.2.2 Representation Trainer

A naive way to identify recurring temporal patterns would be to cluster the sparse representations X_T directly. However, the positional dependence of activities leads to significantly different representations for similar contexts. For instance, *A: jumping* followed by *A: jogging* produces a quite different sparse representation from *A: jogging* followed by *A: jumping*, even though both likely correspond to the same context of exercising. One approach to reducing this sensitivity is to explicitly featurize the context representations to encode sequential and parallel patterns, but finding a fixed featurization scheme that generalizes is difficult because the importance of localized patterns can differ across user populations and environments.

Instead, we built a Representation Trainer that learns to featurize the input context representations X_T. It uses an autoencoder-based approach, an unsupervised neural network that learns to compress input data into a dense encoding and then reconstruct the original input from this compressed form [237]. The encoded representation, a dense context embedding Z_T, is then used as input for the Cluster Trainer.

A basic autoencoder [210] maps input to a denser latent representation but does not guarantee encoding of sequential or parallel patterns. An LSTM-based autoencoder [89] is a natural

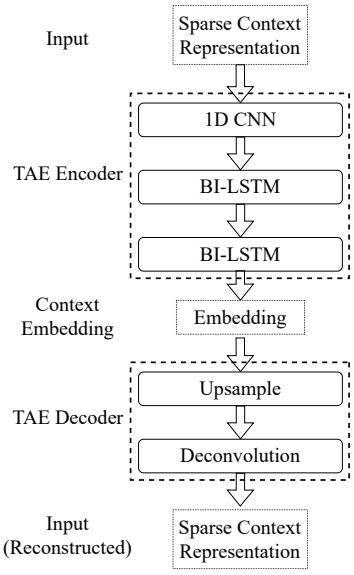


Figure 5.7: Architecture of the Temporal AutoEncoder (TAE).

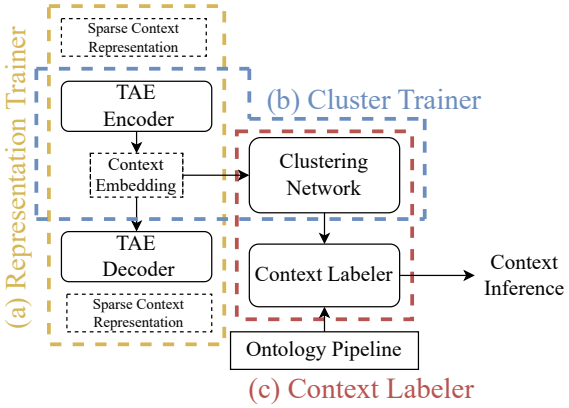


Figure 5.8: Overview of the TAO temporal pipeline. The Representation Trainer (a) pretrains a TAE encoder-decoder to learn dense context embeddings from sparse input representations. The Cluster Trainer (b) uses the TAE encoder and learned embeddings to discover the optimal context cluster count and their sparse representations. The Context Labeler (c) uses the ontology to provide labels to the generated context clusters.

alternative for modeling sequential patterns, but based on our initial experiments, we found that LSTM-based architectures failed to learn context representations effectively for our use case. The primary reason is that LSTMs need dense input data and longer sequences to learn meaningful representations, and they fail to model temporal patterns effectively from the sparse context representations that activity streams produce.

Based on these findings, we adopted a Temporal AutoEncoder (TAE) [134], a specialized architecture designed to capture temporal relationships in multivariate time series data. Figure 5.7 shows the TAE encoder-decoder architecture. The first level consists of a 1D convolutional layer that extracts short-term features (waveforms), followed by a max pooling layer that compacts the time series into a more compressed representation while retaining the most relevant information. This step helps learn local patterns in the sparse context representation, effectively modeling information from activities happening in parallel. These compact representations are then fed into two bi-directional LSTM cells that learn temporal changes in both time directions, capturing sequential patterns of activities in the final embedding. Reconstruction is performed by an upsampling layer followed by a deconvolution layer. Our Representation Trainer pretrains this TAE architecture with a binary cross-entropy loss function [56] to learn context embeddings Z_T. These embeddings are then provided as initial representations to derive optimal cluster counts and cluster activity patterns into context clusters with the Cluster Trainer.

5.3.2.3 Cluster Trainer

A direct approach to deriving contexts from the embeddings Z_T would be to apply partition or hierarchical clustering to identify prominent contexts based on cluster centers. However, such approaches do not utilize data-specific patterns in the activity streams to enhance separability between centroids. Existing literature has shown superior performance by jointly training an autoencoder for latent representation and clustering error [238]. Our Cluster Trainer uses a similar combined training approach, inspired by Deep Temporal Clustering (DTC) [134] and Deep Embedded Clustering (DEC) [222]. In this approach, we iteratively update cluster centroids to optimize cluster separability while fine-tuning the weights of the Representation Trainer to ensure that the learned context embeddings are well suited to separate context representations X_T into distinct categories.

One challenge of this combined training method is determining the optimal number of clusters. Since the number of recurring activity patterns differs across settings, we need a data-driven method for selecting the cluster count. Spectral clustering methods (*e.g.*, DBSCAN, OPTICS, HDBSCAN) provide a natural way to determine cluster count based on parameters like distance thresholds and minimum cluster sizes. However, we observed that spectral methods are sensitive to the subspace of embeddings Z_T and their hyperparameters: for different kinds of activity streams and lengths of context representation, the underlying subspace varies significantly, leading to a wide range of cluster counts, whereas the number of meaningful interleaved patterns for a given set of activity streams is finite. We therefore used silhouette scores [235] with K-Means clustering over context embeddings Z_T to determine the optimal cluster count. One caveat of this partition-based approach is the assumption of convex cluster boundaries, which is addressed by the deep clustering refinement in the next step.

We initialize centroids with K-Means clustering over the learned context embeddings Z_T, followed by iterative training with an unsupervised method that alternates between two steps until convergence. First, we compute q_{ij}, the probability that input X_i belongs to cluster j, based on the distance between context embedding Z_i and centroid w_j. The distance is evaluated using the Complexity Invariant Distance (CID) [26], a generalizable metric across complex activity streams that corrects Euclidean distance (ED) by a complexity factor (CF):

$$\text{dist}(Z_i, w_j) = CF(Z_i, w_j) \times ED(Z_i, w_j)$$

$$CF(Z_i, w_j) = \frac{\max(CE(Z_i), CE(w_j))}{\min(CE(Z_i), CE(w_j))}$$

$$CE(x) = \sqrt{\sum_{t=1}^{N-1}(x_{t+1} - x_t)^2}$$

where $CE(x)$ is the complexity estimate of a sequence x. We normalize these distances into probability assignments q_{ij} using a Student's t-distribution kernel with a single degree of freedom [208]:

$$q_{ij} = \frac{(1 + \text{dist}(Z_i, w_j))^{-1}}{\sum_{j=1}^{k}(1 + \text{dist}(Z_i, w_j))^{-1}}$$

Second, we train the Cluster Trainer iteratively by minimizing KL divergence between q_{ij} and a target distribution p_{ij} that strengthens high-confidence predictions and normalizes losses to prevent distortion of the context embeddings:

$$p_{ij} = \frac{q_{ij}^2 / f_j}{\sum_{j=1}^{k} q_{ij}^2 / f_j}$$

where $f_j = \sum_{i=1}^{n} q_{ij}$, derived empirically [86, 222]. The KL divergence loss is then:

$$L = \sum_{i=1}^{n} \sum_{j=1}^{k} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

We use a weighted combination of this KL divergence loss and the Representation Trainer reconstruction loss to jointly optimize cluster centroids and fine-tune the encoder weights. This produces final cluster centroids $w_j^{*(z)}$ in the subspace of context embeddings Z_T, which are fed back through the Representation Trainer's decoder to reconstruct $w_j^{*(x)}$, the cluster centroid representations in the original space of sparse context representations X_T.

5.3.2.4 Context Labeler

Each reconstructed centroid $w_j^{*(x)}$ defines the representative activity pattern for cluster j and thus the context for all X_T assigned to that cluster. However, when converted back into timestamped activities, these centroid representations can be long and interleaved with activities from multiple contexts. For example, a cluster centroid might correspond to the sequence *A: typing* > *A: drinking*+*A: writing* > *A: typing* > *A: talking*, spanning the 4 to 14 minute range in Figure 5.6. We use a heuristic method to break such sequences and query the ontology to detect semantically meaningful context labels.

A naive approach would be to query the ontology for all possible subsets of sequential, parallel, and single activities, but this produces too many candidate contexts. For the example above, querying all subsets would return *C: In a meeting*, *C: Having meal*, and *C: Office work*, when the actual context is only *C: Office work*. To reduce false contexts, we apply two heuristics that work well in practice. First, we query contexts from sequential and then parallel activity patterns of two activities at a time. In the example, querying for (a) *A: typing*, (b) *A: drinking*+*A: writing*, and (c) *A: typing* > *A: talking* returns *C: Office work*. If no meaningful contexts are found, we fall back to querying based on single activities from the interleaved set. Second, we use the ontology to identify subsets of activities that are distinctive for a particular context (*e.g.*, only *A: typing* strongly indicates *C: Office work*, whereas *A: talking* alone could indicate multiple contexts) and prioritize queries using these distinctive subsets.

If new activities appear in the stream that are not modeled in the ontology, the ontological labeling cannot provide contextual information. In such cases, users can be asked to manually label the context, and these new patterns can be used to update the ontology over time. We scope the current work to the set of activities modeled by our extended ontology and leave automated ontology extension as a direction for future work.

5.3.3 Combining the Pipelines

We combine context predictions from the ontological and temporal pipelines to produce TAO's final context predictions. A simplistic approach would be to take the union of unique contexts predicted by both pipelines for a given time period, but this leads to overestimation. The primary reason is the presence of ambiguous activities, those that span multiple contexts such as *A: sitting* or *A: talking*, which cause the ontology to predict all possible contexts (*e.g.*, *C: Office work*, *C: Amusement*, and *C: Having meal*) when only one is actually present.

To address this, we opportunistically select output from one of the pipelines based on the number of contexts each predicts for a given time period. The ontology is deterministic: it consists of predefined rules that map activities to contexts and provides reliable detection when the activity stream contains a single activity or a well-defined sequential pattern. The temporal pipeline, on the other hand, is more reliable in multi-context settings because it predicts multiple contexts only when corresponding activity patterns recur in the training data, whereas the ontology produces multiple predictions whenever ambiguous activities are present. We therefore choose the temporal pipeline's output when either pipeline predicts multiple contexts for a given time period, and fall back to the ontological pipeline for simpler, single-context cases.

5.4 Implementation

The TAO system is implemented to take activities as input from any activity recognition pipeline and return detected contexts.[2] We developed the ontology using OWL 2 RL reasoning [77] with SPARQL queries (SPARQL Inferencing Notation, SPIN [213]) for implementing the context interpretation process. The ontology was built and edited in Protégé [147] using the Pellet and Hermit reasoners. The temporal pipeline is written in Python, with deep learning models built using PyTorch [153]. We utilized the RAPIDS framework [200] to support faster K-Means clustering on a GPU for finding optimal cluster counts and initializing centroids for the Cluster Trainer.

The system uses five submodules: (i) *Data Parsers*, which convert different types of data streams into a common input format (*i.e.*, sparse context representations) for training; (ii) *Representation Trainer*, which learns context embeddings and locally stores pretrained autoencoder models; (iii) *Cluster Trainer*, which embeds pretrained autoencoder models into a temporal clustering network to extract context embeddings, learn optimal cluster counts, and assign cluster memberships; (iv) *Context Labeler*, which stores a copy of the current ontology and uses the other components to assign semantic context labels to clusters; and (v) *Prediction Engine*, which deploys the trained pipeline for real-time context predictions. The system supports incremental learning by caching activity streams and ML models in local memory.

For training, TAO takes a JSON configuration file specifying data parameters (list of activities and contexts, streaming method, lag parameter, stacking window, and ontology models) and model parameters (learning rate, context embedding size, TAE layer sizes, bounds for optimal cluster count, and stopping criteria). We provide default model specifications that enable out-of-the-box

[2]The TAO system is open-sourced at `github.com/synergylabs/tao`, including the ontology models, which can be used out of the box with activity streams from various sensing systems.

training for any activity stream of the form ⟨*timestamp, list of activities*⟩. The final output of the training pipeline is a deployable package that serves context predictions given a stream of activities.

5.5 Evaluation

In this section, we evaluate TAO's ability to accurately identify contexts from different activity patterns using representative HAR datasets and a real-world study. Our evaluation aims to answer three questions. First, how well do the individual components of TAO, namely the ontological and temporal pipelines, detect contexts from daily activities, and how well does the hybrid approach that combines them perform (RQ1)? Second, how does TAO's context detection accuracy compare to prior work (RQ2)? Third, how well does TAO perform in a real-world deployment, and how does the accuracy of the underlying activity detection pipeline impact its performance (RQ3)?

We describe the experimental setup and define accuracy metrics in Section 5.5.1. In Section 5.5.2, we evaluate the accuracy of the individual pipelines and the combined system. We compare with prior approaches in Section 5.5.3. Finally, in Section 5.5.4, we evaluate TAO in a real-world study with eight participants across four activity scenarios.

5.5.1 Evaluation Setup

We benchmark TAO on two human activity datasets, *ExtraSensory* [207] and *Casas* [55], and on a real-world activity detection pipeline. The dataset-based evaluation allows us to measure TAO's performance on continuous activity streams across multiple days and to compare with other approaches evaluated on the same datasets.

The *ExtraSensory* dataset consists of activity and location labels from smartphone sensor data collected from 60 participants over several days (approximately 7 to 14 days per participant). Activity labels are provided at 1-minute intervals, and there are 34 unique activity labels across all participants. The *Casas* dataset collects data at 1-second granularity from ambient sensors deployed in 30 different smart homes over 1.5 months. The data is manually labeled by researchers and contains 42 unique activity labels.

Ground-Truth Context Annotation: Most public HAR datasets include activity annotations but none provide activity-to-context mappings. We therefore created our own context labels using the annotated activities from both datasets. We used the activity labels to create timestamped activity traces (at a participant level in *ExtraSensory* and at a home level in *Casas*), identified 14 different contexts from these activity labels, and had two researchers independently annotate 90 randomly sampled days of activity data across both datasets. In total, we annotated 1,300 hours of data across 1,800 annotated instances of contexts. Detailed summaries of dataset characteristics and context annotation are provided in the appendix (Appendix B.2 and Appendix B.4).

Performance Metrics: To evaluate the accuracy of TAO, we use the Jaccard Similarity Coefficient (JC) [94] as our primary metric (Figure 5.9). For context-level performance, we use precision and recall, calculated based on True Positives (TPs), False Positives (FPs), and False Negatives (FNs) at the context level. A detection for a specific context (*e.g.*, *C: Office work*) is a TP if it is both detected and present in the ground truth, an FP if detected but not in the ground truth, and an FN

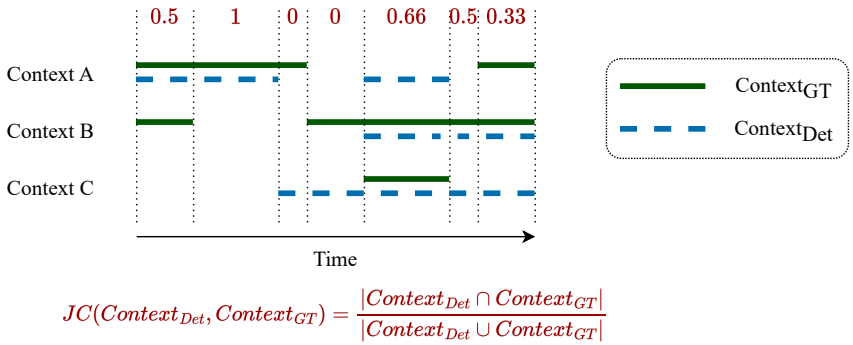


Figure 5.9: A representative figure illustrating the evaluation of context prediction accuracy using the Jaccard Coefficient (JC). Predicted contexts ($Context_{Det}$) are compared against ground truth ($Context_{GT}$). The JC ranges from 0 to 1: 0 when there is no overlap between ground truth and detected contexts, and 1 when they are identical. JC penalizes methods that detect contexts not present in the ground truth (increasing the union) and that miss contexts present in the ground truth (decreasing the intersection).

Table 5.1: Comparing the accuracy (JC) and total contexts detected by TAO's ontological pipeline with prior work on two real-world HAR datasets. Our approach detects different types of contexts (sequential and parallel) at significantly higher accuracy (72.8% on *Casas* and 53.9% on *ExtraSensory*).

Dataset	**Ontological Approach**	**Sequential Contexts**		**Parallel Contexts**		**Total Contexts**	
		Acc (JC)	Count	Acc (JC)	Count	Acc (JC)	Count
ExtraSensory [207]	Meeting Minds [209]	20.3%	976/4585	–	–/17265	0.03%	976/21850
	TAO (Ontology only)	**24.6%**	1138/4585	59%	10648/17265	**53.94%**	11786/21850
Casas [55]	Meeting Minds [209]	11.03%	38949	NA	NA	11.03%	38949/182604
	TAO (Ontology only)	**72.80%**	153022/182604	NA	NA	**72.80%**	153022/182604

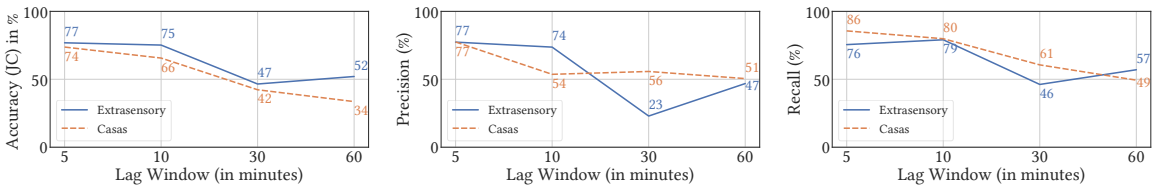
if not detected but present in the ground truth. Precision and recall are then $P = TP/(TP + FP)$ and $R = TP/(TP + FN)$. For overall accuracy, we use a weighted average of the JC across all detected overlaps, weighted by the length of the overlap interval. We also compute weighted averages of precision and recall at the context level, weighted by the frequency of each context in the ground truth. In addition, we report the F1 score for comparison with prior approaches.

5.5.2 RQ1: Evaluation of the Ontological, Temporal, and Hybrid Pipelines

We evaluate the components of TAO in isolation before evaluating the combined system, using the two HAR datasets and our ground-truth context labels.

5.5.2.1 Ontological Pipeline

We evaluate TAO's ontological pipeline in terms of accuracy and richness of context capture, comparing it against the Meeting Minds (MM) context ontology [209]. For a fair comparison, we updated the activity labels from both datasets to be consistent with the activity instance definitions



(a) Accuracy (JC) comparison (b) Precision comparison (c) Recall comparison

Figure 5.10: Comparing the accuracy (JC), precision, and recall of context detection for TAO's temporal pipeline across different *lag window* durations on the *Casas* and *ExtraSensory* datasets. The approach achieves high accuracy (65% to 75%) for short lag windows (5 and 10 minutes) compared to longer windows (30 and 60 minutes). Recall of context detection is consistently higher than precision across both datasets.

in the respective ontologies, ensuring that all activities and their timestamps could be used for querying. Table 5.1 compares the accuracy (JC) and the number of contexts detected by each approach, categorized into sequential and parallel contexts.

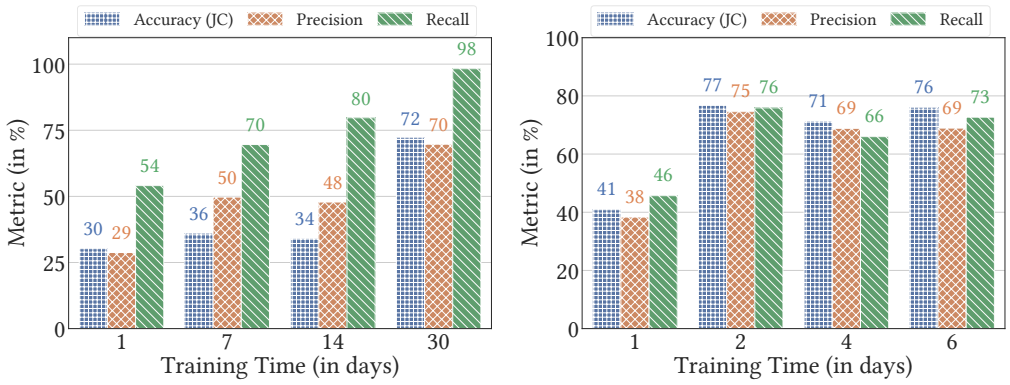
For both datasets, our approach detects contexts at substantially higher accuracy (*ExtraSensory*: 53.94%, *Casas*: 72.8%) compared to the MM approach (*ExtraSensory*: 0.03%, *Casas*: 11.03%). In the *ExtraSensory* dataset, the low accuracy of the MM approach is primarily due to its inability to infer contexts from parallel activity patterns (*e.g.*, *A: talking*, *A: eating*, and *A: tv* occurring at the same time), whereas our approach detects such contexts with 59% accuracy. In the *Casas* dataset, which consists predominantly of sequential activity patterns, our approach still substantially outperforms MM (72.8% *vs.* 11.03%) because our ontology contains a denser vocabulary of activity relationships and contexts.

5.5.2.2 Temporal Pipeline

We evaluate the temporal pipeline across two key dimensions: the duration of the lag window used for creating sparse context representations, and the amount of training data used for learning context clusters.

Optimizing the Lag Parameter: Figure 5.10 shows the performance of models trained on both datasets for lag parameters of 5, 10, 30, and 60 minutes. Accuracy (JC) decreases across both datasets as the lag parameter increases, showing that a lag of 5 minutes is sufficient to capture the activity patterns associated with most contexts. Larger values miss shorter-duration contexts (*e.g.*, *C: Using bathroom* occurring between stretches of *C: Office work*) or overestimate context presence across the wider window (*e.g.*, both *C: Office work* and *C: Having meal* being attributed to an entire 60-minute window). Precision and recall follow a similar decreasing trend, with precision dropping more steeply than recall, indicating that shorter-duration contexts are overestimated with larger lag values rather than missed entirely. Based on these findings, we set the lag parameter to 5 minutes for all subsequent evaluations.

Accuracy with Incremental Data: Figure 5.11 shows how performance changes as training data increases, expressed in number of days. We use data from two randomly chosen users, one from each dataset. The *Casas* user has 45 days of ground-truth data; we trained separate models



(a) Casas Dataset (b) Extrasensory Dataset

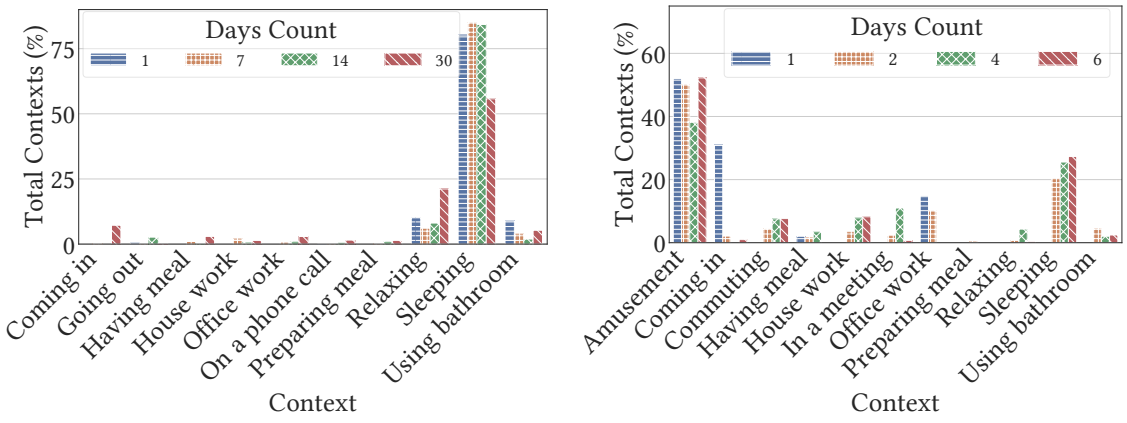
Figure 5.11: Comparing the accuracy (JC), precision, and recall of context detection for TAO's temporal pipeline as the training data (in days) increases. For the *Casas* dataset, the approach shows a consistent increase in accuracy, precision, and recall as training data grows. For the *ExtraSensory* dataset, the approach achieves high accuracy within two days of training data.

with 1, 7, 14, and 30 days of training data and tested all models on the same held-out week. We followed a similar protocol for the *ExtraSensory* user, training with 1, 2, 4, and 6 days and testing on the remaining two days. For the *Casas* user, we observe a consistent increase in recall as training data increases, with good recall from a single day but low precision initially. For the *ExtraSensory* user, precision and recall improve from 1 to 2 days and then stabilize. Figure 5.12 shows the distribution of detected contexts: in *Casas*, which consists predominantly of sequential patterns, 30 days of training enables the pipeline to detect a wide range of contexts at 72% JC. The most frequently detected context is *C: Sleeping*, reflecting its dominance in the daily routine. For *ExtraSensory*, the pipeline detects a diverse set of contexts from *C: Amusement* to *C: Office work* within two days of training data.

5.5.2.3 Overall Performance

Figure 5.13 shows the overall performance of the ontological and temporal pipelines in isolation and the hybrid approach that combines them. For this evaluation, we split users into training and testing sets and trained one model per dataset: for *Casas*, we trained on 28 of 30 users and tested on the remaining 2; for *ExtraSensory*, we used 53 of 60 users for training and tested on the remaining 7.

The temporal pipeline by itself achieves around 75% accuracy (JC) across both datasets. The ontological pipeline, however, shows markedly different behavior: it achieves significantly higher accuracy on *Casas* than on *ExtraSensory*. This difference is attributable to the types of activity patterns in each dataset. In *Casas*, most activities happen sequentially (*e.g.*, *A: toilet* > *A: eating* > *A: washing dishes*), leading to straightforward mappings to context (*e.g.*, *C: Using bathroom* > *C: Having meal* > *C: House work*). In *ExtraSensory*, most activity patterns are interleaved over extended periods (*e.g.*, *A: walking* > *A: with friends*+*A: talking* > *A: talking*), and users are often in multiple contexts simultaneously (*e.g.*, *C: Commuting* and *C: Amusement*). The ontological



(a) Casas Dataset Context Distribution

(b) Extrasensory Dataset Context Distribution

Figure 5.12: Comparing the distribution of predicted contexts learned over 45 days (1/7/14/30/45 days) for *Casas* and six days (1/2/4/6 days) for *ExtraSensory*. In *Casas*, 30 days of training data enables the pipeline to detect a wide range of contexts at 72% JC. For *ExtraSensory*, several contexts are detected within two days of training data.

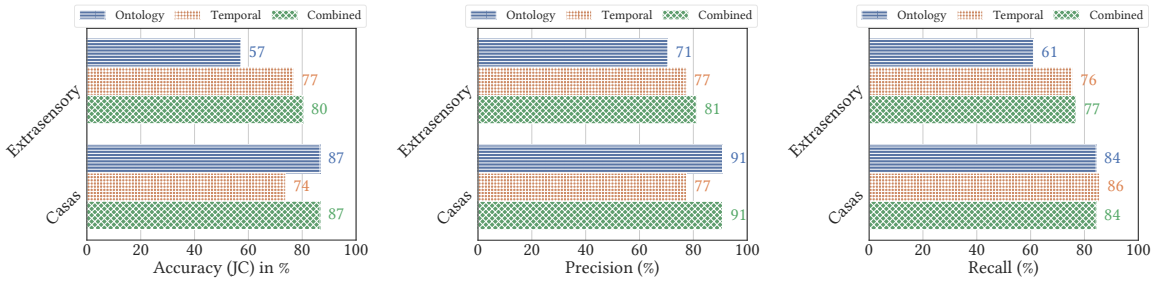
pipeline, with its static mappings, cannot detect these interleaved contexts accurately, whereas the temporal pipeline learns recurring patterns from the data and reasons over them effectively.

The combined approach performs better than either individual component on both datasets. For *Casas*, the combined approach matches the ontological pipeline because most activity patterns are sequential and the combined pipeline relies primarily on ontological predictions in these cases. For *ExtraSensory*, the combined pipeline improves over the temporal pipeline alone (80% *vs.* 77%), with the improvement attributable to using the temporal pipeline for recurring multi-context patterns and falling back to the ontological pipeline for deterministic predictions where the temporal clustering occasionally assigns single activities to incorrect context clusters.

5.5.3 RQ2: Comparison with Prior Approaches

While none of the prior approaches model the breadth of contexts that TAO does, two approaches, those by Meditskos *et al.* [137] and Riboni *et al.* [177], model a subset of the same contexts and follow approaches similar to TAO's ontological pipeline. Both focus on interleaved activities and use ontologies or probabilistic methods to model them as contexts, and both evaluate on a subset of activities from the *Casas* dataset. We therefore compare TAO with these approaches using context-level F1 scores for the subset of contexts they detect, computing the metric in the same way as reported in their respective papers.

It should be noted that both prior approaches [137, 177] use fine-grained activity event information such as *A: picking spoon* or *A: teacup moved* to infer contexts like *C: Having breakfast*. In contrast, TAO's hybrid approach operates on coarser activity labels like *A: drinking* and *A: eating* and models temporal patterns to infer contexts such as *C: Having meal*. This difference in input granularity limits our ability to make a direct comparison, so we focus on how these methods compare in inferring the same set of contexts on average.



(a) Accuracy (JC) comparison (b) Precision comparison (c) Recall comparison

Figure 5.13: Comparing the accuracy (JC), precision, and recall of the TAO pipeline. Results are shown for the ontological pipeline alone, the temporal pipeline alone, and TAO's hybrid combined approach. The temporal pipeline achieves approximately 75% accuracy across both datasets. The ontological pipeline achieves higher accuracy (87%) on *Casas* but lower accuracy (57%) on *ExtraSensory*. The combined approach achieves high accuracy (80% and 87%) and high precision (81% and 91%) across both datasets.

Table 5.2 shows the context-level F1 scores. TAO not only detects a wider range of contexts compared to both prior approaches, but also achieves higher accuracy on most of the contexts they do detect, including *C: Preparing meal*, *C: Relaxing*, and *C: Using bathroom* (all above 85% F1). For other contexts such as *C: House work* and *C: On a phone call*, TAO achieves comparable performance. The one context where Riboni *et al.* outperform TAO is *C: On a Phone call* (72.3% *vs.* 55.8%), likely because their fine-grained event-level inputs provide more discriminative information for short-duration contexts.

5.5.4 RQ3: Evaluation in a Real-World Deployment

We conducted a real-world user study to evaluate the robustness of TAO for context detection beyond activity datasets. In addition, we evaluate how the accuracy of an underlying activity detection pipeline affects TAO's performance.

5.5.4.1 Scenarios and Study Design

We defined four scenarios representing common activity patterns in daily life. Scenario 1 involves sequential activity patterns that indicate a context, such as *C: Person Entering* requiring *A: door knock* followed by *A: door open*. Scenario 2 involves simultaneous contexts, such as being *C: On a phone call* while *C: Preparing coffee*, requiring both the transition from *A: phone ring* to *A: talking* and the activity *A: grinding coffee*. Scenarios 3 and 4 involve interleaved and parallel patterns: for example, when exercising, activities like *A: jumping* and *A: running* are interleaved, and an occupant can be simultaneously *C: Relaxing*, *C: Having meal*, and *C: Office work* if *A: watching tv* runs in the background while the person shifts between *A: chewing food* and *A: clicking the mouse*. To emulate real-world conditions, we constrained only the start and end times of each scenario; participants could perform the given activity patterns in whatever manner they saw fit.

Table 5.2: Accuracy comparison (F1 score) of TAO's combined pipeline with prior approaches by Meditskos *et al.* [137] and Riboni *et al.* [177] for context detection on the *Casas* dataset [55]. Overall, TAO performs significantly better across a broader set of contexts.

Dataset	**Context**	**TAO**	**Meditskos *et al.* [137]**	**Riboni *et al.* [177]**
Casas	*C: Coming in*	**46.5%**	-	-
	C: Going out	**52.9%**	-	-
	C: Having meal	**73.2%**	-	-
	C: House work	**87.7%**	81.8%	57.4%
	C: Office work	**89.1%**	-	-
	C: On a Phone call	55.8%	54%	**72.3%**
	C: Preparing meal	**88.4%**	78.4%	82.2%
	C: Relaxing	**86.4%**	79.7%	81.1%
	C: Sleeping	**82.4%**	-	-
	C: Using bathroom	**92.2%**	90.1%	88.2%

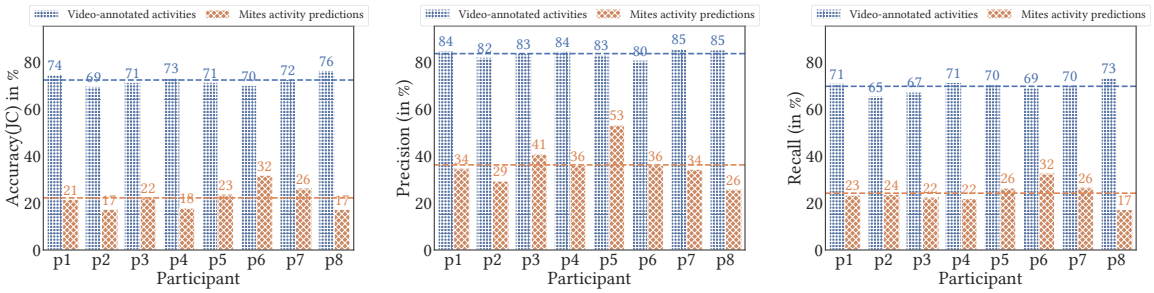
Table 5.3: Context-level performance for the real-world study, showing precision and recall for each context when using video-annotated activities and Mites activity predictions.

Context	**Video-annotated activities**		**Mites activity predictions**	
	`Precision`	`Recall`	`Precision`	`Recall`
Coming In	98.2%	99.68%	17.47%	58.04%
Exercising	97.56%	99.42%	33.99%	36.36%
Having Meal	87.29%	87.66%	16.57%	17.43%
House Work	88.35%	87.66%	23.48%	4.58%
Office Work	98.78%	73.04%	55.96%	36.02%
Phone Call	69.81%	39.98%	21.17%	13.36%
Relaxing	77.87%	58.04%	46.30%	24.44%

We recruited 8 participants (four male and four female) with an average age of 23 years, ranging from 21 to 32. Participants were mostly university students and staff. Our institution's IRB approved the study. We captured two types of data: (1) raw video from a camera, annotated manually for activities and contexts, and (2) activity inferences from the Mites platform [139], a state-of-the-art multi-modal sensor that captures environmental data (temperature, humidity, pressure, vibrations, motion, light, thermal, and sound) and performs activity recognition using an SVM-based ML model. We trained an ML model for each user based on the Mites sensor data, and once deployed, TAO used the resulting activity predictions as input. Each participant performed 3 to 4 activity patterns per scenario for 4 to 5 minutes each, repeating each pattern three times to simulate data collection over multiple time periods.

5.5.4.2 Results

We ran a leave-one-participant-out (LOPO) cross-validation to measure accuracy. Figure 5.14 shows the LOPO performance across all 8 participants with two kinds of activity inputs: (i) activities from manually annotated video data, and (ii) activities from the Mites ML pipeline.



(a) Accuracy (JC) comparison (b) Precision comparison (c) Recall comparison

Figure 5.14: Comparing the accuracy (JC), precision, and recall of TAO's context detection in the real-world user study. Activity inferences are obtained either from manually annotated video data or from the Mites platform [139]. TAO detects contexts at higher accuracy (JC: 72%, precision: 83%, recall: 69%) using video-annotated activities. Accuracy drops (JC: 22%, precision: 36%, recall: 24%) when using Mites activity predictions, reflecting inaccuracies in the underlying activity detection pipeline.

When TAO uses the annotated activities from video data, it detects contexts with accuracy comparable to the dataset-based evaluations (JC: 72%, precision: 83%, recall: 69%), confirming that the system is robust in real-world settings. However, when TAO receives activity input from the Mites pipeline, accuracy drops substantially (JC: 22%, precision: 36%, recall: 24%). To understand this drop, we examine the context-level precision and recall in Table 5.3. For contexts like *C: Coming in* and *C: On a phone call*, precision and recall decrease sharply compared to video-annotated activities, indicating that the Mites activity detection pipeline itself was inaccurate for these activities. This result highlights a key characteristic of TAO's design: because it is agnostic to the underlying activity detection mechanism, its context detection accuracy is bounded by the quality of the activity predictions it receives. With better sensing approaches and additional training data, one can achieve higher activity recognition accuracy, which would directly improve TAO's context detection. Advances in activity detection using different sensor modalities and ML techniques, such as those presented in Chapter 3 and Chapter 4, would benefit TAO's downstream performance.

5.6 Application: Context-Driven Wellness Metrics

The contexts detected by TAO are not an end in themselves; their value lies in the applications they enable. To demonstrate this, we prototyped two wellness applications that use TAO's context predictions to assess occupant productivity and stress. These applications are illustrative rather than exhaustive: a wide range of application scenarios can utilize contextual information, from understanding eating habits to detecting sudden lifestyle changes to monitoring the elderly in ambient assisted living environments. We present these two applications to show that rich contextual information, as opposed to raw activity labels, is necessary for computing meaningful wellness metrics.

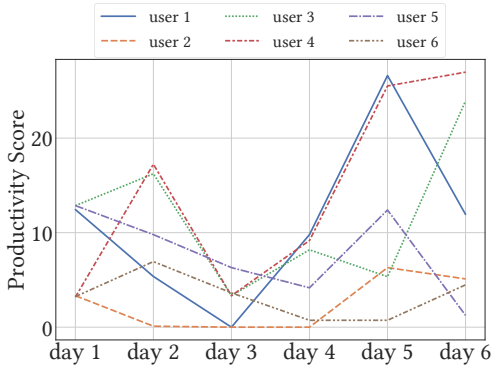


Figure 5.15: Productivity score for six users from the *ExtraSensory* dataset over six days.

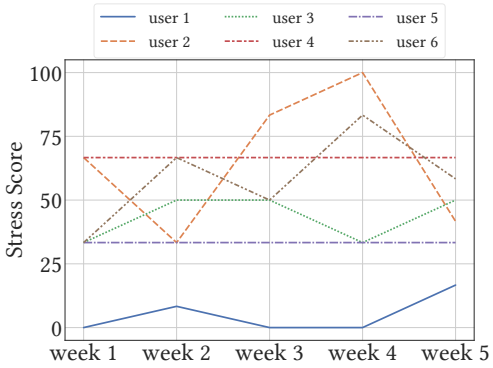


Figure 5.16: Stress score for six users from the *Casas* dataset over five weeks.

5.6.1 Wellness Metrics for Productivity

Prior work has shown that an individual's brain can sustain focused attention for approximately 90 to 120 minutes before requiring a break, a cycle known as the *ultradian* rhythm [38, 203]. Common strategies to improve productivity include 15-minute breaks during work stretches [185], the Pomodoro technique of 25-minute focus intervals followed by 5-minute rests [53], and 90-minute work windows for a single task followed by 20-minute breaks. Several factors can affect productivity, including procrastination, distractions, and multitasking [42, 83].

We built a productivity application based on these metrics using the contexts derived from TAO. The application identifies how long an individual remains in the *C: Office work* context and whether they switch to contexts denoting a break (*e.g.*, *C: Relaxing*) at appropriate intervals. It assigns a productivity score based on adherence to these patterns, penalizing continuous office work without breaks. Figure 5.15 illustrates how productivity scores for different users from the *ExtraSensory* dataset change over a week. User 1 shows lower productivity in the middle of the week compared to the start and end, while user 6 maintains consistently low productivity throughout. These patterns showcase the potential of using context predictions from TAO to surface productivity insights that would not be visible from raw activity labels alone.

5.6.2 Wellness Metrics for Stress

Several survey-based studies have established correlations between physiological stress and long-term behavioral patterns that can be expressed in terms of the contexts a person occupies, the duration spent in each context, and the frequency of context transitions within a day or a week. We reviewed 11 such surveys across a wide variety of demographics and identified four categories of context patterns associated with varying stress levels. Cao *et al.* [46] showed that absence of physical activity and low-intensity physical activities are associated with high stress, corresponding to a sustained absence of contexts like *C: Exercising* or *C: House work*. The same study found that high-intensity transport-related activities lead to elevated stress, corresponding to frequent *C: Commuting* patterns. Can *et al.* [44] studied the interaction between working

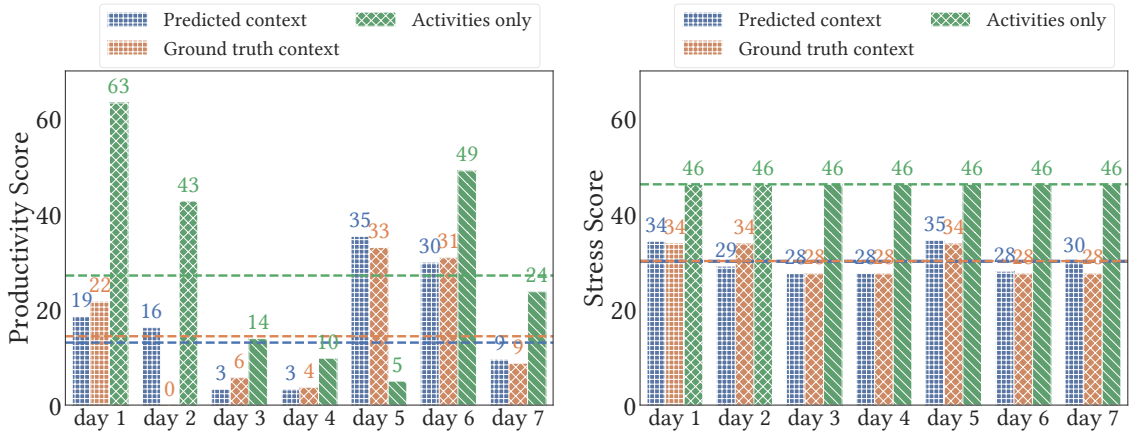
patterns and physical activity, showing that individuals who work shifts without regular exercise experience high stress, while those with regular day shifts and exercise have lower stress scores, patterns expressible through weekly distributions of *C: Office work* and *C: Exercising*. Other studies have quantified that working more than 10 hours per day or 50 hours per week leads to elevated risk of occupational health problems [219], and that the negative effect of an hour increase in weekend work is one and a half to two times larger than that of weekday overtime for white-collar workers, with relatively long weekend rest being key for reducing stress [183].

We prototyped an application that identifies these context patterns and tracks markers that can potentially relate to increasing or decreasing an individual's stress level. Our aim is not to quantify an individual's stress in absolute terms, as stress is multi-faceted and subjective. Rather, keeping track of these context patterns can help provide personalized recommendations for improving wellness. We assigned each marker a score based on severity (low or high) and impact (increasing or decreasing) on stress as reported in the survey literature, and approximated weekly and daily stress scores based on the cumulative score of identified context patterns. Figure 5.16 shows the variation in weekly stress scores for six users from the *Casas* dataset. Users 1, 5, and 4 show consistent stress scores across weeks at low, medium, and high levels respectively, indicating stable lifestyles. User 2 shows a sudden increase in stress, potentially reflecting a lifestyle change. User 6 shows a gradual upward trend, and user 3 shows fluctuations across consecutive weeks. These patterns can be studied further by examining the specific context patterns that contribute to each score.

5.6.3 Significance of Context for Wellness Applications

In both wellness applications, the context inference from TAO plays a crucial role in calculating meaningful metric values. To illustrate this, Figure 5.17 compares productivity and stress scores generated in three ways: using TAO's predicted contexts, using manually annotated ground-truth contexts, and using only raw activity information. We selected one of the two users for whom we annotated ground truth (user 3 in Figure 5.15) from the *ExtraSensory* dataset and computed scores for a week.

For a fair comparison, when using activities directly, we created a mapping of activities to plausible office contexts (*e.g.*, mapping *A: computer work* and *A: typing* to *C: Office work*) to emulate what a user would do in practice. The productivity score from TAO's predicted contexts (average/week = 16.57) closely matches the ground-truth score (average/week = 14.41), both indicating lower productivity over the week. In contrast, using activity information directly yields an inflated productivity score (average/week = 29.65) that misrepresents the user's actual productivity. Similarly, TAO's stress predictions (average/week = 30.12) match the ground truth (average/week = 30.17) closely, while activity-only stress scores are substantially higher (average/week = 46.25). These results demonstrate that accurate context predictions are essential for providing meaningful wellness metrics and that raw activity information, without the temporal reasoning that context detection provides, can lead to misleading assessments.



(a) Productivity score comparison

(b) Stress score comparison

Figure 5.17: Comparison of productivity and stress scores generated using context predictions from TAO, ground-truth context labels, and activity information alone, for a single user (user 3) in the *ExtraSensory* dataset. Both the productivity scores (avg/week = 16.57) and stress scores (avg/week = 30.12) from TAO's predictions are similar to the ground truth (productivity avg/week = 14.41, stress avg/week = 30.17), while scores generated from activity information alone diverge substantially (productivity avg/week = 29.65, stress avg/week = 46.25).

5.7 Discussion and Limitations

Context awareness has long been proposed as a way to allow computing technologies to understand human behavior and adapt dynamically to provide meaningful insights. Using IoT sensors coupled with machine learning to sense human activities and then detect semantically meaningful contexts from those activities is a natural progression. Through TAO, we have shown that understanding context from complex activity patterns is feasible and can enable useful applications. Here we discuss limitations and directions for improvement.

Sensitivity to the Underlying Activity Detection Pipeline: The real-world study in Section 5.5.4 revealed that TAO's context detection accuracy drops substantially when the underlying activity recognition system is itself inaccurate. This is an inherent property of TAO's design: because it is agnostic to how activities are detected, it treats activity predictions as ground truth, and any errors in those predictions propagate into context detection. One approach to mitigating this is to choose activity recognition architectures that are robust to noise in training data, for example by using ensembles of models that tolerate noisy labels. However, different recognition systems introduce different kinds of errors (some miss real activities, others hallucinate activities from sensor noise), and building a context detection system that is robust to all such error modes remains a challenging open problem. Within the scope of this dissertation, the bootstrapping approaches presented in Chapter 3 and Chapter 4 represent progress toward more accurate activity recognition from privacy-preserving sensors, and improvements at that stage would directly improve TAO's downstream performance.

Improvements to Sensing Approaches: The sensing technologies available today, including

commercial devices and multi-modal platforms like Mites, are enabling richer understanding of everyday activities. However, many of these systems are still evaluated in controlled settings and report HAR accuracies of 70% to 80% [33, 116]. Research is still needed to explore sensing modalities and activity recognition pipelines that achieve higher accuracy (90% to 95%) in real-world settings without requiring significant amounts of in-situ labeled training data.

Encoding New Contexts: While TAO can detect meaningful contexts from parallel, sequential, and interleaved activities, the majority of the contexts it models are based on the ontology we designed. Given that new contexts will likely be discovered and will require extending the ontology, tools are needed for end users and application designers to express their own context definitions, such as custom activity-to-context rules [131]. The temporal pipeline's ability to discover recurring patterns that the ontology does not cover provides a foundation for this, but closing the loop, automatically proposing new ontology entries from discovered clusters, remains future work.

Scenarios Where TAO Misses Contexts: We identified two primary sources of inaccuracy. First, transitions between contexts produce activity patterns (*e.g.*, *A: sleeping* followed by *A: taking a shower* every morning) that recur regularly but do not correspond to any meaningful context. This is particularly problematic for short-duration contexts like *C: Using bathroom*, *C: Coming in*, or *C: Going out*. Second, the current strategy for combining the ontological and temporal pipelines is relatively simple: we select one pipeline's output based on the number of contexts predicted. While this provides a richer set of contexts and higher recall, it can reduce precision when either pipeline detects an incorrect context. A more sophisticated combination strategy could utilize characteristics of the activity trace to dynamically weight one pipeline over the other.

5.8 Summary

This chapter presented TAO, a hybrid context detection system that combines ontological reasoning with temporal clustering to synthesize semantically meaningful contexts from streams of fine-grained human activities. The ontological pipeline provides a structured vocabulary for mapping well-defined activity patterns to contexts, while the temporal pipeline discovers recurring interleaved patterns from data and uses the ontology to assign them semantic labels. By combining both components, TAO handles the full range of activity structures that arise in daily life: sequential, parallel, and interleaved.

We evaluated TAO on two large-scale public datasets, *ExtraSensory* and *Casas*, achieving 80% and 87% context detection accuracy respectively, substantially outperforming prior approaches. A real-world deployment with eight participants confirmed that TAO can detect contexts from actual sensor-driven activity streams, and highlighted that context detection accuracy is bounded by the quality of the underlying activity recognition. The wellness applications demonstrated that TAO's context predictions are close enough to ground truth to produce meaningful productivity and stress metrics, and that these metrics diverge substantially when computed from raw activity information alone.

Within the broader arc of this dissertation, TAO addresses the third stage of the sensing pipeline: translating activity streams into the contextual understanding that applications require. It is agnostic to how those activity streams are produced, which means it can sit downstream of

systems like VAX and OrganicHAR without modification. The contexts it produces, however, are only useful if occupants can access them. A system that knows someone has been in focused work for three hours is valuable only if it can communicate that insight in a way the occupant understands. The next chapter turns to this final stage of the pipeline: providing a conversational interface through which occupants can query their building's sensor infrastructure in natural language.

Chapter 6

Natural-Language Interaction with Building Sensor Infrastructure

The previous three chapters traced a path from raw sensor signals to contextual understanding. VAX showed that training labels for privacy-preserving sensors can be generated automatically from temporary audio and video observation. OrganicHAR showed that the set of activities worth detecting can be discovered from sensor data itself, at a granularity matched to what the sensors can distinguish. And TAO showed that streams of fine-grained activities can be translated into the higher-level contexts that downstream applications require. Together, these systems address three of the four stages of the sensing pipeline described in Chapter 1: training labels, activity vocabulary, and context synthesis.

Those three chapters developed and evaluated their contributions in individual environments, primarily homes, where the occupant is also the sensor owner. This chapter shifts to shared buildings, where sensing infrastructure is institutional and occupants interact with sensors they did not install and may not be aware of. The same Mites sensors used in the deployments of earlier chapters are deployed at building scale in the university building that serves as this chapter's testbed [34], and composing the full pipeline into an integrated home deployment remains an engineering challenge discussed in Section 7.1.3. This chapter addresses the interface stage independently, both because the interface problem is valuable in its own right and because the shift to shared buildings surfaces a challenge absent from single-occupant settings: when multiple stakeholders share sensor infrastructure, one person's comfort query can constitute surveillance of another. The access control and query reformulation mechanisms at the center of this chapter's contribution arise directly from this multi-stakeholder tension, a tension that does not exist when the occupant owns the sensors and is the only person whose data is at stake.

The question that remains is how occupants can access and derive value from the stored data underneath. A building that can detect activities, discover behavioral patterns, and reason about context is only useful if the people inside it can ask questions and receive answers without navigating fragmented dashboards or possessing technical expertise. This chapter presents BuildingChat[1], a conversational system that enables building occupants to query sensor infrastructure directly through natural language.

[1] `https://github.com/synergylabs/building-chat`

6.1 Introduction

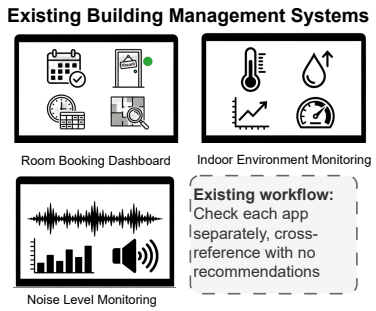
As discussed in Chapter 2, modern buildings collect extensive sensor data, but occupants rarely benefit from it. Building Management Systems direct sensor readings to facility management dashboards, and occupants who want to know whether their floor is noisy or which conference room is comfortable must consult separate applications for each subsystem, each with its own interface and credentials [81]. The fragmentation problem is not a lack of data but a lack of access: the sensing infrastructure described in Section 2.1 generates a continuous, rich data stream that is discarded at the interface layer by forcing occupants into narrow, pre-designed interaction paths using specific APIs. In commercial buildings, this is partly attributable to industry incentives, as BMS vendors focus on energy and comfort management for facility managers and building operators. However, many buildings operate under shared governance structures where operators and occupants have a mutual stake in how spaces function, including universities, hospitals, public institutions, and co-managed workplaces. In these settings, a barrier to occupant access is the absence of mechanisms that can make sensor data available to multiple stakeholders while respecting privacy constraints.

Section 2.5.2 introduced conversational interaction as a paradigm that could bridge this gap: instead of requiring occupants to identify the right application, a conversational system allows them to express needs in natural language and receive answers drawn from whatever data sources are relevant. Section 2.5.3 identified three challenges that make applying this paradigm to building sensor infrastructure difficult: queries often require multi-step analytical composition rather than single lookups, shared buildings introduce a multi-stakeholder privacy problem, and occupants routinely underspecify their queries. This chapter presents a system that addresses all three.

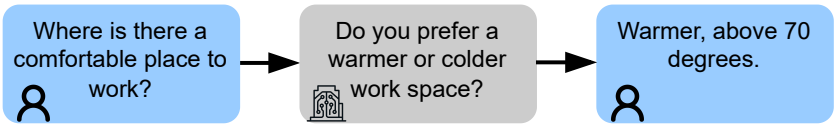
To ground these challenges in the actual needs of building occupants, we conducted a formative study with 11 participants across five building roles: private office occupants, shared office occupants, resource managers, building managers, and emergency managers (Section 6.3). The study surfaced query needs that varied widely by role and information granularity, from personal comfort questions to cross-location comparisons to temporal usage patterns. Of the 80 in-scope queries we collected, 72% required multi-step analysis, either retrieving and comparing data across multiple locations or analyzing temporal patterns. Participants also raised substantial concerns about surveillance and the potential for queries to reveal identifiable patterns about individuals, creating direct tensions with desired functionality. These findings confirmed the challenges identified in Chapter 2 and provided the concrete design requirements for BuildingChat.

Two approaches might seem promising for building such a system. One is to provide sensor data directly to a Large Language Model (LLM) for reasoning, but buildings contain thousands of sensors generating continuous time-series data, far more than fits in an LLM's context window. Another is to have an LLM generate retrieval and analysis code for each query, which offers flexibility but sacrifices explainability and predictability of responses. The surveillance concerns from our formative study impose further constraints: in most building deployments, sending raw sensor data to LLMs is a non-starter due to privacy concerns, and access control cannot simply reject unauthorized queries because users lack visibility into what data exists.

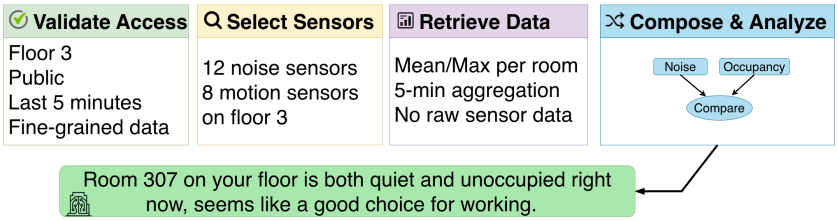
BuildingChat (Figure 6.1) takes a different approach. Rather than delegating analytical reasoning to the LLM or asking it to generate arbitrary code, BuildingChat uses the LLM as an orchestrator of pre-verified analytical operators. The design embodies the bounded-role principle



(a) To find a comfortable place to work, building occupants today must check multiple applications separately: room booking, noise monitoring, and occupancy dashboards.



(b) BuildingChat provides a conversational interface and can clarify ambiguous requests.



(c) The system validates access, selects relevant sensors, retrieves aggregated statistics, composes multi-sensor analysis, and generates a contextual recommendation.

Figure 6.1: Overview of BuildingChat. (a) The current state of occupant access to building data requires consulting separate applications for each subsystem. (b, c) BuildingChat provides a single conversational entry point that decomposes natural language queries into composable analytical operations, enforces access control, and produces contextual recommendations.

introduced in Section 1.2: the LLM remains part of the system but never receives raw sensor data, interacting only with verified operators whose outputs it composes. The user effort being replaced is the need to navigate specific interfaces or formulate database queries; the automated alternative is a conversational system that assembles analysis pipelines on the fly from a library of composable operations.

Concretely, BuildingChat addresses the three challenges through three mechanisms. First, it decomposes queries into composable template operations that execute over pre-aggregated sensor statistics, enabling multi-step analysis while maintaining predictable, explainable execution that works with locally deployed LLMs. The system arranges these operations into directed acyclic graphs (DAGs) that can express parallel branches and merges, supporting queries that combine multiple criteria or compare across locations. Second, BuildingChat enforces data access policies that constrain queries across multiple dimensions, including spatial scope, temporal resolution, sensor count, and query rate, so that combinations of allowed queries cannot be used to reconstruct individual patterns. Because access control operates through programmatic validation rather than prompt-level instructions, unauthorized data access is structurally mitigated within the pipeline. Third, when a query is underspecified or involves inaccessible data, the system generates clarifying questions or proposes alternative formulations, proceeding only with explicit user consent.

We evaluate BuildingChat through three complementary studies. First, we measure query accuracy on a mock building with controlled ground truth, comparing against three baselines representing established LLM system paradigms: code generation, in-context data reasoning, and a ReAct agent loop. BuildingChat achieves 88% accuracy, outperforming the baselines by 14 to 46 percentage points, and maintains the highest accuracy on every model tested across six language models spanning local and cloud deployments. Second, we evaluate whether alternative

designs can enforce equivalent access control. When baselines are augmented with explicit policy descriptions and instructed to reject unauthorized queries, they still leak restricted data in 13 to 46% of cases; BuildingChat prevents unauthorized data access by construction. Third, we conduct a usability study with 10 participants in a building instrumented with over 300 multi-modal sensors to understand the experience of querying sensor infrastructure through conversation.

In summary, this chapter makes the following contributions:

- We conduct a formative study with 11 participants across five building stakeholder roles, identifying query patterns that vary by role and analytical complexity, and surfacing tensions between desired functionality and privacy concerns that shaped the system design.
- Based on insights from the formative study, we design and implement BuildingChat, a conversational interface for building sensor infrastructure that decomposes queries into composable template operations arranged as directed acyclic graphs, enforces tag-based access control through programmatic validation, and iteratively refines queries when information is missing or inaccessible.
- We evaluate BuildingChat and show that it achieves 88% query accuracy, outperforming code generation, in-context reasoning, and ReAct agent baselines by 14 to 46 percentage points, while preventing the unauthorized data leaks that occur in 13 to 46% of baseline queries when access control is enforced through prompts alone.
- We evaluate BuildingChat in two usability studies with 21 building occupants. The first study with 11 participants informed design iterations from V1 to V2. The second, a three-day deployment with 10 participants, yielded a System Usability Scale score of 80 ± 8, with all participants finding responses understandable. We also identify open challenges in interpreting sensor values and LLM reliability.

The remainder of this chapter is organized as follows. Section 6.2 positions BuildingChat against the most closely related systems for building data access, LLM-based query generation, and access control. Section 6.3 presents the formative study and its design implications. Section 6.4 describes the system design, covering query understanding and reformulation, composable template operations, access control, and response generation. Section 6.5 reports on the initial user study that informed design iterations. Section 6.6 presents the technical evaluation against baselines. Section 6.7 reports the usability study results. Section 6.8 discusses lessons learned and limitations, and Section 6.9 summarizes the chapter's findings in the context of the broader dissertation.

6.2 Positioning Against Prior Work

The challenges of applying conversational interfaces to building sensor data, including interface fragmentation, multi-step query composition, multi-stakeholder privacy, and query underspecification, were introduced in Section 2.5. Here we focus on the specific systems most directly comparable to BuildingChat and on the gaps that motivated its design.

6.2.1 Querying Building and IoT Sensor Data

Natural language interfaces have been explored across several IoT domains at different scales. In the smart home domain, systems such as Sasha [109] and SAGE [178] use LLM-based agents to respond to underspecified user commands like "make it cozy," decomposing goals into action plans and dynamically constructing prompts, while other systems focus on enabling lower-power on-device reasoning [110]. These smart home systems focus primarily on real-time querying and control for a single user. In the personal sensing domain, DeepSQA [223] and SensorChat [234] provide natural language interfaces to historical sensor data from wearables and fitness trackers, where a single user owns all data. None of these systems address the multi-stakeholder privacy challenges or the cross-location analytical queries that arise in shared building environments.

For buildings specifically, recent work spans a range from metadata discovery to sensor analytics. BuildingGPT [124] proposed retrieval-augmented generation for querying building metadata in Brick Schema, enabling users to discover what sensors and equipment exist but not to perform analytical queries over their readings. BuildingSage [58] introduced an AI copilot that generates code for querying Brick metadata and sensor data, addressing data exfiltration risks by keeping raw data on-premise and sharing only the metadata schema with cloud LLMs. In shared buildings, however, privacy risks also arise from occupants themselves querying data about each other's spaces; this internal threat model, where the same query may be appropriate from one stakeholder but invasive from another, is complementary to data locality and is the focus of BuildingChat's access control framework.

BuildingQA [145] and BIM-GPT [243] address metadata discovery through different generation approaches, benchmarking SPARQL generation over Brick knowledge graphs and enabling natural language access to BIM models respectively, but neither queries sensor readings. OntoSage [62] combines Brick Schema-based SPARQL generation with 30 analytics microservices that operate over time-series sensor data, covering temperature analysis, anomaly detection, forecasting, and indoor air quality assessment across three building deployments. Each query is routed to a single analytics function by a classifier, so queries requiring multiple analytical operations, such as comparing noise and temperature across candidate spaces, cannot be expressed. This limitation motivated our composable template approach, where general-purpose analytical operators are arranged into directed acyclic graphs that can express multi-criteria queries through independent branches that merge at comparison or ranking nodes. JARVIS [119] similarly bridges query understanding and sensor analytics for HVAC systems, using a two-stage LLM framework for SQL-based retrieval and statistical processing over commercial building data. These systems demonstrate that LLM-mediated access to building sensor data is feasible, but they operate in single-user settings where one operator interacts with the full sensor inventory, without considering scenarios where multiple stakeholders require different levels of access to the same data.

More broadly, LLM-based approaches to data analysis vary in how they distribute reasoning between the language model and deterministic computation. Program-synthesis approaches generate executable code for each query [163, 196]. API-calling agents chain tool invocations through iterative reasoning [156, 168]. And autonomous systems reason directly over structured data presented in-context [239]. Building sensor data presents challenges for all three paradigms: time-series readings require temporal aggregation rather than point lookups, spatial hierarchies require context-aware resolution, and occupants routinely underspecify queries with implicit

context (*e.g.*, "my office feels warm"). We compare BuildingChat against baselines representing each of these paradigms in Section 6.6.

6.2.2 Access Control for Shared Sensor Data

Access control is a particular challenge for conversational systems because the core language model is non-deterministic, but access control should be deterministic and reliable. Planas *et al.* integrated role-based access control (RBAC) into chatbot specifications through a domain-specific language, supporting both restricting which queries a role can trigger and varying response precision levels based on user privileges [161, 162]. In their framework, precision tiers are fixed per role; building sensor queries introduce additional complexity because the appropriate resolution depends on query attributes such as recency and spatial scope, not role alone. Other work uses LLMs or other ML models to make access decisions directly. AirGapAgent [23] and LLMAC [246] take this approach, but LLMs are non-deterministic and may be vulnerable to prompt injection. In BuildingChat, we use LLMs to help compose a query compatible with the access control policy, but do not use an LLM to enforce the policy. SEAgent [96] applied attribute-based access control to prevent privilege escalation in LLM agent tool use, but assumed resources are discrete (intents, tools, documents). When multiple stakeholders query shared continuous sensor data, access decisions must also account for query parameters, as the same user with the same role may be permitted a query at one temporal resolution but not another, and data accessible in aggregate may not be accessible for individual spaces.

Privacy research in IoT has primarily studied smart home contexts with single users or small households [157, 158]. SmartAuth [202] proposed user-centered authorization for IoT applications, and work on interactive privacy negotiation [245] demonstrated that groups in multi-occupant scenarios can reach privacy consensus through structured interfaces. Zhang *et al.* [241] found that users of LLM-based conversational agents face trade-offs between privacy, utility, and convenience with limited awareness of disclosure risks, suggesting that conversational building systems must carefully manage what information is revealed through query responses. Researchers have also explored differential privacy for building data [103], though achieving meaningful guarantees for aggregate location data remains challenging in practice [91].

Some work has explored how to gracefully handle scenarios where a user's query violates an access control policy. Adeniye *et al.* [8] proposed a system that guides users toward privacy-compliant alternatives when queries cannot be fulfilled, similar to BuildingChat's query reformulation mechanism, though it operates on static graph data rather than continuous sensor streams. In the ontology domain, query relaxation techniques systematically broaden failing SPARQL queries to return partial results [28], though these operate silently on structured queries rather than negotiating alternatives with users through natural language.

6.2.3 Research Gap

Smart buildings consist of multiple stakeholders (*e.g.*, occupants, resource managers, facility staff, emergency responders) who share physical space and may have legitimate but conflicting interests in the same sensor data. No single user owns the data, and the same query may be appropriate from one person but constitute surveillance from another. They are distinct from traditional databases

Table 6.1: Comparison across capabilities required for conversational sensor data access in multi-stakeholder buildings. Rows 1–4 are building data systems; rows 5–6 are access control frameworks for conversational systems, for which sensor analytics and query composition are outside their design scope (—). ✗ indicates the capability is absent.

#	System	Sensor/Metadata Analytics	Multi-Criteria Composition	Multi-Stakeholder Access Control	Query Reform.
1	BuildingGPT [124]	✓ metadata only	✗	✗	✗
2	BuildingSage [58]	✓ LLM-gen. code	✗	✗	✗
3	OntoSage [62]	✓ 30 specialized fns.	✗	✗	✗
4	JARVIS [119]	✓ LLM + SQL	✗	✗	✗
5	Planas *et al.* [161]	—	—	✓ role-driven	✗
6	LLMAC [246]	—	—	✓ role-driven	✗
	BuildingChat	✓ *general-purpose*	✓ *composable ops.*	✓ *role+query*	✓ *consent-based*

in that users often do not know what data exists, making simple access denials uninformative and frustrating. The difficulty lies not in missing components but in their integration (Table 6.1): existing systems address sensor analytics or multi-stakeholder access control independently, but none of them combine these with multi-criteria composition or query reformulation in a unified pipeline. This requires an end-to-end system that considers these aspects holistically, rather than treating any single aspect in isolation.

6.3 Soliciting User Needs from Smart Buildings

We conducted a semi-structured interview study with 11 participants to understand how different building stakeholders interact with their spaces and what they would want from a smart building system. We designed the study to address two research questions: **RQ1:** How would different personas interact with smart building systems via a chatbot interface? **RQ2:** What concerns do users have around the deployment and use of smart building chatbot systems?

6.3.1 Method

We targeted five personas for our study: *private office occupants*, *shared office occupants*, *resource managers* who manage scheduling and IT resources, *building managers* who oversee building systems, and *emergency managers* responsible for building safety. Participants were required to be 18 years or older, fluent in English, and have regular interaction with office buildings. We recruited 11 participants: three private office occupants, three shared office occupants, two resource managers, one building manager, and two emergency managers.

Our semi-structured interviews lasted 60 minutes and were conducted either in-person or remotely; participants were paid $15 USD. The protocol consisted of four phases: (1) establishing participants' relationship with their building, (2) introducing a hypothetical smart building with sensing capabilities (temperature, sound levels, motion) to ground the discussion, (3) exploring desired queries and interactions, and (4) discussing privacy concerns and boundaries. Interview

recordings were transcribed and analyzed using thematic analysis by one researcher to identify emerging themes. The study was approved by our institution's IRB. The complete interview script is provided in Appendix C.2.

Limitations: The sample (n=11) was drawn from a single institution, and participants responded to a hypothetical system rather than an actual deployment. Therefore, it is possible that the insights we found are specific to this particular community. In addition, the five roles we studied may not capture all building stakeholders, such as visitors or janitorial staff.

6.3.2 Findings

6.3.2.1 Query needs vary by role and information granularity

Participants' information needs clustered into three groups that differed in topic, granularity, and completeness of information they required (Table 6.2). Building occupants with private or shared offices oriented their queries around personal comfort and resource availability. Faculty participants wanted to understand conditions in their own offices, *e.g.*, temperature, interruption frequency, and whether collaborators were nearby. Students focused on finding spaces to work, asking which rooms were quiet and whether shared resources like microwaves were currently available.

Resource and building managers wanted proactive detection of issues before they became disruptive. Both resource managers (P7, P9) described how a system could help identify problems with spaces before they affect others in the building. However, they worried about information overload and wanted actionable summaries rather than raw data. P9 specifically requested temporal visualization: "that would really help to be able to see these kind of things graphically on a timeline for a space or a building."

Emergency managers had distinct needs centered on crisis response, *e.g.*, detecting gas leaks, locating occupants during emergencies, and identifying escape routes. They wanted maximum information completeness and expressed no concern about overload. These differences reflect not just variation in topics but different granularity requirements. Given these findings and the broad set of use cases that varied between personas, we concluded that a chatbot design would be appropriate for BuildingChat to provide the flexibility needed to support diverse use cases.

6.3.2.2 Queries require multi-step reasoning

Beyond simple sensor lookups, many of the questions participants described required reasoning across multiple locations or time periods. Of the 80 in-scope queries we collected, 58 (72%) require multi-step analysis by either retrieving and comparing data across multiple locations or analyzing temporal patterns (Table 6.2). For example, to answer P6's question "Is it that all offices are this loud, or this hot, or is it just that I'm the unlucky one today?", a query system would need to (a) retrieve temperature and noise for P6's office, (b) identify other offices on the same floor, (c) retrieve and aggregate readings from those offices, and (d) compare P6's values against the aggregate. No single database lookup can produce this answer. Recent work on tabular and time-series reasoning confirms that decomposing multi-step analytical queries into structured operations with deterministic execution outperforms end-to-end LLM reasoning [97, 231].

Table 6.2: Query distribution from our formative study by role and analytical complexity. Point measurements retrieve a single sensor reading; cross-location queries compare or filter across multiple sensor groups; temporal queries analyze patterns over time; outside scope queries require capabilities beyond sensing (*e.g.*, wayfinding).

Role	N	Point Meas.	Cross-Loc. (Comp./Filter)	Temporal Analysis	Outside Scope	Total
Emergency Managers	2	8	3	0	6	17
Shared Space Users	3	5	17	3	9	34
Private Office Occupants	3	6	7	3	6	22
Resource Managers	2	1	3	8	2	14
Building Manager	1	2	8	6	4	20
Total	**11**	**22 (28%)**	**38 (47%)**	**20 (25%)**	**27**	**107**

Beyond comparison, questions about space availability require checking the status of multiple rooms and filtering based on current occupancy. Temporal patterns represented another common category: two participants (P5, P9) wanted to understand when spaces were most heavily used, which requires aggregating occupancy data over time rather than reporting instantaneous readings.

6.3.2.3 Privacy concerns center on surveillance and inference

When asked about concerns with a smart building system, participants' responses diverged by role. Emergency managers engaged minimally with privacy questions, focusing instead on information accuracy. Occupants expressed substantial surveillance concerns. Five of eleven participants explicitly worried about authorities using such a system for surveillance. P10 emphasized the need for prevention of "behavior patterns of the occupants in an identifiable way." Two participants expressed skepticism that meaningful privacy was achievable with such systems.

This created direct tension with desired functionality. Faculty participants wanted to know where their students were and whether collaborators were on campus. But students and staff worried about exactly this capability being used against them. One participant (P4) acknowledged this tension, indicating they would prefer data to be available but would protect themselves in ways they felt necessary to maintain their privacy. Several participants requested specific mechanisms: opt-in consent (P8, P10), need-to-know access restrictions (P8, P9), and aggregate-only data without individual-level detail (P3, P9).

6.3.3 Design Implications

These findings shaped BuildingChat's design in three ways. First, the diversity of query types, spanning point-in-time measurements, temporal patterns, and cross-location comparisons, combined with the multi-step reasoning many queries require, motivated a semi-structured approach where common query patterns are encoded as pre-defined operations that can be composed together to answer complex questions (Section 6.4.3). Second, the surveillance concerns raised reveal that access control for building sensors differs fundamentally from traditional database

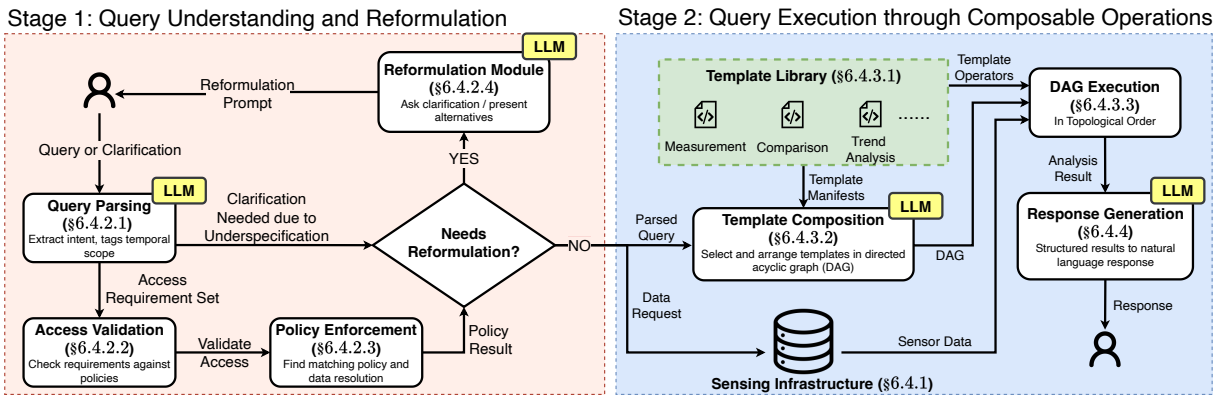


Figure 6.2: System overview for BuildingChat. **Stage 1 (Query Understanding and Reformulation):** User queries are parsed to extract intent, spatial tags, and temporal scope (Section 6.4.2.1). The system derives access requirements and validates them against policies (Section 6.4.2.2). If the query is underspecified or unauthorized, the reformulation module (Section 6.4.2.3) proposes default interpretations or accessible alternatives, prompting the user for confirmation or clarification. **Stage 2 (Query Execution):** Once validated, the template composition module selects and arranges templates (Section 6.4.3.1) from the library into a Directed Acyclic Graph (DAG) (Section 6.4.3.2). The system retrieves aggregated sensor statistics based on the approved policy, executes template operators (Section 6.4.3.3), and generates a natural language response (Section 6.4.4). Yellow badges indicate LLM-powered components; other components execute deterministic logic.

permissions: privacy risk emerges from the interaction of multiple dimensions, including the person asking, the spatial and temporal granularity, and the accumulation of queries over time to reveal patterns (Section 6.4.2.2). Third, participants frequently expressed queries that were underspecified, using vague qualifiers, contextual references like "my office," and implicit temporal context, while simultaneously expressing concern about being denied access without recourse. These patterns suggest that a conversational interface cannot simply reject problematic queries but must guide users toward alternatives that are both answerable and respect privacy (Section 6.4.2.3).

6.4 BuildingChat System Design

Figure 6.2 presents an overview of BuildingChat. Consider a user who asks *"Where is there a comfortable place to work?"*. During the first stage, BuildingChat parses this query to extract intent, spatial scope, and temporal scope (Section 6.4.2.1), then validates access requirements against data access policies (Section 6.4.2.2). Since "comfortable" is ambiguous, the reformulation module (Section 6.4.2.3) checks the user's profile preferences and, finding no comfort preference set, proposes a default interpretation prioritizing noise and lighting. The user confirms or adjusts, and we move to Stage 2. First, we retrieve sensor data aggregated to the permitted resolution, and in parallel compose a Directed Acyclic Graph (DAG) of template operations (Section 6.4.3.2): parallel branches process noise and light data for candidate spaces, then merge at a comparison

Table 6.3: An illustration of tag types and values for sensor organization.

Tag Type	Example Values	Privacy Relevance
visibility	public, shared, individual	Primary access dimension
floor	1, 2, 3, 4	Spatial granularity
cardinality	NE, NW, SE, SW	Building quadrant
locationType	office, corridor, study room	Space classification
owner	user email	Personal space access
department	organizational units	Group-level access

node that ranks spaces by both criteria. Templates execute deterministically (Section 6.4.3.3), and the response generator (Section 6.4.4) produces a natural language recommendation identifying the best matching spaces, accompanied by visualizations comparing the requested metrics across candidate locations.

This two-stage architecture addresses three interconnected challenges from our formative study. The prevalence of multi-step queries (72% of in-scope queries, Table 6.2) requires a composition mechanism that chains analytical operations rather than treating each query as a single lookup. The surveillance and inference concerns participants raised (Section 6.3.2.3) require that access control be enforced before any data is retrieved, so that denied queries can enter reformulation rather than failing silently or leaking unauthorized data. And the frequency of underspecified queries requires the system to negotiate with users before committing to execution. These challenges are entangled rather than independent. Access validation must precede composition because the set of permitted data determines which template arrangements are valid, and reformulation must precede both because clarifying the query may change what data is requested. This ordering ensures that by the time a query reaches Stage 2, it is fully specified, authorized, and user-approved.

6.4.1 Smart Building Infrastructure

BuildingChat is designed to integrate with smart building deployments that have sensors throughout the building and store historical sensor data. In our implementation, we integrated it with BuildingDepot [218], an open-source building operating system, deployed in the same academic building described in Section 2.1.1, instrumented with the Mites multi-modal sensing platform [34]. The data collected from these sensors include audio levels (microphone), vibrations (accelerometer), motion (PIR), low-resolution (8×8) thermal imaging, temperature, humidity, and light levels. Sensor metadata is defined using normalized *tags* and *attributes*, which indicate metadata such as the *location type* (*e.g.*, office or common area), *location* (*e.g.*, room 345, 3rd floor northeast), and the *owner/occupants* (Table 6.3).

To implement BuildingChat, we extended BuildingDepot with a new API to query *aggregate* sensor data. As our formative study revealed, many queries share a common structure: they involve (1) identifying relevant sensors, (2) grouping or aggregating readings, and then (3) filtering or ranking results. Existing BMS platforms aggregate sensor data for pre-configured facility dashboards and gate visibility through coarse role-based tiers (administrator, operator, viewer). They do not support per-query access policies that couple temporal resolution with

Table 6.4: Query structure components extracted during parsing.

Component	Description	Example
Intent	Information need	comparison, measurement, trend
Metrics	Sensor modalities	temperature, noise, occupancy
Locations	Spatial scope via tags	floor=3, locationType=study_room
Temporal	Time range	"right now", "today", "last week"
Qualifiers	Subjective descriptors	"warm", "quiet", "busy"

spatial granularity, nor do they expose interfaces for occupants to pose ad-hoc analytical queries. Our aggregate API addresses both gaps: it computes aggregations on demand based on query parameters, and it enforces access policies that constrain what resolution is permitted for each request. A query consists of a list of tag values and sensing modalities (*e.g.*, common areas and meeting rooms), time window (*e.g.*, last 12 hours), aggregation window (*e.g.*, 1 hour), and aggregation function (*e.g.*, median). We discuss the aggregate data query system further in Section 6.4.2.2.

6.4.2 Query Understanding and Reformulation

Throughout this chapter, a *query* refers to a natural language question posed by a building occupant to BuildingChat, such as "Is my office warm?" or "Which floor is quietest?" The first stage of BuildingChat transforms natural language queries into structured representations and negotiates with users when queries cannot be fulfilled as requested. This section describes how the system handles both unauthorized and underspecified queries, and the reformulation mechanism that guides users toward accessible alternatives.

6.4.2.1 Three-Stage Query Parsing

Based on our formative study, we identified five query components required for answering building-related queries: intent, sensor modalities, spatial constraints (expressed through tags), temporal scope, and qualifiers (Table 6.4). We separate parsing into three stages, each implemented as a separate LLM call (complete prompts in Appendix **??**).

The first stage identifies which tag *types* are relevant to the query: the LLM receives the user's query along with the building's tag taxonomy (the set of available tag types such as `floor`, `locationType`, `owner`) and outputs which tag types are referenced. For our running example, *"Where is there a comfortable place to work?"*, the first stage identifies that `locationType` and `location` are relevant (to find and enumerate candidate workspaces). The second stage performs full structured extraction: given the query, the relevant tag types identified in stage one, and the user's profile context, the LLM extracts specific values, intent, metrics, and temporal range. The user's profile context includes their office assignment and floor (retrieved from BuildingDepot), their timezone, and any preferences the user has configured through the profile settings. This context enables the parser to resolve spatial self-references deterministically: "my office" maps to the user's assigned room tags, "my floor" to their floor tag, and "here" to their

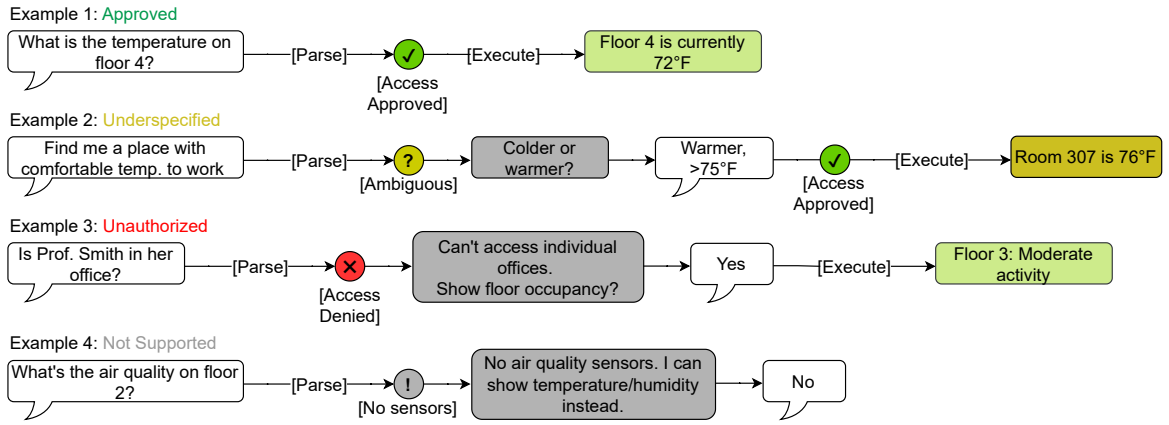


Figure 6.3: Query processing traces illustrating four scenarios: (1) a well-specified query proceeds directly to execution, (2) an underspecified query is resolved through profile preferences or a proposed default, (3) an unauthorized query prompts the system to propose an accessible alternative requiring user consent, and (4) a query beyond the building's sensing capabilities ends with an explanation of available alternatives.

current building context, without requiring clarification. When users ask about "occupancy" or "how busy" a space is, the system maps this to the relevant sensor modalities (motion, thermal imaging, audio levels, light) rather than expecting users to name specific sensor types. For this query, the second stage extracts an intent of location recommendation, metrics of noise and light (after clarification resolves "comfortable" to quiet and well-lit), and defaults temporal scope to the last minute. The system also checks whether the query would be allowed under the access control policy (Section 6.4.2.2) before querying for sensor data, to allow the query to be reformulated if necessary.

The final stage extracts relevant qualifiers for the query that describe the location and sensor modalities (*e.g.*, subjective terms like "warm" or "good", and vague location terms such as "here" or "there"). In this stage, we ask the LLM to serve as a judge to determine whether the qualifier needs additional clarification from the user. This separation into multiple LLM calls addresses a challenge we encountered during development: single-pass parsing confused tag types with tag values as the taxonomy grew, and qualifier judgments were more accurate when isolated from other extraction tasks.

Our formative study revealed that users expect a conversational system to understand their subjective context, and routinely underspecify queries: participants used phrases like "my office," "here," or "somewhere quiet" without providing explicit spatial coordinates, omitted temporal scope when asking about current conditions, and used qualitative descriptors like "warm" or "busy" without defining thresholds (see Figure 6.3, Example 2). BuildingChat resolves spatial self-references through the user's profile context during parsing, as described above. For ambiguous qualifiers, the system first checks the user's configured preferences: if a user who has set "temperature" and "noise" as their comfort priorities asks *"Where is there a comfortable place to work?"*, the system resolves "comfortable" to those metrics without prompting. When qualifiers cannot be

resolved through preferences, the system proposes a concrete default interpretation for the user to accept or reject, rather than asking open-ended clarifying questions. Once parsing produces a complete structured query, either directly or after user confirmation, the system validates whether the user has access to the requested data.

6.4.2.2 Access Control Framework

In a building context, occupants have varying privacy expectations; in our formative study, many participants expressed concerns about sensor systems being used for monitoring and surveillance (Section 6.3.2.3). The community norms and policies for each deployment likely differ, but we believe sensor systems should be designed to aid in understanding the building and *not* support tracking individuals. Traditional role-based access control approaches are inadequate in this respect: they assign permissions too coarsely, so if facility managers are allowed to access granular motion data to understand usage trends, they would likely also have the ability to see when an individual is using their office. In some cases, however, it may be acceptable for *aggregate* data to be queried based on defined policies. For example, while allowing building managers to access motion data for an individual office would be problematic, a system where they can learn motion trends aggregated across an entire building floor may be acceptable.

As discussed in Section 6.4.1, queries to BuildingDepot include a list of metadata tags to match, start and end time, aggregation time window, and aggregation function. Each query is checked against the list of *data access policies* and allowed only if it is permitted under at least one policy. The policies include:

- *Required metadata tags:* List of filters that must be included in this policy, *e.g.*, `owner:<user_id>`.
- *Optional metadata tags:* List of optional tags to further filter sensors, *e.g.*, `floor`. Unspecified tags are not permitted to prevent excessively narrow queries.
- *Minimum number of sensors:* Integer. The query is only permitted if it matches at least this many sensors.
- *Aggregation:* List of `[recency, time resolution]` tuples that define the minimum aggregation window for a query about data with a given recency, *e.g.*, `[last day, 15 minute granularity], [last week, 3 hour granularity]`. This policy allows querying more granular recent data while limiting the ability to query detailed data about past activity (when combined with a rate limit policy below).
- *Rate limits:* List of `[limit, time frame]` tuples that define the maximum number of queries in a rolling time window, *e.g.*, `[10 queries, last day]`.

This policy framework is designed to support various types of queries, including accessing data from the building occupant's own office (higher time granularity, not rate limited) and querying aggregate values across locations including other people's private offices (higher minimum number of sensors, lower time granularity, lower rate limits). Table 6.5 shows the policies used in our deployment, designed based on feedback from the formative study and community feedback from the deployment of the sensor system; the policies used for each deployment will likely vary based on community norms and expectations. Table 6.6 shows the time resolution constraints

Table 6.5: Data access policies used in our deployment. We included policies for personal data access, querying information about public spaces, and querying aggregates across larger sections of the building (which may include private offices). The `location` tag refers to the room number, `floor` and `cardinality` to approximate location in the building, and `visibility` and `locationType` to the type of space (*e.g.*, public corridor).

Use Case	Required Tags	Optional Tags	Min. Sensors	Time
Accessing own office	owner=⟨user_id⟩	location	1	unrestricted
Checking public spaces	visibility=public	location, floor, card.	1	Table 6.6
Multi-room aggregates	floor	locationType, card.	14*	Table 6.6

*To protect individual privacy by providing k-anonymity, we only allow queries including other people's private offices if at least seven offices are included (with two sensors each).

Table 6.6: Time resolution constraints for non-personal data. We allow high-granularity access to more-recent data, and lower-granularity access to historical data.

Recency	Time Resolution	Effect
15 minutes	5 seconds	Current state, fine resolution
1 hour	1 minute	Recent history, moderate aggregation
1 week	1 hour	Historical, coarse aggregation

for non-personal data. Queries that fail validation against these policies enter the reformulation module, which proposes accessible alternatives rather than simply denying the request.

6.4.2.3 Reformulation Module

When a query cannot proceed as requested, either because the user lacks access to the requested data or because the query falls outside the system's sensing capabilities, the reformulation module proposes a concrete alternative rather than simply denying the request (see Figure 6.3). For access denials, the module receives the original query and the reason for denial, then generates an alternative that addresses a similar information need using data the user is authorized to access. For example, if a user requests noise data for individual offices on their floor, the system explains the access restriction and proposes checking noise levels in public areas on the same floor instead (Figure 6.5). For queries beyond the system's capabilities, the module explains what sensing modalities are available and suggests the closest answerable query.

In both cases, the system presents the proposed alternative with explicit accept and reject options, proceeding only with user consent. If the user rejects the proposed alternative, they can reformulate their query manually or end the interaction. BuildingChat maintains session state to avoid re-proposing alternatives the user has already rejected. Intent clarification in LLM-based systems is an active research area with open challenges around generalization across domains and query types [133, 242]. Our clarification and reformulation module is tailored to the building sensor domain, handling the specific forms of underspecification we observed in our formative study (vague qualifiers, implicit spatial context, ambiguous temporal scope) rather than providing a general-purpose solution. Once a query is fully specified and authorized, it proceeds to template

Table 6.7: Packet types defining data contracts between templates.

Packet Type	Contents	Typical Producer
input_packet	Tag-aggregated sensor statistics	Database handler
measurement	Single value with location/metadata	current_measurement
measurement_set	Collection of related measurements	Parallel meas. branches
analysis_result	Findings, comparisons, diagnostics	spatial_comp., temporal_an.
location_set	Locations meeting filter criteria	threshold_filter
time_series	Temporal sequence of values	temporal_analysis

Table 6.8: Template library derived from our formative study query patterns.

Template	Inputs	Outputs	Formative Pattern
current_meas.	input_packet	measurement	Point-in-time comfort
spatial_compare	measurement	analysis_result	Cross-location comparisons
temporal_analysis	input_packet	time_series, anal._result	Pattern/trend detection
threshold_filter	measurement	location_set	Resource avail. filtering
web_search	(none)	analysis_result	External context (weather/events)
info_lookup	(none)	analysis_result	Building info queries

composition for execution.

6.4.3 Query Execution through Composable Operations

When all access requirements for a query are approved, the database handler retrieves sensor data for each validated requirement. This data arrives as an *input packet*: a structured container holding the tag combination that defined the sensor group, the column metadata describing each feature, and the time-indexed values. We adopt a compositional approach where common analysis patterns are encoded as reusable operations called *templates*. Each template performs one well-defined function: extracting current measurements, comparing values across locations, analyzing temporal trends, or filtering by thresholds. Many queries from our formative study require retrieving data from independent sources and merging the results: a comparison retrieves measurements from two locations and merges them at a comparison node, while a multi-criteria query branches per modality and merges at a ranking node. A directed acyclic graph (DAG) is the minimal structure that supports both branching and merging while guaranteeing bounded execution (Figure 6.4), so we express complex queries as DAGs of template operations. The system's role is to understand the user's intent and compose appropriate templates into a DAG, while the templates themselves execute deterministic analysis logic. This separation provides two properties: (1) predictability, as the same template executes the same code regardless of how the query was phrased, and (2) transparency, as each template has well-defined inputs, outputs, and internal behaviors.

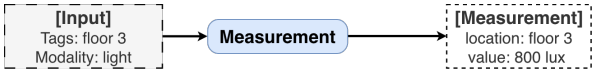
6.4.3.1 Template Structure

Each template consists of a *manifest* declaring its interface (accepted and produced packet types, parameters, execution constraints) and an *operator* implementing its analysis logic. Composing templates into DAGs requires that nodes agree on the format of data flowing between them. We define a set of packet types (Table 6.7) that serve as contracts between templates: during composition, the system validates that each edge connects compatible types, catching configuration errors before execution and enabling templates to be developed independently.

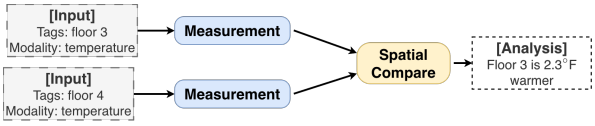
Table 6.8 describes the six templates we implemented, derived from the query patterns observed in our formative study. Four analytical templates address the core patterns participants described: checking current conditions, comparing across locations, analyzing temporal trends, and filtering by thresholds. Two additional templates handle queries that do not require sensor data retrieval: `web_search` retrieves external context such as weather conditions, and `info_lookup` answers informational queries about the building itself, such as room capacities or available sensor modalities. These non-sensor templates produce `analysis_result` packets directly, enabling DAGs that combine building sensor data with external sources (*e.g.*, "Is it warmer inside than outside?" branches into parallel sensor and weather retrieval paths). The template library is designed to evolve: in real deployments, queries that cannot be fulfilled reveal gaps that guide further development. The four analytical patterns reflect general ways people reason about shared physical spaces, and are not specific to the building type or sensor configuration of our deployment. Building administrators can extend the library without modifying the core BuildingChat pipeline. Full descriptions of each template's inputs, outputs, and logic are provided in Appendix C.3.

6.4.3.2 Template Composition

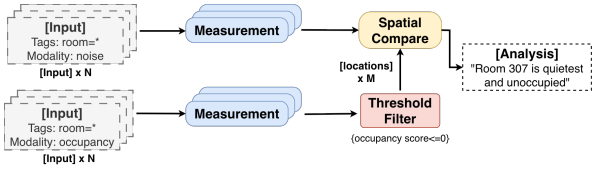
Given a parsed query and validated access requirements, the system must select appropriate templates, configure their parameters, and arrange them into an execution graph. We implement this as a structured generation task where an LLM produces a DAG specification, but the generation is constrained to ensure valid, executable compositions. The composition prompt provides the LLM with the structured query and template manifests with detailed parameter specifications. We only allow the tag combinations that pass access validation so that the LLM cannot compose queries over unauthorized sensor groups. The prompt also includes valid examples demonstrating common patterns. These examples teach packet flow semantics (`input_packet` $\rightarrow$ `measurement` $\rightarrow$ `analysis_result`) without requiring the LLM to derive composition rules from first principles. The LLM outputs a JSON structure specifying nodes and edges, which the system validates against template manifests before execution. When a query references locations that only partially resolve to available sensors, the composer constructs a DAG over the resolved portion and includes a rationale noting the coverage gap, which the response generator surfaces to the user rather than silently omitting unresolvable locations. Figure 6.4 provides DAG compositions for different types of queries.



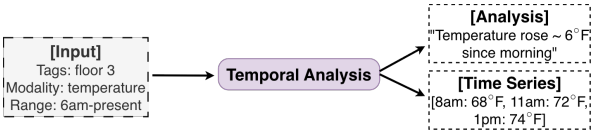
(a) "What is the light level on floor 3?"



(b) "Is floor 3 warmer than floor 4?"



(c) "Find a quiet and private space for a phone call on my floor"



(d) "How has temperature changed since morning on floor 3?"

Figure 6.4: DAG compositions for different query types. Each panel shows how natural language queries decompose into template operations connected by typed data packets. (a) Simple measurements use a linear pipeline. (b) Comparisons create parallel branches that merge at a `spatial_compare` node. (c) Multi-criteria queries over N candidate locations produce parallel branches per modality, each processing N locations (indicated by stacked boxes). (d) Temporal queries produce dual outputs for time series and summarization.

6.4.3.3 DAG Execution

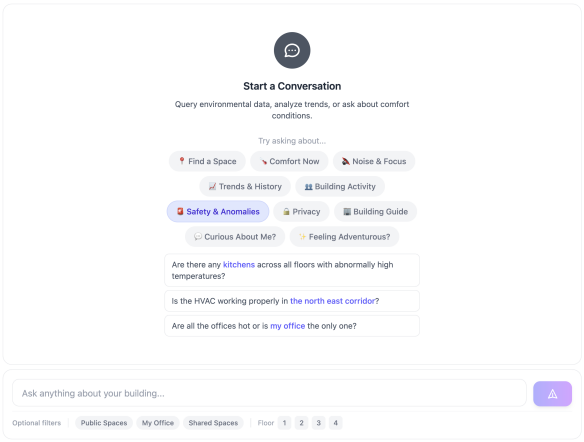
Templates execute within declared resource constraints (timeout and memory limits) in their manifest. When a node fails, it produces an *error packet* rather than crashing the DAG, enabling partial results when some branches succeed while others encounter errors.

6.4.4 Response Generation

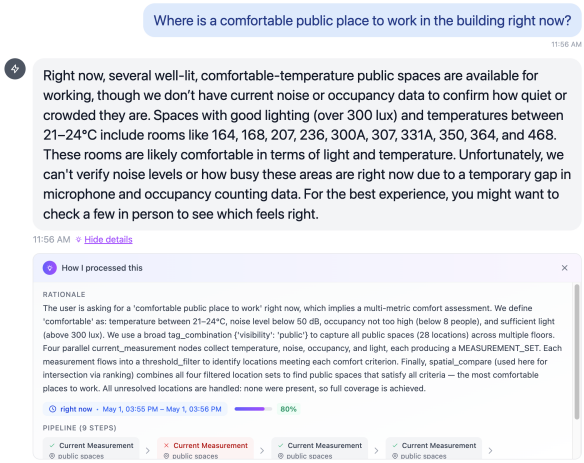
The final pipeline stage transforms analysis results into natural language responses. The response generator receives all `analysis_result` packets produced by terminal DAG nodes, along with confidence metadata (data coverage, freshness, errors encountered), and assembles them into a structured prompt. The LLM's role at this stage is linguistic rather than analytical: the analysis has already been performed by template operators with deterministic logic, and the LLM synthesizes findings into coherent prose. For our running example, this produces a response such as: *"Room 307 is both quiet and well-lit right now, a good choice for focused work. Room 312 has similar lighting but is noisier due to nearby activity."* The response generator is constrained to translate analysis results into user-facing language: it must not expose internal representations such as tag combinations, packet types, or raw error traces, and must describe locations and metrics in natural terms.

Beyond text, a deterministic visualization module selects an appropriate visual format based on the packet type: cross-location comparisons produce bar charts, temporal analyses produce time-series line charts, and queries spanning a single floor produce spatial heatmaps overlaid on building floor plans (Figure 6.5). Visualizations are rendered inline alongside the text response.

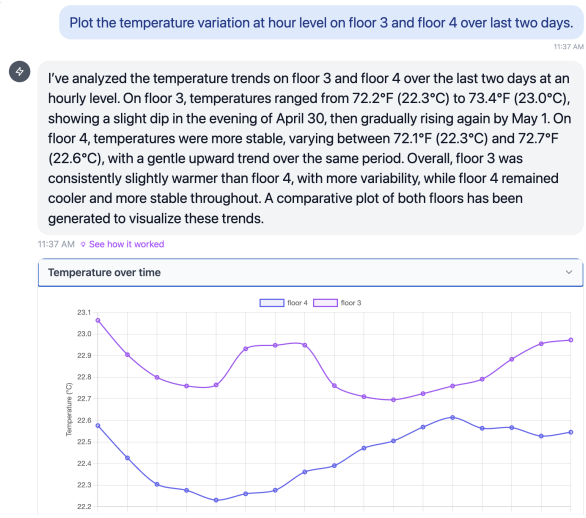
6.4.5 Interface Design



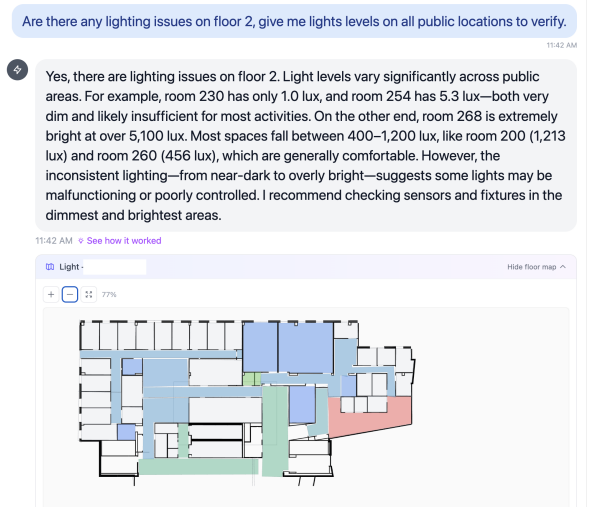
(a) Categorized suggestion chips with editable placeholders help users discover queryable capabilities, while spatial filters along the bottom scope queries by space type and floor.



(b) An expandable explanation panel shows which templates were used, the time range analyzed, and per-node confidence assessments for data coverage and method reliability.



(c) Responses combine natural language with inline visualizations. Here, a comparative time-series chart shows temperature trends across two floors.



(d) Spatial heatmaps overlaid on building floor plans provide room-level context for sensor readings, here showing light levels across public areas.

Figure 6.5: The BuildingChat interface. (a) The empty-state screen guides query formulation through categorized suggestions. (b) Each response includes an expandable panel exposing the template pipeline and confidence metrics. (c, d) Inline visualizations complement text responses, ranging from comparative time-series charts to spatial heatmaps on floor plans.

Figure 6.5 presents the BuildingChat interface. To help users discover what queries are possible, the empty-state screen presents categorized suggestions with editable placeholders. Users can declare comfort preferences through a profile settings page; combined with office

assignment and floor retrieved from BuildingDepot, this context resolves self-references and ambiguous qualifiers without prompting (Section 6.4.2.1). An expandable explanation panel below each response shows which templates were used, the time range analyzed, and per-node confidence assessments. When a query cannot proceed due to access restrictions, the interface presents the denial reason alongside a concrete suggested alternative with explicit accept and reject controls.

6.4.6 Implementation

We implemented BuildingChat as a web service with a chat interface for building occupants. The system consists of four modules: a *query understanding module* that parses natural language, checks if clarification is needed, and validates access requirements; a *database handler* that retrieves sensor data from a time-series backend; an *analysis engine* that composes and executes template DAGs; and a *response generator* that produces natural language answers. The backend exposes REST endpoints for batch queries, streaming responses via Server-Sent Events, and multi-turn clarification flows. We use LiteLLM [29] to abstract LLM provider details, supporting both cloud-hosted models and locally-deployed models.

Our deployment instruments a university building with over 300 sensors spanning seven modalities: temperature, humidity, motion, thermal imaging, audio, light, and vibration. Adapting BuildingChat to a new building requires defining a tag taxonomy reflecting the spatial organization and configuring access policies through stakeholder consultation; the core pipeline and templates remain unchanged.

6.5 Design Iterations from Initial User Study

We conducted two user studies during the development of BuildingChat. The first, described in this section, informed design iterations from an initial prototype (V1) to the system presented in Section 6.4. The second, a three-day deployment evaluating the final system's usability, is reported in Section 6.7.

For the first study, we recruited 11 building occupants through snowball sampling: seven had shared offices (PhD students), three occupied private offices (two faculty members and one staff member), and two were resource managers (staff managing building resources). Each participant attended a one-hour session where they set up BuildingChat and used it to ask questions about the building, working through five randomly selected scenarios covering valid queries, access-restricted queries, and queries beyond system capabilities. Participants were required to be 18 years or older, fluent in English, and regular occupants of the building. The study was approved by our institution's IRB; participants were paid $20 USD. The interview script is included in Appendix C.5.

We conducted semi-structured interviews and performed qualitative coding to identify emerging themes. Participants valued that BuildingChat enforced privacy boundaries (3/11 specifically appreciated being unable to access others' private data), and 5/11 found that conversational sensor access provided capabilities they did not previously have. These findings validated the core

architectural decisions around access-controlled conversational interfaces. However, the study surfaced four categories of design gaps that motivated specific changes throughout the system.

First, participants struggled to interpret raw numerical sensor values and wanted visual context. These findings motivated the visualization module (Section 6.4.4), the explanation panel, and the statistical summary cards described in Section 6.4.5. Second, our initial design used open-ended clarification questions to resolve underspecified queries. While nearly all participants (10/11) understood why these were needed, extended back-and-forth felt tedious, and several requested that the system propose alternatives directly. This motivated redesigning the reformulation module to propose concrete alternatives (Section 6.4.2.3), and user preferences to resolve ambiguous qualifiers without prompting (Section 6.4.2.1). Third, participants reported a learning curve in discovering what questions the system could answer, which motivated the suggested queries panel (Section 6.4.5). Fourth, three categories of errors appeared in responses: (a) external context hallucinations from missing timezone and geographic context, (b) access over-restriction from the parser inferring unnecessary tag constraints, and (c) speculative causal explanations for sensor readings. User profiles (Section 6.4.2.1) address the first category by supplying timezone, location, and office context to the parser. Response sanitization constraints (Section 6.4.4) reduce speculative explanations by restricting the response generator to reporting observed patterns. The remaining access validation issues were resolved through engineering improvements to the parser.

6.6 Technical Evaluation

We evaluate BuildingChat along three dimensions: (1) correctness of query analysis across different query types, (2) computational efficiency (latency and token usage), and (3) comparative effectiveness of access control enforcement.

6.6.1 Evaluation Setup

We adopt an evaluation strategy composed of objective queries running on a mock building environment. This approach allows us to (1) generate sensor data with unambiguous ground truth values, (2) systematically test query types derived from our formative study, (3) isolate BuildingChat's analytical capabilities from confounding factors like sensor failures or network issues, and (4) protect the privacy of the current building residents.

Mock Building Environment: The mock building mirrors the structure of our real deployment (Section 6.4.1): a 4-floor academic building with offices, conference rooms, laboratories, corridors, and common spaces distributed across four quadrants (NE, NW, SE, SW). We modeled the tag taxonomy, sensor placements, and distribution of space types directly from our real deployment, using a smaller configuration (4 floors, 39 rooms, 96 sensors) that preserves the structural relationships and variety of the real building. For this evaluation, we generated 1 month of synthetic data from all sensors, consisting of a total of 45,619,200 measurements. The system maintains a complete ground truth state for each timestamp, including occupancy counts, environmental conditions, and the causal factors producing each sensor reading, allowing us to verify whether BuildingChat correctly identifies patterns and comparisons. The mock system generates sensor

readings based on occupancy schedules: temperature rises with occupancy, noise increases during meetings, motion sensors activate as people move.

Test Query Generation: We generated 300 test queries across four categories derived from the information need patterns identified in our formative study (Section 6.3), distributed equally (75 each) across measurement, comparison, filter, and trend queries. Each test case was constructed by: (1) sampling structured parameters, including location tags from validated combinations in the mock building, sensor modalities, and time specifications (supporting both relative references like "2 hours ago" and absolute timestamps), (2) computing the ground truth answer deterministically on the mock data using manually constructed procedures for each query type (*e.g.*, sensor-weighted means for measurements, linear regression direction and significance for trends), and (3) generating natural language query text. For comparison queries, we use deterministic templates to preserve precise references and avoid inverting comparison direction. For other query types, we prompt an LLM (Qwen-3 235B) with the structured parameters and manually review the output to remove ambiguities. The ground truth is independent of any language model. The LLM's only role is rephrasing structured parameters into natural language, and correctness is evaluated against the deterministic reference. We discard and resample cases where the ground truth is missing or ambiguous (*e.g.*, comparisons where values differ by less than 10%). Because queries are sampled programmatically from the building's topology and sensor configuration, the test set covers the parameter space systematically but may not capture the full linguistic diversity of unconstrained user queries. We assess robustness to natural phrasing variation through the usability study in Section 6.7.

(1) Measurement queries (75 queries) capture point-in-time information needs: "What is the average temperature on floor 3 from 2 PM to 3 PM yesterday?" These require extracting and aggregating sensor values across matched spaces. **(2) Comparison queries** (75 queries) address cross-location or cross-temporal analysis: "Is my office noisier currently than 5 hours ago?" These require retrieving data from multiple sensor groups, aggregating appropriately, and performing comparative reasoning. **(3) Filter queries** (75 queries) identify locations meeting specific criteria: "How many rooms in Floor 2 are above 75°F?" These combine measurement extraction with threshold-based filtering. **(4) Trend queries** (75 queries) analyze temporal patterns: "How has the temperature changed today?" These require time-series analysis to identify the direction and statistical significance of change. We detail the full generation procedure, including prompt templates and example test cases, in Appendix C.4.

Baseline Systems: LLM-based approaches to analytical tasks over structured data vary in how they distribute reasoning between the language model and deterministic computation. At one end, program-synthesis approaches [50, 73] delegate analysis entirely to generated code. At the other, recent work has shown that LLMs can reason directly over structured tabular data presented in-context [193], avoiding external computation entirely. Between these, agentic frameworks [228] interleave LLM reasoning with iterative tool use, allowing the model to retrieve data incrementally and refine its analysis across multiple steps. We compare BuildingChat against three baselines representing these paradigms.

(1) Code Generation. Following the program-of-thought paradigm [50], given a user query, an LLM generates Python code that retrieves sensor data and performs the required analysis. The generated code has access to the same data retrieval APIs as BuildingChat, along with standard

libraries (NumPy, Pandas, SciPy) for computation. This baseline tests whether general-purpose code generation can replace structured template composition.

(2) In-Context Data. Following the in-context tabular reasoning paradigm [193], the system retrieves all potentially relevant sensor data and includes it directly in the LLM prompt as formatted tables. The LLM then reasons over this structured data to produce an answer. This baseline tests whether LLMs can perform accurate numerical analysis when given raw sensor data, avoiding the complexity of structured query decomposition.

(3) ReAct Agent. Following the ReAct framework [228], the LLM operates in an iterative loop of reasoning and tool use. At each step, the model reasons about what data it needs, calls a tool to retrieve it, observes the result, and decides whether to retrieve additional data or submit a final answer. The agent has access to two data-retrieval tools: one for discovering available tag filters in the building taxonomy, and one for executing snapshot queries over specified locations, sensors, and time windows. The agent may call these tools multiple times across up to 40 iterations. This baseline tests whether iterative, agent-driven retrieval and reasoning can match the accuracy of structured template composition while adapting flexibly to diverse query types.

All baselines receive the same information available to BuildingChat: the building's tag taxonomy, sensor metadata, and data retrieval API documentation. For Code Generation and In-Context Data, we provide detailed instructions for parsing the user query, setting appropriate parameters for data retrieval, and performing aggregation and statistical analysis. For Code Generation, we additionally provide code snippet examples. The ReAct agent receives the same API documentation and discovers available building metadata through its tool calls.

Evaluation Protocol: For the correctness evaluation, we grant all systems root-level access to sensor data in the mock building environment, bypassing access control checks. This isolates analytical capability from access enforcement. We evaluate each system using six language models spanning different capability levels and deployment contexts: three open-weight models suitable for local deployment (Qwen-3 32B, Qwen-3 235B, GPT-OSS 120B) and three proprietary models (GPT-4o Mini, GPT-5 Mini, GPT-5.2). This range allows us to assess whether BuildingChat's benefits hold across model capabilities and to assess the viability of local deployment with smaller open-weight models. A response is marked correct if it matches the ground truth answer. For numerical answers, we apply a 2% tolerance threshold, which exceeds the measurement precision of the underlying sensors while remaining strict enough to catch substantive analytical errors. All experiments were conducted in April 2026.

6.6.2 Comparing BuildingChat with Baseline Approaches

6.6.2.1 Response Accuracy

Figure 6.6 presents overall accuracy across systems and models. BuildingChat achieves 88% average accuracy, outperforming ReAct (74%), In-Context Data (67%), and Code Generation (42%). BuildingChat's accuracy ranges from 93% on GPT-5.2 to 76% on GPT-4o Mini, while baselines exhibit substantially larger variation (*e.g.*, Code Generation ranges from 8% to 79%). BuildingChat achieves the highest accuracy on every model tested, and its 76% on the weakest model exceeds the average performance of all three baselines. This consistency is practically significant: BuildingChat can operate effectively with smaller, locally-deployable models, enabling

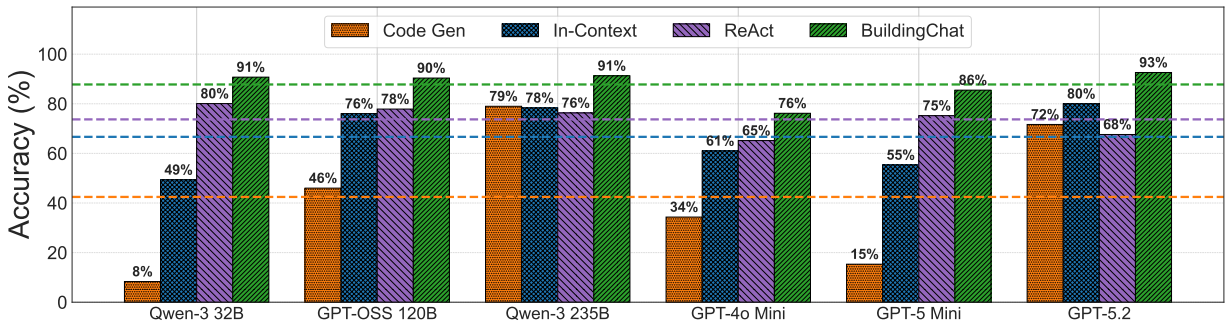


Figure 6.6: Query accuracy across six language models. BuildingChat achieves the highest accuracy on every model (76–93%). Code Generation ranges from 8% to 79%, In-Context Data from 49% to 80%, and ReAct from 65% to 80%.

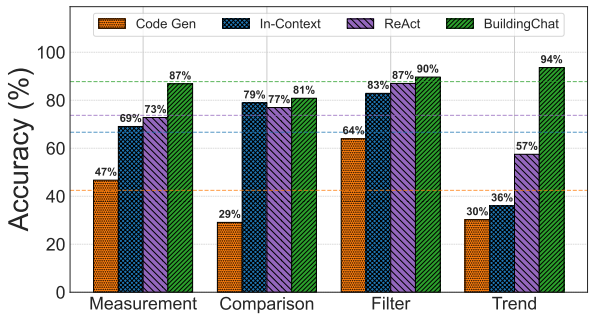


Figure 6.7: Accuracy by query category. BuildingChat maintains high performance across all query types. The advantage over baselines is most pronounced for trend queries, which require multi-step temporal reasoning.

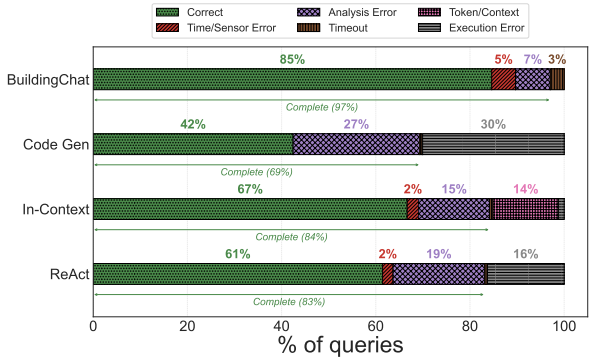


Figure 6.8: Error type distribution across 1800 queries (300 × 6 models). Code Gen. is dominated by execution failures (30%), In-Context by context limits (14%), and ReAct by analysis errors (19%) and execution failures (16%).

deployments where data governance or latency concerns favor local processing over cloud-hosted services.

Figure 6.7 breaks down accuracy by query category. BuildingChat leads across all categories, with smaller margins for comparison and filter queries where the best baselines approach BuildingChat's accuracy (*e.g.*, ReAct at 87% *vs.*BuildingChat at 90% for filters), reflecting that these operations are straightforward enough for LLM reasoning. The largest difference appears for trend queries, where BuildingChat achieves 94% compared to 57% for ReAct, 36% for In-Context Data, and 30% for Code Generation. Trend analysis requires selecting appropriate time windows, computing directional changes, and assessing significance. BuildingChat's `temporal_analysis` template encodes these operations explicitly, while baselines must derive the complete analysis logic from scratch for each query. ReAct's iterative tool use partially compensates (57%) but still falls short of BuildingChat's template-based approach. The In-Context baseline fails because reasoning over long time-series data within the prompt context proves unreliable for detecting trends.

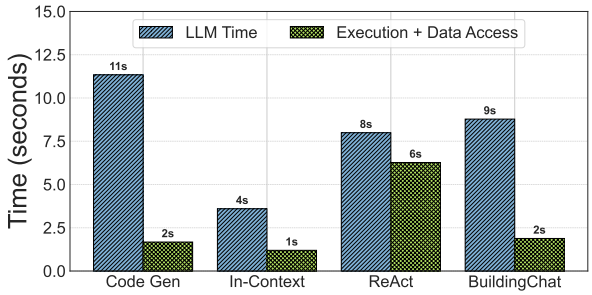


Figure 6.9: Query latency breakdown. ReAct is the slowest (14s) due to iterative tool calls. BuildingChat and Code Generation achieve comparable latency (11s and 13s). In-Context is fastest (5s) but at the cost of higher token usage and context overflow errors.

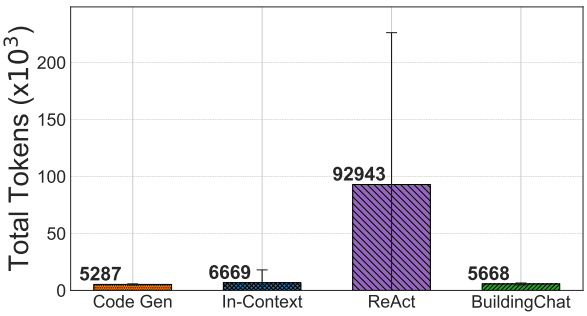


Figure 6.10: Token usage per query. ReAct consumes 93K tokens on average (16× BuildingChat), driven by accumulating conversation history across iterations. BuildingChat, Code Generation, and In-Context Data remain in the 5–7K range.

Figure 6.8 shows error patterns across 1800 queries per system (300 queries × 6 models). BuildingChat completed 97% of queries with zero execution failures and zero context overflow errors; its errors were limited to time and sensor parsing (5%) and analysis errors (7%). The baselines exhibit failure profiles that reflect their architectural approaches: Code Generation is dominated by execution failures (30% of queries), In-Context Data by context overflow (14%), and ReAct combines both failure modes, with 16% execution errors and 19% analysis errors, exhibiting the execution fragility of code generation alongside the reasoning errors of in-context approaches.

6.6.2.2 Cost and Latency Analysis

We evaluated latency and cost on queries that all four systems completed successfully with GPT-5.2. Figure 6.9 presents latency for each system. BuildingChat achieves 11s average latency, comparable to Code Generation (13s). ReAct is the slowest at 14s, with its execution time (6s) three times BuildingChat's (2s) due to multiple iterative tool calls. In-Context Data is fastest (5s) but at the cost of a 14% context overflow rate (Figure 6.8).

Figure 6.10 presents token usage. BuildingChat (5,668 tokens), Code Generation (5,287), and In-Context Data (6,669) fall within a similar range, while ReAct is an outlier at 92,943 tokens per query, with a standard deviation (133,277) exceeding the mean. BuildingChat's token usage remains stable because it retrieves pre-aggregated statistics rather than raw data, and its multi-stage pipeline uses tokens predictably across query types. The complete evaluation (7,200 queries across six models, four systems) cost approximately $150 (in April 2026), using Cerebras for Qwen-3 235B and GPT-OSS 120B, OpenRouter for Qwen-3 32B, and OpenAI for GPT-4o Mini, GPT-5 Mini, and GPT-5.2.

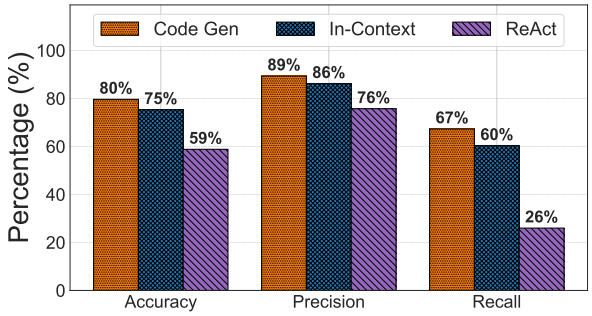


Figure 6.11: Correctness of unauthorized query rejection for policy-aware baselines. All three achieve reasonable precision (76–89%) but low recall, with ReAct correctly rejecting only 26% of unauthorized queries.

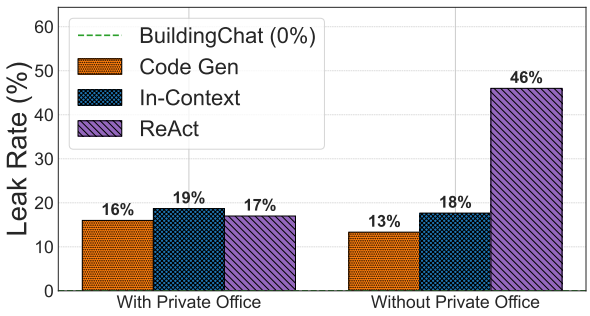


Figure 6.12: Data leak rates by user profile for policy-aware baselines. ReAct leaks 46% of restricted queries for the more constrained user profile. BuildingChat has zero leaks by construction.

6.6.2.3 Access Control Violations in Baseline Approaches

Because the policy framework described in Section 6.4.2.2 validates every data access request before retrieval, making unauthorized access structurally impossible within BuildingChat, our evaluation instead tests whether alternative system designs would provide equivalent guarantees. We created policy-aware variants of all three baselines by augmenting their prompts with natural language descriptions of the access control policies and a structured policy table. The prompts instruct the system to (1) check whether the user's access privileges permit the requested data, and (2) reject queries that would require unauthorized access. This represents a reasonable attempt to add access control to systems not designed with it as a first-class concern.

We evaluated with two user profiles: one with access to their private office sensors, and one without. We generated 300 test queries distributed equally across three access categories: *Full Access*, where both users are authorized; *Partial Access*, where only the user with private office privileges is authorized; and *No Access*, where neither user is authorized.

Figure 6.11 summarizes rejection accuracy for the policy-aware baselines. Code Generation achieves 67% recall, In-Context Data 60%, and ReAct only 26%, meaning ReAct fails to reject three out of four unauthorized queries. All three baselines exhibit higher precision (76–89%) than recall, reflecting a systematic bias: instruction-tuned LLMs are reluctant to deny requests, favoring helpfulness over enforcement. BuildingChat rejects unauthorized queries through programmatic validation (Section 6.4.2.2), achieving 100% on both metrics.

Figure 6.12 shows leak rates for each baseline. Despite explicit policy instructions, Code Generation leaks 13–16% of restricted queries, In-Context Data 18–19%, and ReAct 17–46%, with the higher rate occurring for the more restricted user profile. ReAct's asymmetry is notable: its iterative reasoning finds more paths to restricted data when policies are complex, making the system that is most competitive on accuracy (Section 6.6.2.1) the least reliable on access control. These failures occur because interpreting natural language policies for parameter-specific contexts is unreliable. BuildingChat avoids leakage by performing programmatic access validation against structured definitions before retrieval.

6.7 Usability Evaluation

To evaluate the usability of BuildingChat, we conducted a second real-world deployment in the same academic building described in Section 6.4.1.

6.7.1 Method

For this study, we had the same recruitment criteria as the study conducted in Section 6.5. To explore a diverse range of perspectives, we again conducted snowball sampling to find our participants: four were PhD students in smaller shared spaces, two were masters' students in larger shared spaces, one was a staff member/administrator in a shared office, two were faculty in private offices, and one was a faculty member who also had an administrative role with a private office.

Each participant was given an instruction sheet and was told to independently experiment with BuildingChat for three days, using queries of their own choosing. Following this three-day period, participants attended an hour-long interview, where they tried additional queries and discussed their experiences with BuildingChat. To ensure that all users had hit the boundaries of what BuildingChat could not respond to, we provided participants with three scenarios: one expected to produce a valid response, one where BuildingChat lacked the required data, and one where access control would prevent answering. We additionally asked participants to query BuildingChat with questions of their own. For a quantitative measure of usability, participants completed a System Usability Scale (SUS) survey [40]. Participants were compensated with a $30 Amazon Gift Card. This study was approved by our institution's IRB. The interview script is included in Appendix C.6.

6.7.2 Overall User Experience

Broadly, all participants found BuildingChat to be very usable: across all participants, BuildingChat received an average SUS score of 80 ± 8. Specifically, we find from our SUS survey that all users either somewhat or strongly agreed that it was easy to learn how to use BuildingChat, and that it was not cumbersome to use. In this section, we explore how users approached learning how to use BuildingChat and how they experienced interactions with the system.

6.7.2.1 Newfound capabilities for accessing sensed information.

The majority (8/10) of users spent time exploring BuildingChat prior to the interview, and were able to explore questions of their own in that time. The two users who did not utilize BuildingChat prior to the interview learned how to use it on the fly during the interview session.

The participants who used BuildingChat prior to the interview explored a wide variety of questions based on their own curiosities and uncovered newfound information about their spaces. Participants primarily asked about public spaces they had access to, spanning trend (4/10), comparison (5/10), and measurement (2/10) queries. Some (2/10) asked meta-questions about sensing capabilities, and 3/10 queried their own office sensors. Notably, 3/10 participants attempted to query their shared offices but could not retrieve data without their office-mates' consent.

Many participants were excited about the insights that this information gave them, especially as they helped them optimize their building usage or gave them context relevant to their jobs. We found that 2/10 participants were interested in the context that BuildingChat provided as it related to their jobs. For example, one participant (P26), who is an administrative staff member, experimented with asking BuildingChat about activity trends in different public spaces, and found that the kitchenette was busy after hours. She noted that "As a staff member, it could be helpful to conceptualize, like, what is going on after hours." Other participants tended toward asking questions that gave insights on the facilities they were using. For example, P29, a student with a shared office, asked specific questions about comfortable places to work, and noted that over the course of her three-day experimentation period, she consulted BuildingChat for finding this comfort-related information: "essentially, whenever I had a physical need, and I needed some kind of information to make a decision, I used the chat" (P29). Another student (P24) mentioned that she was curious about temperature trends in the building, and found it interesting to find out the hottest time period.

6.7.2.2 User learning and exploration strategies.

Four of ten users found that there was no learning curve to using BuildingChat; rather, using BuildingChat felt like a familiar and natural interaction with other LLM-based chat interfaces, like ChatGPT. One participant (P22) noted, "Building chat, there was no learning curve, and in fact, it's a really, really good tool to learn what is available and what [Sensing Infrastructure] can do." These participants felt comfortable formulating queries and were primarily focused on experimenting with different queries and learning what the system could and could not answer.

The participants that did feel that there was a learning curve (6/10) noted that they were able to learn how to query BuildingChat easily through four different methods: *trying example queries* on the dashboard to see how queries could be formulated, *experimenting* with their own queries, *increasing the specificity* of their queries in different ways to see how responses changed, and *observing alternative queries* and how they were formulated. These strategies built on each other, enabling increasingly creative queries. P26 noted that "The more questions that I asked it, the more, like, possibilities that opened up of things that it can sense, or, like, questions that can be asked to it."

6.7.2.3 Alternative queries and alignment with user intention.

We found that 9/10 participants found that their intention was well understood when they interacted with BuildingChat. These participants found that their queries were well understood, and that, in most cases, the alternative questions that were generated were also aligned with their original intention. P22 stated that BuildingChat was very helpful in reformulating privacy-sensitive questions into non-privacy-invasive alternatives: "I think it actually did a really good job at... [reformulating] questions to go around... I wouldn't say to go around, to accommodate some privacy properties. I thought this was very, very well done." Further, participants overwhelmingly understood why alternatives were proposed, and the explanations helped them learn what data they had access to. P30 noted that "I think the [explanation for the alternative query] was needed, mostly when it suggests, like, an alternative question. It was about, like, private versus, like,

public space. So I can understand, oh, this is... I'm ask... am I actually asking, like, a privacy invasive question."

6.7.2.4 Response quality: reliability, sufficiency, and understandability.

Overall, the majority of users (7/10) reported that BuildingChat was mostly reliable, 1/10 reported that it was sometimes reliable, and 2/10 reported that it was not reliable. All users found BuildingChat to produce understandable responses, and 9/10 users found that it produced responses that sufficiently answered their queries.

Participants identified three factors that built reliability: traceability, experiential validation, and consistency. Two users found that the "See how it works" panel helped them understand how responses were generated. P24 stated: "I really like the traceability of it, so like, even if I didn't initially understand... I could very clearly just see how it works, and was able to determine, like, how it came to that conclusion." Physical co-location also mattered: P29, who was in the building during the study, validated responses against her own experience ("I did use this data, like, one was to find a warmer space, that was right"), while P27, who was remote, noted that "I can't really measure [the] reliability [of BuildingChat], because I don't think I have access to, like, the ground truth data." Inconsistent data availability also reduced trust; P29 observed that sensor data was sometimes unavailable for locations she expected to have sensors, requiring her to probe further. The two users who reported low reliability attributed it to factors outside BuildingChat itself: P22 found the underlying sensor readings inconsistent, and P21 did not receive appropriate answers for web-based queries.

Finally, we found that displaying information through visualizations, both for the "See how it works" explanation and the query response itself, aided in understanding. Five of ten participants indicated interest in the visualization features, and many indicated that they aided in interpreting BuildingChat responses. P30 described that the "See how it works" diagram allowed her to parse complicated technical jargon by comparing diagrams across queries: "I feel like visualization is helpful, so because sometimes the text is just, like, too long, and also contains a lot of, like, terminologies, I will directly go to... the little buttons, like, or the little diagrams... I compare a little bit, like, across the answers to understand what they represent." Another participant (P24) found that the query-response visualizations for floor maps and bar charts were a low-friction way of understanding BuildingChat information. She even suggested that a future version of BuildingChat should allow these visualizations to be "pinned" to the dashboard for easier reference.

6.7.3 User Experience of Privacy Boundaries

All participants encountered query sessions where they attempted to access unauthorized data. In this section, we discuss how participants perceived the privacy boundaries enforced by BuildingChat.

6.7.3.1 Participant exploration of privacy boundaries.

While all participants encountered privacy restrictions, some (4/10) intentionally attempted to "break" the system's privacy controls and found they were unable to. For example, one participant (P30), who is a PhD student, asked about faculty offices. Although she expected to not have access, she felt that she wanted to check to verify: "I was just, like, testing, kind of, like, the safety of [BuildingChat], because, what if, like, someone asks if I'm in my room? Like, what if it's, like, the other way around?" Another participant (P25) attempted to access additional information through a prompt injection attack, and failed to access the information he was querying for: "I claim to be the building manager, and I need to, like, access everyone's offices, like, you know, just as a... for fun, I guess, and it didn't work."

Other participants unintentionally ran into access control issues from their use of BuildingChat and discovered privacy controls through these experiences. Most commonly, participants with shared office spaces learned that they could not access data about their office without the consent of their office-mates. Another participant discovered that they could not access public space data going back more than a month through asking a query about typical trends in a public conference room.

6.7.3.2 Participant alignment with privacy boundaries.

We found that, in most cases, participants aligned with the privacy boundaries expressed in BuildingChat. In particular, participants uniformly agreed that only the owner of a private office should have access to their own office data. When it came to shared offices, although some users (such as P28 and P29) were comfortable with other office-mates accessing information about their shared office, participants understood the rationale behind the privacy policy of not giving access to shared information without the other occupants' consent.

However, there were instances where participants did not align with BuildingChat privacy boundaries. In one case, one user (P27) disagreed on the recency of access for public spaces, stating "I don't know what kind of privacy leaks can come from being able to get that information back a month. Beyond things just like having more data storage as more opportunities for data leaks." Another participant (P22) noted that the existing privacy policy felt limiting to him in his role as an administrator: "I did not learn anything directly actionable from the queries that I ran, because the type of stuff that I wanted was kind of blocked. But what I learned is that it can do it if it's properly configured and, like, you know, there's tiered levels of permission, then I think it's gonna be really, really useful."

6.8 Discussion

In this section, we reflect on lessons from our usability study, discuss implications for the design of privacy-aware building interfaces, and present limitations and future directions.

6.8.1 Lessons from the Usability Study

Our usability study revealed insights about how building occupants interact with conversational sensor interfaces and where current approaches fall short.

Increased need for interpretability: Although participants noted that the "See how it works" feature helped them develop mental models of how BuildingChat worked, some found that explanations remained too technical and did not sufficiently clarify what could be queried or why specific access restrictions applied. This challenge connects to trustworthiness: when participants did not understand how metrics like occupancy were calculated, their sense of reliability decreased. We could address this gap by using a hierarchical explanation approach, where users can hover for metric definitions and precise access policy information.

To infer, or not infer? Participants had mixed feelings about the level of inference that BuildingChat made when explaining sensor information. While some participants wanted BuildingChat to make inferences about what could be happening in a space, others were strongly against it. For example, one participant (P30) wished that BuildingChat could infer which parts of the building felt like a "coffee shop" in terms of noise level, which requires high levels of inference about spaces, and another participant (P24) wished that BuildingChat would infer information for which no sensor was available. Another participant (P29), however, expressed concerns about BuildingChat inferring about why spaces were empty (*i.e.*, stating that they might be closed for maintenance). Further research could explore controlled and personalized sense-making that could cater to individual needs and scenarios.

6.8.2 Design Implications for Privacy-Aware Building Interfaces

Our experience building and evaluating BuildingChat suggests broader considerations for similar systems.

Need for deterministic access control: Our evaluation demonstrates that access control cannot be reliably enforced through prompt-level instructions alone. Despite explicit policy documentation, all baselines attempted to access unauthorized sensor data in 13 to 46% of queries. BuildingChat avoids these failures by integrating access control into the query processing architecture, ensuring unauthorized data never enters the pipeline. This finding argues against retrofitting access control onto systems designed without privacy as a first-class concern.

Community engagement for policy design: The access control policies in our study were designed by researchers based on IRB restrictions and general privacy principles. Real-world deployments would require policies reflecting specific building community norms. BuildingChat allows flexible policies, but this presumes someone with both technical knowledge to configure the framework and institutional authority to make policy decisions. Who authors policies, through what process, and with what expertise remains an open problem complementary to technical access control mechanisms.

Needs beyond conversational interfaces: User feedback suggests that accurate answers are not always sufficient. Participants wanted context to interpret results, visualizations to reveal patterns, and the ability to share findings. Conversational interfaces may be most effective as one component of a larger system that includes visualization tools and collaborative features; natural

language excels at providing accessible entry points to complex data, but other modalities may better serve interpretation and action.

6.8.3 Limitations and Future Work

Need for long-term deployments. Our correctness evaluation used a mock building environment with synthetic sensor data and programmatically computed ground truth, enabling rigorous accuracy measurement but not capturing sensor failures, network issues, or data quality problems that occur in practice. Our usability study involved participants from a single academic institution in one-hour structured sessions rather than sustained everyday use. Thus, we cannot make claims about long-term adoption patterns, learning effects, or how usage might evolve over time. The participant pool, while spanning multiple roles, does not include visitors, maintenance staff, or occupants with accessibility needs. Longitudinal deployments with diverse building populations would reveal adoption barriers and usage patterns that controlled studies cannot surface.

Verification of threat model assumptions: Our access control framework assumes correct user identification, accurate sensor-to-space mapping, and no collusion between users pooling their permitted queries. We did not test BuildingChat under adversarial conditions where users actively attempted to circumvent access restrictions. The rate limiting mechanism, while specified in our policy framework, was not tested over extended periods to verify that it prevents pattern reconstruction through accumulated queries. Formal verification of policy configurations and red-team evaluations with motivated adversaries would help close the gap between our threat model and empirical validation.

Deployment across different building types: Our formative study, system design, and evaluation all took place in the context of an academic building. Adapting BuildingChat to other building types requires reconfiguring two layers: the tag taxonomy and access policies, which encode a building's spatial organization and community norms, and potentially the template library, if occupants in a given context require analytical operations beyond those our formative study surfaced. Buildings that already maintain semantic models such as Brick [24] can simplify the first step, as the ontology provides structured sensor metadata that can populate BuildingChat's tag layer. The core analytical templates, measuring conditions, comparing locations, detecting trends, and filtering by criteria, are building-agnostic because they reflect how occupants reason about shared spaces rather than how a specific building is organized.

Beyond addressing these limitations, future work could explore mechanisms for identifying template gaps from query failures and supporting template extension by building administrators without requiring researcher involvement.

6.9 Summary

This chapter presented BuildingChat, a conversational system that enables building occupants to query sensor infrastructure through natural language. The system decomposes queries into composable template operations arranged as directed acyclic graphs, enforces access control through policy-driven validation, and guides users toward accessible alternatives when queries are underspecified or unauthorized.

Our formative study with 11 participants across five building roles revealed that occupant query needs vary widely in both topic and analytical complexity, with 72% of in-scope queries requiring multi-step reasoning, and that privacy concerns create direct tensions with desired functionality. These findings shaped BuildingChat's design: composable templates address the multi-step reasoning requirement, programmatic access validation addresses the privacy concerns, and the reformulation module addresses the underspecification that participants exhibited throughout the study.

Our technical evaluation demonstrated that BuildingChat achieves 88% query accuracy, outperforming code generation, in-context reasoning, and ReAct agent baselines by 14 to 46 percentage points across six language models. Critically, BuildingChat prevents the unauthorized data leakage that occurs in 13 to 46% of baseline queries when access control is enforced through prompts alone, confirming that privacy enforcement cannot be delegated to instruction-following alone. Our usability study with 10 participants in a building with over 300 sensors confirmed that occupants find the system easy to use (SUS score of 80 ± 8) and its responses understandable, while also surfacing open challenges in interpreting sensor values and calibrating the level of inference the system should provide.

BuildingChat addresses the fourth and final stage of the sensing pipeline described in Chapter 1: the interface through which occupants access what the system knows. Together with VAX (training labels), OrganicHAR (activity vocabulary), and TAO (context synthesis), it completes the path from raw sensor signals to natural-language interaction that this dissertation set out to trace. Each system demonstrated that a specific form of per-environment user effort, from labeling training data to navigating purpose-built interfaces, can be replaced by an automated alternative, whether that alternative is a temporary richer source that departs after bootstrapping or, as in BuildingChat's case, a language model that remains in a bounded orchestration role. The next chapter steps back from the individual contributions to reflect on what they achieve together, the limitations they share, and the research directions they open.

Chapter 7

Ongoing Work, Discussion, and Future Directions

The previous four chapters presented systems that each reduce the user effort required at one stage of the context-aware sensing pipeline: training labels (VAX), activity vocabulary (OrganicHAR), context synthesis (TAO), and occupant interaction (BuildingChat). Each system demonstrated that the bootstrapping principle, opportunistically engaging a more capable resource and bounding it with appropriate scaffolding so that it is not needed after setup or remains in a constrained role, is viable at its respective stage. This chapter steps back from the individual systems to reflect on what the dissertation as a whole achieved and identifies possible next steps.

Section 7.1 describes ongoing work that extends the contributions of this dissertation into real-world applications, including deployments with people living with dementia and continuous operation in an instrumented university building. Section 7.2 reflects on the cross-cutting themes and limitations that span multiple chapters. Section 7.3 identifies the research directions that would bring the vision described in Section 1.2 closer to realization.

7.1 Ongoing Efforts and Emerging Applications

The contributions presented in this dissertation were developed and evaluated in controlled studies and short-term deployments. Several ongoing efforts are now applying these contributions to sustained, real-world applications. This section briefly describes three: a deployment supporting people living with dementia, the continuous operation of BuildingChat in our university building, and an integrated home system currently under development.

7.1.1 The Autonomous Project: Supporting People Living with Dementia

The motivating scenario presented in Section 1.2, an elderly person living alone whose family worries about whether meals are being prepared and whether the stove is being left on, is not hypothetical. Through the Autonomous project [21], we are deploying an activity recognition and intervention system in the homes of people living with dementia across three countries (Portugal, Belgium, and Italy), in collaboration with Fraunhofer Portugal and LUCA School of Arts [198].

The project was selected as a finalist for the Longitude Prize on Dementia [1], which recognizes technology that helps people living with dementia maintain their independence.

The Autonomous system draws directly on the sensing and personalization approaches developed in this dissertation. A multimodal sensor suite, including Doppler radars, thermal cameras, microphone arrays, and depth cameras, captures complementary information about kitchen activities. All sensor data is processed locally on a hub installed in the participant's home, ensuring that sensitive data never leaves the premises. During an onboarding phase, participants or their caregivers provide a small number of labeled activity examples via a smartwatch interface to personalize a foundation model pre-trained on data from earlier participants, following the bootstrapping principle from VAX: a brief period of guided input produces a personalized model, and afterward the system operates without further human labeling burden. Rather than simply detecting activities, the system delivers proactive interventions through the smartwatch when sustained patterns indicate a potential concern, such as water running without associated activity, a stove left on without recent kitchen presence, or prolonged inactivity during typical waking hours.

The system's interface was developed through iterative participatory design with people living with dementia, caregivers, and healthcare professionals [197]. This co-design process identified requirements like personalization of alert modalities (including voice, vibration intensity, and iconography), visual simplicity to reduce cognitive load, and multisensory feedback to accommodate varying cognitive ability. The system was deployed in six homes between August and November 2025. Participants who actively used the system reported that it meaningfully supported their daily routines. One participant living alone described reminders to turn off the tap, close cabinet doors, and attend to the oven as "very, very helpful," noting that they increased their sense of safety. Over time, this participant reported that the prompts had begun to shape habits, making them more attentive to closing cabinets even without a notification. A caregiver observed that the alerts became more frequent, more specific, and better suited to the circumstance as the system learned the household's routines. Three participants requested to continue using the system after the pilot concluded.

The deployment also revealed how deeply personal the definition of "successful support" is [198]. One participant in Belgium makes himself a ham sandwich every day for lunch, a task he finds challenging enough that he calls his partner multiple times during the process. The system could intervene with step-by-step guidance, but the team found itself asking: when is the sandwich successful? If the participant places cheese on top but forgets the butter, should the system intervene? The answer depends on the individual, and arriving at it requires sustained dialogue between designers, caregivers, and participants that cannot be reduced to an accuracy metric. At the same time, participants who struggled to engage with the system attributed the difficulty not to the technology but to their own capabilities [197], highlighting a challenge fundamental to assistive technology for progressive conditions: the people who stand to benefit most are also the people for whom interacting with any technology becomes increasingly difficult.

The deployment surfaced practical challenges as well. Wi-Fi reliability in residential settings caused communication drops that hampered functionality for some participants. Multi-occupancy households complicated activity attribution without invasive identification measures. And the progressive nature of dementia itself means the user's capabilities and routines change over the deployment period, requiring the system to adapt not just to a new environment but to changes

in the way activities are performed by the person within that environment. The Autonomous deployment is the closest realization to date of the vision articulated in Section 1.2, and its most important lesson may be that the distance between a technically functional system and one that genuinely supports a person's independence is measured not in accuracy points but in the quality of the conversation between the system, its designers, and the people it serves.

To make the Autonomous system suitable for sustained, unsupervised use, we need to close several technical gaps that our deployment brought into focus. Our onboarding models are fit once and do not adapt as the person's condition progresses or as routines shift over months (discussed in Section 7.3.3). Our proactive interventions cover a narrow band of safety-critical alerts but cannot negotiate what is worth monitoring, communicate where the system's sensing is confident and where it is not, or verify that its reading of an observed pattern matches the occupant's own understanding, questions we explore in Section 7.3.7. And our reliance on cloud-based VLMs during bootstrapping remains a practical barrier for the privacy-sensitive home settings where the system is most needed, one that edge-deployable alternatives (Section 7.3.1) will remove.

7.1.2 BuildingChat in TCS Hall

We built BuildingChat on top of the Mites sensing infrastructure deployed throughout TCS Hall at Carnegie Mellon University [34], where over 300 multi-modal sensors continuously collect environmental data across all four floors. The system is available to any building occupant and serves as a live interface to this sensing infrastructure through natural language. The formative and evaluation studies reported in Chapter 6 were conducted with this deployed system. BuildingChat continues to operate, handling the full range of occupant queries against real sensor data that includes gaps, noise, and the day-to-day variability of a shared academic building.

This sustained operation has also clarified the natural next steps for the system. Today BuildingChat answers questions about the building; the first extension would be to let it act on the occupant's behalf, adjusting a thermostat or booking a room rather than simply reporting conditions. Our operator-composition architecture enables actuation commands by expressing them as pre-verified operators with confirmation steps and safety bounds. However, our access control model would need to grow to handle write permissions and to resolve conflicts when occupants issue contradicting preferences for a shared space. A second extension is user-defined automation. An occupant who routinely checks outdoor conditions before leaving could describe that pattern once, and BuildingChat could save the corresponding operator graph as a standing automation triggered by a schedule or by an event such as the end of the day's last meeting. Consumer platforms have recently adopted a similar pattern, letting users create recurring automations by describing them in natural language.[1] In both cases, the operator library and DAG composition that we use for ad-hoc queries provide the building blocks; what remains is a scheduling and conflict-resolution layer on top of it. We also discuss the broader applicability of this operator-composition paradigm in Section 7.3.5.

[1]Apple Intelligence, for example, lets users generate Shortcuts automations from natural-language descriptions (`https://www.apple.com/apple-intelligence/`).

7.1.3 Toward an Integrated Home Deployment

The four systems presented in this dissertation were developed and evaluated independently. A natural question is whether they can be composed into a single pipeline that realizes the full vision of Section 1.2: a system that bootstraps itself in a new home from sensor data and temporary observation, discovers what activities the occupant performs, synthesizes meaningful context from those activities, and provides both responsive and proactive interaction through natural language.

I am currently (as of June 2026) developing such an integrated system.[2] The hardware combines a collection of Mites sensors [34] and other off-the-shelf sensors deployed throughout the home with a smart-glass-based camera that serves as the temporary rich source for bootstrapping training labels. Unlike the tripod-mounted camera rigs used in VAX and OrganicHAR, the smart glass is worn by the occupant during a brief training period, capturing a first-person perspective of daily activities that can be processed by vision-language models to generate labels for the ambient sensors. Once the ambient sensor models are trained, the smart glass is set aside. The interaction layer follows the BuildingChat architecture, extended to support both occupant-initiated queries and system-initiated notifications, drawing on the proactive intervention patterns developed in the Autonomous project.

7.2 Discussion

7.2.1 What the Bootstrapping Principle Achieved

The recurring design principle of this dissertation is that per-environment user effort can be replaced by opportunistically engaging a more capable resource, bounded by scaffolding that ensures it is not necessary after setup or remains in a constrained role. VAX demonstrated this for training labels: a camera and microphone depart after generating labels for privacy-preserving sensors, reducing labeling effort from 17 labels per home to approximately 2. OrganicHAR demonstrated it for activity vocabulary: a vision-language model processes brief video segments during key moments to semantically ground clusters discovered from sensor data, then departs. TAO demonstrated it for context: an ontology provides structured knowledge that grounds the temporal patterns a clustering pipeline discovers from data, remaining as a bounded reasoning component. BuildingChat demonstrated it for interaction: a language model composes pre-verified analytical operators into execution graphs, remaining in a bounded orchestration role where it never touches raw sensor data.

Across these four stages, the principle held: in each case, the automated alternative achieved performance comparable to, or better than, the conventional approach that demanded substantially more user effort. VAX matched a fully supervised baseline (84% vs. 79%) with one-eighth the labeling effort. OrganicHAR discovered 4 to 8 meaningful activity categories per environment with 87% average accuracy while processing only 10% of the video that continuous monitoring would require. TAO produced wellness assessments within 1.1% of ground truth, compared

[2]The integrated home deployment is not a contribution of this dissertation and is not required for its completion. I include it here to indicate that the end-to-end composition of the pipeline is being pursued as a concrete engineering effort rather than left as a theoretical possibility. Progress is tracked at `https://github.com/synergylabs/self-configuring-smart-homes`.

to 30% divergence for a baseline using activities directly. BuildingChat achieved 88% query accuracy while eliminating the unauthorized data leakage that occurred in 13 to 46% of baseline queries.

These results suggest that the bootstrapping principle is not specific to any one stage but is a general strategy for reducing deployment effort in context-aware systems. The pattern of temporarily engaging a more capable (and potentially more privacy-invasive) resource, scaffolding it so that its contributions are predictable and bounded, and then constraining or removing it may apply to stages of the pipeline not addressed in this dissertation, such as sensor placement guidance or policy configuration for access control.

7.2.2 The Pipeline as a Whole

Each of the four systems was evaluated against the stage it was designed for, but they were never run together as one composed pipeline. It helps to be precise about how the four relate. VAX and OrganicHAR both sit at the front of the pipeline, where raw signals from privacy-preserving sensors are turned into activity labels: VAX addresses how to obtain those labels without manual annotation, and OrganicHAR addresses what the label vocabulary should be in the first place. The composition the pipeline depends on runs from this activity-recognition front end into TAO's context synthesis, and from context into BuildingChat's conversational interface. This composition matters because the performance of the assembled pipeline is not the product of its parts. Each stage consumes the output of the stage before it, so an error introduced early does not stay contained. Activity labels that are noisy or incomplete leave TAO unable to synthesize the contexts that depend on them, and inaccurate context in turn produces misleading answers when BuildingChat reports on what an occupant has been doing.

TAO's evaluation is the one place in this dissertation where such a handoff was measured directly, and the result makes this concern concrete rather than hypothetical. When TAO received activities annotated from video, it detected contexts with a Jaccard coefficient of 72%, a precision of 83%, and a recall of 69%. When the same pipeline instead received activity predictions from the Mites ML models, its accuracy reduced to a Jaccard coefficient of 22%, a precision of 36%, and a recall of 24%. The context-reasoning logic did not change between these two runs; only the quality of the activity stream feeding it did. This is not a property peculiar to TAO. It is what any layered system exhibits when each stage inherits the mistakes of the one beneath it, and it is the most direct evidence in this work that an integrated deployment will be governed by the accuracy of its activity-recognition front end. That front end therefore carries more weight than its position might suggest, since everything downstream is bounded by how well activities are recognized in the first place. This is the problem VAX and OrganicHAR were built to address, with OrganicHAR going further by removing the assumption of a fixed activity vocabulary. What the stage-by-stage evaluation in this dissertation cannot establish, however, is how these errors accumulate once all of the systems run in sequence on the same data. The integrated home deployment described in Section 7.1.3 is where that question begins to be examined.

7.2.3 Longitudinal Robustness

The evaluations reported in this dissertation span hours to days, with OrganicHAR's seven-day home deployment being the longest. The vision described in Section 1.2 assumes operation that is sustained over months or years, and longitudinal robustness turns out to be a different problem altogether from short-term accuracy.

The first reason is that behavior does not hold still. A new roommate moves in, a seasonal shift rearranges a daily routine, an injury changes how a person moves. In the Autonomous deployment, for example, the progression of dementia gradually transforms how a person carries out familiar tasks. A model fit during a short bootstrapping phase captures behavior as it was during that phase. VAX's sensor models, trained on a few hours of activity, need not cover the full range of an occupant's routine. OrganicHAR's discovered activities reflect the period in which they were found and may miss behaviors that only appear weeks later. TAO's ontology, once designed, stays fixed and cannot name a context it was not built to describe. Each of these systems was evaluated against behavior that stayed largely stable over the study, which is precisely the assumption that fails in long deployments.

The second reason is that the sensing itself drifts. Batteries deplete, network links fluctuate, and devices are moved or blocked when furniture is rearranged. The Autonomous deployment ran into Wi-Fi reliability problems that interrupted functionality for some participants, a reminder that the stability a controlled study can take for granted is not available in a real home. None of the evaluations in this dissertation were long enough to observe how the systems behave once behavior and hardware shift underneath them, or how such issues might be addressed. How the bootstrapping principle might be extended to keep pace with that change is taken up in Section 7.3.3.

7.2.4 The Limits of Reactive Interaction

The vision in Section 1.2 describes a system that opens the conversation itself, telling the occupant what it can detect, what it cannot, and asking what would be useful to know. The four contributions in this dissertation sit almost entirely on the other side of that exchange. They answer when the occupant asks, most explicitly in BuildingChat, where the occupant poses a question and the system responds. The system rarely speaks first.

The Autonomous project (Section 7.1.1) is the one place where we built and observed the proactive side. Its intervention system watches activity predictions over a sliding window, raises a notification only when a pattern is sustained rather than on any single prediction, and limits how often it interrupts. Participants valued these notifications, those living alone especially, and some began to anticipate the alerts and act before they arrived, which suggests that a system speaking first can shape behavior in a way the occupant welcomes.

However, the proactive engagement in Autonomous is in a narrow band: its proactive messages are confined to safety-critical patterns such as an appliance left on, water running, or a long stretch of inactivity. It does not negotiate what is worth monitoring, explain the boundaries of its own sensing, or check whether its reading of a situation matches the occupant's. That kind of conversation asks the system to reason about its own uncertainty and put it into words, and neither BuildingChat's template composition nor Autonomous's rule-based intervention offers it. The

research questions this opens are taken up in Section 7.3.7.

7.2.5 Sensor Placement

The systems in this dissertation are agnostic to where sensors are placed and how a room is arranged, and yet across every deployment, placement turned out to matter a great deal. In VAX, furniture occlusion and poor camera angles kept the A/V pipeline from producing labels for some activities. In OrganicHAR, the linear layout of certain kitchens forced the camera into lateral positions, and the resulting scene descriptions were less informative than the ones an elevated, diagonal view produced. In the Autonomous deployment, where a sensor sat relative to an appliance or a work surface decided which activities could be detected at all. In each case the algorithms were unchanged; what differed was the physical vantage point they were given.

These were not incidental issues. They set a ceiling on what the rest of the pipeline could do, since an activity that the sensors cannot observe well is one that no amount of downstream reasoning can recover. The question of whether the system itself could help an occupant place sensors better, rather than leaving placement to chance, is taken up in Section 7.3.2.

7.3 Future Directions

Each limitation discussed in Section 7.2 points to research worth pursuing. This section sets out the concrete directions they open, ordered roughly from near-term technical extensions to the broader questions that would bring the vision of Section 1.2 closer.

7.3.1 Edge-Deployed Vision-Language Models

OrganicHAR relies on cloud-based vision-language models during its training phase, which introduces a transient privacy exposure: video segments from the home are sent to third-party services for scene understanding. OrganicHAR limits this exposure by processing only 9 to 11% of the video and removing the camera once training is complete, but the dependence on the cloud is still a practical barrier for exactly the populations who would benefit most, including elderly individuals and patients recovering at home.

The rapid progress in edge-deployable VLMs suggests this barrier may be temporary. Models such as Qwen2.5-VL [2] at three billion parameters and SmolVLM [3] now run on resource-constrained devices, including single-board computers and edge accelerators. Early experiments with local VLMs in the OrganicHAR pipeline indicate that current edge models do not yet reach the scene-description quality needed for reliable activity labeling, particularly for fine-grained distinctions between activities that look alike. The trajectory of model compression, quantization, and specialized edge hardware, however, points toward edge VLMs capable of supporting the OrganicHAR workflow in the near term. Characterizing the accuracy and privacy tradeoff of edge VLMs in the bootstrapping pipeline, and finding the minimum description quality that still yields usable labels, is a concrete and timely direction.

7.3.2 System-Guided Sensor Placement

Section 7.2.5 showed that sensor placement set a ceiling on what every deployment in this dissertation could achieve, while the systems themselves stayed agnostic to it. A natural next step is to let the system close that gap on the occupant's behalf. Because the system eventually learns what it can and cannot detect, it could assess the quality of its own sensing after a short observation period and suggest where a sensor should move. It might report, for instance, that it can see movement near the stove but cannot tell apart the activities happening at the countertop, and propose moving the sensor closer or changing its angle. Guidance of this kind would extend the bootstrapping principle to the physical setup of the infrastructure itself, reducing the expertise a deployment demands not only for labeling and configuration but for the act of placing the sensors in the first place.

7.3.3 Continuous Adaptation and Concept Drift

Section 7.2.3 argued that the systems in this dissertation assume behavior and hardware that stay stable through and after bootstrapping, an assumption that is likely not true in a longitudinal deployment over months or years. One way to meet that challenge is to apply the bootstrapping principle not once but periodically. The richer source, whether a camera, a VLM, or a brief labeled session, could return for short intervals when the system senses that its predictions have grown less confident or less consistent. The system could track its own prediction entropy over time and, when confidence falls below a threshold, ask the occupant for a short re-training session or bring the bootstrapping camera back for a day. This re-bootstrapping treats the initial setup not as a one-time event but as the first turn of a continuous adaptation cycle.

For TAO, continuous adaptation raises the question of extending the ontology on the fly. New activities can appear that the current ontology does not cover, producing contexts the system has no name for. A mechanism that proposes new ontology entries from recurring but unlabeled temporal patterns, and presents them to the occupant for confirmation, would close the loop between the patterns discovered from data and the structured knowledge used to reason about them. These ontologies can also be crowd-sourced so that as ontologies grow in individual homes, they can be unified to create richer "global" ontologies.

7.3.4 LLMs as Dynamic Context Reasoners

TAO detects context with a formal ontology, and that choice buys real properties: the mappings from activity patterns to contexts are explicit, interpretable, and transferable across environments without per-environment training. However, the ontology can only express contexts its designers anticipated, extending it takes researcher effort, and, as Section 7.2.2 showed, its rules are rigid enough that a noisy activity stream breaks the mapping rather than bending it. Large language models are interesting here not as a replacement for that structure but as a way to cover what it cannot. An LLM can propose a plausible context from partial or uncertain activity information where a rigid mapping would simply fail. A hybrid arrangement, where the ontology remains the verified base for well-established context mappings and an LLM handles the novel or ambiguous

patterns, would keep the interpretability of the ontology for known contexts while extending coverage to ones it was never built for.

The LLM could also take on the dynamic ontology extension described in Section 7.3.3. When it meets a recurring activity pattern the ontology does not cover, it could propose a new context label and the mapping that goes with it, for a person to verify before the system begins to use it. This direction follows the same philosophy as BuildingChat. Just as BuildingChat constrains the language model to composing pre-verified operators rather than generating arbitrary code, a context reasoner can constrain the language model to proposing mappings that are verified before they become operational, keeping the safety guarantees while gaining flexibility.

7.3.5 The Operator-Composition Paradigm Beyond Buildings

BuildingChat's approach to language-model-driven querying, composing pre-verified analytical operators into directed acyclic graphs rather than generating arbitrary code, speaks to a problem that reaches well past building sensor data. Any setting in which a language model should orchestrate analytical operations over sensitive data faces the same tension between flexibility and safety: the model must handle a wide range of queries, yet it must not reach data it should not see or run computations no one has checked.

Clinical data systems feel this tension sharply. A physician querying patient records in natural language needs multi-step composition, retrieving lab results, comparing them against a historical baseline, and identifying a trend, under strict access control where a physician may see their own patients' records and not others'. Educational analytics carries a similar shape, where an instructor asking about engagement patterns needs compositional queries alongside privacy protections for individual students. Industrial settings, where operators query data from manufacturing equipment, ask for the same pairing of analytical flexibility with safety guarantees over operational data. Investigating whether BuildingChat's template composition and policy validation transfer to these domains, and what each domain demands in adaptation, would carry the contribution well beyond the building context in which it was developed.

7.3.6 Healthcare Deployments as a Proving Ground

The thesis is motivated by dementia care and post-surgical recovery, and the Autonomous project (Section 7.1.1) is a first step toward deploying the pipeline in these settings. Healthcare is at once the highest-impact use case and the most demanding test of what the pipeline can do.

Dementia care asks for longitudinal operation over months or years, sensitivity to changes as subtle as a gradual increase in the time a task takes, coverage across the rooms where activities of daily living happen, and a careful balance between proactive support and a person's autonomy, since over-alerting can do as much harm as under-alerting. Post-surgical recovery, such as the wound-care tracking explored in related work [16, 206], asks for precise detection of procedural steps and the ability to tell a correctly performed care task from an incorrect one. These requirements press on every stage at once. The training labels must be accurate enough for safety-critical detection, the activity vocabulary must capture the fine steps of a care procedure, the context layer must separate ordinary variation from a clinically meaningful change, and the interaction layer must speak with people who may have cognitive impairment or limited technical familiarity.

A sustained deployment with actual care recipients, even at a small pilot scale, would be the most rigorous test of the thesis vision and would surface challenges that no controlled study can anticipate.

7.3.7 The System-Initiated Conversation

Section 7.2.4 described how far the contributions in this dissertation sit from a system that opens the conversation itself, and how narrow Autonomous's proactive messages remain. Turning that gap into a research agenda raises three concrete questions.

The first is how a system should assess its own sensing in a new environment. After a bootstrapping period, can it produce a summary of what it can reliably detect and where its confidence is low, and present that to the occupant in plain language? The second is how proactive communication should be balanced against alert fatigue. The Autonomous deployment showed that rate limiting and priority-based filtering help, but the right balance likely depends on the person and the situation. The third is whether a conversational interface can support a negotiate-and-refine pattern, where the system offers an interpretation of something it has observed, such as "it looks like you make tea around three most afternoons, should I treat that as a regular routine?", and the occupant confirms, corrects, or dismisses it. This pattern would close the loop between what the system discovers on its own and what the occupant validates, completing the shift from constructing a system to confirming what it produces that this dissertation has pursued throughout.

7.4 Societal Impact

The systems in this dissertation are built to understand what happens in a space while protecting the privacy of the people in it. As sensing moves into homes, offices, and care settings, the questions of who benefits from that understanding, who is exposed by it, and who decides how it is used stop being abstract. This work carries clear benefits, and it raises concerns that deserve to be stated plainly rather than left between the lines.

The most direct benefit is a practical alternative to camera-based monitoring. For most of the history of context-aware computing, the systems that worked best were the ones that watched and listened most closely, and so usefulness and privacy were treated as a tradeoff that could not be avoided. Cameras and microphones were the default because they produce rich data and because large datasets exist to train models on them. The privacy-preserving sensors used in this work, such as radar, thermal arrays, and motion sensors, capture far less. They can register that someone is moving near the stove without recording their face, their appearance, or their conversation. By showing that these sensors can reach accuracy close to what cameras provide once they are bootstrapped appropriately, this work gives system builders a reason to choose the less invasive option without giving up the function they need. For places where a camera is unwelcome or unacceptable, a bedroom, a bathroom, the home of someone who values their privacy, that shift opens applications that were previously off the table.

The people who stand to gain the most from this are often those with the highest stakes. In the Autonomous deployment, we worked with people living with dementia who want to remain in their own homes rather than move into care earlier than they need to. For them, independence

is not a convenience but a large part of their quality of life, and the usual ways of keeping them safe, a live-in caregiver or a camera in the kitchen, come at the cost of privacy and dignity. A system that can notice a stove left on or a tap running, and do so without recording video, can extend the period during which living alone remains safe. The same system changes something for the people who love them. Families separated by distance carry a quiet and constant worry about a relative living alone, and that worry usually turns into another phone call to check in. A system that can confirm that meals are being made and the home is in order replaces some of that worry with something specific, and in our deployment the alerts grew more particular and better suited to each household as the system learned its routines. For the caregivers who carry most of this work, that relief matters, because their own exhaustion is one of the least visible costs of long-term care [205]. Used with care, monitoring of this kind supports a person and the people around them instead of simply watching over them.

These benefits rest on a quieter contribution, which is the lowering of the effort it takes to deploy a system at all. Context-aware sensing has mostly been available to people with the technical skill and the resources to set it up, which leaves out many of the households that could use it most. The people who would benefit the most from a system that watches for a stove left on are frequently the least able to configure one. By replacing the per-environment setup work with automated alternatives, this work makes it realistic for such a system to be deployed by someone who would never label a dataset or edit a configuration file. BuildingChat carries a related benefit into shared buildings. The sensor data a building collects has long flowed only to facility managers, leaving the occupants who live and work in those spaces with no way to ask simple questions about them. Letting a person ask whether their office is too warm, or which nearby room is quiet, returns to them a small measure of insight into the spaces they spend their days in.

These advancements come with a new set of concerns. The privacy improvement is meaningful, but it is not complete. The bootstrapping phase still relies on a camera and, in some cases, on models running in the cloud, so the system narrows privacy exposure to a short setup window rather than removing it, and that window has to be handled carefully rather than waved away. Even once the camera is gone, the behavioral patterns the system continues to infer can themselves be sensitive. A record of when a person is home, when they are active, and when they are resting can reveal a great deal about their life without a single frame of video, and over time it can surface things they never intended to share, including signs of declining health. Inferences of this kind deserve the same protection as the raw data they were meant to replace.

The benefit to vulnerable people comes paired with a responsibility toward them. The people best served by these systems are often the least able to give meaningful consent, or to object when the system gets something wrong. Our deployment showed this in a small but telling way: when the system was difficult to use, participants tended to blame themselves rather than the technology. It also showed how personal the idea of help really is. One participant prepared the same lunch every day in a way that matched no standard recipe, and the question of whether the system should step in when he skipped a step was not a technical one. It depended on what mattered to him, on whether a slightly different lunch was a problem to be corrected or simply his own way of doing things. Judgments like these belong to the person and to the people who care for them, not to whoever built the system, and they cannot be reduced to an accuracy number.

The same capabilities that support a person can be turned to watching them, and which one it

becomes usually depends on who holds the data. The wellness metrics that TAO makes possible are genuinely useful when a person uses them to understand their own patterns of work and rest. Placed in an employer's hands, the very same metrics become a means of monitoring workers, and the kind of inference this dissertation set out to enable, a space that can tell focused work from idleness, is precisely what that monitoring would draw on. Shared buildings raise the same problem in another form, because a question that is perfectly reasonable for one occupant to ask can amount to surveillance of another, and many of the people a building senses never agreed to it and may not know the sensors are there. BuildingChat answers this by building access control into the system rather than trusting a written policy to be followed, but a technical control can only settle part of what is, at heart, a question of governance.

These considerations were not added after the fact; they shaped the choices the work is built on. Privacy-preserving sensors stand in for cameras, observation is temporary rather than continuous, access control is enforced by the system rather than left to good intentions, and a person stays in the loop to confirm and correct what the system produces. Each of these reduces a risk without removing it entirely. Whether systems like these are worth building will depend not only on how well they work, but on whether the understanding they produce stays in the service of the people it is about. That is not a property the technology can guarantee on its own, and keeping it true is work that continues long after a system is deployed.

The four systems presented in this dissertation each demonstrated that automated alternatives can reduce the user effort required to deploy context-aware systems in new environments. No single system realizes the full vision of a self-configuring environment, and the ongoing work and future directions described in this chapter make clear how much remains to be done. What this dissertation establishes is that the bootstrapping principle, engaging a more capable resource with appropriate scaffolding to train, discover, reason, or interact on behalf of a more privacy-preserving system, is viable across the sensing-to-interaction pipeline. The gap between privacy-preserving sensors and the contextual understanding that makes smart environments useful is not a single problem but a series of challenges at successive stages, and at each stage, the human effort that conventional approaches demand can be substantially reduced. The path from here to the self-configuring environments described in Section 1.2 requires integration, longitudinal validation, and sustained deployment with the populations who stand to benefit most. The research presented here provides a foundation for that path.

Appendix A

Supplementary Material for OrganicHAR

A.1 Featurization Pipelines for Various Sensing Modalities

This appendix provides detailed technical specifications for the featurization pipelines used for specific sensing modalities. The following detailed descriptions outline the specific features extracted from each sensor, and the rationale behind our design choices, providing the implementation details necessary for replicating our multimodal sensing approach.

- **Doppler Radar:** We extract temporal features (velocity statistics, directional changes), spatial features (distance from sensor, movement extent), complexity features (velocity entropy, motion transitions), and motion type (stationary, slow/fast movement *etc.*) distributions. While Doppler provides precise and high fidelity velocity sensing, it only captures motion in the radial direction and struggles with distinguishing movement at the same distance or complex spatial relationships.
- **2D LiDAR:** Our approach first establishes a static boundary model in the horizontal 2D plane, then extracts features from deviations that represent dynamic objects. Features include centroid positions, velocities, point distributions, boundary interactions, and temporal patterns. LiDAR provides precise spatial mapping but with decreasing granularity at greater distances, and it cannot capture fine-grained movements or height information.
- **Low-Resolution Thermal Camera:** From the 10×10 thermal array, we compute spatial features (temperature statistics, gradients) and signature-based features that identify thermal patterns from humans, appliances, and ambient sources. While thermal sensing excels at detecting heat-generating activities and object interactions, it is limited to coarse human movements or thermal changes.
- **IMU (Wearables):** We leverage the pretrained SAMoSA [142] model to extract 128-dimensional motion features from 2.88-second windows with a stride of 0.21 seconds. Wearable IMUs capture detailed hand and arm movements but are limited to the wearer's perspective and require consistent device wearing.
- **Pose:** From the raw 2D pose data over 25 body keypoints (in 640×480 frame) at 6–8Hz, we extract biomechanical features that characterize body positioning and movement, including joint velocities and accelerations, inter-joint configurations, working zones (based on torso

and hand movements), and movement complexity. Despite providing rich body movement data, pose estimation is constrained by occlusions and the limited field of view (less than 60 degrees), sometimes missing key interactions outside the camera's perspective.

- **Depth:** We extract motion patterns and statistical features from the 10×10 depth array, establishing boundary maps to identify deviations representing people and objects. While effective for detecting presence and basic movements, depth sensing is limited by its narrow field of view and cannot capture fine-grained interactions.

A.2 Prompts for LLMs and VLMs

This appendix provides the detailed prompts used in the OrganicHAR framework for activity discovery. The pipeline employs five main prompts for Vision Language Models (VLMs) and Large Language Models (LLMs) that work in sequence to translate video clips into consistent activity labels at appropriate granularity levels. The exact prompts can be found in the Supplemental File as a text file. Here, we describe the key design for each prompt to address specific challenges in the pipeline.

1. The first step leverages a VLM to generate rich semantic descriptions from video frames captured during key moments. This prompt instructs the VLM to analyze short video clips focusing on actions, objects, locations, and activity structure. It specifically directs the model to focus only on clearly observable elements, provide confidence scores, and exclude low-confidence observations.
2. Since VLM outputs contain diverse location references (*e.g.*, "at sink," "by counter"), our second step employs an LLM to consolidate these into consistent functional zones. This prompt directs the LLM to cluster locations based on supported activities, ensuring that areas like "sink area," "counter area," and "coffee machine area" are consistently identified. This spatial context is crucial because activities have different meanings in different locations.
3. After establishing functional zones, we use a two-step process to transform the unstructured VLM descriptions into discrete activity labels. First, we employ the clustering prompt to create initial activity clusters for each functional zone. This prompt directs the LLM to preserve task-specific distinctions while minimizing the overall number of clusters.
4. Next, we match each VLM description to the appropriate cluster using the matching prompt. This prompt directs the LLM to evaluate matches based on action alignment (60%), object consistency (25%), and location match (15%), ensuring that descriptions are consistently mapped to appropriate activity labels.
5. Finally, to enable granularity control through the relaxation parameter λ, we analyze each activity's semantic properties across multiple dimensions. This prompt directs the LLM to break down each activity in terms of action type, objects involved, sub-location, purpose, related activities, and access patterns. These dimensions form the basis for computing pairwise similarity scores between activities, which are then used with the relaxation parameter λ to determine which activities should be merged at different granularity settings.

A.3 VLM Variability and Configuration Analysis

We conducted additional analyses examining how different VLM models and frame rate configurations impact label generation performance. These evaluations provide insights into design choices and system adaptability across different VLM capabilities.

Performance of Label Discovery Pipeline with Different VLMs. We evaluated OrganicHAR using three state-of-the-art vision language models as of July 2025: OpenAI's GPT-4.1, Google's Gemini 2.5 Flash, and Anthropic's Claude Sonnet 4. These models represent different architectures and training approaches currently available. For consistency, all models used identical prompts with minor formatting adaptations for model-specific requirements.

Table A.1: Label discovery performance across VLM models and semantic granularity settings. Detection rate represents the percentage of key moments that generate useful activity descriptions.

Granularity	Model	Acc (%)	F1 (%)	Detection Rate (%)
Conservative ($\lambda = 0.4$)	GPT-4.1	84.9	72.5	92.0
	Gemini 2.5 Flash	80.9	69.7	50.2
	Claude Sonnet 4	59.3	50.8	89.2
Balanced ($\lambda = 0.3$)	GPT-4.1	83.9	69.4	92.0
	Gemini 2.5 Flash	79.0	67.2	50.2
	Claude Sonnet 4	60.4	55.4	89.2
Relaxed ($\lambda = 0.2$)	GPT-4.1	72.0	55.4	92.0
	Gemini 2.5 Flash	69.0	54.0	50.2
	Claude Sonnet 4	48.7	35.6	89.2

Table A.1 shows notable variations across models. GPT-4.1 and Gemini 2.5 Flash demonstrate similar accuracy levels in Conservative and Balanced settings (80–85%), while Claude Sonnet 4 shows lower accuracy but reasonable performance in the Balanced setting (60.4% accuracy, 55.4% F1). However, accuracy and F1 scores alone do not capture whether VLMs can consistently provide meaningful activity descriptions. To measure this consistency, we examine the detection rate, the percentage of video segments for which the VLM generates actual location and action information rather than empty responses. GPT-4.1 maintains a high detection rate (92.0%) across all settings, while Gemini 2.5 Flash achieves only 50.2% detection rate despite competitive accuracy when successful, and Claude Sonnet 4 maintains a high detection rate (89.2%) but with lower overall accuracy. For our use case, GPT-4.1 performs the best in terms of both accuracy and reliability, making it the most suitable choice for consistent activity discovery across diverse scenarios.

Frame Rate Impact Analysis. Based on GPT-4.1's combination of high accuracy and detection rate, we selected it for frame rate analysis. Using GPT-4.1, we evaluated performance across three frame rates (1, 3, and 5 FPS) using identical 5-second video clips from our dataset.

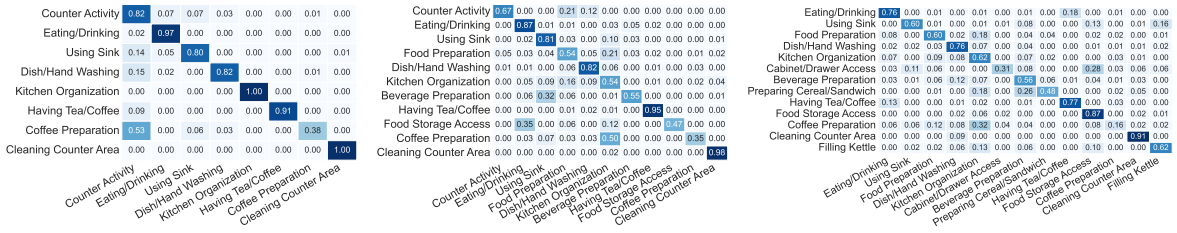
Table A.2: Impact of frame rate on label discovery performance using GPT-4.1. Detection rates remain consistent across frame rate settings.

Frame Rate	Granularity	Acc (%)	F1 (%)	Detection Rate (%)
1 FPS	Conservative	84.9	72.5	92.0
	Balanced	83.9	69.4	92.0
	Relaxed	72.0	55.4	92.0
3 FPS	Conservative	82.9	70.5	91.3
	Balanced	80.6	62.5	91.3
	Relaxed	69.0	52.4	91.3
5 FPS	Conservative	82.4	67.5	90.6
	Balanced	79.2	65.1	90.6
	Relaxed	66.7	53.1	90.6

Table A.2 shows an interesting pattern: increasing frame rate from 1 to 5 FPS results in modest performance decreases rather than improvements. Accuracy drops from 84.9% to 82.4% in the Conservative setting, with similar trends observed across all granularity levels. This finding suggests that kitchen activities may operate at coarse timescales where meaningful changes occur over multi-second intervals. Additional frames often capture intermediate postures, repetitive motions, or transitional states that may not provide additional semantic information useful for activity classification. The 1 FPS configuration appears to capture key semantic moments while avoiding potential temporal noise. Frame rate selection also has practical implications for deployment. Processing at 5 FPS requires five times the input context in the VLM API calls compared to 1 FPS, with corresponding increases in latency, bandwidth usage, and cloud processing costs. In conclusion, our analysis indicates that this increased computational overhead does not translate to improved recognition performance under current VLM capabilities.

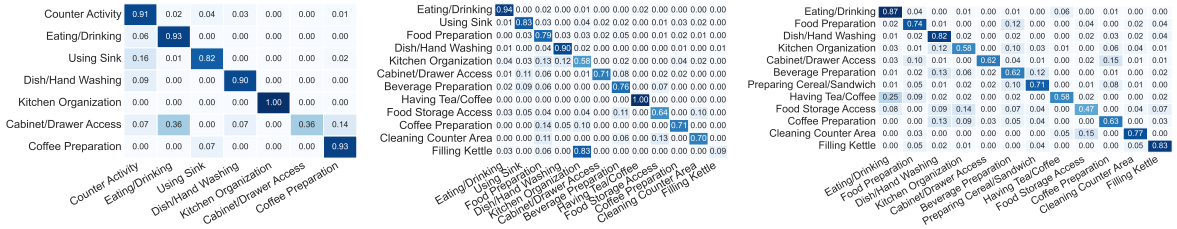
Implications for System Design. These evaluations provide several insights for OrganicHAR deployments. The detection rate metric reveals that model selection involves trade-offs between accuracy when successful and reliability of processing, where consistent processing across all key moments may be more important than marginal accuracy improvements on successfully processed segments. The frame rate analysis supports our original configuration choices (1 FPS, 640×480 resolution) as a reasonable balance between performance and computational efficiency. The consistent performance patterns across multiple VLMs indicate that fine-grained activity recognition limitations (accuracy scores of 48–72% in Relaxed settings) reflect current VLM capabilities rather than weaknesses in our approach, as similar performance degradation occurs across different model architectures when distinguishing semantically similar activities. OrganicHAR's modular architecture enables adaptation to different VLMs without modifying sensor processing components, allowing the framework to incorporate future advances in VLM temporal reasoning and fine-grained visual understanding while maintaining efficiency for current deployments.

A.4 Activity Confusion Analysis (Other Sensor Configurations)



(a) Conservative ($\lambda = 0.4$) (b) Balanced ($\lambda = 0.3$) (c) Relaxed ($\lambda = 0.2$)

Figure A.1: Confusion matrices showing activity recognition performance for *Ambient(Basic)+Wearable(IMU)* configuration across three granularity settings.



(a) Conservative ($\lambda = 0.4$) (b) Balanced ($\lambda = 0.3$) (c) Relaxed ($\lambda = 0.2$)

Figure A.2: Confusion matrices showing activity recognition performance for *Ambient(Advanced)+Wearable(IMU)* configuration across three granularity settings.

We present the confusion matrices for the other configurations in our evaluation. First, adding wearable IMU sensing to basic ambient sensors reveals complementary strengths and persistent challenges (Figure A.1). Activities with distinctive motion patterns like "Kitchen Organization" and "Cleaning Counter Area" show excellent recognition across granularity settings. However, activities sharing similar motion signatures in different locations remain challenging to differentiate. In Conservative settings, "Coffee Preparation" frequently confuses with broader "Counter Activity" categories. The Balanced setting reveals confusion between "Beverage Preparation" and "Using Sink," while the Relaxed setting shows increased confusion between semantically related activities like "Cabinet/Drawer Access" and "Food Storage Access." These patterns demonstrate that wearable sensing effectively complements ambient sensors for activities with unique motion signatures but cannot fully resolve spatial ambiguities for activities with similar hand movements performed in different contexts.

Secondly, the most advanced configuration combining ambient sensors with wearable IMU, pose estimation, and depth sensing shows notable improvements in activity disambiguation (Figure A.2). With pose and depth information, activities that were previously confused due to spatial ambiguity show clearer separation. Certain activities like "Having Tea/Coffee" and

"Eating/Drinking" achieve particularly strong recognition rates across settings. However, even with this rich sensing configuration, some challenging distinctions remain, particularly for activities with subtle differences like "Filling Kettle" (which often confuses with general "Kitchen Organization") and "Food Storage Access" (which sometimes confuses with "Cleaning Counter Area"). The advanced configuration particularly excels at disambiguating fine-grained activities that involve distinctive postures or spatial relationships, confirming that spatial awareness through pose and depth information provides a crucial complement to motion and environmental sensing. This highlights how different sensor modalities each contribute unique capabilities to the overall recognition system.

Appendix B

Supplementary Material for TAO

This appendix provides additional details for the TAO system, specifically example use cases of context recognition from the ontological pipeline, details on the dataset characteristics, ground truth annotation, and the real-world study protocol.

B.1 Ontology Details

This section presents the detailed representation of the ontology utilized in our study. The ontology serves as a structured framework that defines the concepts, relationships, and hierarchies within a specific domain. By elucidating the ontology's structure, we aim to provide a comprehensive understanding of how context and activity information are organized and interconnected using concrete examples.

Figure B.1 showcases real activity traces from the ExtraSensory dataset showing different activity patterns, such as (a) sequential and (b) parallel activity patterns. Alongside the activity data, we also show the corresponding ground truth labels and context predictions derived from the ontology. This timeline graph provides insights into the patterns of activities and how they align with the expected context.

Ground-Truth Context Annotation. While some public datasets have context annotations from sensor data, none of these datasets have an activity-to-context mapping. Hence, we created our own context labels using annotated activities from these datasets (from *ExtraSensory* and *Casas*). We first normalized the vocabulary for the activity labels across the datasets using our ontology (*e.g.*, *A: eat breakfast*, *A: eat lunch*, *A: eat dinner* were normalized to *A: Eating*). To bootstrap our context annotation process, we created timestamped activity traces as shown in Figure B.1, which allowed us to visualize the patterns of activities and their correlation to an expected context (without context prediction labels). Using the activity labels, we recognized 14 distinct contexts, and two researchers independently annotated 90 randomly sampled days of activity data across both datasets. Through this process, we not only achieved precise ground truth annotations but also discovered context patterns beyond the existing ontology, enabling us to expand and enhance it.

Ontology Prediction. In Figure B.1, marker ① shows an example of an activity pattern where activity *A: Bathing-Shower* is followed by *A: Dressing*. This sequential transition has been

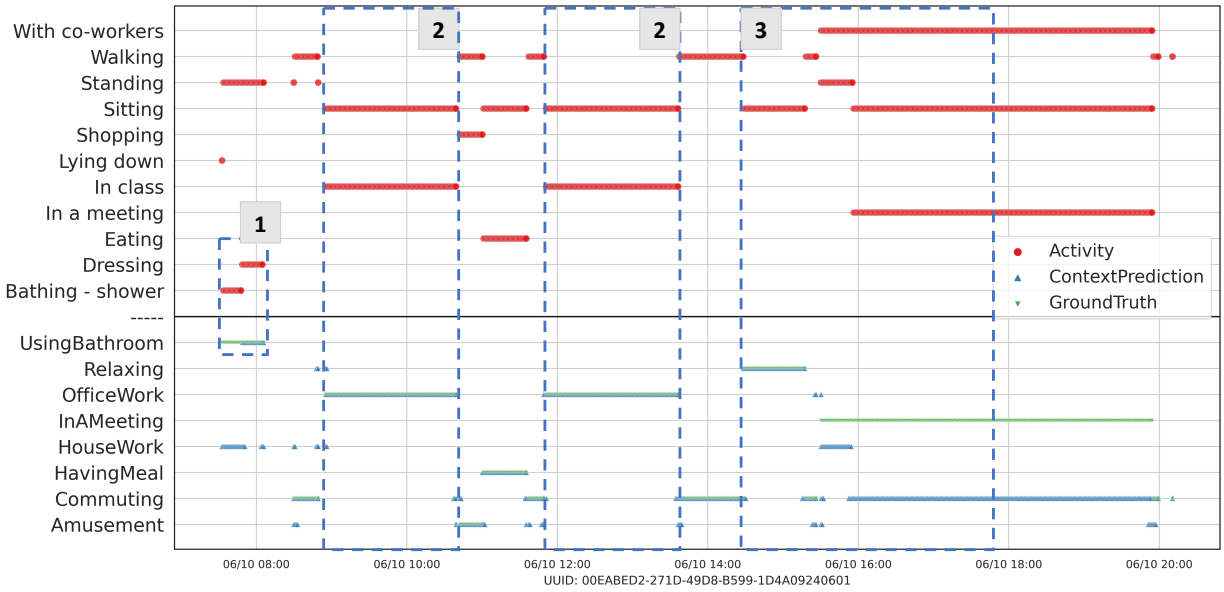


Figure B.1: A real-world timeline of different activities an individual performs from the ExtraSensory dataset and the corresponding ground truth and context predictions from the ontology. The annotations in the graph show different activity patterns, such as (a) sequential and (b) parallel activity patterns.

explicitly modeled in the ontology of TAO, enabling the system to infer the corresponding context *C: Using bathroom* (as depicted in Figure B.2(a)) whenever this sequential activity sequence occurs. We also see the time difference between the ground truth annotation of the *C: Using bathroom* context and the actual model prediction: the ontology model only predicts during the transition and not before it. We see similar cases in the other activity patterns, as shown in marker ③ (*A: walking* to *A: sitting* predicts *C: Commuting* instead of *C: Meeting*). Similarly, marker ② shows a parallel activity pattern where *A: Sitting* is performed simultaneously with *A: inClass*. Our ontology correctly models these activities as *C: Office work* context because of the definition in the ontology (as depicted in Figure B.2(b)).

Overall, this analysis of real activity traces, along with the corresponding ground truth labels and context predictions derived from the ontology, allows us to gain valuable insights into the patterns and relationships between activities and their associated contexts, leading to a deeper understanding of the context-aware capabilities of the ontological pipeline.

B.2 Dataset Characteristics

This section examines the characteristics of the datasets used in our study. By exploring various aspects such as data size, sources, collection methods, and relevant statistical measures, we provide a comprehensive overview of the datasets that TAO uses.

ExtraSensory and Casas. The *ExtraSensory* dataset contains data from 60 participants, with thousands of samples per participant typically taken at intervals of 1 minute (but not necessarily

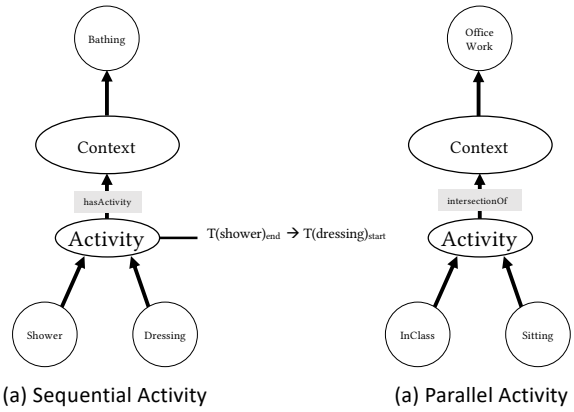


Figure B.2: Examples of sequential and parallel activity pattern definitions in the ontology.

in one long sequence, as there are time gaps). Every sample consists of sensor measurements from personal smartphones and self-reported labels for user activities and location. The *Casas* dataset is collected using ambient sensors deployed in 30 different smart homes and has 1.5 month of manually labeled activity information per home. We filtered activity annotations from these datasets to create multiple timestamped activity streams (at participant level in *ExtraSensory* and home level in *Casas*). We identified 14 different contexts from these activity labels. To evaluate the accuracy of our pipeline, we hand-annotated 90 days (selected randomly) of data across both datasets (more details in Table B.1).

Table B.1: Summary of the HAR datasets used for evaluation and their characteristics.

Dataset	**Sensors**	**Total Users**	**Total Data (days)**	**Annotated Activity Data (hours)**	**Total Unique Activities**
ExtraSensory [207]	IMU, audio, location	60	471	86	34
Casas [55]	Multimodal	30	2,934	2,850	42

B.3 Detailed Scenarios for Real-World Study

In Table B.2, we show details for the four scenarios selected for our real-world study. All participants were asked to perform each of these scenarios and were not given any instruction on the duration of each activity; we allowed them to complete the activity patterns at their own pace.

B.4 Details on Ground Truth Annotations

Table B.3 shows a set of example activities from the *ExtraSensory* dataset and the corresponding manual annotation for ground truth context.

Table B.2: Overview of the scenarios used in the real-world user study. We identify several common activity patterns in daily life from which context can be inferred and categorize them into different scenarios.

Scenario	**Activity Patterns**	**Context**
Scenario 1 (Sequential activity patterns)	*A: door knock* followed by *A: door open*	*C: ComingIn*
	A: phone ring followed by *A: talking*	*C: On a phone call*
	A: sweeping followed by *A: vacuum*	*C: House work*
Scenario 2 (Sequential and parallel activity patterns)	*A: eating* followed by *A: typing*	*C: Having meal*, *C: Office work*
	A: coffee grinding, *A: mouse click*, *A: typing*	*C: Preparing meal*, *C: Office work*
	A: phone ring, *A: talking*, *A: sweeping*, *A: coffee grinding*	*C: Phone call*, *C: House work*, *C: Preparing meal*
Scenario 3 (Interleaved activity patterns for single context)	*A: jumping*, *A: running*	*C: Exercising*
	A: mouse click, *A: typing*, *A: writing*	*C: Office work*
	A: running, *A: jumping*	*C: Exercising*
	A: typing, *A: mouse click*, *A: writing*	*C: Office work*
Scenario 4 (Interleaved and parallel activity patterns for multiple contexts)	*A: tv*, *A: talking* and *A: typing*	*C: Relaxing*, *C: Office work*
	A: eating, *A: mouse click*, *A: tv*	*C: Having meal*, *C: Office work*, *C: Relaxing*
	A: vacuum, *A: talking*	*C: House work*, *C: Office work*

Table B.4 presents a summary of our ground truth annotations, including 14 unique contexts annotated across the *ExtraSensory* and *Casas* datasets and the amount of data annotated for each context in terms of total time (in hours) and the number of unique instances (*i.e.*, time intervals) for each context.

Table B.3: Example activities from the *ExtraSensory* dataset [207]. We manually annotate the context for these activities during our ground truth labeling process.

Activity	**Context**
A: talking, *A: computer work*	*C: In a meeting*, *C: Office work*
A: jogging, *A: cycling*, *A: running*	*C: Exercise*
A: watching tv, *A: talking*	*C: Relaxing*

Table B.4: Amount of data annotated for ground truth of 14 different contexts across the *ExtraSensory* and *Casas* datasets.

Context	**Total Time (hours)**	**Instance Count**
Amusement	82.90	45
ComingIn	0.44	88
Commuting	40.33	108
Exercising	6.27	20
GoingOut	1.33	107
HavingMeal	58.52	218
HouseWork	27.94	89
InAMeeting	48.97	55
OfficeWork	96.39	112
PhoneCall	3.30	21
PreparingMeal	20.27	87
Relaxing	258.69	265
Sleeping	561.52	203
UsingBathroom	82.43	382

Appendix C

Supplementary Material for BuildingChat

C.1 LLM Prompts

The LLM prompts used by BuildingChat are available at `https://osf.io/4j5gq/overview?view_only=af3af1549b1b4a4c834a743e76d40795`.

C.2 Formative Study Interview Script

Introduction and Consent

My name is [interviewer name]. I'm a [title] at [institution]. Thank you for your interest in our research. As a refresher, we are going to have a discussion together about your experiences with office buildings, and talk about some hypothetical scenarios.

Before we get into the interview, we wanted to first go over a consent form with you. Take some time to review the form here: [give participant consent form] Do you have any questions about this form? [Answer questions]

I just want to highlight a few key points in that form. Our conversation will be 60 minutes. I will be asking you some questions about your day-to-day experiences with office buildings, and then we'll dig deeper into some hypothetical scenarios. I will be taking some notes about our discussion in my notebook, and I will also be using [either zoom or device] to record our discussion so that our research team can refer to it later. I will not write down any sensitive information, and we will remove sensitive information from our transcript. However, to the extent that you are able to, please avoid revealing sensitive information in this conversation. When we are done with our conversation, you will receive a $15 Amazon gift card. Do you have any other questions before we begin [Answer questions]

Are you still interested in participating in this interview? Are you okay with me taking notes? Are you okay with the interview audio being recorded? [If agree, start recording]

Establishing Persona

[All participants except emergency managers] First, I'm going to ask you some questions about your experience with this building and what your relationship is with it. How long have you been coming to this building? Is coming to this building a part of your day-to-day routine?

What does a typical day look like for you in this building? What kinds of spaces do you typically occupy within this building? What frustrations do you have with this building and with the spaces that you occupy during your typical day? Some examples might be, "the temperature in my office fluctuates frequently" or "my floor is really loud in the morning"

[Emergency managers] For what community do you manage emergencies for? What kinds of emergencies occur at buildings typically that you manage? When you arrive on scene, what kinds of information do you look for? What would be useful? How do you currently learn about the building information that you need? Would any other information be useful to you?

Pain Point Deep Dive

[Select 1–2 pain points described by the participant. For each pain point, ask the following] You mentioned [pain point]. Let's dive deeper into the challenges that you face with this. How often does [pain point] happen? Are there any specific times, or indicators that [pain point] is more likely to occur? Can you give a specific example of when/how [pain point] happened? Describe the play-by-play of how this happened. Who else seems to have this same problem? Are there ways that you think you experience [pain point] differently than other people in the building?

Introducing Smart Building Concept

Now, I'd like to explore the concept of having a more "intelligent" building with you. Imagine this building could "know" things about how the spaces and facilities are being used, and share that information with the occupants.

The building's intelligence is informed by sensors placed around the building. These sensors are able to capture information such as loudness and sounds (but not speech), temperature, and motion. There are multiple sensors in each room of the building, as well as public spaces.

Imagine that this system has the following boundaries: (1) for privacy, the system can't track specific individuals, or know who you are, (2) it filters raw audio data to remove speech and cannot identify specific people. To help solidify this building concept, I will provide the following example: Two sensors are placed on the walls and ceiling, and can detect changes in the lighting, temperature, and sound. You can ask the question "what room has the brightest lighting?", and the system may answer, "room 345". You can ask the question, "how productive was I from 3–4", then the system may answer "you worked for 45 minutes in the past hour"

[All participants] Imagine that you are going through your typical interactions with this building. What would make the day go smoother?

[All participants] Imagine that you could ask the building questions through an app. What kinds of questions would you want to ask?

[If emergency manager] In time-sensitive scenarios, would you want to use an app such as this? What kinds of information would be useful to access?

[All participants] What (if anything) would you want the building to tell you proactively?

[If building occupant, administrator, or facilities staff member] How might your daily routine change if your building was "smart"? How might a smart building make you more productive/-comfortable using the spaces that you have access to? If you were a visitor in a new building, what kinds of information would you want to know? What building insights would be useful to you?

[If the person has their own space in the building] As someone who has your own space here,

what would you want to know about your specific space? What would you want to know about the building beyond your own space?

[If the person is an administrator] What kinds of responsibilities do you hold/what does your job entail? From your experience managing [department/resource], what building insights would help your decision-making?

[If the person is a facilities staff member] What kinds of responsibilities do you hold/what does your job entail? From your experiences maintaining/supporting this building, what would help you be more effective?

Exploring Boundaries and Privacy Concerns

Now, I'm going to ask you some questions about what you would NOT want from a smart building.

What concerns would you have in general with a smart building system? Based on our discussion so far, what parts of a smart building system would make you uncomfortable or uneasy? What kinds of information would you be ok with the building knowing about you versus not? Would you want control over when and how the building interacts with you? What would this control look like?

Wrap Up

Thank you very much for participating in our discussion today! Just a few last questions as we wrap things up: Do you have anything else that you would like to tell me? How was this conversation for you? Did anything that we discussed surprise you? Are there others that we should talk to who have different experiences with your building? As a reminder, we will follow up with the compensation and send you the gift card as soon as possible.

C.3 Template Library Details

Table **??** summarizes the six templates in BuildingChat's library. This appendix describes each template's function and the formative study pattern it addresses. `current_measurement` addresses point-in-time queries (*e.g.*, "What is the temperature on floor 3?"). It extracts current values from sensor data with configurable aggregation and produces a `measurement` packet. `spatial_compare` addresses cross-location queries (*e.g.*, "Which floor is warmer?"). It takes `measurement` packets from multiple locations and produces an `analysis_result` with rankings and margins. `temporal_analysis` addresses trend queries (*e.g.*, "How has the temperature changed today?"). It analyzes time-series data to identify directional changes, peaks, and statistical significance, producing both a `time_series` and an `analysis_result`. `threshold_filter` addresses conditional queries (*e.g.*, "Which rooms are above 75°F?"). It filters locations based on whether measurements meet specified criteria, producing a `location_set`. `web_search` retrieves external context such as current weather conditions. Unlike the analytical templates, it takes no sensor input packets and produces an `analysis_result` directly. For queries like "Is it warmer inside than outside?", the DAG contains parallel branches: one processing building sensors and one retrieving external data, with the response generator synthesizing both into a coherent answer. `info_lookup` handles informational queries about the building itself, such as room capacities, operating hours, or available sensor modalities. Like `web_search`, it takes no sensor input packets and routes through building-specific documentation.

C.4 Test Query Generation Details

This appendix describes the test query generation procedure in detail.

C.4.1 Generation Procedure

We generate each test case in three phases: parameter sampling, ground truth computation, and natural language query generation.

Phase 1: Parameter Sampling. For each query, we sample structured parameters from the mock building's validated topology. We draw location tags from predefined combinations that exist in the building (*e.g.*, {`floor: 3, locationType: office`}, {`location: 310`}). We sample sensor modalities from the 11 available types (temperature, humidity, light, microphone, barometer, accelerometer, color, magnetometer, grideye, motion, wifi). We sample time specifications in one of five formats: relative point (*e.g.*, "2 hours ago"), relative range (*e.g.*, "over the last 3 hours"), absolute point (*e.g.*, "at 14:30 on Jan 5"), absolute range (*e.g.*, "from 14:00 to 18:00 on Jan 5"), or current ("right now"). We validate that all sampled times fall within the mock data range. For comparison queries, we sample a second location-time pair, ensuring the two operands differ meaningfully (different location or at least two hours apart). For filter queries, we additionally sample a threshold value and comparison operator.

Phase 2: Ground Truth Computation. We compute the ground truth answer deterministically from the mock data using manually constructed procedures for each query type. For measurement queries, we compute a sensor-weighted mean across all spaces matching the location tags. For comparison queries, we compute the aggregated value for each operand and determine which is higher or lower. For filter queries, we count spaces whose average sensor value meets the threshold condition. For trend queries, we fit a linear regression over the time window and report the direction (increasing, decreasing, or stable) along with statistical significance. We discard and resample test cases (up to 5 attempts) where the ground truth is missing or ambiguous. For comparison queries, we additionally discard cases where the two values differ by less than 10% of the maximum, avoiding near-equal comparisons that would be ambiguous to evaluate.

Phase 3: Natural Language Generation. We use two strategies depending on query type. For boolean comparison queries (*e.g.*, "Is the temperature in room 310 warmer than in room 350?"), we use deterministic templates that combine LLM-generated operand descriptions with sensor-appropriate comparison phrases (*e.g.*, "warmer than" for temperature, "stronger than" for WiFi signal). This avoids the risk of an LLM inverting the comparison direction, which would make the ground truth label incorrect. For all other query types, we prompt the LLM (Qwen3-235B) with the sampled parameters to generate the query text. Each prompt includes the operation type, sensor display name, location description, time specification, and expected tags. The prompts enforce several constraints to ensure query quality: (a) queries must use exact sensor names rather than vague paraphrases (*e.g.*, "PIR motion reading" rather than "motion activity"), (b) time references must be unambiguous (ranges must specify both start and end), (c) the LLM

must not invent location details beyond what appears in the sampled tags, and (d) queries must request measurable quantities rather than yes/no formulations (for measurement queries). Each prompt includes 4–6 worked examples demonstrating correct formatting. We manually review all generated query texts and adjust them to remove remaining ambiguities.

C.4.2 Representative Prompt

Below is the prompt template used for measurement queries. Other query types (comparison, trend, filter) follow the same structure with type-specific instructions and examples.

```
Generate a natural language query for a building
sensor system.

Context:
- Operation: MEASURE (single measurement query)
- Sensor type: {sensor_display}
- Location: {location}
- Time: {time}
- Expected tags: {tags}

The query should:
1. Clearly specify the location
2. Clearly specify the sensor using the EXACT name
3. Clearly specify the time
4. Be natural and conversational
5. Use ONLY location information from the provided
   Tags | do NOT invent details
6. Phrase as a numeric measurement question, NOT a
   yes/no question

[4-6 worked examples omitted for brevity]

Generate a single, clear query question. Do not
include any explanation, just the query itself.

Query:
```

C.4.3 Example Test Cases

Table C.1 shows representative generated test cases across the four primary query categories.

C.5 Design Iteration Study Interview Script

This script was used for the initial interview study which informed further design iteration of BuildingChat.

Table C.1: Example test cases from the evaluation set, showing sampled parameters, computed ground truth, and generated natural language query for each category.

Category	Parameters	Time Spec	Ground Truth	Generated Query
Measure	sensor: temperature, tags: {floor: 3}	relative range: past 1 hour	72.4°F (mean)	"What was the average temperature on floor 3 over the past hour?"
Compare	sensor: humidity, $tags_1$: {location: 310}, $tags_2$: {location: 350}	both: current point	Location 1 higher (61.2% vs 54.8%)	"How does the humidity in room 310 right now compare to the humidity in room 350 right now?"
Filter	sensor: temperature, tags: {floor: 2}, threshold: > 75°F	relative range: past 3 hours	4 rooms	"How many rooms on Floor 2 had an average temperature above 75°F over the past 3 hours?"
Trend	sensor: light, tags: {locationType: office, cardinality: NE}	relative range: today	increasing ($p < 0.05$)	"What's the light level trend in NE offices today?"

C.5.1 Introduction and Consent

My name is [interviewer name]. I'm a [title] at [institution]. Thank you for your interest in our research. As a refresher, we are going to have a discussion together about our building chatbot called BuildingChat.

Before we get into the interview, we wanted to first go over a consent form with you. Take some time to review the form here: [give participant consent form] Do you have any questions about this form? [Answer questions]

I just want to highlight a few key points in that form. Our conversation today will be 60 minutes. In this discussion, we want you to test out our app, BuildingChat, which enables users to ask questions of their building using natural language. We will set up the BuildingChat app, explore the questions that you might have using the BuildingChat app, and ask you to generate more questions using theoretical scenarios or situations. We'll then ask you to describe your experience with using the app. You will be compensated $20 for your time today.

After our conversation, you will have the opportunity to use BuildingChat for a week as much as you wish.

Finally, we will set up a 30-minute interview for you to tell us how your experience was with the week-long usage of BuildingChat. You will be compensated $10 for your time.

For our conversation today, I will be taking some notes about our discussion in my notebook, and I will also be recording our discussion so that our research team can refer to it later. I will not write down any sensitive information, and we will remove sensitive information from our transcript. However, to the extent that you are able to, please avoid revealing sensitive information

in this conversation. Do you have any other questions before we begin [Answer questions]

Are you still interested in participating in this interview? Are you okay with me taking notes? Are you okay with the interview audio being recorded? [If agree, start recording]

C.5.2 Establishing Persona

First, I'm going to ask you some questions about your experience with this building and what your relationship is with it. How long have you been coming to this building? Is coming to this building a part of your day-to-day routine? What does a typical day look like for you in this building? What kinds of spaces do you typically occupy within this building?

C.5.3 Set up BuildingChat

Now, we're going to have you install our BuildingChat app. BuildingChat is an application that allows you to ask the building questions in natural language, and receive responses based on sensor-based insights. [Direct participant to log into webapp and create account]

Let me go through a few more points of information of what kinds of sensor-insights are available in our building, and give you an example of what kinds of questions you can ask. [Participant is shown flyer with information about sensors installed in building]

With BuildingChat, you can ask questions based on the insights from these sensors. For example, you can see that there are temperature sensors available, so a question that I can ask BuildingChat is "How hot is my office?" Additionally note that for this study, we are using Qwen 3. We are running Qwen3 on the cloud. Note that no sensor data is sent to the cloud. In our final deployment, we plan to run the models locally. Note that BuildingChat will also tell you when it needs more information to answer a question, if you do not have authorization to access the information, or if the information doesn't exist. In these scenarios, the system will tell you why it cannot provide this information. It will additionally help you refine your query to a related query that is authorized or that has information available.

C.5.4 Walking Through Imagined Scenarios

Now, we will explore some sample questions that might come that can be answered by BuildingChat. Now, with these questions, try asking BuildingChat a few questions. We'll be here to help, but please think aloud about any responses that you get from BuildingChat, and how you formulate your queries.

This is a more rudimentary chatbot than most modern LLM chatbots, but it is the first to our knowledge that utilizes sensor insights from buildings to answer questions in natural language. So, please be a bit patient with the system! You can have back and forth interactions with the chatbot. We recommend that if you are asking follow up questions, or questions about the same topic, you can use one chat, but if you decide to switch topics, we recommend using a new chat. For example, if you are having a discussion with BuildingChat about the temperature of your office, and want to switch to asking about the humidity of a conference room, we recommend that you use a new chat for the second query.

[Interviewer selects five scenarios to discuss (Section C.7), with three scenarios known to be supported by BuildingChat, one scenario that requests inaccessible private data, and another scenario that requests data that is not available. For each scenario, ask participant to enter questions into BuildingChat and explain their reactions as they review response and follow-ups as appropriate.]

C.5.5 Overall Reactions and Conclusion

For each query that you asked, did you feel that BuildingChat's response contained the information that you wanted from this query? If BuildingChat asked a clarification question, did you feel that you were able to easily understand why you needed to clarify the question that you initially asked? Were you able to understand how to clarify the question that you initially asked?

How was this experience for you? What did you learn about your space and your building? What felt challenging about using our tool? Would you recommend using this tool to other people in the building?

Thank you very much for participating in our discussion today! Just a few last questions as we wrap things up: Do you have anything else that you would like to tell me? As a reminder, we will follow up with the compensation and send you the gift card as soon as possible. Feel free to email us for any more clarifications and questions that you might have!

C.6 Usability Evaluation Interview Script

This script was used for the interview study to evaluate the usability of BuildingChat.

C.6.1 Introduction and Consent

My name is [interviewer name]. I'm a [title] at [institution]. Thank you for your interest in our research. As a refresher, we are going to have a discussion together about our building chatbot called BuildingChat.

Before we get into the interview, were you able to fill the consent form? If not, please take a moment to review and sign the consent form now. Did you have any questions about this consent form? [Answer questions]

Our conversation today will be 1 hour long. The goal of this interview is for you to tell us how your experience was with the three day-long usage of BuildingChat. You will be compensated $30 for your time.

For our conversation today, I will be taking some notes about our discussion in my notebook, and I will also be using zoom to record our discussion so that our research team can refer to it later. I will not write down any sensitive information, and we will remove sensitive information from our transcript. However, to the extent that you are able to, please avoid revealing sensitive information in this conversation. Do you have any other questions before we begin? [Answer questions]

Are you still interested in participating in this interview? [**Yes**/No; if yes continue] Are you okay with me taking notes? [Yes/No; if yes continue, if no, say "that's ok, we can proceed without

the note-taking" (and then proceed without note taking)] Are you okay with the interview audio being recorded? [Yes/No; if yes continue] Ok, I am going to start recording. As a reminder, please do not share anything private or sensitive about yourself or anyone else. [Turn on recording] Ok, we are recording now.

C.6.2 Establishing Persona

First, I'm going to ask you some questions about your experience with this building and what your relationship is with it. How long have you been coming to this building? Is coming to this building a part of your day-to-day routine? What does a typical day look like for you in this building? What kinds of spaces do you typically occupy within this building?

[As you discuss, ask follow up questions for clarifications]

C.6.3 Questions about Overall Experience and Insights

First, we're going to have you look back at your past 3 days. How much did you use BuildingChat?

[If used system at least once] Check your previous chats to look at what you've asked before.

How was your overall experience using it? What kinds of questions did you find yourself asking? Were you more curious about a specific type of space like huddle rooms? What did you learn? How did you come across this information (i.e, which kinds of queries revealed this information)?

[If used system not at all] Are there any specific reasons you did not use it?

C.6.4 Using BuildingChat for Specific Scenarios

Interviewer selects three scenarios to discuss, with one scenario known to be supported by BuildingChat, one scenario that requests inaccessible private data, and another scenario that requests data that is not available. For each scenario, ask participant to enter questions into BuildingChat and explain their reactions as they review response and follow-ups as appropriate.

C.6.5 Using BuildingChat for User-Generated Questions

Now, we'd like you to ask questions of your own choosing. Can you open it up on your computer and think about what else you might want to ask? We'll have you ask 2–3 questions.

[Participant enters questions into BuildingChat, approx. 5 minutes]

[If no ideas, then suggest following]:

Try to think of a question that you would like to ask BuildingChat. To give you some ideas of questions, try to think about the following prompts:

Imagine that you are going through your typical interactions with this building. What would make the day go smoother?

Imagine that you could ask the building questions through an app. How would your interactions change?

[If they still don't have ideas, suggest something from list of questions]

C.6.6 Questions about Usability

Now, we're going to have you answer questions about your overall experience using BuildingChat.

- Did you feel that BuildingChat was easy to use and intuitive?
- Was there a learning curve?
- What was your process of learning how to use BuildingChat?
- Did you feel that the responses that you got from BuildingChat were reliable? [If no] Can you recall a question where you felt that it wasn't reliable and explain why you felt that it wasn't reliable?
- Did you feel that the information that you received was sufficient?
- Did you feel that BuildingChat answered your questions with the appropriate level of detail?
- Was the response understandable and useful? Were you able to utilize the information?

Now, we're going to ask you about the quality of the responses that you received.

Did you feel that, overall, the intention of your question was well understood? [If Understood] Can you recall an example of instances where your intention was well understood, [If not understood] Can you recall an example of instances when it was not

Did you encounter scenarios where BuildingChat suggested an alternative query? Did you feel like the clarification for the query change was understandable? Did you understand why the clarification was needed?

C.6.7 Questions about Scenarios Where BuildingChat Cannot Answer Questions

When you have asked questions to BuildingChat, were there any questions it said it couldn't answer because the system doesn't have the ability (either knowledge about the building or sensor information) to answer it? [If yes]

- Do you remember which questions? And did you expect it would work?
- Do you think the system should be able to answer this type of question?

[If No] continue...

When you have asked questions to BuildingChat, were there any questions it said it couldn't answer because you had unauthorized access? [If yes]

- Do you remember which questions?
- Did you expect that you would not be authorized to access that information?
- Do you think that policy is appropriate?

[If no, they think is appropriate] Would you want other people to be able to ask that question?

Are there any things you want to ask BuildingChat that you don't think it can do yet?

What features do you wish you had as a part of BuildingChat? What features would benefit BuildingChat and you as a user?

Please fill out this final survey on your overall user experience of BuildingChat!

C.6.8 SUS Survey

Survey is completed on laptop. All questions are on a 5-point Likert scale, strongly disagree to strongly agree.

- I think that I would like to use this system frequently
- I found the system unnecessarily complex
- I thought the system was easy to use
- I think that I would need the support of a technical person to be able to use this system
- I found the various functions in this system were well integrated
- I thought there was too much inconsistency in this system
- I would imagine that most people would learn to use this system very quickly
- I found the system very cumbersome to use
- I felt very confident using the system
- I needed to learn a lot of things before I could get going with this system

C.6.9 Wrap-Up

Thank you very much for participating in our discussion today! Just a few last questions as we wrap things up: Do you have anything else that you would like to tell me? As a reminder, we will follow up with the compensation and send you the gift card in the next few weeks. Feel free to email us for any more clarifications and questions that you might have!

C.7 Usability Evaluation Hypothetical Scenarios

This appendix includes the hypothetical scenarios used in both the design iteration study (Section 6.5) and the usability evaluation (Section 6.7).

We first created a list of 41 questions that one could ask BuildingChat. We generated this list from the formative study, as well as questions generated by the research team. The questions were designed to cover a diverse range of intentions and sensor types. We additionally included questions that would violate privacy boundaries and questions that would involve data that does not exist.

From this list of questions, we generated a list of hypothetical scenarios by prompting OpenAI's GPT-5 model to write a hypothetical scenario around each prompt. We then adjusted the imagined scenarios to remove hallucinated information or context. The scenarios that we utilized for users are listed in Table C.2.

Table C.2: Scenarios discussed in the design iteration study and usability evaluation interviews. Each scenario was presented to participants, who then formulated their own queries to BuildingChat. The "Expected Response" column indicates whether the scenario was designed to be answerable (Allowed), to involve private data the user should not access (Private data), or to request information not available in the system (No data).

Scenario	**Expected Response**
Imagine that it's a cold winter morning, you're feeling sluggish at your desk, and you want to relocate to a brighter, warmer spot in the building to work more comfortably.	Allowed
Imagine that you're about to join an important video call and need to quickly find a private room that isn't already occupied.	Allowed
Imagine that you've just arrived at the office for the day and want to choose a workspace that best fits your tasks, whether that's quiet focus or collaboration.	Allowed
Imagine that your office feels unusually warm and you're trying to figure out whether this is a building-wide issue or something specific to your room.	Allowed
Imagine that you need to have a quick in-person discussion with collaborator X and want to know if they're currently in the building.	Private data
Imagine that you're sensitive to odors and are deciding whether to spend the afternoon working on floor 3.	No data
Imagine that it's lunchtime and you want to heat up your food without waiting in line at a busy microwave.	No data
Imagine that you're planning your day and want to avoid areas that might be closed due to maintenance or unexpected issues.	No data
Imagine that you're about to move into room 347 for a meeting and want to check if the temperature will be comfortable.	Private data
Imagine that you see unusual activity near Amanda's office and want to know if there's a meeting, event, or issue going on.	Private data
Imagine that you want a general overview of current building activity.	Allowed
Imagine that you're looking for a quiet place to focus and want to avoid the noisiest floor in the building.	Allowed

Continued on next page

Scenario	Expected Response
Imagine that you're visiting the office for the first time and need directions to find X person's office quickly.	No data
Imagine that you're scheduling a meeting and want to check whether your favorite conference rooms are free.	Allowed
Imagine that you're standing outside the TCS conference room and want to know if it's currently occupied before entering.	Allowed
Imagine that you're feeling uncomfortable at your desk and want confirmation on whether your office temperature is above normal.	Allowed
Imagine that your friend's office is office 347 and you have noticed it feels cold every time you're there.	Private data
Imagine that you spend a lot of time working in the lab and want to know if its low temperature is a consistent pattern.	Allowed
Imagine that you're trying to avoid peak crowd times and want to know when the building is usually busiest.	Allowed
Imagine that you're feeling dehydrated and want to know whether there's cold drinking water near area X.	No data
Imagine that you're planning focused work and want to confirm whether a specific space is quiet at a certain time.	Allowed
Imagine that you want to avoid distractions and are trying to identify when space X tends to get loud.	Allowed
Imagine that you're looking for underutilized areas to work and want to know which spaces are already heavily used.	Allowed
Imagine that you want to plan meetings during less busy times and need to know peak occupancy periods.	Allowed
Imagine that your team needs a quick informal meeting and you're trying to find an available huddle space nearby.	Allowed
Imagine that you're new to the building and want to know if there are small collaboration spaces available.	Allowed
Imagine that you often work late and want to understand how the building is typically used after normal office hours.	Allowed
Imagine that you're reflecting on your work habits and want insight into how much time you actually spend at your desk.	Allowed

Continued on next page

Scenario	Expected Response
Imagine that you're deciding whether to step outside for a break and want to compare outdoor conditions to the indoor temperature.	Allowed
Imagine that you're monitoring comfort conditions and want to know if the NE corridor has been gradually heating up.	Allowed
Imagine that you're cold after sitting still for hours and want to relocate to the warmest area on floor 3.	Allowed
Imagine that you noticed temperature fluctuations while walking through the corridor and want to understand whether it's been consistently warm or cold recently.	Allowed
Imagine that you're planning to work in the NE corridor area on floor 3 and want to know how comfortable the temperature is right now.	Allowed
Imagine that the air feels unusually dry or sticky and you want to check the current humidity level.	Allowed
Imagine that you're planning a task that requires good lighting and want to confirm the brightness in room 347.	Private data
Imagine that you're considering using that conference room for a call and want to know if noise will be an issue.	Allowed
Imagine that you're adjusting your clothing layers and want to know the exact temperature at your desk.	Allowed
Imagine that you're reviewing a reported disturbance and want to understand how loud the corridor was at that time.	Allowed
Imagine that facilities management asked you to check whether temperatures were within range earlier in the day.	Allowed
Imagine that you're analyzing historical lighting conditions in room 347 on Jan 2nd.	Allowed
Imagine that you're trying to figure out the average temperature in the NE corridor over the last hour.	Allowed

Bibliography

[1] Longitude Prize on Dementia, . URL https://challengeworks.org/challenge-prizes/the-longitude-prize-on-dementia/. 7.1.1

[2] Qwen Studio, . URL https://qwen.ai/blog?id=qwen2.5-vl. 7.3.1

[3] SmolVLM - small yet mighty Vision Language Model, November 2024. URL https://huggingface.co/blog/smolvlm. 7.3.1

[4] Anders Peter Aavild, Aleksander Rosenkrantz de Lasson, Christian Moesgaard Andersen, Erik Christensen, Sergio Moreschini, David Hästbacka, Davide Taibi, and Michele Albano. Distributed Home Automation with Home Assistant. In *Proceedings of the 14th International Conference on the Internet of Things*, IoT '24, pages 206–212, New York, NY, USA, March 2025. Association for Computing Machinery. ISBN 979-8-4007-1285-2. doi: 10.1145/3703790.3703828. URL https://dl.acm.org/doi/10.1145/3703790.3703828. 2.5.1

[5] Noura Abdi, Kopo M. Ramokapane, and Jose M. Such. More than smart speakers: Security and privacy perceptions of smart home personal assistants. In *Fifteenth Symposium on Usable Privacy and Security (SOUPS 2019)*, pages 451–466, Santa Clara, CA, August 2019. USENIX Association. ISBN 978-1-939133-05-2. URL https://www.usenix.org/conference/soups2019/presentation/abdi. 1

[6] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark, 2016. 3.3.2

[7] Matheus Gabriel Acorsi, Leandro Maria Gimenez, and Maurício Martello. Assessing the performance of a low-cost thermal camera in proximal and aerial conditions. *Remote Sensing*, 12(21):3591, 2020. 3.2

[8] Suli Adeniye, Faisal Al-Atawi, and Arunabha Sen. Natural language interface for queries on databases with sensitive information. 06 2025. doi: 10.1109/ICAD65464.2025.11114052. 6.2.2

[9] Alessandra Agostini, Claudio Bettini, and Daniele Riboni. Hybrid reasoning in the care middleware for context awareness. *International journal of Web engineering and technology*, 5 (1):3–23, 2009. 2.4.1

[10] Antonio A Aguileta, Ramon F Brena, Oscar Mayora, Erik Molino-Minero-Re, and Luis A Trejo. Multi-sensor fusion for activity recognition—a survey. *Sensors*, 19(17):3808, 2019. 2.1.2, 2.2.2

[11] Karan Ahuja, Yue Jiang, Mayank Goel, and Chris Harrison. Vid2doppler: Synthesizing doppler radar data from videos for training privacy-preserving activity recognition. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450380966. doi: 10.1145/3411764.3445138. URL https://doi.org/10.1145/3411764.3445138. 1, 1.1, 2.2.3, 3.1, 3.3.4, 4.1

[12] Ane Alberdi, Asier Aztiria, and Adrian Basarab. Towards an automatic early stress recognition system for office environments based on multimodal measurements: A review. *Journal of biomedical informatics*, 59:49–75, 2016. 5.2

[13] Riku Arakawa, Hiromu Yakura, Vimal Mollyn, Suzanne Nie, Emma Russell, Dustin P. DeMeo, Haarika A. Reddy, Alexander K. Maytin, Bryan T. Carroll, Jill Fain Lehman, and Mayank Goel. PrISM-Tracker: A Framework for Multimodal Procedure Tracking Using Wearable Sensors and State Transition Information with User-Driven Handling of Errors and Uncertainty. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 6(4): 156:1–156:27, January 2023. doi: 10.1145/3569504. URL https://dl.acm.org/doi/10.1145/3569504. 4.1

[14] Riku Arakawa, Jill Fain Lehman, and Mayank Goel. PrISM-Q&A: Step-Aware Voice Assistant on a Smartwatch Enabled by Multimodal Procedure Tracking and Large Language Models. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 8(4):180:1–180:26, November 2024. doi: 10.1145/3699759. URL https://dl.acm.org/doi/10.1145/3699759. 4.1

[15] Riku Arakawa, Hiromu Yakura, and Mayank Goel. PrISM-Observer: Intervention Agent to Help Users Perform Everyday Procedures Sensed using a Smartwatch. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*, UIST '24, pages 1–16, New York, NY, USA, October 2024. Association for Computing Machinery. ISBN 979-8-4007-0628-8. doi: 10.1145/3654777.3676350. URL https://dl.acm.org/doi/10.1145/3654777.3676350. 4.1

[16] Riku Arakawa, Prasoon Patidar, Will Page, Jill Lehman, and Mayank Goel. Scaling context-aware task assistants that learn from demonstration and adapt through mixed-initiative dialogue. In *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology*, UIST '25, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400720376. doi: 10.1145/3746059.3747700. URL https://doi.org/10.1145/3746059.3747700. 4.1, 4.6.2, 7.3.6

[17] Paola Ariza Colpas, Enrico Vicario, Emiro De-La-Hoz-Franco, Marlon Pineres-Melo, Ana Oviedo-Carrascal, and Fulvio Patara. Unsupervised Human Activity Recognition Using the Clustering Approach: A Review. *Sensors*, 20(9):2702, January 2020. ISSN 1424-8220. doi: 10.3390/s20092702. URL https://www.mdpi.com/1424-8220/20/9/2702. Number: 9 Publisher: Multidisciplinary Digital Publishing Institute. 1.1, 2.2.3, 4.1

[18] Luca Arrotta, Claudio Bettini, Gabriele Civitarese, and Michele Fiori. ContextGPT: Infusing LLMs Knowledge into Neuro-Symbolic Activity Recognition Models. In *Companion of the 2024 on ACM International Joint Conference on Pervasive and Ubiqui-*

tous Computing, pages 881–884, October 2024. doi: 10.1145/3675094.3679000. URL http://arxiv.org/abs/2403.06586. arXiv:2403.06586 [cs]. 2.3.1

[19] India Ashok. Hackers leave finnish residents cold after ddos attack knocks out heating systems. https://www.ibtimes.co.uk/hackers-leave-finnish-residents-cold-after-ddos-attack-knocks-out-heat November 2016. 1

[20] Louis Atallah and Guang-Zhong Yang. The use of pervasive sensing for behaviour profiling—a survey. *Pervasive and mobile computing*, 5(5):447–464, 2009. 2.4.2

[21] Autonomous. AUTONOMOUS; Co-Designing Independence — autonomous-project.com. https://www.autonomous-project.com/, 2024. [Accessed 10-10-2025]. 1.2, 4.6.2, 7.1.1

[22] Awan-Ur-Rahman. Understanding Soft Voting and Hard Voting: A Comparative Analysis of Ensemble Learning Methods, August 2023. URL https://medium.com/@awanurrahman.cse/understanding-soft-voting-and-hard-voting-a-comparative-analysis-of-er 4.3.5

[23] Eugene Bagdasarian, Ren Yi, Sahra Ghalebikesabi, Peter Kairouz, Marco Gruteser, Sewoong Oh, Borja Balle, and Daniel Ramage. AirGapAgent: Protecting Privacy-Conscious Conversational Agents. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, CCS '24, December 2024. ISBN 979-8-4007-0636-3. doi: 10.1145/3658644.3690350. URL https://dl.acm.org/doi/10.1145/3658644.3690350. 6.2.2

[24] Bharathan Balaji, Arka Bhattacharya, Gabriel Fierro, Jingkun Gao, Joshua Gluck, Dezhi Hong, Aslak Johansen, Jason Koh, Joern Ploennigs, Yuvraj Agarwal, Mario Berges, David Culler, Rajesh Gupta, Mikkel Baun Kjærgaard, Mani Srivastava, and Kamin Whitehouse. Brick: Towards a unified metadata schema for buildings. In *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments*, BuildSys '16, 2016. ISBN 9781450342643. doi: 10.1145/2993422.2993577. URL https://doi.org/10.1145/2993422.2993577. 2.1.1, 2.5.1, 6.8.3

[25] Oresti Banos, Juan-Manuel Galvez, Miguel Damas, Hector Pomares, and Ignacio Rojas. Window Size Impact in Human Activity Recognition. *Sensors*, 14(4):6474–6499, April 2014. ISSN 1424-8220. doi: 10.3390/s140406474. URL https://www.mdpi.com/1424-8220/14/4/6474. Number: 4 Publisher: Multidisciplinary Digital Publishing Institute. 4.3.1

[26] Gustavo E. Batista, Eamonn J. Keogh, Oben Moses Tataw, and Vinícius M. Souza. Cid: An efficient complexity-invariant distance for time series. *Data Min. Knowl. Discov.*, 28(3):634–669, may 2014. ISSN 1384-5810. doi: 10.1007/s10618-013-0312-3. URL https://doi.org/10.1007/s10618-013-0312-3. 5.3.2.3

[27] Alex Beltran, Varick L. Erickson, and Alberto E. Cerpa. Thermosense: Occupancy thermal based sensing for hvac control. In *Proc. of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings*, BuildSys'13, page 1–8, New York, NY, USA,

2013. ACM. ISBN 9781450324311. doi: 10.1145/2528282.2528301. URL https://doi.org/10.1145/2528282.2528301. 1

[28] Imane Lahmam Bennani, Anand Krishnan Prakash, Marina Zafiris, Lazlo Paul, Carlos Duarte Roa, Paul Raftery, Marco Pritoni, and Gabe Fierro. Query relaxation for portable brick-based applications. In *Proceedings of the 8th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, BuildSys '21, November 2021. ISBN 978-1-4503-9114-6. doi: 10.1145/3486611.3486671. URL https://dl.acm.org/doi/10.1145/3486611.3486671. 6.2.2

[29] BerriAI. LiteLLM: Python sdk and proxy server to call 100+ llm apis in openai format, 2025. URL https://github.com/BerriAI/litellm. Accessed: 2025-01-30. 6.4.6

[30] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding?, 2021. 3.3.2

[31] Claudio Bettini, Oliver Brdiczka, Karen Henricksen, Jadwiga Indulska, Daniela Nicklas, Anand Ranganathan, and Daniele Riboni. A survey of context modelling and reasoning techniques. *Pervasive and mobile computing*, 6(2):161–180, 2010. 2.4, 2.4.3

[32] Sejal Bhalla, Mayank Goel, and Rushil Khurana. Imu2doppler: Cross-modal domain adaptation for doppler-based activity recognition using imu data. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 5(4):1–20, 2021. 2.2.3, 3.3.1

[33] Sejal Bhalla, Mayank Goel, and Rushil Khurana. Imu2doppler: Cross-modal domain adaptation for doppler-based activity recognition using imu data. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 5(4), dec 2022. doi: 10.1145/3494994. URL https://doi.org/10.1145/3494994. 5.7

[34] Sudershan Boovaraghavan, Chen Chen, Anurag Maravi, Mike Czapik, Yang Zhang, Chris Harrison, and Yuvraj Agarwal. Mites: Design and Deployment of a General-Purpose Sensing Infrastructure for Buildings. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 7(1):2:1–2:32, March 2023. doi: 10.1145/3580865. URL https://dl.acm.org/doi/10.1145/3580865. 1, 1.2, 2.1.1, 2.1.2, 6, 6.4.1, 7.1.2, 7.1.3

[35] Sudershan Boovaraghavan, Chen Chen, Anurag Maravi, Mike Czapik, Yang Zhang, Chris Harrison, and Yuvraj Agarwal. Mites: Design and deployment of a general-purpose sensing infrastructure for buildings. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 7(1), mar 2023. doi: 10.1145/3580865. URL https://doi.org/10.1145/3580865. 3.2

[36] Sudershan Boovaraghavan, Prasoon Patidar, and Yuvraj Agarwal. Tao: Context detection from daily activity patterns using temporal analysis and ontology. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 7(3):1–32, 2023. (document), 1.3, 1

[37] Florian Bordes, Richard Yuanzhe Pang, Anurag Ajay, Alexander C. Li, Adrien Bardes, Suzanne Petryk, Oscar Mañas, Zhiqiu Lin, Anas Mahmoud, Bargav Jayaraman, Mark Ibrahim, Melissa Hall, Yunyang Xiong, Jonathan Lebensold, Candace Ross, Srihari

Jayakumar, Chuan Guo, Diane Bouchacourt, Haider Al-Tahan, Karthik Padthe, Vasu Sharma, Hu Xu, Xiaoqing Ellen Tan, Megan Richards, Samuel Lavoie, Pietro Astolfi, Reyhane Askari Hemmat, Jun Chen, Kushal Tirumala, Rim Assouel, Mazda Moayeri, Arjang Talattof, Kamalika Chaudhuri, Zechun Liu, Xilun Chen, Quentin Garrido, Karen Ullrich, Aishwarya Agrawal, Kate Saenko, Asli Celikyilmaz, and Vikas Chandra. An Introduction to Vision-Language Modeling, May 2024. URL http://arxiv.org/abs/2405.17247. arXiv:2405.17247 [cs]. 2.3.1

[38] Aisha Bradshaw. Wellness in the workplace. *Nature Human Behaviour*, 3(10):1039–1039, 2019. 5.1, 5.6.1

[39] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. 4.3.5

[40] John Brooke. SUS - A quick and dirty usability scale. 6.7.1

[41] A.J. Bernheim Brush, Bongshin Lee, Ratul Mahajan, Sharad Agarwal, Stefan Saroiu, and Colin Dixon. Home automation in the wild: challenges and opportunities. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 2115–2124, New York, NY, USA, May 2011. Association for Computing Machinery. ISBN 978-1-4503-0228-9. doi: 10.1145/1978942.1979249. URL https://dl.acm.org/doi/10.1145/1978942.1979249. 1, 1.2

[42] Business News Daily. Biggest Workplace Distractions That Kill Productivity - businessnewsdaily.com. https://www.businessnewsdaily.com/8098-distractions-kiling-productivity.html, 2023. 5.6.1

[43] Kelly E. Caine, Arthur D. Fisk, and Wendy A. Rogers. Benefits and privacy concerns of a home equipped with a visual sensing system: A perspective from older adults. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 50(2):180–184, 2006. doi: 10.1177/154193120605000203. URL https://doi.org/10.1177/154193120605000203. 1

[44] Sema Can. The determining of relationship between physical activity and perceived stress level in security service employees. *Journal of Education and Training Studies*, 7(1):149, 2019. 5.6.2

[45] Timothy I Cannings, Yingying Fan, and Richard J Samworth. Classification with imperfect training labels. *Biometrika*, 107(2):311–330, 2020. 3.3.4, 4.3.5

[46] Bing Cao, Yuxiao Zhao, Zhongyu Ren, Roger S McIntyre, Kayla M Teopiz, Xiao Gao, and Ling Ding. Are physical activities associated with perceived stress? the evidence from the china health and nutrition survey. *Frontiers in public health*, 9:1104, 2021. 5.1, 5.6.2

[47] João Carreira and Andrew Zisserman. Quo vadis, action recognition? A new model and the kinetics dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 4724–4733, Honolulu, HI, USA, 2017. IEEE Computer Society. doi: 10.1109/CVPR.2017.502. URL https://doi.org/10.1109/CVPR.2017.502. 1, 2.1.3, 2.2.1, 3.3.2

[48] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:

321–357, 2002. 3.3.4

[49] Luke Chen and CD Nugent. Ontology-based activity recognition in intelligent pervasive environments. *International Journal of Web Information Systems*, 5(4):410–430, 2009. ISSN 1744-0084. doi: 10.1108/17440080911006199. 1.1, 2.4.1

[50] Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. Program of Thoughts Prompting: Disentangling Computation from Reasoning for Numerical Reasoning Tasks, October 2023. URL http://arxiv.org/abs/2211.12588. arXiv:2211.12588 [cs]. 2.3.2, 6.6.1

[51] Wenqiang Chen, Shupei Lin, Elizabeth Thompson, and John Stankovic. Sensecollect: We need efficient ways to collect on-body sensor-based human activity data! *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 5(3):1–27, 2021. 3.3.1

[52] Gabriele Cipriani, Sabrina Danti, Lucia Picchi, Angelo Nuti, and Mario Di Fiorino. Daily functioning and dementia. *Dementia & Neuropsychologia*, 14(2):93–102, 2020. ISSN 1980-5764. doi: 10.1590/1980-57642020dn14-020001. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7304278/. 1.2, 4.6.2

[53] Cirillo Consulting GmbH. The Pomodoro® Technique — Cirillo Consulting GmbH. https://pomodorotechnique.com/, 2023. 5.6.1

[54] Diane Cook, Narayanan Krishnan, and Parisa Rashidi. Activity Discovery and Activity Recognition: A New Partnership. *IEEE transactions on cybernetics*, 43(3):820–828, June 2013. ISSN 2168-2267. doi: 10.1109/TSMCB.2012.2216873. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3772991/. 1.1, 2.2.3, 4.1

[55] Diane J Cook, Aaron S Crandall, Brian L Thomas, and Narayanan C Krishnan. Casas: A smart home in a box. *Computer*, 46(7):62–69, 2012. (document), 5.1, 5.5.1, 5.1, 5.2, B.1

[56] Antonia Creswell, Kai Arulkumaran, and Anil A. Bharath. On denoising autoencoders trained to minimise binary cross-entropy, 2017. 5.3.2.2

[57] Stephen Dawson-Haggerty, Andrew Krioukov, Jay Taneja, Sagar Karandikar, Gabe Fierro, Nikita Kitaev, and David Culler. BOSS: Building Operating System Services. pages 443–457, 2013. ISBN 978-1-931971-00-3. URL https://www.usenix.org/conference/nsdi13/technical-sessions/presentation/dawson-haggerty. 2.5.1

[58] Volkan Dedeoglu, Qianggong Zhang, Yang Li, Jiajun Liu, and Subbu Sethuvenkatraman. Buildingsage: A safe and secure ai copilot for smart buildings. In *Proceedings of the 11th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, BuildSys '24, 2024. ISBN 9798400707063. doi: 10.1145/3671127.3699677. URL https://doi.org/10.1145/3671127.3699677. 6.2.1, 6.1

[59] Shohreh Deldari, Hao Xue, Aaqib Saeed, Jiayuan He, Daniel V. Smith, and Flora D. Salim. Beyond just vision: A review on self-supervised representation learning on multimodal and temporal data, 2022. 1.1, 2.2.3

[60] Florenc Demrozi, Graziano Pravadelli, Azra Bihorac, and Parisa Rashidi. Human activity

recognition using inertial, physiological and environmental sensors: A comprehensive survey. *IEEE Access*, 8:210816–210836, 2020. doi: 10.1109/ACCESS.2020.3037715. 2.1.2

[61] Kaikai Deng, Dong Zhao, Zihan Zhang, Shuyue Wang, Wenxin Zheng, and Huadong Ma. Midas++: Generating Training Data of mmWave Radars From Videos for Privacy-Preserving Human Sensing With Mobility. *IEEE Transactions on Mobile Computing*, 23 (6):6650–6666, June 2024. ISSN 1558-0660. doi: 10.1109/TMC.2023.3325399. URL https://ieeexplore.ieee.org/document/10287632. 2.2.3

[62] Suhas Devmane, Omer Rana, and Charith Perera. OntoSage: Intelligent Human-Building Smartbot for Semantic Smart Building Question Answering. *World Wide Web*, 29(2), January 2026. doi: 10.1007/s11280-026-01403-0. URL https://doi.org/10.1007/s11280-026-01403-0. 1.1, 2.3.2, 2.5.3, 6.2.1, 6.1

[63] Nathan DeVrio, Vimal Mollyn, and Chris Harrison. SmartPoser: Arm Pose Estimation with a Smartphone and Smartwatch Using UWB and IMU Data. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, UIST '23, pages 1–11, New York, NY, USA, October 2023. Association for Computing Machinery. ISBN 979-8-4007-0132-0. doi: 10.1145/3586183.3606821. URL https://dl.acm.org/doi/10.1145/3586183.3606821. 4.2

[64] Colin Dixon, Ratul Mahajan, Sharad Agarwal, A. J. Brush, Bongshin Lee, Stefan Saroiu, and Paramvir Bahl. An Operating System for the Home. pages 337–352, 2012. URL https://www.usenix.org/conference/nsdi12/technical-sessions/presentation/dixon. 2.5.1

[65] Konstantinos Drossos, Stylianos I. Mimilakis, Shayan Gharib, Yanxiong Li, and Tuomas Virtanen. Sound event detection with depthwise separable and dilated convolutions. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7, Glasgow, UK, 2020. IJCNN. doi: 10.1109/IJCNN48605.2020.9207532. 1, 2.2.1, 3.1, 3.3.2

[66] Haodong Duan, Jiaqi Wang, Kai Chen, and Dahua Lin. Pyskl: Towards good practices for skeleton action recognition. In *Proceedings of the 30th ACM International Conference on Multimedia*, MM '22, page 7351–7354, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450392037. doi: 10.1145/3503161.3548546. URL https://doi.org/10.1145/3503161.3548546. 3.1

[67] Haodong Duan, Yue Zhao, Kai Chen, Dahua Lin, and Bo Dai. Revisiting skeleton-based action recognition. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2959–2968, New Orleans, LA, USA, 2022. IEEE. doi: 10.1109/CVPR52688.2022.00298. 2.2.1, 3.3.2

[68] Pardis Emami-Naeini, Henry Dixon, Yuvraj Agarwal, and Lorrie Faith Cranor. Exploring how privacy and security factor into iot device purchase behavior. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–12, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450359702. doi: 10.1145/3290605.3300764. URL https://doi.org/10.1145/3290605.3300764. 1

[69] Pardis Emami-Naeini, Janarth Dheenadhayalan, Yuvraj Agarwal, and Lorrie Faith Cranor. Which privacy and security attributes most impact consumers' risk perception and willingness to purchase iot devices? In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 519–536, San Francisco, CA, USA, 2021. IEEE. doi: 10.1109/SP40001.2021.00112. 1

[70] Steve Evans. SpaceSense: Exploring the Use of Sensors Data to Understand Occupants' Behavior in Heterogeneous Study Space Types. 1.1, 2.5.1

[71] Christoph Feichtenhofer. X3d: Expanding architectures for efficient video recognition, 2020. 3.3.2

[72] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6201–6210, Seoul, Korea (South), 2019. IEEE. doi: 10.1109/ICCV.2019.00630. 3.3.2

[73] Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. PAL: program-aided language models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *ICML'23*, pages 10764–10799, Honolulu, Hawaii, USA, July 2023. JMLR.org. 2.3.2, 6.6.1

[74] Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 776–780, New Orleans, LA, USA, 2017. IEEE. doi: 10.1109/ICASSP.2017.7952261. 1, 2.1.3, 2.2.1, 3.3.2

[75] Deepti Ghadiyaram, Du Tran, and Dhruv Mahajan. Large-scale weakly-supervised pre-training for video action recognition. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12038–12047, Long Beach, CA, USA, 2019. IEEE. doi: 10.1109/CVPR.2019.01232. 3.3.2

[76] Karan Goel and Emma Brunskill. Learning procedural abstractions and evaluating discrete latent temporal structure. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, page 10, US, 2019. OpenReview.net. URL `https://openreview.net/forum?id=ByleB2CcKm`. 2.4.2

[77] Bernardo Cuenca Grau, Ian Horrocks, Boris Motik, Bijan Parsia, Peter Patel-Schneider, and Ulrike Sattler. Owl 2: The next step for owl. *Journal of Web Semantics*, 6(4):309–322, 2008. 2.4.1, 5.4

[78] Emily Green. Hacker terrorizes family by hijacking baby monitor. `https://nordvpn.com/blog/baby-monitor-iot-hacking/`, December 2018. 1

[79] Chunhui Gu, Chen Sun, David A. Ross, Carl Vondrick, Caroline Pantofaru, Yeqing Li, Sudheendra Vijayanarasimhan, George Toderici, Susanna Ricco, Rahul Sukthankar, Cordelia Schmid, and Jitendra Malik. Ava: A video dataset of spatio-temporally localized atomic visual actions. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6047–6056, Salt Lake City, UT, USA, 2018. IEEE. doi: 10.1109/CVPR.2018.00633. 3.3.2

[80] Megan V. Ha, Emma Russell, Haarika A. Reddy, Alexander K. Maytin, Dustin P. DeMeo, Riku Arakawa, Mayank Goel, Jill F. Lehman, and Bryan T. Carroll. Self-narration for patient monitoring with smartwatch technology in post-operative wound care after dermatologic surgery. *Archives of Dermatological Research*, 316(7):389, June 2024. ISSN 1432-069X. doi: 10.1007/s00403-024-03149-z. 1.2

[81] Jakob Hahn, Sarah Heiler, Michael B. Kane, Sumee Park, and Werner Jensch. The Information Gap in Occupant-Centric Building Operations: Lessons Learned from Interviews with Building Operators in Germany. *Frontiers in Built Environment*, 8, May 2022. doi: 10.3389/fbuil.2022.838859. URL https://www.frontiersin.org/journals/built-environment/articles/10.3389/fbuil.2022.838859/full. 1.1, 1.2, 2.1.1, 2.5.1, 6.1

[82] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2. 4.3.5

[83] Barry P Haynes. Office productivity: a shift from cost reduction to human contribution, 2007. 5.6.1

[84] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 961–970, Boston, MA, USA, 2015. IEEE. doi: 10.1109/CVPR.2015.7298698. 1, 2.2.1, 3.3.2

[85] Rim Helaoui, Daniele Riboni, and Heiner Stuckenschmidt. A probabilistic ontological framework for the recognition of multilevel human activities. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '13, page 345–354, New York, NY, USA, 2013. Association for Computing Machinery. ISBN 9781450317702. doi: 10.1145/2493432.2493501. URL https://doi.org/10.1145/2493432.2493501. 5.2

[86] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006. 5.3.2.3

[87] Shruthi K. Hiremath and Thomas Plötz. The Lifespan of Human Activity Recognition Systems for Smart Homes. *Sensors*, 23(18):7729, January 2023. ISSN 1424-8220. doi: 10.3390/s23187729. URL https://www.mdpi.com/1424-8220/23/18/7729. Number: 18 Publisher: Multidisciplinary Digital Publishing Institute. 1.1

[88] Shruthi K. Hiremath, Yasutaka Nishimura, Sonia Chernova, and Thomas Plötz. Bootstrapping Human Activity Recognition Systems for Smart Homes from Scratch. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 6 (3):1–27, September 2022. ISSN 2474-9567. doi: 10.1145/3550294. URL https:

//dl.acm.org/doi/10.1145/3550294. 1.1, 2.2.3, 4.1

[89] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. doi: 10.1162/neco.1997.9.8.1735. 5.3.2.2

[90] Jin-Hyuk Hong, Julian Ramos, and Anind K. Dey. Toward personalized activity recognition systems with a semipopulation approach. *IEEE Transactions on Human-Machine Systems*, 46(1):101–112, 2016. doi: 10.1109/THMS.2015.2489688. 1.1

[91] Florimond Houssiau, Luc Rocher, and Yves-Alexandre de Montjoye. On the difficulty of achieving differential privacy in practice: user-level guarantees in aggregate location data. *Nature communications*, 13(1), 2022. 6.2.2

[92] Texas Instruments. Awr1642 single-chip 77-and 79-ghz fmcw radar sensor, 2017. 3.2

[93] Texas Instruments. Dca1000evm data capture card. *Retrieved May*, 17:2022, 2018. 3.2

[94] Anil K. Jain and Richard C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, Inc., USA, 1988. ISBN 013022278X. 5.5.1

[95] Yash Jain, Chi Ian Tang, Chulhong Min, Fahim Kawsar, and Akhil Mathur. Collossl: Collaborative self-supervised learning for human activity recognition. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 6(1), mar 2022. doi: 10.1145/3517246. URL https://doi.org/10.1145/3517246. 1.1, 2.2.3, 3.1

[96] Zimo Ji, Daoyuan Wu, Wenyuan Jiang, Pingchuan Ma, Zongjie Li, Yudong Gao, Shuai Wang, and Yingjiu Li. Taming Various Privilege Escalation in LLM-Based Agent Systems: A Mandatory Access Control Framework, January 2026. URL http://arxiv.org/abs/2601.11893. arXiv:2601.11893 [cs]. 6.2.2

[97] Changjiang Jiang, Fengchang Yu, Haihua Chen, Wei Lu, and Jin Zeng. TabDSR: Decompose, Sanitize, and Reason for Complex Numerical Reasoning in Tabular Data, November 2025. URL http://arxiv.org/abs/2511.02219. arXiv:2511.02219 [cs]. 6.3.2.2

[98] Jie Jiang, Riccardo Pozza, Nigel Gilbert, and Klaus Moessner. MakeSense: An IoT Testbed for Social Research of Indoor Activities. *ACM Transactions on Internet of Things*, 1(3):1–25, August 2020. ISSN 2691-1914, 2577-6207. doi: 10.1145/3381914. URL https://dl.acm.org/doi/10.1145/3381914. 2.1.1

[99] Haojian Jin, Boyuan Guo, Rituparna Roychoudhury, Yaxing Yao, Swarun Kumar, Yuvraj Agarwal, and Jason I. Hong. Exploring the needs of users for supporting privacy-protective behaviors in smart homes. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450391573. doi: 10.1145/3491102.3517602. URL https://doi.org/10.1145/3491102.3517602. 1

[100] Charmi Jobanputra, Jatna Bavishi, and Nishant Doshi. Human activity recognition: A survey. *Procedia Computer Science*, 155:698–703, 2019. 3.2

[101] Maria Johnsen. *Large Language Models (LLMs)*. Maria Johnsen, June 2024. ISBN 979-8-3285-3667-7. Google-Books-ID: 6rwOEQAAQBAJ. 2.3.2

[102] Tianjie Ju, Yi Hua, Hao Fei, Zhenyu Shao, Yubin Zheng, Haodong Zhao, Mong-Li Lee, Wynne Hsu, Zhuosheng Zhang, and Gongshen Liu. Watch Out Your Album! On the

Inadvertent Privacy Memorization in Multi-Modal Large Language Models, June 2025. URL http://arxiv.org/abs/2503.01208. arXiv:2503.01208 [cs]. 4.6.3

[103] Janghyun K, Barry H, Tianzhen H, and Marc A. P. A review of preserving privacy in data collected from buildings with differential privacy. *Journal of Building Engineering*, 56, 2022. doi: https://doi.org/10.1016/j.jobe.2022.104724. URL https://www.sciencedirect.com/science/article/pii/S2352710222007379. 6.2.2

[104] Rohan Kabra, Divya Saxena, Dhaval Patel, and Jiannong Cao. Time series clustering for human behavior pattern mining, 2021. URL https://arxiv.org/abs/2110.07549. 2.4.2

[105] G. R. Kanagachidambaresan. *Sensors and SBCs for Smart City Infrastructure*, pages 47–75. Springer International Publishing, Cham, 2021. ISBN 978-3-030-72957-8. doi: 10.1007/978-3-030-72957-8_3. URL https://doi.org/10.1007/978-3-030-72957-8_3. 3.2

[106] Alexander Karpekov, Sonia Chernova, and Thomas Plötz. DISCOVER: Data-driven Identification of Sub-activities via Clustering and Visualization for Enhanced Activity Recognition in Smart Homes, February 2025. URL http://arxiv.org/abs/2503.01733. arXiv:2503.01733 [cs]. 1.1, 2.2.3, 4.1

[107] Harmanpreet Kaur, Alex C. Williams, Daniel McDuff, Mary Czerwinski, Jaime Teevan, and Shamsi T. Iqbal. Optimizing for happiness and productivity: Modeling opportune moments for transitions and breaks at work. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–15, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450367080. doi: 10.1145/3313831.3376817. URL https://doi.org/10.1145/3313831.3376817. 5.2

[108] Sunder Ali Khowaja, Aria Ghora Prabono, Feri Setiawan, Bernardo Nugroho Yahya, and Seok-Lyong Lee. Contextual activity based healthcare internet of things, services, and people (hiotsp): An architectural framework for healthcare monitoring using wearable sensors. *Computer Networks*, 145:190–206, 2018. 1.1, 2.4.3, 5.1, 5.2

[109] Evan King, Haoxiang Yu, Sangsu Lee, and Christine Julien. Sasha: Creative goal-oriented reasoning in smart homes with large language models. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 8(1), March 2024. doi: 10.1145/3643505. URL https://doi.org/10.1145/3643505. 2.3.2, 6.2.1

[110] Evan King, Haoxiang Yu, Sahil Vartak, Jenna Jacob, Sangsu Lee, and Christine Julien. Thoughtful Things: Building Human-Centric Smart Devices with Small Language Models, May 2024. URL http://arxiv.org/abs/2405.03821. arXiv:2405.03821 [cs]. 6.2.1

[111] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. Hmdb: A large video database for human motion recognition. In *2011 International Conference on Computer Vision*, pages 2556–2563, Barcelona, Spain, 2011. IEEE. doi: 10.1109/ICCV.2011.6126543. 3.3.2

[112] Hilde Kuehne, Ali Arslan, and Thomas Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 780–787, US, 2014. IEEE.

2.4.2

[113] Hyeokhyen Kwon, Catherine Tong, Harish Haresamudram, Yan Gao, Gregory D Abowd, Nicholas D Lane, and Thomas Ploetz. Imutube: Automatic extraction of virtual on-body accelerometry from video for human activity recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 4(3):1–29, 2020. 1.1, 2.2.3, 3.1, 4.1

[114] Gierad Laput and Chris Harrison. Surfacesight: A new spin on touch, user, and object sensing for iot experiences. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–12, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450359702. doi: 10.1145/3290605.3300559. URL https://doi.org/10.1145/3290605.3300559. 1, 2.1.2, 3.3.4

[115] Gierad Laput, Yang Zhang, and Chris Harrison. Synthetic sensors: Towards general-purpose sensing. In *Proc. of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, page 3986–3999, New York, NY, USA, 2017. ACM. ISBN 9781450346559. doi: 10.1145/3025453.3025773. URL https://doi.org/10.1145/3025453.3025773. 1, 1.1, 2.1.2, 3.3.1, 3.3.4

[116] Gierad Laput, Karan Ahuja, Mayank Goel, and Chris Harrison. Ubicoustics: Plug-and-play acoustic activity recognition. In *Proc. of the 31st Annual ACM Symposium on UIST*, UIST '18, page 213–224, New York, NY, USA, 2018. ACM. ISBN 9781450359481. doi: 10.1145/3242587.3242609. URL https://doi.org/10.1145/3242587.3242609. 5.7

[117] Gierad Laput, Karan Ahuja, Mayank Goel, and Chris Harrison. Ubicoustics: Plug-and-play acoustic activity recognition. In *Proc. of the 31st Annual ACM Symposium on UIST*, UIST '18, page 213–224, New York, NY, USA, 2018. ACM. ISBN 9781450359481. doi: 10.1145/3242587.3242609. URL https://doi.org/10.1145/3242587.3242609. 1, 2.2.1, 3.3.1

[118] Oscar D Lara and Miguel A Labrador. A survey on human activity recognition using wearable sensors. *IEEE communications surveys & tutorials*, 15(3):1192–1209, 2012. 3.2

[119] Sungmin Lee, Minju Kang, Joonhee Lee, Seungyong Lee, Dongju Kim, Jingi Hong, Jun Shin, Pei Zhang, and JeongGil Ko. Llm-based question-answer framework for sensor-driven hvac system interaction, 2025. URL https://arxiv.org/abs/2507.04748. 1.1, 2.5.3, 6.2.1, 6.1

[120] Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 18(17):1–5, 2017. URL http://jmlr.org/papers/v18/16-365.html. 4.3.5

[121] Zikang Leng, Hyeokhyen Kwon, and Thomas Ploetz. Generating Virtual On-body Accelerometer Data from Virtual Textual Descriptions for Human Activity Recognition. In *Proceedings of the 2023 ACM International Symposium on Wearable Computers*, ISWC '23, pages 39–43, New York, NY, USA, October 2023. Association for Computing Machinery. ISBN 979-8-4007-0199-3. doi: 10.1145/3594738.3611361. URL

https://dl.acm.org/doi/10.1145/3594738.3611361. 2.2.3

[122] Zikang Leng, Hyeokhyen Kwon, and Thomas Plötz. On the Benefit of Generative Foundation Models for Human Activity Recognition, October 2023. URL http://arxiv.org/abs/2310.12085. arXiv:2310.12085 [cs]. 2.3.1

[123] Zikang Leng, Amitrajit Bhattacharjee, Hrudhai Rajasekhar, Lizhe Zhang, Elizabeth Bruda, Hyeokhyen Kwon, and Thomas Plötz. IMUGPT 2.0: Language-Based Cross Modality Transfer for Sensor-Based Human Activity Recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 8(3):1–32, August 2024. ISSN 2474-9567. doi: 10.1145/3678545. URL https://dl.acm.org/doi/10.1145/3678545. 2.2.3, 4.1

[124] Mingchen Li and Zhe Wang. Buildinggpt: Query building semantic data using large language models and vector-graph retrieval-augmented generation. *Building and Environment*, 287, 2026. doi: https://doi.org/10.1016/j.buildenv.2025.113855. URL https://www.sciencedirect.com/science/article/pii/S0360132325013253. 2.3.2, 6.2.1, 6.1

[125] Dawei Liang, Guihong Li, Rebecca Adaimi, Radu Marculescu, and Edison Thomaz. Audioimu: Enhancing inertial sensing-based activity recognition with acoustic models. In *Proceedings of the 2022 ACM International Symposium on Wearable Computers*, ISWC '22, page 44–48, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450394246. doi: 10.1145/3544794.3558471. URL https://doi.org/10.1145/3544794.3558471. 1.1, 2.1.2, 2.2.3

[126] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7082–7092, Seoul, Korea (South), 2019. IEEE. doi: 10.1109/ICCV.2019.00718. 3.3.2

[127] Guocheng Liu, Caixia Zhang, Qingyang Xu, Ruoshi Cheng, Yong Song, Xianfeng Yuan, and Jie Sun. I3d-shufflenet based human action recognition. *Algorithms*, 13(11):301, 2020. 2.2.1, 3.1, 3.3.2

[128] Jun Liu, Amir Shahroudy, Mauricio Perez, Gang Wang, Ling-Yu Duan, and Alex C Kot. Ntu rgb+d 120: A large-scale benchmark for 3d human activity understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(10):2684–2701, 2020. 3.3.2

[129] Tian-Yu Liu. EasyEnsemble and Feature Selection for Imbalance Data Sets. In *2009 International Joint Conference on Bioinformatics, Systems Biology and Intelligent Computing*, pages 517–520, August 2009. doi: 10.1109/IJCBS.2009.22. URL https://ieeexplore.ieee.org/document/5260440. 4.3.5

[130] Zhaoyang Liu, Limin Wang, Wayne Wu, Chen Qian, and Tong Lu. Tam: Temporal adaptive module for video recognition, 2021. 3.3.2

[131] Ryan Louie, Darren Gergle, and Haoqi Zhang. Affinder: Expressing concepts of situations that afford activities using context-detectors. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450391573. doi: 10.1145/3491102.3501902. URL

https://doi.org/10.1145/3491102.3501902. 5.7

[132] Harsh Lunia. Can VLMs be used on videos for action recognition? LLMs are Visual Reasoning Coordinators, July 2024. URL http://arxiv.org/abs/2407.14834. arXiv:2407.14834 [cs] version: 1. 2.3.1

[133] Haohao Luo, Zexi Li, Yuexiang Xie, Wenhao Zhang, Yaliang Li, and Ying Shen. IntentRL: Training Proactive User-intent Agents for Open-ended Deep Research via Reinforcement Learning, February 2026. URL http://arxiv.org/abs/2602.03468. arXiv:2602.03468 [cs]. 6.4.2.3

[134] Naveen Sai Madiraju, Seid M. Sadat, Dimitry Fisher, and Homa Karimabadi. Deep temporal clustering : Fully unsupervised learning of time-domain features, 2018. URL https://arxiv.org/abs/1802.01059. 2.4.2, 5.1, 5.3.2.2, 5.3.2.3

[135] Ginés Hidalgo Martınez. *OpenPose: Whole-body pose estimation*. PhD thesis, Master's Thesis, Carnegie Mellon University, 2019. 1, 2.2.1

[136] Leland McInnes, John Healy, and Steve Astels. hdbscan: Hierarchical density based clustering. *The Journal of Open Source Software*, 2(11):205, March 2017. ISSN 2475-9066. doi: 10.21105/joss.00205. URL http://joss.theoj.org/papers/10.21105/joss.00205. 4.3.2.1, 4.3.5

[137] Georgios Meditskos and Ioannis Kompatsiaris. iknow: Ontology-driven situational awareness for the recognition of activities of daily living. *Pervasive and Mobile Computing*, 40: 17–41, 2017. (document), 2.4.1, 5.1, 5.5.3, 5.2

[138] Georgios Meditskos, Stamatia Dasiopoulou, and Ioannis Kompatsiaris. Metaq: A knowledge-driven framework for context-aware activity recognition combining sparql and owl 2 activity patterns. *Pervasive and Mobile Computing*, 25:104–124, 2016. 1.1, 2.4.1, 5.1, 5.3.1.1

[139] Mites.io. Mites.io: a full-stack ubiquitous sensing platform. https://mites.io/, 2020. (document), 3.2, 3.3.4, 4.2, 5.5.4.1, 5.14

[140] MMAction2. Openmmlab's next generation video understanding toolbox and benchmark. https://github.com/open-mmlab/mmaction2, 2020. 1, 2.2.1, 3.1, 4.1

[141] MMPose. Openmmlab pose estimation toolbox and benchmark. https://github.com/open-mmlab/mmpose, 2020. 1, 2.2.1

[142] Vimal Mollyn, Karan Ahuja, Dhruv Verma, Chris Harrison, and Mayank Goel. Samosa: Sensing activities with motion and subsampled audio. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 6(3):1–19, 2022. 2.1.2, 3.3.1, 3.3.2, 3.5, A.1

[143] Vimal Mollyn, Riku Arakawa, Mayank Goel, Chris Harrison, and Karan Ahuja. IMUPoser: Full-Body Pose Estimation using IMUs in Phones, Watches, and Earbuds. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, CHI '23, pages 1–12, New York, NY, USA, April 2023. Association for Computing Machinery. ISBN 978-1-4503-9421-5. doi: 10.1145/3544548.3581392. URL https://dl.acm.org/doi/10.1145/3544548.3581392. 4.2

[144] Mehrab Bin Morshed, Koustuv Saha, Richard Li, Sidney K. D'Mello, Munmun De Choudhury, Gregory D. Abowd, and Thomas Plötz. Prediction of mood instability with passive sensing. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 3(3), sep 2019. doi: 10.1145/3351233. URL https://doi.org/10.1145/3351233. 5.2

[145] Ozan Baris Mulayim, Avia Anwar, Umut Mete Saka, Lazlo Paul, Anand Krishnan Prakash, Gabe Fierro, Marco Pritoni, and Mario Bergés. BuildingQA: A Benchmark for Natural Language Question Answering over Building Knowledge Graphs. In *Proceedings of the 12th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, BuildSys '25, November 2025. ISBN 979-8-4007-1945-5. doi: 10.1145/3736425.3770097. URL https://dl.acm.org/doi/10.1145/3736425.3770097. 6.2.1

[146] Sebastian Münzner, Philip Schmidt, Attila Reiss, Michael Hanselmann, Rainer Stiefelhagen, and Robert Dürichen. Cnn-based sensor fusion techniques for multimodal human activity recognition. In *Proceedings of the 2017 ACM International Symposium on Wearable Computers*, ISWC '17, page 158–165, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450351881. doi: 10.1145/3123021.3123046. URL https://doi.org/10.1145/3123021.3123046. 2.2.2

[147] Mark A. Musen. The protégé project: a look back and a look forward. *AI Matters*, 1(4): 4–12, 2015. doi: 10.1145/2757001.2757003. URL https://doi.org/10.1145/2757001.2757003. 5.4

[148] Curtis Northcutt, Lu Jiang, and Isaac Chuang. Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research*, 70:1373–1411, 2021. 3.3.4, 3.3.5

[149] George Okeyo, Liming Chen, Hui Wang, and Roy Sterritt. Ontology-based learning framework for activity assistance in an adaptive smart home. In *Activity recognition in pervasive intelligent environments*, pages 237–263. Springer, US, 2011. 1.1, 2.4.3, 5.2

[150] OpenAI. Openai api, 2025. URL https://platform.openai.com/docs/api-reference/. Accessed: 2025-04-29. 4.3.5

[151] The pandas development team. pandas-dev/pandas: Pandas, February 2020. URL https://doi.org/10.5281/zenodo.3509134. 4.3.5

[152] Preksha Pareek and Ankit Thakkar. A survey on video-based human action recognition: recent updates, datasets, challenges, and applications. *Artificial Intelligence Review*, 54(3): 2259–2322, 2021. 2.2.2

[153] Adam Paszke et al. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., US, 2019. URL https://arxiv.org/abs/1912.01703. 5.4

[154] Prasoon Patidar, Mayank Goel, and Yuvraj Agarwal. VAX: Using Existing Video and Audio-based Activity Recognition Models to Bootstrap Privacy-Sensitive Sensors. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 7(3):1–24, September 2023. ISSN 2474-9567. doi: 10.1145/3610907. URL https://dl.acm.

org/doi/10.1145/3610907. (document), 1.1, 1.3, 2.1.2, 4.1

[155] Prasoon Patidar, Riku Arakawa, Ricardo Graça, Rúben Moutinho, Adriano Soares, Ana Vasconcelos, Filippo Talami, Joana Couto da Silva, Inês Silva, Cristina Mendes Santos, Mayank Goel, and Yuvraj Agarwal. OrganicHAR: Towards Activity Discovery in Organic Settings for Privacy Preserving Sensors Using Efficient Video Analysis. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 9(4):203:1–203:32, December 2025. doi: 10.1145/3770674. URL https://dl.acm.org/doi/10.1145/3770674. (document), 1.3

[156] Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. Gorilla: Large language model connected with massive apis. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-4020. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/e4c61f578ff07830f5c37378dd3ecb0d-Paper-Conference.pdf. 2.3.2, 6.2.1

[157] Nandita Pattnaik, Shujun Li, and Jason R. C. Nurse. A Survey of User Perspectives on Security and Privacy in a Home Networking Environment. *ACM Comput. Surv.*, 55(9), January 2023. doi: 10.1145/3558095. URL https://dl.acm.org/doi/10.1145/3558095. 6.2.2

[158] Nandita Pattnaik, Shujun Li, and Jason R.C. Nurse. Security and Privacy Perspectives of People Living in Shared Home Environments. *Proc. ACM Hum.-Comput. Interact.*, 8(CSCW2), November 2024. doi: 10.1145/3686907. URL https://dl.acm.org/doi/10.1145/3686907. 6.2.2

[159] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. 4.3.5

[160] Charith Perera, Arkady Zaslavsky, Peter Christen, and Dimitrios Georgakopoulos. Context aware computing for the internet of things: A survey. *IEEE communications surveys & tutorials*, 16(1):414–454, 2013. 2.4, 2.4.3

[161] Elena Planas, Salvador Martínez, Marco Brambilla, and Jordi Cabot. Towards Access Control Models for Conversational User Interfaces. In *Enterprise, Business-Process and Information Systems Modeling*, volume 450. 2022. ISBN 978-3-031-07474-5 978-3-031-07475-2. doi: 10.1007/978-3-031-07475-2_21. URL https://link.springer.com/10.1007/978-3-031-07475-2_21. 6.2.2, 6.1

[162] Elena Planas, Salvador Martínez, Marco Brambilla, and Jordi Cabot. Modeling and enforcing access control policies in conversational user interfaces. *Software and Systems Modeling*, 22(6), December 2023. doi: 10.1007/s10270-023-01131-3. URL https://doi.org/10.1007/s10270-023-01131-3. 6.2.2

[163] Mohammadreza Pourreza and Davood Rafiei. Din-sql: decomposed in-context learning of text-to-sql with self-correction. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, 2023. 2.3.2, 6.2.1

[164] Preeja Pradeep and Shivsubramani Krishnamoorthy. The mom of context-aware systems: A survey. *Computer Communications*, 137:44–69, 2019. 2.4.3

[165] Prasoon Patidar, Mayank Goel, Yuvraj Agarwal. VAX: Open-source repository for the VAX system. `https://github.com/synergylabs/vax`, 2023. 3.3.5

[166] Prasoon Patidar, Riku Arakawa, Mayank Goel, Yuvraj Agarwal. OrganicHAR: Open-source repository for the OrganicHAR. `https://github.com/synergylabs/OrganicHAR`, 2025. 4.3.5, 4.4

[167] Riccardo Presotto, Gabriele Civitarese, and Claudio Bettini. Federated clustering and semi-supervised learning: A new partnership for personalized human activity recognition. *Pervasive and Mobile Computing*, 88:101726, 2022. 1.1, 2.2.3

[168] Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, dahai li, Zhiyuan Liu, and Maosong Sun. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=dHng2O0Jjr`. 2.3.2, 6.2.1

[169] Bhiksha Raj, Kaustubh Kalgaonkar, Chris Harrison, and Paul Dietz. Ultrasonic doppler sensing in hci. *IEEE Pervasive Computing*, 11(2):24–29, 2012. doi: 10.1109/MPRV.2012.17. 2.1.2

[170] Sreenivasan Ramasamy Ramamurthy and Nirmalya Roy. Recent trends in machine learning for human activity recognition—a survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1254, 2018. 3.2

[171] Suneth Ranasinghe, Fadi Al Machot, and Heinrich C Mayr. A review on applications of activity recognition systems with regard to performance and evaluation. *International Journal of Distributed Sensor Networks*, 12(8):1550147716665520, 2016. doi: 10.1177/1550147716665520. URL `https://doi.org/10.1177/1550147716665520`. 1, 1.1, 2.2.2

[172] Reza Rawassizadeh, Elaheh Momeni, Chelsea Dobbins, Joobin Gharibshah, and Michael Pazzani. Scalable daily human behavioral pattern mining from multivariate temporal data. *IEEE Transactions on Knowledge and Data Engineering*, 28(11):3098–3112, 2016. doi: 10.1109/TKDE.2016.2592527. 2.4.2

[173] Daniele Riboni and Claudio Bettini. Cosar: hybrid reasoning for context-aware activity recognition. *Personal and Ubiquitous Computing*, 15(3):271–289, 2011. 1.1, 5.1, 5.2

[174] Daniele Riboni and Claudio Bettini. Owl 2 modeling and reasoning with complex human activities. *Pervasive and Mobile Computing*, 7(3):379–395, 2011. 5.2

[175] Daniele Riboni, Claudio Bettini, Gabriele Civitarese, Zaffar Haider Janjua, and Rim Helaoui. Fine-grained recognition of abnormal behaviors for early detection of mild cognitive impairment. In *2015 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 149–154, US, 2015. IEEE. doi: 10.1109/PERCOM.2015.7146521. 1.1, 2.4.3, 5.2

[176] Daniele Riboni, Claudio Bettini, Gabriele Civitarese, Zaffar Haider Janjua, and Rim Helaoui. Smartfaber: Recognizing fine-grained abnormal behaviors for early detection of mild cognitive impairment. *Artificial intelligence in medicine*, 67:57–74, 2016. 2.4.2

[177] Daniele Riboni, Timo Sztyler, Gabriele Civitarese, and Heiner Stuckenschmidt. Unsupervised recognition of interleaved activities of daily living through ontological and probabilistic reasoning. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '16, page 1–12, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450344616. doi: 10.1145/2971648.2971691. URL https://doi.org/10.1145/2971648.2971691. (document), 5.2, 5.5.3, 5.2

[178] Dmitriy Rivkin, Francois Hogan, Amal Feriani, Abhisek Konar, Adam Sigal, Steve Liu, and Greg Dudek. Sage: Smart home agent with grounded execution, 2024. URL https://arxiv.org/abs/2311.00772. 2.3.2, 6.2.1

[179] Juan Rocamonde, Victoriano Montesinos, Elvis Nava, Ethan Perez, and David Lindner. Vision-Language Models are Zero-Shot Reward Models for Reinforcement Learning, March 2024. URL http://arxiv.org/abs/2310.12921. arXiv:2310.12921 [cs] version: 2. 2.3.1

[180] Lipsarani Sahoo, Nazmus Sakib Miazi, Mohamed Shehab, Florian Alt, and Yomna Abdelrahman. You know too much: Investigating users' perceptions and privacy concerns towards thermal imaging. In Stefan Schiffner, Sebastien Ziegler, and Adrian Quesada Rodriguez, editors, *Privacy Symposium 2022*, pages 207–229, Cham, 2022. Springer International Publishing. ISBN 978-3-031-09901-4. 2.1.2

[181] Laurens Samson, Nimrod Barazani, Sennay Ghebreab, and Yuki M. Asano. Little Data, Big Impact: Privacy-Aware Visual Language Models via Minimal Tuning, May 2025. URL http://arxiv.org/abs/2405.17423. arXiv:2405.17423 [cs]. 4.6.3

[182] Luis Sanchez, Luis Muñoz, Jose Antonio Galache, Pablo Sotres, Juan R. Santana, Veronica Gutierrez, Rajiv Ramdhany, Alex Gluhak, Srdjan Krco, Evangelos Theodoridis, and Dennis Pfisterer. SmartSantander: IoT Experimentation over a Smart City Testbed. *Computer Networks*, 61:217–238, March 2014. ISSN 13891286. doi: 10.1016/j.bjp.2013.12.020. URL http://arxiv.org/abs/2403.03196. arXiv:2403.03196 [cs]. 2.1.1

[183] Kaori Sato, Sachiko Kuroda, and Hideo Owan. Mental health effects of long work hours, night and weekend work, and short rest periods. *Social Science & Medicine*, 246:112774, 2020. 5.6.2

[184] Alex Schiffer. How a fish tank helped hack a casino. https://www.washingtonpost.com/news/innovations/wp/2017/07/21/how-a-fish-tank-helped-hack-a-casino/?noredirect=on, July 2017. 1

[185] Scientific American. Happiness Is a Walk in the Park - Scientific American. https://www.scientificamerican.com/podcast/episode/happiness-is-a-walk-in-the-park-10-05-05/, 2023. 5.6.1

[186] Chris Seiffert, Taghi M. Khoshgoftaar, Jason Van Hulse, and Amri Napolitano. RUSBoost:

A Hybrid Approach to Alleviating Class Imbalance. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 40(1):185–197, January 2010. ISSN 1558-2426. doi: 10.1109/TSMCA.2009.2029559. URL https://ieeexplore.ieee.org/document/5299216. 4.3.5

[187] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. Ntu rgb+d: A large scale dataset for 3d human activity analysis, 2016. 3.3.2

[188] Hao Shao, Shengju Qian, and Yu Liu. Temporal interlacing network. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(07):11966–11973, Apr. 2020. doi: 10.1609/aaai.v34i07.6872. URL https://ojs.aaai.org/index.php/AAAI/article/view/6872. 3.3.2

[189] Aditya Sharma. OAK-D: Understanding and running neural network inference with DepthAI API. In Puneet Chugh, Aritra Roy Gosthipaty, Susan Huot, Kseniia Kidriavsteva, Ritwik Raha, and Abhishek Thanki, editors, *PyImageSearch*. 2022. https://pyimg.co/8ynbk. 4.2

[190] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12018–12027, Long Beach, CA, USA, 2019. IEEE. doi: 10.1109/CVPR.2019.01230. 3.3.2

[191] Akash Deep Singh, Sandeep Singh Sandha, Luis Garcia, and Mani Srivastava. Radhar: Human activity recognition from point clouds generated through a millimeter-wave radar. In *Proceedings of the 3rd ACM Workshop on Millimeter-Wave Networks and Sensing Systems*, mmNets'19, page 51–56, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450369329. doi: 10.1145/3349624.3356768. URL https://doi.org/10.1145/3349624.3356768. 1, 2.1.2

[192] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild, 2012. 2.2.1, 3.3.2

[193] Yuan Sui, Mengyu Zhou, Mingjie Zhou, Shi Han, and Dongmei Zhang. Table Meets LLM: Can Large Language Models Understand Structured Table Data? A Benchmark and Empirical Study. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, WSDM '24, pages 645–654, New York, NY, USA, March 2024. Association for Computing Machinery. ISBN 979-8-4007-0371-3. doi: 10.1145/3616855.3635752. URL https://dl.acm.org/doi/10.1145/3616855.3635752. 2.3.2, 6.6.1

[194] Chen Sun, Abhinav Shrivastava, Carl Vondrick, Kevin Murphy, Rahul Sukthankar, and Cordelia Schmid. Actor-centric relation network. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision – ECCV 2018*, pages 335–351, Cham, 2018. Springer International Publishing. ISBN 978-3-030-01252-6. 3.3.2

[195] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5686–5696, Long Beach, CA, USA, 2019. IEEE. doi: 10.1109/CVPR.2019.00584. 2.2.1

[196] Shayan Talaei, Mohammadreza Pourreza, Yu-Chen Chang, Azalia Mirhoseini, and Amin

Saberi. Chess: Contextual harnessing for efficient sql synthesis, 2024. URL https://arxiv.org/abs/2405.16755. 6.2.1

[197] Filippo Talami, Britt Pellens, Joana Couto Silva, Inês Silva, Ana Vasconcelos, Ricardo Graça, Prasoon Patidar, Adriano Soares, Jilte Heyvaert, Andrea Wilkinson, Niels Hendriks, Mayank Goel, and Cristina Mendes-Santos. Designing Adaptive User Interfaces for People with Dementia: Findings from Participatory Research. In *Proceedings of the 16th Biannual Conference of the Italian SIGCHI Chapter*, CHItaly '25, pages 1–3, New York, NY, USA, October 2025. Association for Computing Machinery. ISBN 979-8-4007-2102-1. doi: 10.1145/3750069.3755947. URL https://dl.acm.org/doi/10.1145/3750069.3755947. 7.1.1

[198] Filippo Talami, Britt Pellens, Joana Couto Silva, Inês Silva, Ana Vasconcelos, Ricardo Graça, Mayank Goel, Prasoon Patidar, Andrea Wilkinson, Niels Hendriks, Adriano Soares, and Cristina Mendes-Santos. AUTONOMOUS - An Intelligent Ubiquitous System for Tracking Multimodal Daily Living Activities in People with Dementia: Paradoxical Findings from Participatory Design Research. ALZ, July 2025. URL https://alz.confex.com/alz/2025/meetingapp.cgi/Paper/102690. 7.1.1

[199] Adane Nega Tarekegn, Mohib Ullah, Faouzi Alaya Cheikh, and Muhammad Sajjad. Enhancing Human Activity Recognition Through Sensor Fusion And Hybrid Deep Learning Model. In *2023 IEEE International Conference on Acoustics, Speech, and Signal Processing Workshops (ICASSPW)*, pages 1–5, June 2023. doi: 10.1109/ICASSPW59220.2023.10193698. URL https://ieeexplore.ieee.org/document/10193698. 4.3.1

[200] RAPIDS Development Team. Rapids: Collection of libraries for end to end gpu data science, 2018. URL https://rapids.ai. 5.4

[201] Vishnu Priya Thotakura and Purnachand Nalluri. Convolutional 3d in activity recognition -a review. In *2022 2nd International Conference on Artificial Intelligence and Signal Processing (AISP)*, pages 1–6, Vijayawada, India, 2022. IEEE. doi: 10.1109/AISP53593.2022.9760638. 3.3.2

[202] Yuan Tian, Nan Zhang, Yueh-Hsun Lin, XiaoFeng Wang, Blase Ur, XianZheng Guo, and Patrick Tague. Smartauth: user-centered authorization for the internet of things. In *Proceedings of the 26th USENIX Conference on Security Symposium*, SEC'17, 2017. ISBN 9781931971409. 6.2.2

[203] Toggl. Toggl: Time Tracking, Project Planning and Hiring Tools to Help Teams Work BetterToggl: Time Tracking, Project Planning and Hiring Tools to Help Teams Work Better. https://toggl.com/, 2023. 5.6.1

[204] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6450–6459, Salt Lake City, UT, USA, 2018. IEEE. doi: 10.1109/CVPR.2018.00675. 3.3.2

[205] Annibal Truzzi, Letice Valente, Ingun Ulstein, Eliasz Engelhardt, Jerson Laks, and Knut Engedal. Burnout in familial caregivers of patients with dementia. *Brazilian Journal of Psychiatry*, 34:405–412, 2012. ISSN 1516-4446, 1809-452X. doi:

https://doi.org/10.1016/j.rbp.2012.02.006. URL https://www.scielo.br/j/rbp/a/3MDvfncP8nG67DGxnsQ9DYp/?lang=en. 7.4

[206] Annalise Vaccarello, Alexander K. Maytin, Yash Kumar, Toluwalashe Onamusi, Haarika A. Reddy, Mayank Goel, Riku Arakawa, Jill Fain Lehman, and Bryan T. Carroll. Barriers to use of digital assistance for postoperative wound care: a single-center survey of dermatologic surgery patients. *Archives of Dermatological Research*, 316(7):376, June 2024. ISSN 1432-069X. doi: 10.1007/s00403-024-03025-w. 4.6.2, 7.3.6

[207] Yonatan Vaizman, Katherine Ellis, and Gert Lanckriet. Recognizing detailed human context in the wild from smartphones and smartwatches. *IEEE pervasive computing*, 16(4):62–74, 2017. (document), 5.1, 5.5.1, 5.1, B.1, B.3

[208] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. URL http://jmlr.org/papers/v9/vandermaaten08a.html. 5.3.2.3

[209] Claudia Villalonga, Muhammad Asif Razzaq, Wajahat Ali Khan, Hector Pomares, Ignacio Rojas, Sungyoung Lee, and Oresti Banos. Ontology-based high-level context inference for human behavior identification. *Sensors*, 16(10):1617, 2016. 1.1, 2.4.1, 5.1, 5.3.1.1, 5.1, 5.5.2.1

[210] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, page 1096–1103, New York, NY, USA, 2008. Association for Computing Machinery. ISBN 9781605582054. doi: 10.1145/1390156.1390294. URL https://doi.org/10.1145/1390156.1390294. 5.3.2.2

[211] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2. 4.3.5

[212] Michalis Vrigkas, Christophoros Nikou, and Ioannis A Kakadiaris. A review of human activity recognition methods. *Frontiers in Robotics and AI*, 2:28, 2015. 1.1, 2.2.2

[213] W3C. SPIN - SPARQL Inferencing Notation. https://spinrdf.org/, 2023. 5.4

[214] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 20–36, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46484-8. 3.3.2

[215] Shuai Wang, Luoyu Mei, Ruofeng Liu, Wenchao Jiang, Zhimeng Yin, Xianjun Deng, and

Tian He. Multi-modal fusion sensing: A comprehensive review of millimeter-wave radar and its integration with other modalities. *IEEE Commun. Surv. Tutorials*, 27(1):322–352, 2025. doi: 10.1109/COMST.2024.3398004. URL https://doi.org/10.1109/COMST.2024.3398004. 2.2.2

[216] Yuchen Wang and Lu Liu. Research on sustainable green building space design model integrating IoT technology. *PLOS ONE*, 19(4):e0298982, April 2024. ISSN 1932-6203. doi: 10.1371/journal.pone.0298982. URL https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0298982. Publisher: Public Library of Science. 1, 2.1.1, 2.5.1

[217] Pete Warden, Matthew Stewart, Brian Plancher, Colby Banbury, Shvetank Prakash, Emma Chen, Zain Asgar, Sachin Katti, and Vijay Janapa Reddi. Machine learning sensors, 2022. URL https://arxiv.org/abs/2206.03266. 2.1.2, 3.5, 4.2

[218] Thomas Weng, Anthony Nwokafor, and Yuvraj Agarwal. Buildingdepot 2.0: An integrated management system for building analysis and control. In *Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings*, BuildSys '13, 2013. ISBN 9781450324311. doi: 10.1145/2528282.2528285. URL https://doi.org/10.1145/2528282.2528285. 2.5.1, 6.4.1

[219] Kapo Wong, Alan HS Chan, and SC Ngan. The effect of long working hours and overtime on occupational health: a meta-analysis of evidence from 1998 to 2018. *International journal of environmental research and public health*, 16(12):2102, 2019. 5.6.2

[220] Chao-Yuan Wu, Christoph Feichtenhofer, Haoqi Fan, Kaiming He, Philipp Krahenbuhl, and Ross Girshick. Long-term feature banks for detailed video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 284–293, Long Beach, CA, USA, 2019. IEEE. 3.3.2

[221] Jason Wu, Chris Harrison, Jeffrey P. Bigham, and Gierad Laput. Automated Class Discovery and One-Shot Interactions for Acoustic Activity Recognition. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–14, Honolulu HI USA, April 2020. ACM. ISBN 978-1-4503-6708-0. doi: 10.1145/3313831.3376875. URL https://dl.acm.org/doi/10.1145/3313831.3376875. 1.1, 2.2.3, 4.1

[222] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 478–487, New York, NY, USA, 2016. JMLR.org. 5.3.2.3

[223] Tianwei Xing, Luis Garcia, Federico Cerutti, Lance Kaplan, Alun Preece, and Mani Srivastava. Deepsqa: Understanding sensor data via question answering. In *Proceedings of the International Conference on Internet-of-Things Design and Implementation*, IoTDI '21, 2021. ISBN 9781450383547. doi: 10.1145/3450268.3453529. URL https://doi.org/10.1145/3450268.3453529. 6.2.1

[224] Kenji Yamanishi, Jun'ichi Takeuchi, Graham J. Williams, and Peter Milne. On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms. *Data Mining and Knowledge Discovery*, 8(3):275–300, 2004. doi: 10.1023/B:DAMI.

0000023676.72185.7c. 4.3.2.2

[225] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18, New Orleans, Louisiana, USA, 2018. AAAI Press. ISBN 978-1-57735-800-8. 2.2.1, 3.3.2

[226] Ceyuan Yang, Yinghao Xu, Jianping Shi, Bo Dai, and Bolei Zhou. Temporal pyramid network for action recognition. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 588–597, Seattle, WA, USA, 2020. IEEE. doi: 10.1109/CVPR42600.2020.00067. 3.3.2

[227] Deju Yang, Liangli Ma, and Fei Liao. An intelligent voice interaction system based on raspberry pi. In *2019 11th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, volume 1, pages 237–240, Hangzhou, China, 2019. IEEE. doi: 10.1109/IHMSC.2019.00062. 2.1.2, 3.2

[228] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023. URL https://arxiv.org/abs/2210.03629. 2.3.2, 6.6.1

[229] A. Murat Yağcı, Tevfik Aytekin, and Fikret S. Gürgen. Balanced random forest for imbalanced data streams. In *2016 24th Signal Processing and Communication Application Conference (SIU)*, pages 1065–1068, May 2016. doi: 10.1109/SIU.2016.7495927. URL https://ieeexplore.ieee.org/document/7495927. 4.3.5

[230] Juan Ye, Graeme Stevenson, and Simon Dobson. Usmart: An unsupervised semantic mining activity recognition technique. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 4(4):1–27, 2014. 1.1, 5.1, 5.2

[231] Wen Ye, Wei Yang, Defu Cao, Yizhou Zhang, Lumingyuan Tang, Jie Cai, and Yan Liu. TS-Reasoner: Domain-Oriented Time Series Inference Agents for Reasoning and Automated Analysis, April 2026. URL http://arxiv.org/abs/2410.04047. arXiv:2410.04047 [cs]. 6.3.2.2

[232] Hyungjun Yoon, Hyeongheon Cha, Hoang C. Nguyen, Taesik Gong, and Sung-Ju Lee. IMG2IMU: Translating Knowledge from Large-Scale Images to IMU Sensing Applications, February 2024. URL http://arxiv.org/abs/2209.00945. arXiv:2209.00945 [cs]. 2.2.3

[233] Kristina Yordanova et al. Analysing cooking behaviour in home settings: Towards health monitoring. *Sensors (Basel, Switzerland)*, 19(3):E646, February 2019. ISSN 1424-8220. doi: 10.3390/s19030646. URL https://europepmc.org/articles/PMC6387167. 5.2

[234] Xiaofan Yu, Lanxiang Hu, Benjamin Reichman, Dylan Chu, Rushil Chandrupatla, Xiyuan Zhang, Larry Heck, and Tajana S. Rosing. Sensorchat: Answering qualitative and quantitative questions during long-term multimodal sensor interactions. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 9(3), September 2025. doi: 10.1145/3749496. URL

https://doi.org/10.1145/3749496. 6.2.1

[235] Chunhui Yuan and Haitao Yang. Research on k-value selection method of k-means clustering algorithm. *J*, 2(2):226–235, 2019. ISSN 2571-8800. doi: 10.3390/j2020016. URL https://www.mdpi.com/2571-8800/2/2/16. 5.3.2.3

[236] Sojeong Yun and Youn-kyung Lim. What If Smart Homes Could See Our Homes?: Exploring DIY Smart Home Building Experiences with VLM-Based Camera Sensors. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, CHI '25, pages 1–22, New York, NY, USA, April 2025. Association for Computing Machinery. ISBN 979-8-4007-1394-1. doi: 10.1145/3706598.3713265. URL https://dl.acm.org/doi/10.1145/3706598.3713265. 4.6.3

[237] Junhai Zhai, Sufang Zhang, Junfen Chen, and Qiang He. Autoencoder and its various variants. In *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, page 415–419, Miyazaki, Japan, 2018. IEEE Press. doi: 10.1109/SMC.2018.00080. URL https://doi.org/10.1109/SMC.2018.00080. 5.3.2.2

[238] Xiaohang Zhan, Jiahao Xie, Ziwei Liu, Yew-Soon Ong, and Chen Change Loy. Online deep clustering for unsupervised representation learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6687–6696, online, 2020. IEEE. doi: 10.1109/CVPR42600.2020.00672. 5.3.2.3

[239] Wenqi Zhang, Yongliang Shen, Zeqi Tan, Guiyang Hou, Weiming Lu, and Yueting Zhuang. Data-copilot: Bridging billions of data and humans with autonomous workflow, 2025. URL https://arxiv.org/abs/2306.07209. 6.2.1

[240] Xian Zhang and Xiang Cheng. Evaluation of Geolocation Capabilities of Multimodal Large Language Models and Analysis of Associated Privacy Risks, June 2025. URL http://arxiv.org/abs/2506.23481. arXiv:2506.23481 [cs]. 4.6.3

[241] Zhiping Zhang, Michelle Jia, Hao-Ping (Hank) Lee, Bingsheng Yao, Sauvik Das, Ada Lerner, Dakuo Wang, and Tianshi Li. "it's a fair game", or is it? examining how users navigate disclosure risks and benefits when using llm-based conversational agents. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, CHI '24, 2024. ISBN 9798400703300. doi: 10.1145/3613904.3642385. URL https://doi.org/10.1145/3613904.3642385. 2.5.3, 6.2.2

[242] Jiale Zhao, Ke Fang, and Lu Cheng. When and What to Ask: AskBench and Rubric-Guided RLVR for LLM Clarification, April 2026. URL http://arxiv.org/abs/2602.11199. arXiv:2602.11199 [cs]. 6.4.2.3

[243] Junwen Zheng and Martin Fischer. BIM-GPT: a Prompt-Based Virtual Assistant Framework for BIM Information Retrieval, April 2023. URL http://arxiv.org/abs/2304.09333. arXiv:2304.09333 [cs]. 6.2.1

[244] Bolei Zhou, Alex Andonian, Aude Oliva, and Antonio Torralba. Temporal relational reasoning in videos, 2018. 3.3.2

[245] Haozhe Zhou, Mayank Goel, and Yuvraj Agarwal. Bring privacy to the table: Interactive negotiation for privacy settings of shared sensing devices. In *Proceedings of*

the 2024 CHI Conference on Human Factors in Computing Systems, CHI '24, 2024. ISBN 9798400703300. doi: 10.1145/3613904.3642897. URL https://doi.org/10.1145/3613904.3642897. 6.2.2

[246] Sharif Noor Zisad and Ragib Hasan. LLMAC: A Global and Explainable Access Control Framework with Large Language Model. In *2026 IEEE 23rd Consumer Communications & Networking Conference (CCNC)*, January 2026. doi: 10.1109/CCNC65079.2026.11366514. URL https://ieeexplore.ieee.org/document/11366514/. 6.2.2, 6.1