From Gap to Synergy: Enhancing Contextual Understanding through Human-Machine Collaboration in Personalized Systems

Weihao Chen
chenwh20@mails.tsinghua.edu.cn
Tsinghua University
Beijing, China

Chun Yu*
chunyu@tsinghua.edu.cn
Tsinghua University
Beijing, China

Huadong Wang
wang-hd16@tsinghua.org.cn
Tsinghua University
Beijing, China

Zheng Wang
zhengwan20@mails.tsinghua.edu.cn
Tsinghua University
Beijing, China

Lichen Yang
ylc20@mails.tsinghua.edu.cn
Tsinghua University
Beijing, China

Yukun Wang
wang-yk21@mails.tsinghua.edu.cn
Tsinghua University
Beijing, China

Weinan Shi
swn@mail.tsinghua.edu.cn
Tsinghua University
Beijing, China

Yuanchun Shi
shiyc@tsinghua.edu.cn
Tsinghua University
Beijing, China

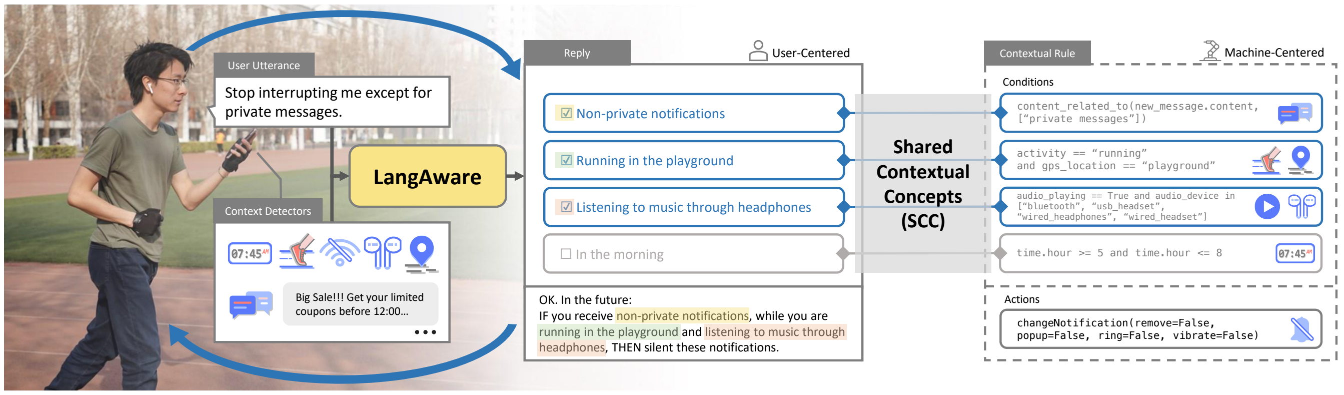

User Utterance
Stop interrupting me except for private messages.
LangAware
Context Detectors
07:45
Big Sale!!! Get your limited coupons before 12:00...
Reply
User-Centered
☑ Non-private notifications
☑ Running in the playground
☑ Listening to music through headphones
☐ In the morning
OK. In the future:
IF you receive non-private notifications, while you are running in the playground and listening to music through headphones, THEN silent these notifications.
Shared Contextual Concepts (SCC)
Contextual Rule
Machine-Centered
Conditions
content_related_to(new_message.content, ["private messages"])
activity == "running" and gps_location == "playground"
audio_playing == True and audio_device in ["bluetooth", "usb_headset", "wired_headphones", "wired_headset"]
time.hour >= 5 and time.hour <= 8
Actions
changeNotification(remove=False, popup=False, ring=False, vibrate=False)

Figure 1: A user is interrupted by notifications while running and listening to music on a playground. He talks in natural language with LangAware (left), which considers extra contexts and replies with an interactive interface displaying a natural language explanation (lower-middle) and several selectable phrases (upper-middle), each linked to a boolean expression (right). These pairs form human-machine Shared Contextual Concepts (SCCs). The user continues conversing until personalized contextual rules are established.

ABSTRACT

This paper presents LangAware, a collaborative approach for constructing personalized context for context-aware applications. The need for personalization arises due to significant variations in context between individuals based on scenarios, devices, and preferences. However, there is often a notable gap between humans and machines in the understanding of how contexts are constructed, as observed in trigger-action programming studies such as IFTTT. LangAware enables end-users to participate in establishing contextual rules in-situ using natural language. The system leverages large language models (LLMs) to semantically connect low-level sensor detectors to high-level contexts and provide understandable natural language feedback for effective user involvement. We conducted a user study with 16 participants in real-life settings, which revealed an average success rate of 87.50% for defining contextual rules in a variety of 12 campus scenarios, typically accomplished within just two modifications. Furthermore, users reported a better understanding of the machine's capabilities by interacting with LangAware.

*Corresponding author.

UIST '23, October 29–November 01, 2023, San Francisco, CA, USA
© 2023 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0132-0/23/10.
https://doi.org/10.1145/3586183.3606741

CCS CONCEPTS

• **Human-centered computing → Interactive systems and tools; Ubiquitous and mobile computing systems and tools.**

KEYWORDS

End User Context Construction, Context-Aware Systems, Large Language Models, Personalization

ACM Reference Format:
Weihao Chen, Chun Yu, Huadong Wang, Zheng Wang, Lichen Yang, Yukun Wang, Weinan Shi, and Yuanchun Shi. 2023. From Gap to Synergy: Enhancing Contextual Understanding through Human-Machine Collaboration in Personalized Systems. In *The 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23), October 29–November 01, 2023, San Francisco, CA, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3586183.3606741

1 INTRODUCTION

With the development of AI and IoT technologies, intelligent systems have increasingly powerful context-aware capabilities. For example, smartphones can provide automated services by sensing users' physical contexts such as locations and activities, and social contexts such as contacts and messages. Context-aware applications face different user scenarios, devices, and preferences, and developers cannot anticipate every possible user context and make targeted adaptations [38]. Simultaneously, data-driven machine learning algorithms struggle to learn user preferences in the face of sparse personalized contexts [25, 28, 36, 45, 46]. These approaches do not effectively involve end-users in leveraging their personalization knowledge for context-aware applications.

Trigger Action Programming (TAP) is a popular paradigm that allows users to customize their own IF-THEN rules for personalized systems. However, end-users often struggle to construct personalized contexts without knowledge of the device's perceptual capabilities [12, 39, 40]. There is a significant gap between human and machine understanding of context construction [5].

To bridge this human-machine context gap, we investigate a collaborative approach in which the machine assistant and the user continuously align their understanding of context through natural language within the context. This in-context approach differs from out-of-context methods, as users are more likely to express their needs and adapt their expressions within the context. We identify several challenges in realizing this collaborative process through a formative study, such as ambiguity in context expressions, omission of context information in user input language, and providing comprehensible interfaces.

We propose LangAware, a system that addresses these challenges and enables end-users to establish in-situ contextual rules using natural language. LangAware leverages large language models (LLMs) to semantically bridge high-level contextual concepts expressed by users with the underlying sensing capabilities that machines can perceive and implement. We propose Shared Contextual Concepts (SCCs) as the medium for contextual understanding alignment between humans and machines. An SCC is a pair of a natural language phrase and its corresponding machine boolean expression.

In an example usage scenario of LangAware, as shown in Figure 1, a user is running in the playground, wearing headphones to listen to music, but becomes annoyed by frequent message alerts. The user tells the assistant, "*Stop interrupting me except for private messages.*" The assistant combines contextual information from the phone to understand the user's linguistic input and generates candidate SCCs. These include both "non-private notifications" derived from the user's utterance and "running in the playground" obtained from contextual information. The user selects the appropriate SCC in an interactive interface or further modifies it through dialogues until a satisfactory contextual rule is generated.

To evaluate LangAware, we conducted a user study with 16 participants using Android smartphones in real-life settings, covering 12 campus contextual tasks. We compared LangAware with a baseline system that only uses language **without** incorporating in-situ context information. Results show that users generated high-quality SCCs (average expert sufficiency score 4.50/5 and necessity score 4.50/5) with an average success rate of 87.50% using LangAware, reflecting a 18.75% improvement over the baseline. LangAware also significantly outperformed the baseline in several subjective scoring metrics such as user satisfaction, perceived machine understanding, mental effort, and physical demand. The SCCs generated by LangAware during the experiment also exhibited personalization characteristics.

In conclusion, our contribution is LangAware, a novel collaborative approach that empowers end-users to construct personalized contexts in situ using natural language. This approach addresses the human-machine context gap by using LLMs to connect different levels of semantics and SCCs to align human-machine contextual understanding. This work contributes to improving the intuitiveness and adaptability of human-machine interactions, facilitating the development of more personalized context-aware applications in IoT scenarios.

2 BACKGROUND AND RELATED WORK

Most existing devices provide limited personalization settings that are predefined and simplistic, which is hard to accommodate the diverse needs of users. Solutions like IFTTT or iOS's Shortcuts and Focus mode, while supporting automated services based on contextual profiles, necessitate user interaction through a graphical user interface (GUI) and involves complex operations that may be less user-friendly. This paper explores ways for end users to better define personalized contexts by employing natural language dialogue and in-situ context information. In this section, we revisit related work on context-aware systems and end-user context construction and conclude by presenting how LLMs are promising in bridging contextual semantics.

2.1 Context-Aware Systems

Context-aware systems provide customized services based on the user's context [16]. With advancements in IoT and AI technologies, smart devices can better utilize physical sensing information such as time, location, activity, app usage, nearby Wi-Fi and Bluetooth devices, and social information embedded in contacts and message content to identify and describe the user's context. Rich contextual information allows context-aware systems to offer various automated services, such as recommendation systems [42], message or caller management [20, 23, 25, 28, 32], app recommendations [10], power management [24, 33], and configuration management for volume or brightness [1, 2].

Many context-aware systems use data-driven machine learning approaches to learn users' contextual preferences [4, 34]. However, these approaches face several issues when geared toward personalized contexts. Firstly, they require a substantial amount of user data gathered over an extended period ranging from several weeks to months [25, 28, 36, 45, 46], which may be inefficient for obtaining personalized knowledge when personalized contextual and behavioral data are sparse. Secondly, many context-aware systems act as black-box AI for users, making it difficult for them to understand the system's behavior and participate in controlling it [3, 6, 17, 44]. Therefore, it is essential to involve users effectively in the contextual preference learning process, which enables them to provide personalized knowledge and have more understanding and control over the system.

2.2 End User Context Construction

Much work has been done on involving end-users and supporting them to customize context-aware systems tailored to their needs [26]. There is a need to lower the programming threshold given that most end-users lack programming skills. Trigger-action programming (TAP), i.e., "IF an event happens, WHILE conditions are true, THEN take actions", is one important paradigm. Studies show that end-users primarily use IF-THEN rules to express their contextual needs [18], and TAP rules can express users' needs in smart homes [39]. TAP is also adopted by many commercial platforms, such as IFTTT [21].

However, TAP programming still presents a threshold for end-users. One issue is that TAP rules involve more underlying machine details, making it difficult for end-users to compose context conditions [12, 39, 40]. To address this, some works propose higher-level abstractions for machine context awareness to support end-users in constructing personalized contexts. Examples include providing better visual programming environments [15, 19]. However, these approaches still require users to navigate through the GUI at a higher learning cost after understanding the context-aware capabilities supported by the system. Another work [27] proposes a block-based programming environment specifically for designers (instead of end-users) to develop location-based context-aware applications that translate human concepts into machine feature expressions. Some works [13, 14] allow users to input natural language instructions and recommend a collection of TAP rules that match the abstract concepts therein; however, these approaches use expert-defined semantic ontologies for a specific platform (i.e. IFTTT) to convert users' abstract concepts into concrete machine semantics, which limits their scalability. The aforementioned works primarily focus on constructing contexts out of real-world situations and do not address users' creation of contexts in situ.

In contrast, our work targets end-users without specialized knowledge and supports constructing personalized contexts in situ. We consider both natural language input and contextual information to make user interaction more natural and less costly. Simultaneously, we leverage the NLU capabilities of LLMs and the embedded generic knowledge to complete the semantic connection, providing greater scalability and flexibility compared to expert-defined platform-specific semantic networks.

2.3 Large Language Models

Language models are used to model the probability of text sequences, while generative language models predict likely output sequences based on given inputs. With the significant increase in model size and amount of text data used to train transformer-based [41] language models in recent years, large language models (LLMs) such as GPT-3 [8], LaMDA [37], PaLM [11], and ChatGPT [30] have emerged with the ability to generalize across new tasks without requiring re-training. To use LLMs effectively, users need to carefully write natural language instructions, or prompts, to enable LLMs to perform the required tasks on demand. The excellent performance of LLMs for many new tasks shows that it embeds rich semantic knowledge into the model while learning a huge amount of corpus, and the process of prompting is the process of extracting the knowledge from LLMs.

LLMs offer new opportunities for human-computer interaction [7]. They are capable of handling the diverse and rich natural language expressions of end-users and understanding their intentions more accurately, enabling users to build their applications more easily without specialized knowledge. LLMs embedded with rich knowledge can also serve as a glue for data from multiple modalities and use it to enhance contextual understanding and better task execution [29].

In this paper, we leverage LLMs to translate users' natural language into machine expressions and abstract user-understandable concepts from machine expressions. By feeding contextual data into LLMs, we can enable them to more effectively understand the user's context, comprehend their utterances, and assist in the collaboration of contextual rules.

3 THE HUMAN-MACHINE CONTEXT GAP

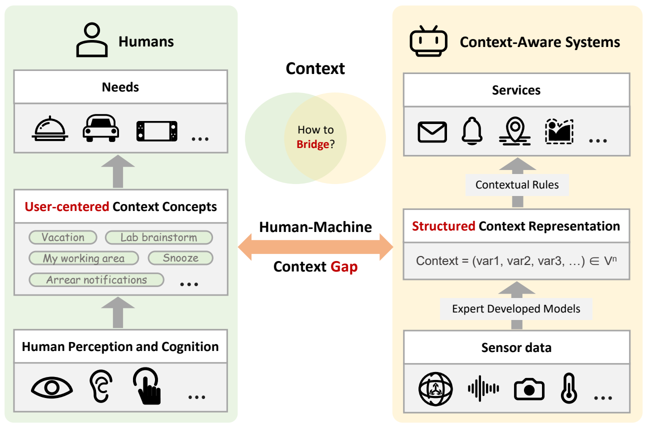


Figure 2: The human-machine context gap.

There is a gap between human understanding and that of context-aware systems on how contexts are constructed [5], as illustrated in Figure 2. Firstly, humans and machines have different perceptual channels, resulting in differences in the input information they collect. Secondly, humans tend to comprehend situations through concepts that can be abstract (e.g., "*at work*") or concrete (e.g., "*at the office*"), which are not strictly independent of each other. In contrast,

Table 1: Challenges in natural language-based human-machine context bridging.

Challenges	Examples
Understanding User Language	
Ambiguity in Context Expressions	"*At work*" could refer to being engaged in work-related activities, being physically present at the workplace, or using work-related apps.
Omission of Context Information	Users might say "*in such situations*" without specifying what the situation is, or mention a service, such as simply stating "*mute*", without indicating the relevant contextual conditions.
Providing Comprehensible Interfaces	
Explanatory Affordance	Users want to understand the meaning of `nearby phone number > 0` through natural language (potential human presence); users expect feedback when the system cannot detect conditions like "*an unattended item*".
Concept-level Controls	Users want to remove "*morning*" from "*running in the morning*" or add a condition like "*listening to music*", even if these concepts may not directly correspond to preset detectors.

machines construct contextual rules using structured contextual variables and service interfaces that are semantically explicit and can be combined into complex contextual expressions and service chains. This discrepancy leads to usability issues for end-users, such as an insufficient understanding of machine capabilities, difficulty in accurately anticipating results from machine rules, and a lack of expertise in transforming unstructured needs into structured machine rules when expressing their needs. Additionally, machines often struggle to effectively summarize high-level semantics in context-aware services, presenting users with relatively low-level specific rules instead. This human-machine context gap limits users' ability to transfer personalized knowledge to machines.

To bridge this gap, we aim to find an effective way for users to interact with machines. As humans predominantly use natural language to communicate with human assistants, we explore how human and machine assistants can generate contextual rules through natural language in specific situations. We conducted a formative study on smartphone automation tasks to understand how end-users express their contextual preferences to machine assistants and collaborate in generating rules in daily life situations. This study provides insights into the challenges of addressing the human-machine context gap and serves as a guide for subsequent system design.

3.1 Study Design

We adopted a Wizard of Oz design, where participants imagined themselves in various daily life situations and interacted with a smartphone assistant controlled by a human wizard behind the scenes. The wizard had expertise in developing context-aware systems.

Six participants (5 males and a female, aged between 22 to 25) completed the study, which took each participant 40 minutes to 1 hour. We compensated participants for their time. Before the study, we asked about their experiences with automated services based on contextual profiles. One participant had no relevant experience; two participants had set up services to send messages or activate Do Not Disturb mode at specific times; another two participants had further configured location-based services such as turning on a QR code; and the final participant had used the Smart Activation feature in iOS's Focus mode, but found that it sometimes did not fit with their daily routines.

During the study, we introduced participants to a smartphone assistant with advanced context-awareness and human-like comprehension. Each participant completed 4-6 tasks, where they were asked to imagine themselves in a specific context and express their preferences for configuring automation rules. Participants were presented with preset tasks in a random order that included story narratives and illustrative background photos, and they were allowed to adjust the tasks according to their own experiences. Communication with the assistant took place through a chat interface where a wizard simulated the assistant's response. Feedback was gathered at the end of each task, and a concluding interview was conducted to gather additional insights from the participants.

The preset tasks provided to participants were divided into four categories: lab work, library study, playground sports, and dormitory break. Each task might concern services such as message management, schedule management, volume control, and network switching. The wizard generated contextual IF-THEN rules that could be implemented on the smartphone by considering both the natural language expressed by the user and the contextual information that could be perceived on the phone. The wizard then communicated the rules back to the user in natural language that was understandable to the end-user.

3.2 Findings

The study involved a total of 30 tasks. Of these tasks, only 7 were confirmed by the user after the assistant's first response, while 21 tasks involved collaboration with the wizard to modify the context conditions (19) or services (2). Additionally, 7 tasks required clarification of the context conditions. Our findings suggest that users are typically able to describe their required services accurately. However, there are significant challenges in communicating context conditions between the user and the assistant, as we have outlined and presented in Table 1. These challenges include difficulties with understanding user language and providing comprehensible interfaces.

At the input level, there are two primary challenges regarding the language used by end-users within the context.

Ambiguity in Context Expressions: Users may employ vague or ambiguous terms that are difficult to convert into machine-readable expressions. In the study, the wizard relied on their own comprehension to identify potentially relevant machine boolean expressions from the user's ambiguous expressions. This underscored the need for machine assistants to possess strong language comprehension abilities and to be able to link high-level natural language with low-level context detectors. Moreover, since machine expressions corresponding to user language are not always unique, the machine must allow for modifications of the generated results by the user.

Omission of Context Information: Users may inadvertently leave out critical contextual information, making it challenging for the machine to comprehend their language. As users are aware of the wizard assistant's context-awareness, they may sometimes omit some or all of the contextual descriptions, such as simply stating the service instructions or using contextual descriptions with referents (e.g., "*in this case*"). Users also commented that this approach reduces the cost of verbal expression and feels more natural. This highlights the importance of machine assistants incorporating contextual data to understand the user's language.

Providing comprehensible interfaces to users in the feedback and subsequent interaction stages also poses two challenges.

Explanatory Affordance: Users require natural language explanations of machine expressions and possible failures. During the study, participants expressed a desire to understand how a specific context was recognized by the machine, and the wizard was required to use their expert knowledge to provide user-friendly explanations. This highlights the need for machine assistants to possess the ability to interpret low-level machine rules, especially boolean expressions composed of multiple detectors, into high-level language that is meaningful to the user. Additionally, users require feedback when the system fails to construct machine expressions for certain contexts, necessitating that the machine assistant has a clear understanding of the limits of the system's context-aware capabilities and can determine when the context referred to by a particular user language input is beyond the system's perceptual abilities.

Concept-level Controls: In addition to modifying boolean expressions, users need the ability to make modifications at the concept level. When modifying context conditions, users generally do not use instructions that are consistent with the machine's predefined detectors (nor are they necessarily aware of the machine's capabilities), but instead, use high-level concepts as units to modify. In rare cases, they may be interested in the machine's boolean expressions corresponding to the concept. Therefore, the machine needs to provide the user with a more flexible modification interface to accommodate the use of high-level concepts for modifying context conditions.

4 LANGAWARE

Inspired by the formative study, we present LangAware, a context-aware natural language dialogue approach that allows end-users to actively construct personalized context rules within a context through natural language and simple interactions on the GUI. With the help of LLM's natural language understanding capabilities, end-users can effectively interact with the machine using comprehensible natural language concepts, bridging the human-machine context gap. In this way, end-users can continuously teach the machine to build their personalized context-aware applications. For additional LangAware use cases, please refer to Appendix A.

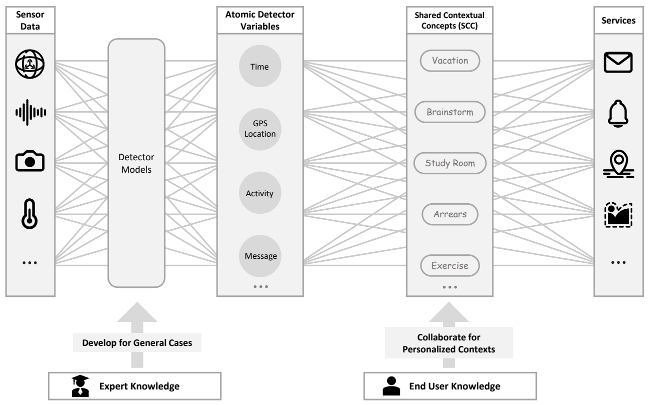


Figure 3: Building personalized context-aware systems involves different layers of context abstraction and knowledge from both experts and end users.

LangAware is designed to be adaptable to various context-aware systems, allowing for seamless integration once the detectors and services are predefined. Our approach can effectively shield users from machine details while better catering to their personalized needs. As shown in Figure 3, we decompose the context-aware system into several layers: sensor data, atomic detectors based on the processed sensor data, Shared Contextual Concepts (which we propose and will discuss further below), and the services they ultimately connect to.

Expert knowledge is used to build connections from sensor data to atomic context detectors, encoding the system's basic explainable context-aware capabilities (with semantic labels) as building blocks for subsequent context-aware applications. End-user knowledge is used to construct mappings from context detectors to Shared Contextual Concepts (context conditions in the context rules, i.e., the IF part), as well as connections from Shared Contextual Concepts to the required services (service actions in the context rules, i.e., the THEN part).

As observed in the formative study, the main gap between humans and machines lies in context construction. In terms of services, since users usually communicate with machine assistants based on known system-provided services, it is relatively easy to match services that meet user utterances from a predetermined list through semantics. For more complex service recommendation scenarios where users may not be aware of all services, the difficulties in human-machine language communication are similar to the difficulties in human-machine context understanding. Therefore, we focus on the context construction process from context detectors to Shared Contextual Concepts.

We consider giving machine assistants the ability to perceive context beyond language, allowing human users and machine assistants to communicate through natural language within the context. To address the challenges raised in the formative study, we further explore important design features in LangAware.

4.1 Shared Contextual Concepts

One of the major challenges in human-machine communication is that the context-aware semantics are different based on varying context-aware capabilities. Human concepts and machine expressions, which are combinations of detector variables, do not necessarily equate. However, the end-users' utterances of context must be grounded in the context-aware capabilities supported by the system. To address the Explanatory Affordance Challenge mentioned in the formative study, we introduce human-machine Shared Contextual Concepts (SCCs).

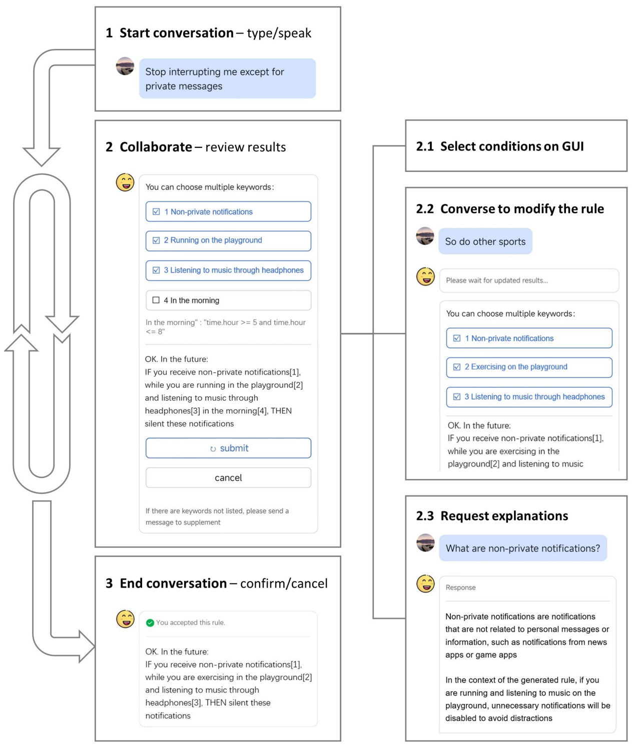
Specifically, we define an SCC as a pair of a natural language phrase and a machine Boolean expression, where the phrase serves as an interpretation of the expression, and the expression provides a concrete description of the phrase condition based on the machine's capabilities (see the middle and right sections of Figure 1). In the context rules generated by LangAware, the IF situational condition part will be composed of several such SCCs connected by `and`.

SCCs facilitate human-machine understanding of context and serve as an intermediary between human concepts and machine expressions. They are derived from both users' utterances as well as the machine's perception and comprehension capabilities, while still maintaining a high level of human understandability. For phrases in user utterances that cannot be perceived by the system, they do not have corresponding expressions and therefore cannot form SCCs. This will be reflected during the SCC generation process (see Section 4.3) and be fed back to the user.

When explaining contextual rules to users, the natural language phrases in SCCs can shield the details of machine expressions, allowing users to focus on high-level semantics, which is particularly helpful when multiple concepts are involved in the rules. At the same time, this design is scalable. The natural language phrases in SCCs can be abstract or specific, which is an inherent flexibility of natural language. As the capabilities of context-aware systems change in the future (e.g., by adding new detectors), the corresponding machine expressions for the same natural language semantics will also be expanded.

4.2 Conversational Interface

As we can see from the formative study, even human-simulated assistants cannot generate results that fully meet users' intentions in a single round of interaction. Users need to modify and check the machine-generated results through multiple rounds of interaction, continuously aligning their understanding of the context with the machine. At the same time, the formative study also mentioned the need to provide users with Concept-level Controls. Therefore, LangAware builds a natural language dialogue interface based on the proposed SCCs, allowing end-users to create and modify contexts in multiple rounds of interaction, ultimately generating contextual rules.


1 Start conversation – type/speak
2 Collaborate – review results
2.1 Select conditions on GUI
2.2 Converse to modify the rule
2.3 Request explanations
3 End conversation – confirm/cancel
1 Non-private notifications
2 Running on the playground
3 Listening to music through headphones
4 In the morning
submit
cancel
1 Non-private notifications
2 Exercising on the playground
3 Listening to music through headphones
Response

Stop interrupting me except for private messages

You can choose multiple keywords:

In the morning" : "time.hour >= 5 and time.hour <= 8"

OK. In the future:
IF you receive non-private notifications[1], while you are running in the playground[2] and listening to music through headphones[3] in the morning[4], THEN silent these notifications

If there are keywords not listed, please send a message to supplement

So do other sports

Please wait for updated results...

You can choose multiple keywords:

OK. In the future:
IF you receive non-private notifications[1], while you are exercising in the playground[2] and listening to music

What are non-private notifications?

Non-private notifications are notifications that are not related to personal messages or information, such as notifications from news apps or game apps

In the context of the generated rule, if you are running and listening to music on the playground, unnecessary notifications will be disabled to avoid distractions

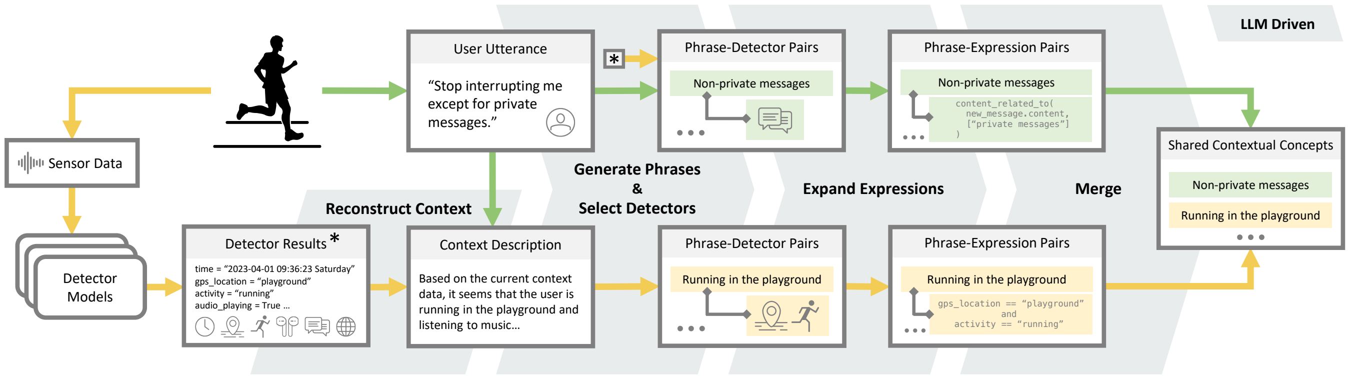
You accepted this rule.

OK. In the future:
IF you receive non-private notifications[1], while you are exercising in the playground[2] and listening to music through headphones[3], THEN silent these notifications

Figure 4: Interaction workflow with LangAware. The "Collaborate" step consists of three options.

As shown in Figure 4, users actively initiate a conversation with LangAware within the context. LangAware will respond with an interactive interface based on the current context and the user's language input. This includes a natural language interpretation of the generated contextual rule and the SCCs involved in the interpretation. Each SCC is displayed as a selectable checkbox, and users can also view the corresponding machine boolean expressions if they wish. If there are phrases in the user's language input that cannot be perceived, the interface will indicate this information. Users can directly continue the conversation to modify or supplement SCCs, or the services used. LangAware will adjust based on the generated results and then resend a modified result. Users can also ask about the meanings of various parts of the contextual rule using natural language, and LangAware will reply with a natural language explanation. Finally, when the user is satisfied with the collaboratively generated contextual rule, they can click "confirm" or "cancel" if not.

Based on this design, we hope that users can effectively generate and edit contextual rules through natural language as much as possible, without being constrained by a lack of professional knowledge. At the same time, this collaborative process not only brings the machine closer to the context concepts intended by the user but also allows users to continuously understand the machine's capabilities in perceiving context, gradually closing the human-machine context gap.



time = "2023-04-01 09:36:23 Saturday"
gps_location = "playground"
activity = "running"
audio_playing = True ...

Based on the current context data, it seems that the user is running in the playground and listening to music...

```
content_related_to(
  new_message.content,
  ["private messages"]
)
```

```
gps_location == "playground"
          and
activity == "running"
```

Figure 5: The generation process of Shared Contextual Concepts. Green arrows represent the data flow of user's natural language input, yellow arrows represent the data flow of context awareness, and they eventually merge into Shared Contextual Concepts. The steps highlighted with a gray background are driven by the LLM.

4.3 Context Construction by Integrating Language and Contextual Data

LangAware generates SCCs that meet users' needs based on their natural language input and real-time context data. In response to the challenge of Ambiguity in Context Expressions discovered in the formative study, we leverage the powerful NLP capabilities of LLM to extract structured information, converting the semantics expressed by users into detector expressions supported by context-aware systems. To address the challenge of Omission of Context Information, we need to supplement it with context data and utilize the semantic connection capabilities inherent in the LLM to abstract into more general user concepts.

In more detail, we have designed the generation process as shown in Figure 5. When a user initiates a request to build a contextual rule in a specific context, LangAware first obtains the contextual state values (such as current time, location, foreground app on the phone, user's physical activities, etc.) and recent detector events (e.g., receiving a new message, location change, noise change, etc.) from the detector models in the context-aware system.

These models are expert-built computational models that can recognize semantically meaningful text labels from sensor data (not only hardware sensors but also graphical interfaces, messages, social media, and other forms of software sensors). These models may be deep (e.g., using neural networks to compute user motion states from IMU time series data) or white-box (e.g., determining user location information based on GPS coordinates). There has already been extensive research on these types of problems [9, 22, 31, 35]. Then, we convert the contextual state values into "variable_name = value" text and express detectable events on the phone within a short period (e.g., 5 minutes) as a list of text, including event name, event change content, and timestamp. Timestamps are described relative to the current time (e.g., "2 min 35 sec ago"). We concatenate the above text, calling it context detector results.

An essential step in our generation process is context reconstruction. Context detector results provide a low-level description of the user's context, but to more accurately capture user-intended SCCs, we need a high-level, integrated context description. Thus, we use the LLM to combine context detector results and user utterance to generate a short natural language text, inferring the user's situation. This step is effective due to both the general knowledge possessed by the LLM and the interpretable naming of detector variables by experts.

Next, we process the user utterance and context reconstruction description separately. From the high-level natural language text, we extract key phrases as context conditions, first finding the corresponding detector variables for each phrase through detector selection, and then expanding a boolean expression that matches the phrase's semantics based on the chosen variables. It is worth noting that if a phrase cannot be detected, the output of the detector selection will be empty, which will serve as the basis for our subsequent feedback to the user. Additionally, the expression expansion step utilizes the LLM's general knowledge to recommend as many possible values for each detector (e.g., motion includes running, cycling, jumping, etc.). The results generated from both the language and context data streams will eventually merge into SCCs and fill in the user interface.

After the first round of results is generated, LangAware takes into account the generated rules when the user subsequently modifies the contextual rule (whether it's an SCC or a service). Based on this design, we utilize the LLM to connect user natural language semantics with contextual data semantics from sensors, enabling a better understanding and fulfillment of user intentions, and allowing the contextual assistant to continuously engage with users in a more comprehensive context.

5 EVALUATION

We conducted a real-world user study to evaluate the usability of our system. To test the effectiveness of in-situ context information in enhancing the generation of contextual rules, we implemented LangAware on Android smartphones and compared it with a baseline version that only uses language input without considering context information. We used a within-subject design, asking all participants to use both the LangAware and baseline versions of the assistant.

5.1 Implementation

Both LangAware and the baseline version use Feishu Bot[1] to build the front-end conversation interface, allowing users to input text through either typing or voice. Feishu Bot is controlled by our remote server, which is also connected to the context library running in the background on our smartphones.

Context rule generation is implemented on the remote server, including two parts: context condition generation and service mapping. The context condition part consists of a series of SCCs, where the boolean expressions are in Python syntax, and the context detector variables used are shown in Appendix Table 2. Specifically, we introduce a new detector for natural language text: `content_related_to(content, keywords)`, which determines whether the `content` is related to the keywords in the `keywords` list. For example, for "*advertisement message*", it can be expressed as `content_related_to(new_message.content, ['advertisement'])`.

The service action component can either be chosen from our pre-defined service APIs (including volume adjustment, modification of notification modes for new messages, call mode alteration, network switching, etc.) or be generated by the LLM itself, adhering to the Python function call syntax.

We use the `gpt-3.5-turbo-0301` model from the ChatGPT API[2] as our driving LLM, due to its strong capabilities in following instructions. Our prompts are designed using an instruction-examples structure that is well-suited for LLMs without relying on version-specific features. This design allows our system for easy adaptation to newer versions of the LLM, such as GPT-4.

Compared to LangAware, the baseline version does not accept the context data when users input language, i.e., the context data path represented by the yellow arrow at the bottom of Figure 5 is removed. Both versions share the pre-defined detector variables and service APIs.

Further implementation details are provided in Appendix B.

5.2 Participants

We recruited 16 participants (different from those in the formative study) to evaluate these two systems in real-world scenarios. The participants consisted of 8 males and 8 females, aged between 18 and 27 years old. Each session took 40 minutes to 1 hour, and all participants received compensation for their time.

Before the study, we inquired about participants' experience with IF-THEN rule programming (e.g., IFTTT). Only 2 participants reported having used such tools, while 4 participants indicated familiarity with them but no prior usage. The main reasons cited by participants for not using these tools included high usage costs and perceived limited practical benefits.

5.3 Procedure

Our study comprised two stages.

In the first stage, we engaged end-users with the assistants in real-world scenarios. To avoid engineering problems caused by device differences, we asked participants to use the Android phones we provided uniformly. Initially, we introduced the functions and usage of the assistants to the participants. To aid participants in effectively expressing their needs based on the assistant's capabilities, we outlined the functional differences between the LangAware and the baseline version. From 12 preset university campus scenario tasks, We randomly assigned 4 tasks to each participant, with every two tasks conducted at the same location. In each task, both LangAware and the baseline version were used (the order was counterbalanced across participants), resulting in 8 trials for each participant and 128 trials in total.

We employed a customized NASA-TLX based evaluation for each trial. After each trial, participants rated the following aspects (excluding the evaluation of waiting time for assistant response) on a 7-point Likert scale, with 1 being the most negative experience and 7 the most positive:

- How mentally demanding was it to use this assistant?
- How physically demanding was it to use this assistant?
- How hurried or rushed was the pace of the interaction with this assistant?
- How well do you think this assistant understands your intentions?
- How satisfactory do you think the final result generated by this assistant is?

Once all 8 trials were completed, participants evaluated their overall experience with each version of the assistant on the same 7-point Likert scale:

- To what extent did this assistant increase your understanding of its context-aware capabilities?
- How easy was it to learn how to use this assistant?
- How easy was it to generate context rules using this assistant?
- How likely are you to use this assistant again in the future?

Concluding interviews were conducted to acquire a deeper understanding of participants' thought processes and verbal challenges; to assess their comprehension of SCC descriptions and identify any potential discrepancies in assistant's comprehension; and to ascertain whether they paid attention to, understood, and expressed a desire to edit the boolean expressions.

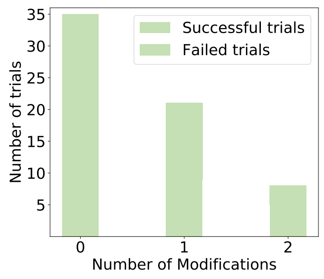
In the second stage, we invited three experts with experience in designing and developing context-aware systems to evaluate the contextual rules generated by LangAware. Referring to the pre-defined detector variable list and user utterances, experts assessed the sufficiency (how well a machine expression implies a phrase) and necessity (how well a phrase implies a machine expression) of the generated SCCs on a 5-point Likert scale. Additionally, they consulted the service API list to evaluate the accuracy of function selection and parameter generation in the action part of the rules.

5.4 Tasks

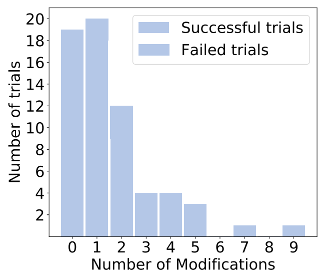
We have preset 12 campus scenario tasks divided into four categories:

- Quiet library/study room with nearby students: prevent sudden audio output from the phone while watching videos casually; automatically play light music when using a focus app with headphones on; don't output audio when the Bluetooth headset suddenly disconnects; block advertisement notifications to avoid disturbing studies.

[1]Feishu Open Platform: https://open.feishu.cn/
[2]ChatGPT API: https://openai.com/blog/introducing-chatgpt-and-whisper-apis



(a) baseline



(b) LangAware

Figure 6: Histogram of the quantity of trials based on the number of modifications made using (a) the baseline and (b) LangAware.

- Noisy sports field: avoid being disturbed by non-private chat messages while running and listening to music with headphones on; automatically start sports check-in records when entering the sports field and start running, and automatically end records when stopping exercise.
- Dormitory with/without roommates present: set ringtone reminders for messages from teachers; loudly remind about food delivery-related messages; ensure the alarm clock is turned on when there are morning classes; do not output audio from videos when others are present in the dormitory.
- Noisy cafeteria: ring and vibrate when receiving messages; increase the volume when listening to music or making phone calls.

In addition, if users do not have real-life experiences related to the preset task, we allow them to adjust it according to their personal circumstances or propose a new task that genuinely addresses their needs.

5.5 Results

5.5.1 *Success rate.* Participants were allowed to discontinue rule generation according to their actual needs and when the perceived cost of generation exceeded their tolerance. Out of 64 trials performed by 16 participants using the baseline system, 44 trials were successfully completed. In contrast, the number of successful trials using LangAware increased to 56 out of 64 trials. The difference in success rates between the baseline (M = 68.75%, SD = 21.41%) and LangAware (M = 87.50%, SD = 15.81%) was confirmed to be statistically significant ($t_{15} = -3.22$, $p < .01$) through a paired-samples t-test. This enhanced success rate highlights the practical value of LangAware in effectively incorporating context information.

5.5.2 *Collaboration.* The average interaction time for successful trials (excluding waiting time for the assistant's response) was 43.04s (SD = 27.72) for the baseline and 78.50s (SD = 50.11) for LangAware.

During the successful trials using LangAware, there were 82 modifications, with 67 (81.71%) being click modifications (each change in the selected state of an SCC upon submission is counted as a click modification), and 15 (18.29%) being natural language modifications (each text message sent, excluding the first input, is counted as a language modification). This shows that most modifications were done via clicks, indicating that LangAware effectively reduced the user interaction cost. As shown in Figure 6(b), the average number of modifications made in successful trials using LangAware was 1.46, and 46 successful trials (82.14%) were completed with two or fewer modifications. This demonstrates the efficiency of the collaboration process. However, one participant, P7, strayed from the experimental instructions during initial use and explored the interface freely, resulting in 9 modifications. This behavior was not repeated in his subsequent trials.

Comparatively, the baseline version had a lower average modification number of 0.43 (19/44) in successful trials, as shown in Figure 6(a). However, only 2 (10.53%) of these modifications were clicks on the GUI, with the majority completed through natural language. This might mean a higher cost of supplementing necessary information compared to LangAware. Considering that the success rate of the baseline is lower than LangAware, this suggests that the baseline version, which generates SCCs based solely on user utterances, may struggle to help users complete their intended goals.

5.5.3 *Subjective ratings.* Figure 7 presents the subjective rating results for both the baseline and LangAware. We utilized the Wilcoxon signed-rank test to measure the impact of context augmentation on various metrics. The results show that users' ratings of low mental effort (W = 106.00, $p < .05$) and low physical demand (W = 90.00, $p < .05$) significantly improved with the adoption of the context-augmented LangAware system. Ratings of the perceived machine understanding (W = 122.50, $p < .05$) and users' satisfaction with the final generated rules (W = 130.50, $p < .05$) were also significantly improved. After completing all trials, LangAware outperformed the baseline version in terms of users' enhanced understanding of machine capabilities (W = 107.50, $p < .05$) and their willingness to reuse the system (W = 125.00, $p < .05$).

These improvements indicate that reconstructing user contexts based on sensor data effectively supplements important information potentially absent from user utterances, enabling users to generate more comprehensive and thorough SCCs with lower costs. As a result, LangAware allows users to construct their personalized contexts more quickly, accurately, and naturally.

5.5.4 *Generated rules.* Expert evaluations of the successfully generated rules from LangAware indicate high accuracy in mapping high-level user concepts to low-level machine expressions. The SCCs achieved an average sufficiency score (how well a machine expression implies a phrase) of 4.50/5 (SD = 0.77) and an average necessity score (how well a phrase implies a machine expression) of 4.50/5 (SD = 0.71); the actions had an average accuracy score of 4.71/5 (SD = 0.54). These scores demonstrate the system's ability to effectively detect corresponding user concepts in real environments, bridging the user's high-level semantics and the machine's low-level semantics.

Additionally, out of the 325 SCCs generated by LangAware, 195 (60.00%) involved either combinations of multiple detector variables (abstraction) or multiple value conditions for the same variable, which cannot be directly obtained from the current context. Despite their complexity, these SCCs received an average sufficiency score of 4.41/5 (SD = 0.82) and an average necessity score of 4.37/5 (SD =

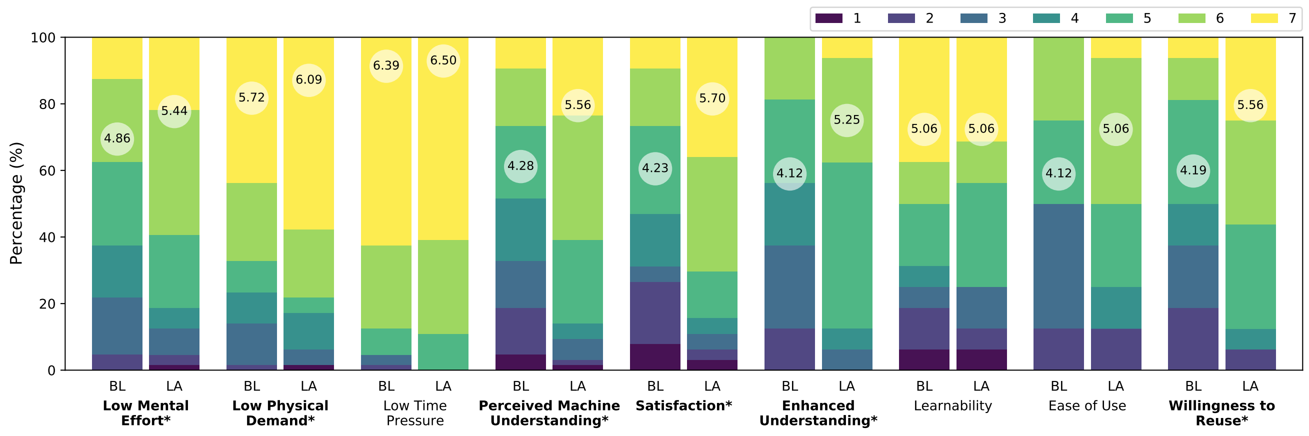


Figure 7: Subjective ratings of the baseline (BL) and LangAware (LA) represented in a stacked bar chart. Higher ratings indicate better performance. The numbers within the white circles denote the mean value of each bar. Measurements marked with an asterisk (*) denote a significant difference between the two systems.

0.76) from experts, further demonstrating the system's effective abstraction and generalization capabilities without compromising the quality of generated rules.

5.6 Discussion

In general, participants were satisfied with LangAware. During interviews, 11 out of 16 participants reported that expressing their intentions to the assistant was not difficult and that they could communicate in a conversational manner. The majority of participants (excluding P1, P6, and P8) appreciated the context-aware assistant, which could provide them with SCCs that they had not considered but actually needed. The assistant sometimes exceeded their expectations. Participants found it easier to select and edit natural language phrases than to create a phrase from scratch.

LangAware demonstrates strong support for personalization in many aspects. Each task scenario in the study was completed by 5 or 6 participants. Interestingly, different participants exhibited a wide range of personalized characteristics within the same context and task. It is not surprising that users have different utterances for the same concept, such as "*having a meal*", "*in the canteen*", or "*in the restaurant*", which all correspond to the boolean expression `gps_position == 'canteen A'`. What is worth noting is that different participants reasoned different context conditions for the same task. For example, in the task of avoiding disturbing others, P6 used "*in a quiet public place*", generating the SCC using location and noise detectors, while P15 used "*when there are people around*", generating the SCC using nearby phone number detectors. Similarly, for the task of not being disturbed by messages while wearing headphones to run on the playground, P7 used "*when I'm running and listening to music*", P2 used "*when running on the playground*", and P1 used "*when running with headphones on*". These differences reflect different users' reasoning for solving the problem, resulting in the selection of different detector variables. Finally, it is worth emphasizing that the introduction of the `content_related_to` detector, which specifically handles text semantics, provides sufficient personalization, as the keywords it handles are inherently personalized. For instance, different participants proposed judging whether the sender was a superior, colleague, or mentor to filter important messages, each with their understanding of importance.

During the study, it was observed that non-professional users did not pay much attention to the machine expressions component of SCCs during system usage. Among the 8 participants with technical backgrounds, P2 and P13 expressed a desire to directly modify the expressions, while P7, P8, and P9 preferred to adjust numerical values in the expressions. On the other hand, P3 and P15 found that they were unwilling to modify machine expressions directly in a text editor and preferred to complete SCC modifications through further communication with the assistant. Despite the availability of a natural language interface to inquire about the SCC's implementation, few participants used this function during the study, even though they had received sufficient instruction. This indicates that the SCC-based interaction design of the system effectively hides machine details while ensuring efficient communication with users. However, it also highlights the importance of having accurate SCC machine expressions to ensure maximum usability, especially in cases where users do not inspect the expressions.

The user interviews revealed potential issues with the system in real-world applications. P5, P6, and P7 mentioned that expressing their intentions and checking rules might be inconvenient in some cases, especially when they were busy. This implies that not all contexts are suitable for expressing preferences within the context and that alternative input methods may be needed. Moreover, the limitation of LLM computation speed causes our assistant to respond slowly, which can negatively impact the user experience. The randomness of LLM also affects system performance, as seen in P8 and P13 expressing that the assistant's responses were not always stable in certain situations. Finally, P4 expressed concerns about the system's actual execution effectiveness, stating that she would not use automation systems if their performance were not reliable enough. These observations suggest that the system needs to strike a balance between usability and reliability to ensure optimal user satisfaction in real-world applications.

6 LIMITATIONS AND FUTURE WORK

The current work is an initial exploration of helping end-users build personalized contextual rules. LangAware considers the user's expressed intentions through language, but it does not address the user's underlying goals, which may not be explicitly stated or even recognized by the user. For instance, a user may request a reminder to wake up at 6 am every workday, but their underlying goal is to arrive at work on time. This rule may fail during holidays, resulting in the user being unnecessarily woken up. To address this issue, we plan to analyze and cluster SCC semantics after users have accumulated a set of SCCs using LangAware. This will enable the creation of an interpretable SCC network and the provision of SCC recommendations to users. Furthermore, we plan to investigate data-driven user behavior pattern mining methods for long-term operation, which will complement the proactive user-teaching approach of LangAware. Overall, these future directions aim to improve the system's ability to capture users' underlying goals and provide personalized solutions to users' real-world problems.

Despite its promising results, the practical application of LLMs still faces limitations. Their relatively slow computation speed can significantly affect the assistant's response time, thereby compromising the user experience. Additionally, LLMs' token limit can restrict the number of examples in prompts and consequently impede their ability to effectively handle complex tasks. These challenges could be alleviated by using fine-tuned models in future work. Furthermore, the randomness in LLM's outputs can lead to instability, even though our pipeline design has mitigated this issue to some extent. To address these limitations, we will continue to monitor the latest developments in LLM research and strive to use more efficient, accurate, and practical LLMs in our system.

7 CONCLUSION

In this paper, we presented LangAware, a novel collaborative approach enabling end-users to construct personalized contexts in situ through natural language. Addressing the human-machine context gap, LangAware leverages large language models (LLMs) to semantically connect low-level sensor detectors with high-level natural language. We introduced Shared Contextual Concepts (SCCs) as a medium for human-machine dialogue, fostering mutual understanding and consensus.

Through a user study conducted in real-life settings with 16 participants across 12 campus scenarios, we demonstrated LangAware's effectiveness with an average success rate of 87.50% in defining contextual rules. LangAware outperforms the baseline in terms of success rate, user satisfaction, and other aspects, highlighting the effectiveness of incorporating in-situ contextual data for context collaboration. Additionally, users reported that the collaboration process with LangAware enhanced their understanding of the machine's capabilities.

Our work contributes to the development of personalized context-aware applications in IoT scenarios, paving the way for future research on collaborative human-machine approaches.

ACKNOWLEDGMENTS

This work is supported by the Natural Science Foundation of China under Grant No. 62132010, Beijing Key Lab of Networked Multimedia, the Institute for Guo Qiang, Tsinghua University, the Institute for Artificial Intelligence, Tsinghua University (THUAI), the 2025 Key Technological Innovation Program of Ningbo City under Grant No. 2022Z080, as well as by the Beijing Municipal Science & Technology Commission and the Administrative Commission of Zhongguancun Science Park under No. Z221100006722018.

We thank Meizhu Chen for her valuable assistance in improving the visual impact of the figures in our paper.

REFERENCES

[1] J.S.D.M.D.S Abeywardhane, E.M.W.N de Silva, I.G.A.G.S Gallanga, L.N. Rathnayake, Jagath Wickramarathne, and Disni Sriyaratna. 2018. Optimization of Volume & Brightness of Android Smartphone through Clustering & Reinforcement Learning ("RE-IN"). In *2018 IEEE International Conference on Information and Automation for Sustainability (ICIAfS)*. IEEE, 1–6. https://doi.org/10.1109/ICIAFS.2018.8913391

[2] V. D. Ambeth Kumar, S. Malathi, Abhishek Kumar, Prakash M, and Kalyana C. Veluvolu. 2020. Active Volume Control in Smart Phones Based on User Activity and Ambient Noise. *Sensors* 20, 15 (July 2020), 4117. https://doi.org/10.3390/s20154117 Publisher: Multidisciplinary Digital Publishing Institute.

[3] Saleema Amershi, Dan Weld, Mihaela Vorvoreanu, Adam Fourney, Besmira Nushi, Penny Collisson, Jina Suh, Shamsi Iqbal, Paul N. Bennett, Kori Inkpen, Jaime Teevan, Ruth Kikin-Gil, and Eric Horvitz. 2019. Guidelines for Human-AI Interaction. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, USA, 1–13. https://doi.org/10.1145/3290605.3300233

[4] Asier Aztiria, Juan Carlos Augusto, Rosa Basagoiti, Alberto Izaguirre, and Diane J. Cook. 2012. Discovering frequent user–environment interactions in intelligent environments. *Personal and Ubiquitous Computing* 16, 1 (Jan. 2012), 91–103. https://doi.org/10.1007/s00779-011-0471-4

[5] Louise Barkhuus. 2004. *The Context Gap: An Essential Challenge to Context-Aware Computing*. Ph. D. Dissertation.

[6] Louise Barkhuus and Anind Dey. 2003. Is Context-Aware Computing Taking Control away from the User? Three Levels of Interactivity Examined. In *UbiComp 2003: Ubiquitous Computing (Lecture Notes in Computer Science)*, Anind K. Dey, Albrecht Schmidt, and Joseph F. McCarthy (Eds.). Springer, Berlin, Heidelberg, 149–156. https://doi.org/10.1007/978-3-540-39653-6_12

[7] Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Ben Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, Julian Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Rob Reich, Hongyu Ren, Frieda Rong, Yusuf Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishnan Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. 2022. On the Opportunities and Risks of Foundation Models. https://doi.org/10.48550/arXiv.2108.07258 arXiv:2108.07258 [cs].

[8] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya

Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, Vol. 33. Curran Associates, Inc., 1877–1901. https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html

[9] Kaixuan Chen, Dalin Zhang, Lina Yao, Bin Guo, Zhiwen Yu, and Yunhao Liu. 2021. Deep Learning for Sensor-based Human Activity Recognition: Overview, Challenges, and Opportunities. *Comput. Surveys* 54, 4 (2021), 77:1–77:40. https://doi.org/10.1145/3447744

[10] Xinlei Chen, Yu Wang, Jiayou He, Shijia Pan, Yong Li, and Pei Zhang. 2019. CAP: Context-aware App Usage Prediction with Heterogeneous Graph Embedding. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 3, 1 (March 2019), 1–25. https://doi.org/10.1145/3314391 Publisher: ACM PUB27 New York, NY, USA.

[11] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. PaLM: Scaling Language Modeling with Pathways. https://doi.org/10.48550/arXiv.2204.02311 arXiv:2204.02311 [cs].

[12] Fulvio Corno, Luigi De Russis, and Alberto Monge Roffarello. 2019. A high-level semantic approach to End-User Development in the Internet of Things. *International Journal of Human-Computer Studies* 125 (May 2019), 41–54. https://doi.org/10.1016/j.ijhcs.2018.12.008 Publisher: Academic Press.

[13] Fulvio Corno, Luigi De Russis, and Alberto Monge Roffarello. 2020. HeyTAP: Bridging the Gaps Between Users' Needs and Technology in IF-THEN Rules via Conversation. In *Proceedings of the International Conference on Advanced Visual Interfaces*. ACM, New York, NY, USA, 1–9. https://doi.org/10.1145/3399715.3399905

[14] Fulvio Corno, Luigi De Russis, and Alberto Monge Roffarello. 2021. From Users' Intentions to IF-THEN Rules in the Internet of Things. *ACM Transactions on Information Systems* 39, 4 (Oct. 2021), 1–33. https://doi.org/10.1145/3447264 Publisher: ACM PUB27 New York, NY.

[15] Giuseppe Desolda, Carmelo Ardito, and Maristella Matera. 2017. Empowering End Users to Customize their Smart Environments: Model, Composition Paradigms, and Domain-Specific Tools. *ACM Transactions on Computer-Human Interaction* 24, 2 (2017), 12:1–12:52. https://doi.org/10.1145/3057859

[16] Anind K. Dey. 2001. Understanding and Using Context. *Personal and Ubiquitous Computing* 5, 1 (Feb. 2001), 4–7. https://doi.org/10.1007/s007790170019 Publisher: Springer.

[17] Anind K. Dey and Alan Newberger. 2009. Support for context-aware intelligibility and control. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. Association for Computing Machinery, New York, NY, USA, 859–868. https://doi.org/10.1145/1518701.1518832

[18] Anind K. Dey, Timothy Sohn, Sara Streng, and Justin Kodama. 2006. iCAP: Interactive Prototyping of Context-Aware Applications. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 3968 LNCS. Springer Verlag, 254–271. https://doi.org/10.1007/11748625_16

[19] Giuseppe Ghiani, Marco Manca, Fabio Paternò, and Carmen Santoro. 2017. Personalization of Context-Dependent Applications Through Trigger-Action Rules. *ACM Transactions on Computer-Human Interaction* 24, 2 (April 2017), 1–33. https://doi.org/10.1145/3057861 Publisher: ACM PUB27 New York, NY, USA.

[20] Hyungik Oh, Laleh Jalali, and Ramesh Jain. 2015. An intelligent notification system using context from real-time personal activity monitoring. In *2015 IEEE International Conference on Multimedia and Expo (ICME)*, Vol. 2015-August. IEEE, 1–6. https://doi.org/10.1109/ICME.2015.7177508

[21] IFTTT. 2023. IFTTT. https://ifttt.com

[22] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. 2019. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery* 33, 4 (July 2019), 917–963. https://doi.org/10.1007/s10618-019-00619-1

[23] Ashraf Khalil and Kay Connelly. 2005. Context-Aware Configuration: A Study on Improving Cell Phone Awareness. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 3554 LNAI. Springer Verlag, 197–209. https://doi.org/10.1007/11508373_15

[24] Joohyun Lee, Kyunghan Lee, Euijin Jeong, Jaemin Jo, and Ness B. Shroff. 2016. Context-aware application scheduling in mobile systems: what will users do and not do next?. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '16)*. Association for Computing Machinery, New York, NY, USA, 1235–1246. https://doi.org/10.1145/2971648.2971680

[25] Tianshi Li, Julia Katherine Haines, Miguel Flores Ruiz de Eguino, Jason I. Hong, and Jeffrey Nichols. 2022. Alert Now or Never: Understanding and Predicting Notification Preferences of Smartphone Users. *ACM Transactions on Computer-Human Interaction* (Feb. 2022). https://doi.org/10.1145/3478868 Publisher: ACM PUB27 New York, NY.

[26] Henry Lieberman, Fabio Paternò, Markus Klann, and Volker Wulf. 2006. End-User Development: An Emerging Paradigm. In *End User Development*, Henry Lieberman, Fabio Paternò, and Volker Wulf (Eds.). Springer Netherlands, Dordrecht, 1–8. https://doi.org/10.1007/1-4020-5386-X_1

[27] Ryan Louie, Darren Gergle, and Haoqi Zhang. 2022. Affinder: Expressing Concepts of Situations that Afford Activities using Context-Detectors. In *CHI Conference on Human Factors in Computing Systems*, Vol. 18. ACM, New York, NY, USA, 1–18. https://doi.org/10.1145/3491102.3501902

[28] Abhinav Mehrotra, Robert Hendley, and Mirco Musolesi. 2016. PrefMiner: mining user's preferences for intelligent mobile notification management. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, New York, NY, USA, 1223–1234. https://doi.org/10.1145/2971648.2971747

[29] Grégoire Mialon, Roberto Dessì, Maria Lomeli, Christoforos Nalmpantis, Ram Pasunuru, Roberta Raileanu, Baptiste Rozière, Timo Schick, Jane Dwivedi-Yu, Asli Celikyilmaz, Edouard Grave, Yann LeCun, and Thomas Scialom. 2023. Augmented Language Models: a Survey. (Feb. 2023). http://arxiv.org/abs/2302.07842 arXiv:2302.07842.

[30] OpenAI. 2022. Introducing ChatGPT. https://openai.com/blog/chatgpt

[31] Charith Perera, Arkady Zaslavsky, Peter Christen, and Dimitrios Georgakopoulos. 2014. Context Aware Computing for The Internet of Things: A Survey. *IEEE Communications Surveys & Tutorials* 16, 1 (March 2014), 414–454. https://doi.org/10.1109/SURV.2013.042313.00197

[32] Martin Pielot, Bruno Cardoso, Kleomenis Katevas, Joan Serrà, Aleksandar Matic, and Nuria Oliver. 2017. Beyond Interruptibility: Predicting Opportune Moments to Engage Mobile Phone Users. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 3 (Sept. 2017), 1–25. https://doi.org/10.1145/3130956 Publisher: ACM PUB27 New York, NY, USA.

[33] Xin Qi, Qing Yang, David T. Nguyen, and Gang Zhou. 2013. Context-aware frame rate adaption for video chat on smartphones. In *Proceedings of the 2013 ACM conference on Pervasive and ubiquitous computing adjunct publication (UbiComp '13 Adjunct)*. Association for Computing Machinery, New York, NY, USA, 111–114. https://doi.org/10.1145/2494091.2494122

[34] P. Rashidi and D.J. Cook. 2009. Keeping the Resident in the Loop: Adapting the Smart Home to the User. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 39, 5 (Sept. 2009), 949–959. https://doi.org/10.1109/TSMCA.2009.2025137

[35] Iqbal H. Sarker. 2019. Context-aware rule learning from smartphone data: survey, challenges and future directions. *Journal of Big Data* 6, 1 (Dec. 2019), 95. https://doi.org/10.1186/s40537-019-0258-4 Publisher: SpringerOpen.

[36] Vijay Srinivasan, Saeed Moghaddam, Abhishek Mukherji, Kiran K. Rachuri, Chenren Xu, and Emmanuel Munguia Tapia. 2014. MobileMiner: mining your frequent patterns on your phone. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, New York, NY, USA, 389–400. https://doi.org/10.1145/2632048.2632052

[37] Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, YaGuang Li, Hongrae Lee, Huaixiu Steven Zheng, Amin Ghafouri, Marcelo Menegali, Yanping Huang, Maxim Krikun, Dmitry Lepikhin, James Qin, Dehao Chen, Yuanzhong Xu, Zhifeng Chen, Adam Roberts, Maarten Bosma, Vincent Zhao, Yanqi Zhou, Chung-Ching Chang, Igor Krivokon, Will Rusch, Marc Pickett, Pranesh Srinivasan, Laichee Man, Kathleen Meier-Hellstern, Meredith Ringel Morris, Tulsee Doshi, Renelito Delos Santos, Toju Duke, Johnny Soraker, Ben Zevenbergen, Vinodkumar Prabhakaran, Mark Diaz, Ben Hutchinson, Kristen Olson, Alejandra Molina, Erin Hoffman-John, Josh Lee, Lora Aroyo, Ravi Rajakumar, Alena Butryna, Matthew Lamm, Viktoriya Kuzmina, Joe Fenton, Aaron Cohen, Rachel Bernstein, Ray Kurzweil, Blaise Aguera-Arcas, Claire Cui, Marian Croak, Ed Chi, and Quoc Le. 2022. LaMDA: Language Models for Dialog Applications. https://doi.org/10.48550/arXiv.2201.08239 arXiv:2201.08239 [cs].

[38] Shiu Lun Tsang and Siobhan Clarke. 2007. Mining User Models for Effective Adaptation of Context-Aware Applications. In *The 2007 International Conference on Intelligent Pervasive Computing (IPC 2007)*. 178–187. https://doi.org/10.1109/IPC.2007.108

[39] Blase Ur, Elyse McManus, Melwyn Pak Yong Ho, and Michael L. Littman. 2014. Practical trigger-action programming in the smart home. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. Association for Computing Machinery, New York, NY, USA, 803–812. https://doi.org/10.1145/2556288.2557420

[40] Blase Ur, Melwyn Pak Yong Ho, Stephen Brawner, Jiyun Lee, Sarah Mennicken, Noah Picard, Diane Schulze, and Michael L. Littman. 2016. Trigger-Action Programming in the Wild: An Analysis of 200,000 IFTTT Recipes. In *Proceedings*

of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16). Association for Computing Machinery, New York, NY, USA, 3227–3231. https://doi.org/10.1145/2858036.2858556

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

[42] Norha M. Villegas, Cristian Sánchez, Javier Díaz-Cely, and Gabriel Tamura. 2018. Characterizing context-aware recommender systems: A systematic literature review. *Knowledge-Based Systems* 140 (Jan. 2018), 173–200. https://doi.org/10.1016/j.knosys.2017.11.003

[43] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. https://doi.org/10.48550/arXiv.2201.11903 arXiv:2201.11903 [cs].

[44] Rayoung Yang and Mark W. Newman. 2013. Learning from a learning thermostat: lessons for intelligent systems for the home. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing (UbiComp '13)*. Association for Computing Machinery, New York, NY, USA, 93–102. https://doi.org/10.1145/2493432.2493489

[45] Lefan Zhang, Weijia He, Olivia Morkved, Valerie Zhao, Michael L. Littman, Shan Lu, and Blase Ur. 2020. Trace2TAP: Synthesizing Trigger-Action Programs from Traces of Behavior. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 3 (Sept. 2020), 1–26. https://doi.org/10.1145/3411838

[46] Hengshu Zhu, Enhong Chen, Hui Xiong, Kuifei Yu, Huanhuan Cao, and Jilei Tian. 2015. Mining Mobile User Preferences for Personalized Context-Aware Recommendation. *ACM Transactions on Intelligent Systems and Technology* 5, 4 (Jan. 2015), 1–27. https://doi.org/10.1145/2532515 Publisher: ACM PUB27 New York, NY, USA.

A EXAMPLE USE CASES OF LANGAWARE

A.1 Accidental Headphone Disconnection

Suppose a user is watching a video with headphones in a study room. An accidental disconnection of the Bluetooth headphones (e.g., due to low battery) can result in the video audio suddenly playing from the device's speakers, thereby disturbing other students in the study room.

To avoid this embarrassing situation, users can instruct LangAware: "*Mute if earphones disconnected*".

LangAware integrates the user's utterance with contextual information from the detector and presents four context condition phrases in the form of checkboxes: "*Earphones disconnected*" (derived from user utterance), "*In a classroom*", "*Quiet environment sound*", and "*In the afternoon*" (the latter three are generated based on detector data). Each phrase corresponds to an executable detector expression.

At this point, if the user feels there are additional conditions that can be added and they have the time, they can send LangAware an additional message: "*When others are around*". LangAware then supplements the "*Others around*" condition and generates the corresponding machine-executable expression.

After confirming that the conditions meet their requirements, the user validates the generated rule: "*IF the earphones get disconnected and you are in a classroom with other people around and the environment sound is quiet, THEN set media volume to 0.*" When the future situation satisfies these conditions, this rule will be automatically executed.

A.2 Never Miss a Courier Call

To avoid interruptions from advertisements or sales calls during important work or meetings, users sometimes adjust their phone's volume settings. However, some calls are very important: for example, if a delivery driver cannot contact you for a long time, they will move on to the next delivery location, which means you might go hungry for a long time. Users might want their phones to pay special attention to incoming messages and calls related to food delivery, even when the phone volume is set low.

To achieve this, a user can instruct LangAware: "*Remind me when I receive messages related to food delivery.*"

LangAware generates the condition "*Related to food delivery*" and binds it to the `content_related_to` and `new_message.source_app` detectors, generating keywords like `'food delivery'`, `'delivery man'` and a list of food delivery service apps such as `'Doordash'`, `'Uber Eats'`, `'Grubhub'`, `'Ricepo'` as parameters for the two detectors. This condition offers a reasonable abstraction of the detectors and provides a generalizable solution. Additionally, it adds the condition "*User is at home*" based on the user's context information perceived by the current detector, making the rule more precise and meeting the user's real needs.

After the rule generation is complete, in the future when the user's context meets the conditions "*Related to food delivery*" and "*User is at home*", the system will execute the operation `adjustVolumeTo(volumeType=0, target=15)` to set the phone message alert volume to 15.

A.3 Proper Temperature Control

Smart home systems, integrated with LangAware, can automatically control home devices, such as heaters or air conditioners, by recognizing environmental conditions and user behavior patterns.

For example, a user might prefer their room to be warm on cold mornings, which can make getting out of bed easier. They can instruct LangAware: "*Please turn on the heater in advance on cold mornings.*" LangAware generates the conditions "*Cold weather*" and "*Early morning*" and binds them to the corresponding weather and time detectors available in the system. Also, since the user may not want to turn on the heater when they're not at home, LangAware generates a third condition "*User is at home*" based on the detector data and common sense.

The final generated rule is: "*IF it is a cold morning and the user is at home, THEN turn on the heater.*" Once these conditions are met, the heater automatically turns on.

B IMPLEMENTATION DETAILS

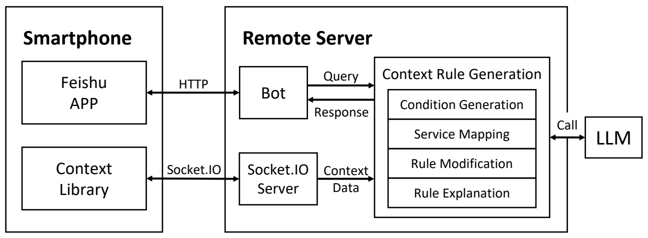


Figure 8: System architecture diagram.

LangAware and the baseline system were both implemented using Android smartphones and a remote server, as shown in Figure 8. For the baseline version which does not need context information, the context library module is disabled.

Table 2: Context detector variables used in the implemented systems.

Context Detectors	Description	Values
Date & time	Current date and time.	e.g., "2023-03-24 16:26:58 Friday"
GPS position	Mapped building names on campus based on GPS coordinates.	e.g., "No. 6 canteen"
Activity	Activity recognized by custom developed deep model using IMU data.	"still", "walking", "running", "cycling" or "others"
Noise	Continuous noise level in db detected using microphones every 2 seconds.	>0
Audio device	Current using audio device type.	"speaker", "wired_headphones", "wired_headset", "bluetooth", "usb_headset" or "unknown"
Audio playing	Whether the phone is playing audio, detected using Android AudioPlaybackCapture API.	True or False
Audio stream	Audio stream type.	"media", "voice_call" or "ring"
Volume	Volumes of media, voice call, notification, ring and alarm.	e.g., media_volume = 2, voice_call_volume = 5
APP	Current foreground APP, detected using Android AccessibilityEvent.	e.g., "Zoom", "YouTube"
Network	Network status.	"no internet connection", "connected to Wi-Fi" or "using mobile data"
Network delay	Network delay in ms.	>= 0
Wifi name	Connected Wifi AP SSID.	e.g., "Starbucks-5G"
Wifi BSSID	Connected Wifi AP BSSID.	e.g., "a8:58:40:d7:13:b2"
Screen orientation	Screen orientation.	"vertical" or "horizontal"
Nearby PC	Nearby PC number detected by Bluetooth scanning.	>= 0
Nearby phone	Nearby phone number detected by Bluetooth scanning.	>= 0
Latest message	Latest received notification within 5 minutes.	e.g., new_message.title = "Daddy: how to make..."; new_messaage.content = "how to make my phone ring when i'm out of the house?"; new_message.source_app = "WhatsApp"; new_message.sender = "Daddy: How to make..."; new_message.type = "APP messages"

The front-end dialogue interface was built using the Feishu Bot[3]. The Feishu app facilitates bidirectional HTTP communication with our server through the Feishu server, supporting user interactions with our customized Feishu Bot. Users can send messages either through typing or the speech-to-text function available in the Feishu app, which then forwards them to our server. Our bot responds to the users via message cards that contain explanatory text, checkbox controls for SCC selection, and button controls for confirmation or cancellation. Interactions with the message card lead to corresponding updates on the card content, while new messages from the users are met with fresh message cards in response.

The context library persistently runs in the background on the smartphone while maintaining a Socket.IO connection with the remote server. This library implements a variety of detectors that process raw sensor data into semantically meaningful results, such as time, location, activity, messages, and more. Refer to Table 2 for a comprehensive list of detectors.

The context rule generation module is executed on the remote server and consists of four components: condition generation, service mapping, rule modification, and rule explanation. The condition generation component corresponds to the generation of SCCs proposed in Section 4.3. Each boolean expression of an SCC conforms to Python syntax and is composed of pre-defined context detector variables (see Table 2) and the newly introduced text detector `content_related_to`. The service mapping component transforms natural language into service API calls in Python syntax, which can be selected from a heuristically pre-defined list (including volume adjustment, modification of notification modes for new messages, call mode alteration, network switching, etc.), or generated by the LLM itself. The rule modification component alters the generated rule based on the user's instructions, while the rule explanation component provides a natural language explanation in response to the user's queries.

In both our LangAware and baseline implementations, we utilized the `gpt-3.5-turbo-0301` model via the ChatGPT API[4], with

[3]Feishu Open Platform: https://open.feishu.cn/

[4]ChatGPT API: https://openai.com/blog/introducing-chatgpt-and-whisper-apis

a temperature setting of 0.2 to generate a relatively deterministic output. Our prompts follow a dialog format, where the "system" section presents the core requirements of each task, and the "user" section provides detailed instructions. Once the instructions are set, we supplement user-assistant dialogues with illustrative examples, promoting a clearer understanding of the expected outcomes and task requirements. Additionally, we incorporate a Chain-of-Thought [43] strategy in some prompts to aid the LLM in reasoning effectively. For instance, during the logical analysis stage of user utterances, we first prompt the LLM to mark keywords, then proceed to semantic analysis. Our findings indicate that this keyword annotation practice enhances the LLM's comprehension of inputs and the quality of outputs. Please refer to the supplementary materials for the prompts used in the paper.