Empirical Software Engineering (2026) 31:132
https://doi.org/10.1007/s10664-026-10859-7

Using large language models to test voice user interfaces

Emanuela Guglielmi[1] · **Angelica Spina**[1] · **Gabriele Bavota**[2] · **Rocco Oliveto**[1] · **Simone Scalabrino**[1]

Received: 24 March 2025 / Accepted: 30 March 2026

Abstract
Voice-based virtual assistants enable hands-free operation, allowing users to perform tasks, access information, and control smart home devices through simple voice commands. Their growing ubiquity in smartphones, smart speakers, and other devices led to the flourish of more and more apps taking advantage of a Voice User Interface (VUI). VUI testing is far from trivial due to the wide variability in human speech (e.g., different accents, dialects, speech patterns), and the fact that users can express the same command in numerous ways, using different (but semantically equivalent) wordings and phrases. For this reason, techniques have been proposed to support VUI testing. The basic idea behind these specialized approaches is to generate paraphrases for the set of voice commands for which developers implemented support in the VUI. Preliminary results from a recent study suggest that specialized models can outperform a general-purpose LLM (ChatGPT). However, a simple prompt and interaction strategy with ChatGPT has been adopted. In other words, it is still unknown whether optimizing the LLM usage allows to obtain better results. In this paper, we aim to thoroughly study to what extent LLMs (ChatGPT, specifically) can be adopted to test VUIs. We focused on optimizing the used prompt and the interaction with the model. Our results show that an optimized use of LLMs results in new state-of-the-art performance for VUI testing in terms of number of correct and bug-revealing paraphrases. While introducing the generated paraphrases into the Voice Interaction Models of the skills allows to fix some bugs, we observe that many bugs remain, and some are even introduced by the generated paraphrases. Our results call for specialized approaches for fixing bugs in VUIs.

Keywords Voice user interfaces · Software testing · Empirical study

Communicated by: Hadi Hemmati.

Extended author information available on the last page of the article

Published online: 02 May 2026

1 Introduction

Voice-based virtual assistants use natural speech to execute required tasks and, more in general, to interact with users. These assistants are seamlessly integrated into daily life through smartphones, smart speakers, and other connected devices. Prominent representatives of this technology are Amazon Alexa, Apple Siri, and Google Assistant. With the aim of offering a broader range of services and capabilities tailored to diverse user needs, these systems provide an open development environment through which third-party developers can create custom apps featuring a Voice User Interface (VUI). For example, the Alexa Skills (i.e., apps built for Amazon Alexa) store (Amazon 2018) hosts over 100k skills.

When creating these apps, developers must define a Voice Interaction Model (VIM) which defines the functionalities to be executed as consequence of voice commands received by the user. In other words, the VIM can be seen as a set of pairs featuring a *seed sentence* and the related feature to trigger, known as "*intent*". The developer can specify multiple seed sentences for a single intent (i.e., the same functionality is triggered with different sentences) to account for different wordings that may be used during the voice interaction. On top of that, the AI integrated in the voice-based virtual assistants is usually able to deal with textual variations of the seed sentence not explicitly encoded in the VIM. For example, if the developer coded in the VIM the seed sentence "*what's the current status of the US stock market?*" it is likely that the assistant can recognize also its variant "*can you tell me the current status of the US stock market?*", by correctly running the expected intent. However, problems may arise when the input utterance is too far from what expected (e.g., "*give me an update on the financial markets in the United States*").

The variety of linguistic expressions available in the human language makes the testing of VUI extremely challenging. For this reason, techniques and tools have been proposed to support developers in VUI testing (Guglielmi et al. 2022, 2023), with the main goal of helping them in identifying utterances being semantically equivalent to a specific seed sentence but that, however, do not result in the running of the same intent (as an user would expect).

The Alexa Developer Console (ADC) used to include an AI-based paraphrase generator, highlighting the value of synthetic linguistic variation for improving intent matching.. Nowadays, this tool is not available anymore. Amazon's guidelines, however, still emphasize the importance of enhancing voice interaction models with diverse natural expressions to increase NLU coverage and reduce fallbacks. They suggest to use LLMs as a natural solution for this step because they are able to efficiently synthesize large, varied, human-like sets of paraphrases that rule-based or synonym replacement methods struggle to achieve, thus highlighting edge cases and fragilities in intent mapping that would otherwise remain hidden (Alexa 2025). In a previous work (Guglielmi et al. 2022), some of the authors proposed VUI-UPSET,[1] an approach that adapts chatbot testing techniques (Guichard et al. 2019) to VUI testing. Given the VIM of an Alexa skill, VUI-UPSET starts by generating as many paraphrases as possible of the seed sentences by replacing words with synonyms. Since some of these paraphrases are unlikely to represent utterances that a human would actually use during the voice interaction (e.g., "*save our natal day*" as a paraphrase for "*record birthday*"), in a second step VUI-UPSET exploits a RoBERTa model (Liu et al. 2019) to filter-out irrelevant paraphrases. In an empirical evaluation conducted on 40 Alexa skills, Guglielmi et al. (2023) compared the VUI-UPSET to ChatGPT in terms of their ability to

[1] In the following we refer previous work as Guglielmi et al. because the set of authors only partially overlaps.

generate paraphrases that are (i) imperatively equivalent to the seed sentence, and (ii) able to reveal bugs in the VUIs. They show that, while ChatGPT generates a higher percentage of valid paraphrases (59% *vs* 45%), VUI-UPSET generates a higher absolute number of imperatively equivalent paraphrases (2,752 *vs* 2,039), since it produces a higher number of paraphrases as output. In turn, this results in VUI-UPSET generating a higher number of bug-revealing paraphrases.

While the above-summarized findings seem to suggest that a specialized approach (VUI-UPSET) is, from a practical point of view, more effective for VUI testing than a general-purpose LLM, these results might be driven by the way in which ChatGPT has been used Guglielmi et al. 2023. Indeed, the authors used a very simple prompt asking the LLM to generate "*all possible paraphrases*". This usually resulted in a quite low number of paraphrases generated by ChatGPT, since the LLM is designed to output a maximum number of tokens (thus limiting the number of proposed paraphrases even when more would have been possible). Also, the exploited prompt might be suboptimal for the task, since no prompt engineering was performed to define it. For example, in-context learning via few-shot examples has been shown to substantially boost the performance of LLMs in several tasks related to both natural language processing (Hegde and Patil 2020; Witteveen and Andrews 2019) and software engineering. In this work, we replicate the work by Guglielmi et al. (2023) by exploring, however, a different interaction procedure with the LLM.

First, we experimented with four different prompts being: (i) the *baseline* (same used by Guglielmi et al. (2023)); (ii) a *few-shot* prompt providing the LLM with examples of the task to perform; (iii) a *contextual* prompt including information for the LLM about the specific usage scenario for the paraphrases to generate; and (iv) a prompt combining *few-shot* examples and *contextual* information. Second, we adopt an iterative interaction procedure, in which the LLM is continuously asked for more and more paraphrases until it does not produce any new paraphrase. We studied to what extent how each improvement contributes in achieving better results on a set of 10 Alexa skills. Our results show that (i) a prompt combining *few-shot* examples and *contextual* information allows to achieve the best results, and (ii) that the iterative interaction procedure is by far the most important step.

Then, we replicated the large-scale study conducted by Guglielmi et al. (2023) on the same set of 40 Alexa skills with the previously discussed optimizations. Such a study required the manual validation of 25,298 paraphrases, for a total of about 500 hours of human evaluation. Our results show that ChatGPT with the optimizations (we call it ChatGPT+ from now on) generates a total of 109k paraphrases, as compared to the 40k of VUI-UPSET, and to the 5k of the ChatGPT *baseline* as used in Guglielmi et al. (2023). Also, ChatGPT+ generates the highest percentage of imperatively equivalent paraphrases (80%, as compared to the 58% of the *baseline* ChatGPT, and to the 44% of VUI-UPSET). These improvements result in ChatGPT+ being able to identify the highest number of bug-revealing paraphrases in the tested Alexa skills (over $\sim$4$\times$ VUI-UPSET and $\sim$25$\times$ the ChatGPT baseline). Then, we tried to introduce the paraphrases generated by the best approach (ChatGPT+, in our study) into the VIM of the Alexa skills to understand if it is possible to improve them (i.e., reduce the number of bugs). On the one hand, we observed that this practice allows to reduce the number of bugs. On the other hand, our results show that some bugs are still there and some might be even introduced when adding the generated paraphrases. Thus, we qualitatively studied the root causes of bugs in VUIs defining four main categories of bugs: (i) *Ambiguous Intent* (i.e., there are two or more intents in the original skill that have unclear

differences); (ii) *Fallback Intent* (i.e., the fallback intent is activated instead of the desired intent); (iii) *No-activation Intent* (i.e., no intent is activated whatsoever); and (iv) *Unexplained* (i.e., we found no clear reason for the issue). Such a study required the classification of 3,735 paraphrases into this four bug categories with independent double labeling (7,470 in total). We spent about 500 human-hours overall. While generating paraphrases can help for *Fallback Intent* and *No-activation Intent* bugs, it does not work for the other categories. Specifically, *Unexplained* bugs are particularly concerning because we found that a single word in an activation sentence of the VIM (e.g., "about") can confuse the intent-matching model behind the skill and cause bugs.

Finally, we assess whether a smaller, open-weight model (GPT-oss-20B) could replace GPT-4 for VUI paraphrase generation. We replicated the same evaluations of ChatGPT+. Findings show that while GPT-4 naturally achieves a slightly higher equivalence rate, the open-weight model produces a larger number of valid paraphrases, confirming the approach is model-agnostic and using a smaller model is the best choice in terms of cost-effectiveness.

In summary, our contribution is twofold. We present not only the results of an empirical study in which we properly use LLMs to generate paraphrases to test VUIs, but also a grounded, manually validated taxonomy of VUI bugs we identified.

Our findings support the usage of LLMs as effective VUI-testing techniques, but they also call for the introduction of specialized approaches for fixing bugs caused by ambiguous intents and unexplained bugs. We release all data used in our study (Guglielmi et al. 2024).

2 Background and Related Work

In this section, we first provide an overview of the basic concepts behind VUIs and VUI testing. Then, we discuss work using LLMs for paraphrasing.

2.1 Voice User Interface Testing

A VUI is a type of human-computer interface that allows users to interact with computer systems through spoken commands or questions, receiving primarily spoken responses. VUIs involve two key components: (i) A speech recognition module that translates spoken words into text, and (ii) an NLP module that interprets natural language inputs to match it with predefined commands (*utterances*) to run the appropriate function (*intent*). In the following, we focus on the latter component since our work regards this aspect.

To define VUIs, developers are required to create a VIM for the app. A VIM consists of a set of pairs $\langle U, I \rangle$, where $U \in \Upsilon$ represents an utterance from all the possible utterances that a user might pronounce, and $I \in \Lambda$ is the intent they are supposed to trigger from the set of intents defined in the app. When a user pronounces an utterance $U^* \in \Upsilon$, the AI integrated in the voice-based virtual assistants matches U^* to the nearest utterance U from the VIM to activate its associated intent. However, such a component sometimes fails to perform the match and either activates a wrong intent $I \in \Lambda$ or activates a special *fallback* intent that indicates that no utterance in the VIM matches U^*.

Testing VUIs requires to define couples $\langle U^t, I^t \rangle$, where $U^t \in \Upsilon$ is an utterance and $I^t \in \Lambda$ is the intent expected to be activated (i.e., the oracle). Manually writing test cases for VUIs can require a significant effort since the input space (U) can be very large. Thus, recent

work explored the feasibility of automated test case generation. The main strategy consists of using a methodology inspired by metamorphic testing (Segura et al. 2016). For each couple $\langle U, I \rangle$ in the VIM, some paraphrases of U are generated through a function p, and for each $U^t \in p(U)$ new test cases $\langle U^t, I \rangle$ are defined. Specifically, the function $p : \Upsilon \rightarrow \mathcal{P}(\Upsilon)$ transforms a given utterance in a set of *imperatively equivalent* utterances, i.e., utterances that a user could use as an alternative to the original one to express the same command. A paraphrase is "imperatively equivalent" to a seed sentence if a human might use the two sentences interchangeably to express the same intent and, therefore, to trigger the same action. For example, the sentence "record my birthday" is semantically different from the sentence "don't forget my birthday"; however, if users adopt the latter they most likely want to trigger the same action they would express with the former (Guglielmi et al. 2023).

Figure 1 shows how we derive the oracle from the VIM (NLU Annotation Set) by pairing each generated, imperatively equivalent paraphrase couples $\langle U^t$,with the seed's intent $I \rangle$. The NLU evaluation then returns the Actual intent I^a for each U^t and marks the test *PASS* if $I^a = I$, otherwise *FAIL*.

Previous work defined several functions p to automatically generate tests for VUIs. Guichard et al. (2019) presented a rule-based approach that generates paraphrases aimed at evaluating the robustness of chatbots, which are conceptually similar to VUI apps. Such an approach first generates candidate paraphrases through lexical substitution, and then filters out candidates probably not equivalent to the original utterance through metrics such as translation pivoting, language model, word vectros, web search, word-sense disambiguation and lexical frequency.

The work most closely related to ours is the one by Guglielmi et al. (2022, 2023). They introduced VUI-UPSET, which builds upon the approach provided by Guichard et al. VUI-UPSET first generates a set of candidate paraphrases mainly through synonym substitution, and then filters out non-equivalent ones through a ML model. They compared VUI-UPSET to a simple baseline based on GPT-3.5-turbo to generate paraphrase for VUI-testing. They show that VUI-UPSET achieves better results than such an approach. However, as previously explained, they only tested a simple prompt for generating paraphrases through such

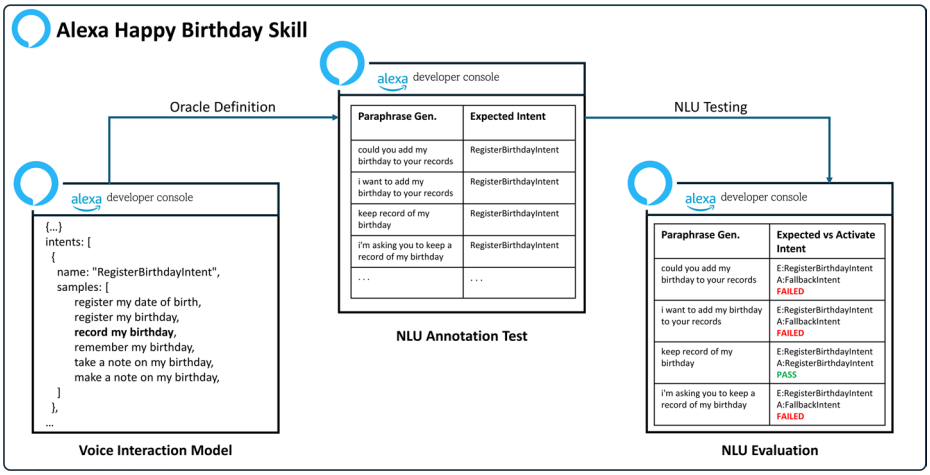

Fig. 1 Overview of the VUI testing workflow

a model, in which they asked to autonomously decide how many paraphrases to output. Differently from such a work, we systematically perform prompt tuning to make sure that the full capabilities of such models are exploited.

VUIs are complex interfaces, and some intents can only be triggered when the app is in a specific state. For this reason, other previous work defined techniques to automatically discover and reach a given status of the app that later allows to activate a given intent. Zhang et al. (2019) examine semantic interpretation errors in voice assistants after Automatic Speech Recognition (ASR) by targeting the NLU intent classifier. They introduce LipFuzzer, a linguistic-model-guided, that builds adversarial language models (Bayesian networks on pronunciation, vocabulary, and grammar) to drive the mutation of voice command models and verifies the effects using Text-to-Speech (TTS), thus isolating NLU errors from ASR. This work underscores that post-ASR intent mapping is a primary failure point and motivates methods that stress text-to-intent behavior. Li et al. (2022) introduced VITAS, a framework for testing VUIs by simulating a multi-state interaction. Later on, Li et al. (2024) built on top of VITAS and introduced Elevate, a framework based on LLMs for testing voice personal assistant (VPA) applications. Elevate uses LLMs to extract states from app outputs and generate context-related inputs, incrementally building a behavioral model during automated interactions. While related to our work, we exploit LLMs for the paraphrases generation phase rather than for exploring the apps' state. ReCode (Wang et al. 2022) defines robustness under input rewriting conditions while preserving meaning, systematically paraphrasing prompts (e.g., documentation strings, identifiers, syntax/format changes), verifying semantic validity, and demonstrating that even small valid changes can compromise model behavior. NLPerturbator (Chen et al. 2024) reaches a similar conclusion for the variations in natural language that practitioners observe in practice, finding significant declines in code-based LLM models. We adopt the same principle for VUI: instead of verifying the correctness of the code, we test the consistency between text and intent by generating imperatively equivalent paraphrases and verifying whether the skill still activates the expected intent.

Several VUI testing approaches rely on automatically generated expressions to explore the behavior of large-scale applications, particularly to detect policy, privacy, or security violations. SkillExplorer by Guo et al. (2020) performs grammar-based dialog exploration: it extracts seed utterances from store pages, classifies a skill's prompts (Yes/No, Wh, etc.), auto-generates admissible replies, and then flags policy/behavior issues at scale (e.g., requesting PII without disclosure; continuing to listen after "stop"). Shezan et al. (2020) introduced VerHealth, a framework for auditing health-related Alexa skills for privacy and safety compliance. VerHealth combines pass over descriptions with a dynamic tester to detect violations (e.g., storing medical info, missing disclaimers). Young et al. (2022) proposed SkillDetective, which performs large-scale detection of policy violations through dynamic dialogue exploration (text/audio/image analysis, question type classification, and tree-based navigation) to identify issues such as the collection of undisclosed data or the lack of disclaimers. More recently, Liao et al. (2023) introduced SkillScanner, which detects policy non-compliance in Alexa skills pre-deployment via static code analysis, combining NLP heuristics and taint/data-flow tracking to surface privacy-policy problems and front/back-end inconsistencies. Even if paraphrase generation isn't the goal, these papers operate in the same VUI testing sub-domain and do generate paraphrases. Although these methods often involve grammar-based or dialogue-driven variations of user inputs, they

do not explicitly investigate whether semantically or imperatively equivalent expressions consistently correspond to the same intent. In addition, Yigit and Amasyali (2024) report that previous work in question answering has explored adversarial perturbations of text, which sometimes include paraphrase-like transformations to test model robustness; however, these are crafted to expose vulnerabilities in QA systems rather than to systematically study paraphrase generation as a mechanism for testing semantic consistency in VUI intent classification.

2.2 LLMs for Paraphrase Generation

Large-scale language models like GPT-4 have revolutionized natural language processing (NLP). These models, designed to understand and generate human language, are able to solve a wide range of tasks, including natural language understanding, text generation, language translation, and more (Wei et al. 2022; Roumeliotis and Tselikas 2023). One of the most notable models at date is GPT-4, which stands for "Generative Pre-trained Transformer 4". Introduced by OpenAI, GPT-4 has great ability to comprehend and produce human-like text across various languages and domains (Egli 2023).

Related to our work are mostly applications of LLMs for paraphrases generation coming from the NLP field. Hegde and Patil (2020) fine-tuned a GPT-2 model for the specific task of paraphrases generation. Their empirical study show that the fine-tuned model works better than solutions based on machine learning techniques (SVM). Also Witteveen and Andrews (2019) fine-tuned a GPT-2 model for the task of paraphrasing, this time targeting not only single sentences but also entire paragraphs.

LLMs may generate results of different quality depending on the style and content of the prompt. For this reason, *prompt engineering* – "the formal search for prompts that retrieve desired outcomes from language models" (Liu and Chilton 2022) – is an active field of research. The output quality of LLMs largely depends on the specificity and clarity of the prompts provided by the user. Effective prompt engineering is therefore essential to ensure accurate and useful results from these models (Liu and Chilton 2022; Lo 2023). The style of the prompt, as well as its content, may determine the level of correctness of the response. This is also confirmed in the domain of paragraphs generation (Meier et al. 2024). In a nutshell, prompt engineering can be seen as a kind of programming in natural language, where programming statements are the natural language sentences in the prompts (Reynolds and McDonell 2021). Understanding how to write good prompts is also relevant when considering code generation tasks. Denny et al. (2023) studied the impact of changes implemented in prompts used for solving Python programming problems. The results show that prompt engineering has been useful in the vast majority of the cases to improve the correctness of the resulting code. The effectiveness of prompt engineering varies widely depending on the specific model and task, making it difficult to find the right prompt for complex tasks. To solve this problem, more advanced techniques such as prompt tuning and prompt optimization have been developed. Prompt tuning involves tuning the model with a specific prompt to improve and optimize its responses to that particular prompt, ensuring more accurate and relevant results (Xu et al. 2023).

In our study we exploit a newer LLM (GPT-4) without performing any fine-tuning for the paraphrasing task, with the aim of disclosing its capabilities for the testing of VUI. Given the lack of fine-tuning, we pay particular attention to the adopted prompt. Indeed, it is well-

documented that the output quality of LLMs largely depends on the specificity and clarity of the prompts provided by the user (Liu and Chilton 2022; Lo 2023), also in the context of text generation (Meier et al. 2024).

3 Optimizing VUI Testing through LLMs

We discuss in the following the LLM interaction procedure we devised to overcome the limitations of the work by Guglielmi et al. (2023). In particular, we detail in the following: (i) the definition of enriched prompts which may help the LLM in generating better paraphrases; and (ii) the iterative interaction aimed at maximizing the number of paraphrases output of the LLM.

3.1 Prompting Optimization for Paraphrase Generation

Guglielmi et al. (2023) relied on a simple prompt template to generate paraphrases. We defined three additional prompt templates and studied which one allows to achieve the best results in terms of absolute number of imperatively equivalent paraphrases generated (i.e., paraphrases being semantically equivalent to the input sentence).

Basic Prompt With "*basic prompt*" we refer to the prompt template used in the work by Guglielmi et al. (2023). Such a prompt simply instructs the LLM to generate paraphrases starting from a given utterance in the VIM (`seed-utterance`).

> *Given the following input sentence '*`{seed-utterance}`*', generate all possible paraphrases that you consider semantically equivalent to it.*

Few-shot Prompt The first optimization of the basic prompt template consists of providing examples of how the model should generate paraphrases through few-shot learning. The quality of the examples is fundamental to obtain good results. Thus, we used a systematic procedure to determine the examples (shots) to use. The first author manually analyzed the utterances in the VIMs of 40 Alexa skills and clustered them based on their grammatical form. As a result, we identified six types of utterances:

1. *Interrogative*: An open question posed to the virtual assistant, e.g., "What is ...?";
2. *Polite*: A cordial request in which words like "please" are used;
3. *Indirect*: The user expresses their personal will with phrases like "I am interested in ...";
4. *Imperative*: The utterance is expressed as an actual command for the app;
5. *Inquisitive*: A specialization of the interrogative form in which the user inquires about something and uses phrases like "any news about ...?";
6. *Exploratory*: A specialization of the indirect form with an explicit indication that the user is searching for something, e.g., starting with "I am looking for ...".

We adopt these six types of utterances to build the following prompt:

`basic-prompt`. *For example, given the input sentence* "`es-utter`", *a sample of semantically equivalent paraphrases expected in output includes:* `ep-interrogative`; `ep-polite`; `ep-indirect`; `ep-imperative`; `ep-inquisitive`; `ep-exploratory`.

We indicate with `basic-prompt` the basic prompt described above. The `es-utter` is a specific **e**xample **s**eed utterance and the `ep-X` fields indicate the various **e**xamples of **p**araphrases of such an utterance in the six forms previously listed. The example seed utterance is skill-agnostic since we are interested in instructing the model to generate paraphrases in the different grammatical forms, regardless of the content of the examples. Thus, we always use the same example for all the skills. We choose an example that is outside the domain of all the skills used in the experiments we performed, i.e., "give me the daily news". We use the following example paraphrases for the various categories: "what's in the daily news today?" (interrogative), "could you please give me the daily news?" (polite), "I'm interested in hearing the daily news" (indirect), "show me the daily news" (imperative), "any highlights in the daily news today?" (inquisitive), and "I'm looking to find out what's in the daily news" (exploratory).

Contextual Prompt The second variation of the basic prompt consists of providing additional context to the LLM. First, we provide the model with more information about the task it has to perform by not simply asking to generate paraphrases but by also reporting the context for which they will be used (i.e., VUI-based apps). Second, we provide a more complete background on the specific skill for which we require paraphrases. As for the latter, we include (i) the name of the skill, (ii) the generic description of the skill, (iii) the name of the intent that the utterance is supposed to activate, and (iv) the list of the other utterances that activate the same intent. This information may help ensuring that the generated paraphrases are aligned with the expected functionality of the skill.

You have to generate paraphrases for an activation sentence of an Alexa skill. The activation sentence starts a feature in the Alexa skill. In this case, the activation sentence you will receive belongs to a skill named '`skill-name`'. *Such a skill* `skill-description`. *The activation sentence you will receive starts a feature named* '`intent-name`'. *Other activation sentences that starts this feature are:* '`other-seed-utterances`'. `basic-prompt*`

Where all the fields are replaced with the respective information reported above. In this case, we slightly modify the basic prompt (`basic-prompt*`) and use "activation sentence" instead of "input sentence" since we defined in the prompt what an activation sentence is, while we do not do that in the basic prompt.

Full Prompt The last optimized prompt template includes both contextual information and examples. Thus, such a prompt template is simply the concatenation of the contextual prompt and a variation of the few-shot prompt from which we remove the basic prompt (since it is already contained in the former).

Finding the Best Prompt Template To find out which template allows to achieve the best results, we run an evaluation on a set 10 Alexa Skills. To select the set of skills, we adopted the same methodology that Guglielmi et al. (2023) used for the selection of the skills subject of their study. In detail, we (i) extracted the most popular open source skills on GitHub (ii) verified that both the speech interaction model and programming logic were available, (iii) considered only the interaction models defined for the English language (en-US), and (iv) focused on the skills written in JavaScript. We instantiate the four prompt templates for each seed utterance and used ChatGPT (specifically, GPT-4) to generate the paraphrases.

Two of the authors independently analyzed a significant stratified sample (5% margin of error, 95% confidence level) of the paraphrases generated for each skill and each prompt template. This means that (i) prompts that generated a higher number of paraphrases resulted in a larger sample to manually analyze and (ii) similarly, the sample of paraphrases from each skill is proportional to the number of paraphrases generated for that skill. The goal of the manual analysis was to report if the paraphrase was imperatively equivalent to the seed utterance from which it was generated. We analyzed a total of 3,490 paraphrases generated by the basic prompt (618), the few-shot prompt (1,000), the contextual prompt (707), and the full prompt (1,165). The two evaluators discussed the cases in which they disagreed aiming at reaching consensus. In total, the two evaluators disagreed on 734 cases (21%), mostly because of the different level of strictness they used in the determination of what is to consider imperatively equivalent. After discussion, they reached consensus on all of them.

We report in Table 1 the results of this analysis. While the basic template achieves the highest percentage of equivalence, all the optimized versions allow to generate a higher absolute number of imperatively equivalent paraphrases, based on the estimation we can make on the whole population s (see columns "#equiv$\star$"). To do so, we consider the percentage of imperatively equivalent generated paraphrases C^s computed on the sample, the total number of paraphrases generated for s, G^s, and the 5% margin of error given by the considered sample size. We compute the confidence interval of the number of imperatively equivalent generated paraphrases for s as $P^s = (C^s \pm 0.05) \times G^s$. For example, while the basic prompt is expected to generate 657±127 imperatively equivalent paraphrases across the ten skills, the full prompt goes up to 1,469±334. When comparing two types of prompt (e.g., basic *vs* full prompt) for a given skill (e.g., Apl-video, first row in Table 1) we can state that one generates a statistically significant higher number of imperatively equivalent paraphrases than the other (95% confidence level) if there is no overlap between the intervals estimated for the whole population. In our running example (Apl-video), the basic prompt is expected to generate between 40 (50-10) and 60 (50+10) paraphrases, while the full prompt between 66 (90-24) and 114 (90+24). Since there is no overlap between these intervals (i.e., 60<66), we can claim a statistically significant difference. These significant differences are reported in bold in Table 1. In case more than one prompt is highlighted in bold, this indicates that there is no significant difference among them, while the difference is significant when comparing the ones in boldface against the others (non in boldface). These cases are indicated with a *.

As it can be seen, the full prompt is the one ensuring the highest number of imperatively equivalent paraphrases. This holds both overall, with statistically significant difference when comparing it to the three other prompts (see last row in Table 1) as well as when looking in isolation at the ten subject skills. Indeed, (i) it never happens that another prompt generate a

Table 1 Comparison of different prompt templates in terms of imperatively equivalent of generated paraphrases. ○ indicates that the column refers to the evaluated sample; ⋆ indicates that it refers to the estimated value on the whole population. We report in boldface the significantly best results, while we mark with a * the cases in which the best result is shared by two approaches

	Basic Prompt					Few shot prompt				
Skill	Generated	Sample	#equiv○	%equiv	#equiv⋆	Generated	Sample	#equiv○	%equiv	#equiv⋆
Apl-video	60	52	32	61.5%	50 ± 10	61	53	37	69.8%	53 ± 8
Device location	175	120	89	74.2%	***157 ± 18**	157	112	79	70.5%	138 ± 19
Discord	100	80	47	58.8%	82 ± 18	569	230	126	54.8%	***455 ± 114**
heroku-sales	48	43	25	58.1%	39 ± 9	102	81	52	64.2%	***86 ± 16**
Mirror wall	26	24	18	75.0%	23 ± 3	112	87	69	79.3%	***103 ± 9**
Pet tales	27	25	16	64.0%	23 ± 4	101	80	36	45.0%	***76 ± 25**
Purchase	85	70	49	70.0%	74 ± 11	89	72	46	63.9%	75 ± 14
Recipes	111	86	49	57.0%	90 ± 21	134	100	48	48.0%	103 ± 31
Reminders	27	25	13	52.0%	21 ± 6	83	68	42	61.8%	69 ± 14
Streaming	125	93	49	52.7%	99 ± 26	167	117	49	41.9%	***123± 44**
Overall	784	618	387	62.6%	657 ± 127	1,575	1,000	584	58.4%	1,287 ± 288
	Contextual prompt					Full prompt				
	Generated	Sample	#equiv○	%equiv	#equiv⋆	Generated	Sample	#equiv○	%equiv	#equiv⋆
Apl-video	66	56	28	50.0%	51 ± 15	114	88	47	53.4%	**90 ± 24**
Device location	71	60	45	75.0%	64 ± 7	215	138	100	72.5%	***191 ± 24**
Discord	254	153	102	66.7%	218 ± 36	365	187	117	62.6%	***306 ± 49**
heroku-sales	41	37	20	54.1%	33 ± 8	100	80	37	46.3%	***76 ± 24**
Mirror wall	46	41	19	46.3%	35 ± 11	109	85	54	63.5%	* **92 ± 17**
Pet tales	87	71	35	49.3%	67 ± 20	108	84	35	41.7%	***79±29**
Purchase	137	101	50	49.5%	***106± 31**	309	171	64	37.4%	***220 ± 89**
Recipes	131	98	62	63.3%	110 ± 21	217	139	90	64.6%	**184± 33**
Reminders	26	24	13	54.2%	21± 5	184	125	94	75.2%	**166 ± 18**
Streaming	80	66	31	47.0%	61 ± 19	82	68	37	54.4%	***65 ± 17**
Overall	939	707	405	57.3%	762 ± 177	1,803	1,165	675	57.9%	**1,469 ± 334**

significant higher number of equivalent paraphrases than the full prompt; and (ii) for three skills, the full prompt generates a significant higher number of equivalent paraphrases than all other prompts. Given these findings and the (rather small) price to pay in terms of lower percentage of semantically equivalent paraphrases generated by the full prompt (i.e., 57.9% *vs* 62.6% of the basic prompt), the full prompt is our choice for the subsequent experiments.

3.2 Iterative Paraphrase Generation

To increment the number of valid paraphrases generated by the LLM, we use an iterative approach. We ask several times the LLM to generate new paraphrases through the same prompt in order to obtain a larger number of paraphrases. At each iteration, we collect only the new paraphrases generated, i.e., we remove the ones already generated in previous iterations. We stop when no new paraphrase is generated in a given iteration. This process allows us to also include the paraphrases for which the model might have relatively low confidence in terms of imperative equivalence with the original sentence, but that can still help highlighting bugs.

While we never observed this behavior in our experiments, it is theoretically possible that the iterative paraphrase generation process never converges (i.e., the LLM always generates at least a new paraphrase). We expect that a tool integrating this approach allows developers to manually set a maximum budget (e.g., time, number of interactions, or number of tokens) to allocate for each generation so that it is possible to have a forced termination strategy that occurs in this unlikely event.

4 Empirical Study Design

The *goal* of the study is to understand whether an optimized interaction with an LLM can allow to better support VUI-testing via paraphrases generation as compared to specialized state-of-the-art techniques. As baselines, we use the approach by Guglielmi et al. (2023), namely VUI-UPSET, as well as ChatGPT as used in their work (i.e., using the basic prompt and a single interaction modality). For simplicity, we use "ChatGPT+" to refer to our interaction modality with ChatGPT (i.e., full prompt and iterative generation).

Our study is steered by the following research questions (RQs):

- RQ_0: *To what extent does the sampling temperature affect the generation of imperatively equivalent paraphrases?*
- RQ_1: *To what extent can ChatGPT+ generate imperatively equivalent paraphrases?*
- RQ_2: *To what extent do the generated paraphrases reveal bugs?*
- RQ_3: *To what extent can the generated paraphrases reduce bugs?*
- RQ_4: *Which categories of bugs can be detected and reduced using ChatGPT+?*
- RQ_5: *What is the role of the different improvements we included in ChatGPT+?*
- RQ_6: *To what extent can a smaller, open-weights model substitute GPT-4 in generating paraphrases for VUI-testing?*RQ_0 aims to evaluate the impact of the sampling temperature. RQ_{1-3} are shared with the previous study by Guglielmi et al. (2023) since we want to replicate such a study with ChatGPT+. RQ_4 aims at understanding what categories of bugs can be prevented by introducing the generated paraphrases into the VIM of the

skills, while RQ_5 is a sort of ablation study where we investigate the role played by the improvements we included in ChatGPT+. Finally, RQ_6 aims to assess whether a smaller, open-weight model can effectively replace GPT-4 for VUI paraphrase generation, thus obtaining a more cost-effective alternative of the approach.

The context of the study is the same of the study by Guglielmi et al. (2023) and consists of 40 Amazon Alexa skills totaling 631 selected seed utterances used for the experiments. Table 2 describes all the Alexa skills selected for our study.

4.1 Manual Evaluation of Imperative Equivalence

The identification of imperatively equivalent paraphrases is carried out manually through human annotation. Each generated paraphrase is evaluated against its corresponding seed sentence by human evaluators following a well-defined and consistent set of guidelines. In particular, annotators independently assess each ⟨seed sentence, paraphrase⟩ pair according to the following criteria.

1. **Intent preservation:** A paraphrase is marked as imperatively equivalent if it can reasonably be used by a human to trigger the same intent as the seed sentence in the VUI. The focus is on the expected action performed by the system, not on strict semantic similarity. For example, the seed utterance "when is the next game?" and the paraphrase "I need to know when the next game is" differ in syntactic form (interrogative vs. indirect declarative) but would be used by a user to request the same information and therefore to activate the same intent.
2. **Interchangeability in realistic usage:** Annotators evaluate whether a user could naturally substitute the paraphrase for the seed sentence in an actual voice interaction scenario, without changing their underlying goal. For example, the seed utterance "when is the next game" can be naturally replaced by the paraphrase "tell me about the next game schedule", as both would plausibly be used by a user to obtain the same scheduling information and therefore to trigger the same intent in the VUI.
3. **Tolerance to semantic reformulation:** Paraphrases are allowed to differ semantically, stylistically, or syntactically (e.g., indirect, polite, interrogative, or idiomatic forms), as long as they plausibly express the same command. For example, the seed utterance "tell me about this place" and the paraphrase "I'm curious about this place, can you tell me about it?" differ in wording and sentence structure, but both naturally express the user's intent to request information about the same entity and would be expected to trigger the same intent in the VUI.
4. **Naturalness and plausibility:** The paraphrase must represent a formulation that a human user could realistically pronounce in a voice interaction. Utterances that are grammatically incorrect, unnatural, incomplete, or unlikely to be used in practice are rejected, even if loosely related in meaning. For example, while a command such as "delete from favorites" is natural in a voice interaction, a paraphrase like "expunge from favorites collection" is considered not imperatively equivalent due to its unnatural and overly formal wording, which makes it unlikely to be used by a real user.
5. **No change in constraints or scope:** Paraphrases that introduce additional constraints, remove essential information, or alter the scope of the request (e.g., adding conditions

Table 2 Skills used for the empirical evaluation and their characteristics

		# seeds		#generated paraphrases		
Skill	Description	Total	Selected	ChatGPT+	ChatGPT	VUI-UPSET
Berry bash	Quiz game about a topic	25	20	2264	188	769
Boys basketball	Info on basketball match	5	5	1738	44	6
Button trivia	Single- and multi-player quiz game	24	17	2123	165	523
Charity roster	Allows to get to know charities and make donations	12	8	1491	91	159
City guide	Recommends activities in the city	7	6	1290	68	23
College finder	Query the public dataset of colleges	102	61	13691	576	6115
Cyclist assistant	Provides info to cyclist	5	5	1077	50	155
Feed reader	Allows you to play your feed	108	51	2979	402	4693
Forget me not	VUI app for last minute reminders	11	4	470	40	84
Girls volleyball	Info on volleyball match	14	9	2825	81	15
Greeting sender	Order management and sharing	17	16	2697	154	319
Happy birthday	Stores birthday information	6	6	603	54	211
History facts	Provides interesting historical facts	7	4	378	34	13
Humour me	Provides funny stories	12	10	973	93	994
Internet archive	VUI app of Internet Archive	19	12	1506	106	263
Ilama facts	Provides facts and trivia	8	8	726	70	35
Math facts	Provides arithmetic exercises	7	4	632	36	44
Memory	Memory-matching game	4	4	332	33	171
Multistream audio	Provides multiple audio, event and control managers	15	5	338	45	836
My barkeeper	Store information on cocktails	14	110	21098	81	12385
Name the show	Guess game about TV shows	18	15	1638	143	167
Next prayer	Shows prayer times for Japan	4	3	140	30	47
Notification	Demonstrates messaging for different errors	14	9	1304	80	78
Particle cloud	Provides access to Particle devices	19	14	3433	128	391
Piano player	Teaches the piano	9	7	1182	66	201
Pocket	Manages lists of items	21	11	1304	100	1421
Premium fact	Allows you to add new categories	16	14	1232	120	35
Premium hello world	Adds greetings in several languages	23	21	3167	193	329

Table 2 (continued)

		# seeds		#generated paraphrases		
Skill	Description	Total	Selected	ChatGPT+	ChatGPT	VUI-UPSET
Quiz game	Quiz game about a topic	6	5	241	48	268
Radio	Allows the radio to be managed	15	11	4564	101	41
Salesforce	Identifies opportunities on Salesforce	12	8	1280	78	1203
Scheduling	Schedules calls and appointments	9	9	1323	84	102
Store Amazon pay	VUI app for the Amazon Store	45	24	3668	235	1168
Tasty beverage	Allows you to create a list	18	13	5131	115	52
Timers	Manages custom timers	16	11	1432	78	473
Trading teacher	Teaches options trading to end users	31	23	2180	212	78
Video app	Allows to play a video	8	8	837	78	39
Voice developer	Allows the voice to be edited and provides news	48	36	7575	346	3826
Weather wear	Offers climate-appropriate clothing options	13	11	7450	106	791
Week calendar	Allows to plan the vacation	12	7	690	62	1342
Total		779	631	109,002	4,814	39,865

or changing objects) are not considered imperatively equivalent. For example, the seed utterance "how should I dress?" expresses a general request for clothing advice, whereas the paraphrase "can you suggest what attire I need for my ride into the wild?" introduces a specific contextual constraint (the ride into the wild), affecting the scope of the request and modifying its underlying intent. As a result, the two utterances are not considered imperatively equivalent.

4.2 Experimental Procedure

To answer the first three RQs, we ran ChatGPT+ on the 40 skills and compared it with the baseline by replicating the exact procedure adopted by Guglielmi et al. (2023), which is reported below. Note that running ChatGPT+ on all 40 skills required about 63 hours (1.6 hours per skill, on average) and 29,338 API requests, consuming about 11.6M tokens (184.69$ at the pricing available during the experiments, i.e., ∼4.6$ per skill on average).

To answer RQ_0, we assess the extent to which paraphrases generated with three temperature settings are imperatively equivalent to their seed utterances. We vary only the sampling temperature in ChatGPT+ (i.e., full prompt and iterative generation) and compare $t_0 = 0.5$, $t_1 = 1.0$ (default t), $t_2 = 1.5$. We run this comparison on 10 randomly selected skills from the 40 used in our main study. We do not include all 40 because of the economic and human-annotation costs of ground-truthing imperative equivalence. For each temperature, we randomly sample, per skill, a number of generated paraphrases that guarantees a 5% margin of error at 95% confidence level. We obtain the following overall sample sizes: $t_0 = 1,989$, $t_1 = 2,685$ (default t), $t_2 = 3,254$. Three of the authors were involved in the evaluation.

Each generated paraphrase was assigned and independently evaluated by two authors (evaluators). Each evaluator checked whether the generated paraphrase and the respective seed utterance were imperatively equivalent. If the two original evaluators disagreed, the third one independently evaluated the paraphrase and the three authors discussed the case, aiming at reaching consensus. The two evaluators originally assigned to each paraphrase had disagreements in 357 cases (i.e., 17.95%) in t_0, 639 cases (i.e., 23.8O%) in t_1 and in 528 cases (i.e., 16.22%) in t_2. After discussion, consensus was reached for all the paraphrases. In total, 15,378 manual inspections were needed ((1,989 + 2,684 + 3,254) $\times$2 evaluators + 1,524 further checks). Most of the conflicts arose in borderline situations in which the paraphrases were interpreted differently by the evaluators. In details, the conflicts analyzed often related to cases in which unusual phrases or slang were used in the paraphrases. For each temperature we report: (i) percentage of imperatively equivalent paraphrases, and (ii) the absolute number of accepted paraphrases.

To answer RQ_1, we assess the extent to which the paraphrases generated by the three approaches are imperatively equivalent to the seed utterance. We run ChatGPT+ with the full prompt template on each seed utterance. Since the experimental procedure is the same used in the previous work, we do not re-ran VUI-UPSET and ChatGPT and took the results from the original paper (Guglielmi et al. 2023).

ChatGPT+ generated a very high number of total paraphrases (109,002). Therefore, we randomly selected a sample of the utterances it generated for manual evaluation. Specifically, we choose from each skill a sample ensuring a 5% margin of error with 95% confidence level. In total, we selected 11,615 paraphrases for ChatGPT+. This is substantially more than the 6,291 inspected for VUI-UPSET and 3,475 for ChatGPT in the previous work Guglielmi et al. 2023. Three of the authors were involved in the evaluation. Each generated paraphrase was assigned and independently evaluated by two authors (evaluators). Each evaluator checked whether the generated paraphrase and the respective seed utterance were imperatively equivalent. If the two original evaluators disagreed, the third one independently evaluated the paraphrase and the three authors discussed the case, aiming at reaching consensus. The two evaluators originally assigned to each paraphrase had disagreements in 2,068 cases (i.e., $\sim$ 18%).

After discussion, consensus was reached for all the paraphrases. In total, 25,298 manual inspections were needed (11,615 $\times$ 2 evaluators + 2,068 further checks). Most of the conflicts arose in borderline situations in which the paraphrases were interpreted differently by the evaluators. In details, the conflicts analyzed often related to cases in which unusual phrases or slang were used in the paraphrases. For example, “let’s kick off the game” generated as paraphrase for the seed utterance “get started”, with “kick off” assessed as an invalid paraphrase by one of the evaluators due to its unlikely usage to indicate the willing to start a game.

We compare the three approaches in terms of percentage imperatively equivalent paraphrases. To this end, we use the Wilcoxon Signed-Rank test (Woolson 2007) in which the null hypothesis is that there is no significant difference between ChatGPT+ and the VUI-UPSET and ChatGPT+ and ChatGPT. We do not perform a comparison between VUI-UPSET and ChatGPT since it has been already done in the original study. We reject the null hypothesis if the *p*-value is lower than 0.05. We also compute the effect size to quantify

the magnitude of the significant differences we find. We use Cliff's Delta (Macbeth et al. 2011) since it is non-parametric. Cliff's delta lays in the interval [-1, 1]: The effect size is **negligible** for $|\delta| < 0.148$, **small** for $0.148 \leq |\delta| < 0.33$, **medium** for $0.33 \leq |\delta| < 0.474$, and **large** for $|\delta| \geq 0.474$. As we did in prompt tuning evaluation, we also estimate the absolute number of equivalent paraphrases within the original population of paraphrases: If there is no overlap between the intervals of the two approaches, we say that one generates a significantly higher number of imperatively equivalent paraphrases for a given skill as compared to the other (with 95% confidence level).

To answer RQ_2, we only consider the imperatively equivalent paraphrases identified in RQ_1 and define and execute test cases for such paraphrases. We used the *NLU evaluation* function available in the Amazon Developer Console. We compute for each skill the percentage of generated bug-revealing paraphrases by computing the number of *FAILED* test cases divided by the total number of valid paraphrases. The evaluation aims to assess whether one of the approaches outperforms the others in terms of the quality of paraphrases generated. Next, as we do in RQ_1, we estimate the absolute number of paraphrases that reveal bugs for each approach within the original population of paraphrases using an analogous procedure. Also in this case, we consider 5% margin of error for both such values. We use the Wilcoxon signed-rank test (Woolson 2007) to compare the percentage of bug-revealing paraphrases. The null hypothesis is that there is no difference in the percentage of bug-revealing paraphrases generated by the approaches. Also in this case, we report the Cliff's Delta (Macbeth et al. 2011). To compare the absolute number of bug-revealing paraphrases, we use the same approach used in RQ_1: If there is no overlap between the confidence intervals of the three approaches, we say that the one that generates a larger number of bug-revealing paraphrases for a given skill is significantly better than the others for such a skill.

To answer RQ_3, we conducted two separate assessments where we add the paraphrases generated with the best approach resulting from RQ_2 in the VIM for the respective skills, to check whether they help fixing the detected bugs. In the first assessment, we added to the VIM all the imperatively equivalent paraphrases we found in RQ_1 and we used them to test whether this helps reducing the bugs we observed in RQ_2. In such an analysis, however, the same sentences added to the VIM are also used for testing, thus not fully representing a real application scenario. To fix this, in the second assessment we perform a two-fold cross validation, with half of the paraphrases used for testing and half added to the VIM. In particular, for each skill, we randomly divided the set of paraphrases generated for each intent in two sets, using one of them (50% of the instances) to improve the speech interaction model of the skill, and the remaining one (the other 50%) to test this new version of the skill. If we still observe a reduction of the bugs in this second assessment scenario as compared to RQ_2 's findings, we can conclude that the generated paraphrases included in the VIM are useful. For both the assessment scenarios, we compute and report the percentage and absolute number of failed test cases and the estimated number of bugs that could be revealed by using the complete set of paraphrases generated by ChatGPT+ (similarly to RQ_2 and RQ_1).

To answer RQ_4, we manually analyzed all bugs identified by the best approach resulting from RQ_2 aiming at characterizing them and explain why they occurred. Two of the authors were involved in the evaluation. Each bug was evaluated and assigned to a category independently by each evaluator, who analyzed the buggy paraphrase comparing the expected

activation intent with the activated intent. If the two evaluators disagreed, they discussed the case with the goal of reaching a consensus. The two evaluators disagreed in 41 cases (1,18%), but reached consensus in all such cases. Besides analyzing the bug categories, we also studied how introducing paraphrases in the VIM (RQ_3) can help improve the skill for each category of bug. To do this, we also categorized the bugs identified after introducing paraphrases in the VIM (RQ_3) in the more realistic evaluation (two-fold cross validation) and compared the distributions in the bug categories.

To answer RQ_5, we run an ablation study on a set of 10 randomly selected skills of the 40 we considered to answer RQ_{1-2}. We run four variants of ChatGPT+ on such a set of skills. Each of such variations has an improvement as compared to the previous one. The first one is ChatGPT+ $_{\text{-MPI}}$, i.e., ChatGPT+ without the improved model (we use GPT 3.5 instead of GPT 4), the full prompt (we use the basic prompt), and the iterative generation (we only ask once). Note that this is exactly the basic approach used by Guglielmi et al. (2023) that we also used in RQ_{1-2} (reported as ChatGPT). The second one, ChatGPT+ $_{\text{-PI}}$, is ChatGPT+ without the advanced prompt and the iterative generation, but with the more recent model (GPT 4). The third one, ChatGPT+ $_{\text{-I}}$, is ChatGPT+ without iterative generation, but with the updated model and the full prompt. Finally, the last one is the full ChatGPT+. We performed the same analysis we conducted to answer (i) RQ_1, i.e., we study the percentage and absolute number of imperatively equivalent paraphrases, and (ii) RQ_2, i.e., we study the percentage and absolute number of bug-revealing paraphrases. We perform pair-wise comparisons between ChatGPT+ $_{\text{-MPI}}$ and ChatGPT+ $_{\text{-PI}}$, ChatGPT+ $_{\text{-PI}}$ and ChatGPT+ $_{\text{-I}}$, and ChatGPT+ $_{\text{-I}}$ and ChatGPT+, to understand how each step improves over the previous one.

To answer RQ_6, we replicate the analyses performed to answer RQ_1 (imperative equivalence), RQ_2 (bug finding capabilities) and RQ_3 (bug reduction). We use GPT-oss-20B as the underlying model instead of GPT-4. We use the full prompt template on each seed utterance. The only difference is that we focus on the 10 skills used in the ablation study instead of the full set of 40 skills. We did this because fully replicating the whole experiment would have required a huge manual investigation effort. We adopted the same design used to answer the previously-mentioned RQ_{1-3}. As GPT-4, GPT-oss-20B generated a large number of paraphrases (35,067 in total). So, we drew a random sample from each skill sufficient to ensure a 5% margin of error at 95% confidence, resulting in a sample of 3,036 paraphrases overall. The two original evaluators disagreed in 520 cases (i.e., 17.14%), solved via open discussion. In our analysis of paraphrases generated with GPT-oss-20B, we observed a subset of invalid outputs, i.e., grammatically incorrect generations, containing extraneous symbols, or leaving completion to the user. In the validated sample, we found 14 invalid paraphrases. The most common error patterns were prompts requiring completion (e.g., “find a school called —”), slot placeholders (e.g., “find schools in [state]”), incorrect punctuation (e.g., “[recommend an attraction,”), and HTML tags (e.g., “
”). Based on preliminary evidence prior to launch, our hypothesis is that explicit reasoning behavior occasionally distracted the model from the task, increasing the probability of such invalid outputs. As for the first analysis (imperative equivalence), we compare the results obtained with GPT-oss-20B with the ones obtained by ChatGPT+ in terms of: (i) percentage of imperatively equivalent paraphrases, and (ii) the absolute number of accepted paraphrases. Finally, as for the third analysis (bug reduction), we base our comparison on the percentage and absolute number of failed test cases and the estimated number of bugs that could be revealed by using the complete set of paraphrases generated.

5 Empirical Study Results

This section reports the results of the four research questions of our study.

5.1 RQ_0: Temperature Impact

Table 3 reports the results of the sampling temperature in ChatGPT+ related to the outcomes of imperative equivalence between the seed utterance and the generated paraphrases. ChatGPT+ t_1 obtain the best results on five skill (City Guide, College Finder, History Facts, Next Prayer, Video App), while t_0 obtain the best results on the remaining five. As we can see in Table 3 t_2 never achieves the best percentage for any skill. Overall results show that t_1 achives the highest percentage of imperatively equivalent paraphrases (70.95%), only slightly higher than t_0 (70.09%). Although t_2 achieves the lowest percentage of equivalents, it produces the highest absolute number of imperatively equivalent paraphrases generated (46,888 ± 3,754) across the entire population thanks to the higher overall number of paraphrases generated 3,254. However, although ChatGPT+ t_2 achieves the best overall result in terms of absolute numbers, we observed a notable drawback compared to ChatGPT+ t_0 and ChatGPT+ t_1. Specifically, paraphrases generated by ChatGPT+ t_2 occasionally included *invalid outputs*, such as paraphrases with completely unrelated context to the seed, grammatical errors, or unnecessary punctuation. In ChatGPT+ t_2 sample we identified 23 such invalid cases. For instance, the seed *"go out"* was paraphrased as *"alignna obs'),"*, and the seed *"lookup schools by location"* was paraphrased as *"tell auga ery joven"*. Although ChatGPT+ t_2 achieves the highest absolute number of imperatively equivalent paraphrases, it also introduces invalid results. In contrast, ChatGPT+ t_1 provides the best balance, delivering the highest percentage of imperatively equivalent paraphrases overall and outperforming ChatGPT+ t_0 without the drawbacks observed in ChatGPT+ t_2.

Answer to RQ_0. ChatGPT+ t_2 maximizes the absolute number of paraphrases generated but occasionally generates paraphrases that are out of context or improperly formatted. We therefore adopt ChatGPT+ t_1 which achieves the highest overall equivalence rate and a better balance than ChatGPT+ t_0 and ChatGPT+ t_2.

5.2 RQ_1: Paraphrase Equivalence

Table 4 reports the results of the imperative equivalence between the seed utterance and the generated paraphrases. While there are a few skills for which the percentage of equivalent paraphrases is higher for VUI-UPSET and ChatGPT as compared to ChatGPT+, for the majority of them (32 out of 40 skills) ChatGPT+ ensures the best performance. Indeed, when considering the overall results (last row in Table 4), ChatGPT+ generates a higher percentage of imperatively equivalent paraphrases (79.5%) than ChatGPT (58.7%) and VUI-UPSET (44.8%).

The statistical test confirms that the difference between ChatGPT+ and the two other approaches is significant (p-value < 0.001 for both), with a *large* effect size ($\delta = 0.72$ for VUI-UPSET and $\delta = 0.68$ for ChatGPT).

Table 3 Results for RQ_0: effect of sampling temperature ($t_0 = 0.5$, $t_1 = 1.0$, $t_2 = 1.5$) on the generation of *imperatively equivalent* paraphrases. ○ indicates that the column refers to the evaluated sample; ⋆ indicates that it refers to the estimated value on the whole population. We report in boldface the significantly best results, while we mark with a * the cases in which the best result is shared by two approaches

	RQ_1: Paraphrase equivalence											
	ChatGPT+ $t_0 = 0.5$				ChatGPT+ $t_1 = 1.0$				ChatGPT+ $t_2 = 1.5$			
Skill	#gen○	#equiv○	%equiv	#equiv⋆	#gen○	#equiv○	%equiv	#equiv⋆	#gen○	#equiv○	%equiv	#equiv⋆
City guide	200	93	46.50%	193 ± 21	294	151	**51.36%**	663 ± 65	348	162	46.55%	**1,834 ± 197**
College finder	356	231	64.89%	3,144 ± 242	374	263	**70.32%**	9,628 ± 685	379	194	51.19%	**20,255 ± 1,979**
Forget me not	94	89	**94.68%**	118 ± 6	212	188	88.68%	417 ± 24	272	176	64.71%	**612 ± 47**
History Facts	151	125	82.78%	213 ± 13	191	169	**88.48%**	334 ± 19	296	212	71.62%	**922 ± 64**
Next Prayer	66	59	89.39%	71 ± 4	103	96	**93.20%**	130 ± 7	224	196	87.50%	**501 ± 29**
Notification	175	131	**74.86%**	239 ± 16	297	205	69.02%	900 ± 65	340	169	49.71%	**1,486 ± 149**
Premium fact	209	151	**72.25%**	335 ± 23	292	187	64.04%	789± 62	340	194	57.06%	**1,775 ± 156**
Salesforce	214	141	**65.89%**	321 ± 24	296	181	61.15%	783 ± 64	346	221	63.87%	**2,318 ± 181**
Video app	203	117	57.64%	248 ± 22	261	181	**69.35%**	580± 42	334	217	64.97%	**1,671 ± 129**
Voice developer	321	257	**80.06%**	1,590± 99	365	284	77.81%	5,894 ± 379	375	291	77.60%	**12,780 ± 823**
Overall	1,989	1,394	70.09%	6,594 ± 470	2,685	1,905	**70.95%**	20,006 ± 1,410	3,254	2,032	62.45%	**46,888 ± 3,754**

ChatGPT+ also achieves significantly better results in terms of absolute number of equivalent paraphrases for most of the skills (39 out of 40), with the only exception being "Week calendar" for which ChatGPT+ cannot significantly outperform ChatGPT.

Answer to RQ_1. ChatGPT+ achieves significantly better results than VUI-UPSET and ChatGPT in terms of percentage and absolute number of generated imperatively equivalent paraphrases.

5.3 RQ_2: Paraphrase Capability of Finding Bugs

We report in Table 5 the results of the analysis conducted to answer RQ_2. ChatGPT+ achieves the best results in terms of percentage of generated paraphrases able to unveil bugs in the tested skills, achieving an overall 37.6% of bug-revealing paraphrases as compared to the 27.4% of ChatGPT and 19.2% of VUI-UPSET. The statistical hypothesis test reports the difference as statistically significant (p-value = 0.008 *vs* ChatGPT) and (< 0.001 *vs* VUI-UPSET). The effect size is *large* ($\delta = 0.50$) for the comparison with VUI-UPSET and *small* ($\delta = 0.23$) for the comparison with ChatGPT.

ChatGPT+ generates the highest absolute number of bug-revealing paraphrases compared to both baselines for 39 out of 40 skills, with VUI-UPSET being the best on the remaining skill. Overall, ChatGPT+ generates between 39,361 and 50,169 bug-revealing paraphrases, against the 4,351 to 6,053 of VUI-UPSET and the 726 to 1,008 of ChatGPT.

Figure 2 shows an example of bugs found in the `RegisterBirthdayIntent` in *Alexa Happy Birthday Skill*. In Step 1 we show the seed sentences integrated by the developer in the VIM to record the birthday of the user. When providing the seed "record my birthday" to ChatGPT+, it generates 16 paraphrases, four of which are shown in Step 2. Three of them result in a bug (i.e., all but "keep record of my birthday"), as shown in Step 3.

Answer to RQ_2. ChatGPT+ generates a higher percentage and absolute number of bug revealing paraphrases than VUI-UPSET and ChatGPT for most of the skills.

5.4 RQ_3: Bug Reduction

Since ChatGPT+ achieves the best results in terms of number of bug-revealing paraphrases (RQ_2), we use this approach to answer RQ_3. Table 6 reports the results for such an RQ, and we discuss them below, divided by scenario.

Including all the Paraphrases When we include all the paraphrases in the VIMs of the skills and use them as tests as well, as expected, only a very small number of bug (37) remains in two skills (Internet archive and Voice developer). Such a number must be compared with the 3,464 bugs identified in the original 40 skills. As for *Internet Archive*, the inclusion of all semantically equivalent paraphrases reduced the number of bugs from 107 to 8. Instead, in the skill *Voice Developer*, the number of bugs remained unchanged at 29. By looking at

Table 4 Results of RQ_1. ○ indicates that the column refers to the evaluated sample, while ⋆ indicates that it refers to the estimated value of the whole population (95% confidence level). We report in boldface the significantly best results, while we mark with a * the cases in which the best result is shared by both of the approaches

	ChatGPT+					ChatGPT					VUI-UPSET				
Skill	#gen	#gen○	#equ○	%equ	#equ⋆	#gen	#gen○	#equ○	%equ	#equ⋆	#gen	#gen○	#equ○	%equ	#equ⋆
Berry bash	2,264	329	301	**91.49%**	**2,071 ± 113**	188	126	99	78.57%	148 ± 9	769	256	97	37.89%	291 ± 38
Boys basketball	1,738	315	294	**93.33%**	**1,622 ± 87**	44	40	32	80.00%	35 ± 2	6	6	3	50.00%	3 ± 0
Button trivia	2,123	324	227	**70.06%**	**1,487 ± 106**	165	114	51	44.74%	74 ± 8	523	222	103	46.40%	243 ± 26
Charity roster	1,491	305	293	**96.07%**	**1,424 ± 67**	91	74	52	70.27%	64 ± 5	159	112	81	72.32%	115 ± 8
City guide	1,290	294	151	51.36%	**663 ± 65**	68	57	26	45.61%	31 ± 3	23	22	17	**77.27%**	18 ± 1
College finder	13,691	374	263	**70.32%**	**9628 ± 685**	576	230	99	43.04%	248 ± 29	6,115	362	138	38.12%	2,331±306
Cyclist assistant	1,077	281	224	79.72%	**859 ± 54**	50	44	39	**88.64%**	44 ± 3	155	111	91	81.98%	127 ± 8
Feed reader	2,979	340	255	**75.00%**	**2,234 ± 149**	402	197	76	38.58%	155 ± 20	4,693	355	108	30.42%	1,428±235
Forget me not	470	212	188	88.68%	**417 ± 24**	40	36	33	**91.67%**	37 ± 2	84	69	25	36.23%	30 ± 4
Girls volleyball	2,825	338	308	**91.12%**	**2,574 ± 141**	81	67	51	76.12%	62 ± 4	15	14	10	71.43%	11 ± 1
Greeting sender	2,697	333	252	**75.68%**	**2,041 ± 135**	154	110	40	36.36%	56 ± 8	319	175	32	18.29%	58 ± 16
Happy birthday	603	235	178	**75.74%**	**457 ± 30**	54	48	23	47.92%	26 ± 3	211	136	47	34.56%	73 ± 11
History facts	378	191	169	88.48%	**334 ± 19**	34	31	27	87.10%	30 ± 2	13	13	12	**92.31%**	12 ± 1
Humour me	973	273	209	**76.56%**	**745 ± 49**	93	75	26	34.67%	32 ± 5	994	277	185	66.79%	664 ± 50
Internet archive	1,506	305	239	**78.36%**	**1,180 ± 75**	106	84	62	73.81%	78 ± 5	263	156	66	42.31%	111 ± 13
llama facts	726	251	184	**73.31%**	**532 ± 36**	70	61	26	42.62%	30 ± 2	35	32	32	100.00%	34 ± 1
Math facts	632	239	213	**89.12%**	**563 ± 32**	36	33	20	60.61%	22 ± 4	44	40	32	80.00%	35 ± 2
Memory	332	178	158	**88.76%**	**295 ± 17**	33	30	15	50.00%	17 ± 2	171	119	64	53.78%	92 ± 9
Multistream audio	338	180	162	**90.00%**	**304 ± 17**	45	40	20	50.00%	23 ± 2	836	263	40	15.21%	127 ± 42
My barkeeper	21,098	377	329	**87.27%**	**18,412±1,055**	81	252	165	65.48%	53 ± 4	12,385	373	178	47.72%	5,910±619
Name the show	1,638	310	249	**80.32%**	**1,316 ± 82**	143	104	52	50.00%	72 ± 7	167	117	84	71.79%	120 ± 8
Next prayer	140	103	96	**93.20%**	**130 ± 7**	30	28	21	75.00%	23 ± 2	47	42	21	50.00%	24 ± 2
Notification	1,304	297	205	69.02%	**900 ± 65**	80	66	40	60.61%	48 ± 4	78	65	46	**70.77%**	55 ± 4
Particle cloud	3,433	346	281	81.21%	**2,788 ± 172**	128	96	79	**82.29%**	105 ± 6	391	194	53	27.32%	107 ± 20
Piano player	1,182	290	226	**77.93%**	**921 ± 59**	66	56	29	51.79%	34 ± 3	201	132	50	37.88%	76 ± 10
Pocket	1,304	296	227	**76.69%**	**1,000 ± 65**	100	80	47	58.75%	59 ± 5	1,421	303	86	28.38%	403 ± 71
Premium fact	1,232	292	187	64.04%	**789 ± 62**	120	92	51	55.43%	67 ± 6	35	32	23	**71.88%**	25 ± 2

Table 4 (continued)

	ChatGPT+					ChatGPT					VUI-UPSET				
Skill	#gen	#gen○	#equ○	%equ	#equ⋆	#gen	#gen○	#equ○	%equ	#equ⋆	#gen	#gen○	#equ○	%equ	#equ⋆
Premium hello word	3,167	343	238	**69.39%**	**2,198 ± 158**	193	129	79	61.24%	118 ± 10	329	177	70	39.55%	130 ± 16
Quiz game	241	147	144	**97.96%**	**233 ± 8**	48	43	26	60.47%	29 ± 2	268	158	64	40.51%	109 ± 13
Radio	4,564	354	293	**82.77%**	**3,778 ± 228**	101	80	41	51.25%	52 ± 5	41	37	18	48.65%	20 ± 2
Salesforce	1,280	296	181	61.15%	**783 ± 64**	78	64	40	**62.50%**	49 ± 4	1,203	291	134	46.05%	554 ± 60
Scheduling	1,323	298	243	**81.54%**	**1,079 ± 66**	84	69	53	76.81%	65 ± 4	102	81	54	66.67%	68 ± 5
Store Amazon pay	3,668	348	301	**86.49%**	**3,173 ± 183**	235	146	90	61.64%	145 ± 12	1,168	289	61	21.11%	247 ± 58
Tasty beverage	5,131	356	252	**70.79%**	**3,632 ± 257**	115	89	47	52.81%	61 ± 6	52	46	20	43.48%	23 ± 3
Timers	1,432	301	246	**81.73%**	**1,170 ± 72**	78	65	36	55.38%	43 ± 4	473	205	95	46.34%	219 ± 24
Trading teacher	2,180	327	251	**76.76%**	**1,673 ± 109**	212	136	77	56.62%	120 ± 11	78	65	47	72.31%	56 ± 4
Video app	837	261	181	**69.35%**	**580 ± 42**	78	64	29	45.31%	35 ± 4	39	35	23	65.71%	26 ± 2
Voice developer	7,575	365	284	**77.81%**	**5,894 ± 379**	346	182	121	66.48%	230 ± 17	3,826	349	106	30.37%	1,162±191
Weather wear	7,450	365	328	**89.86%**	**6,695 ± 373**	106	83	57	68.67%	73 ± 5	791	259	219	84.56%	669 ± 40
Week calendar	690	246	225	**91.46%**	***631 ± 35**	62	54	42	77.78%	48 ± 3	1342	299	117	39.13%	***525 ± 67**
Overall	109,002	11,615	9,235	**79.51%**	**86,667±5,450**	4,814	3,475	2,039	58.68%	2,785 ± 292	39,865	6,289	2,752	44.76%	16,331 ± 1,957

Table 5 Results of RQ_2. ○ indicates that the column refers to the evaluated sample, while ⋆ indicates that it refers to the estimated value on the whole population (95% confidence level). We report in boldface the significantly best results, while we mark with a * the cases in which the best result is shared by two approaches

	ChatGPT+				ChatGPT				VUI-UPSET			
Skill	#equ○	#bugs○	%bugs	#bugs⋆	#equ○	#bugs○	%bugs	#bugs⋆	#equ○	# bugs○	%bugs	#bugs⋆
Berry bash	301	81	**26.91%**	**609 ± 113**	99	16	16.16%	25 ± 8	97	0	0.00%	8 ± 8
Boys basketball	294	199	**67.69%**	**1,176 ± 87**	32	15	46.88%	18 ± 2	3	0	0.00%	0 ± 0
Button trivia	227	35	15.42%	**327 ± 106**	51	17	**33.33%**	27 ± 4	103	14	13.59%	37 ± 13
Charity roster	293	133	**45.39%**	**677 ± 75**	52	21	40.38%	28 ± 3	81	1	1.23%	4 ± 3
City guide	151	84	55.63%	**718 ± 65**	26	18	**69.23%**	24 ± 2	17	0	0.00%	0 ± 0
College finder	263	130	**49.43%**	**6,767 ± 685**	99	38	38.38%	106±14	138	51	36.96%	975± 132
Cyclist assistant	224	127	**56.95%**	**613 ± 54**	39	7	17.95%	9 ± 2	91	0	0.00%	3 ± 3
Feed reader	255	16	6.27%	187 ± 149	76	11	14.47%	25 ± 9	108	18	**16.67%**	**277± 83**
Forget me not	188	12	6.38%	**30 ± 24**	33	0	0.00%	1 ± 1	25	3	**12.00%**	4 ± 1
Girls volleyball	308	170	**55.19%**	**1,559 ± 141**	51	21	41.18%	27 ± 3	3	0	0.00%	0 ± 0
Greeting sender	252	89	**35.32%**	**953 ± 135**	40	8	20.00%	13 ± 3	32	5	15.63%	12 ± 4
Happy birthday	178	84	**47.19%**	**285 ± 30**	23	2	8.70%	3 ± 1	47	13	27.66%	23 ± 4
History facts	169	86	**51.50%**	**195 ± 19**	27	11	40.74%	13 ± 2	12	1	8.33%	1 ± 0
Humour me	209	42	**20.10%**	196 ± 49	26	1	3.85%	2 ± 1	185	5	2.70%	30 ± 25
Internet archive	239	107	**44.77%**	**674 ± 75**	62	14	22.58%	19 ± 4	66	26	39.39%	49 ± 6
llama facts	184	165	**89.67%**	**651 ± 36**	26	23	88.46%	29 ± 2	32	21	65.63%	23 ± 2
Math facts	213	12	5.63%	**36 ± 32**	20	7	**35.00%**	8 ± 1	32	7	21.88%	9 ± 2
Memory	158	82	51.90%	**172 ± 17**	15	8	53.33%	10 ± 1	64	39	**60.94%**	61 ± 5
Multistream audio	162	36	**22.22%**	**75 ± 17**	20	0	0.00%	1 ± 1	40	7	17.50%	30 ± 8
My barkeeper	329	158	**48.02%**	**10,132± 1,055**	165	30	18.18%	22 ± 8	178	71	39.89%	2604± 326
Name the show	249	91	36.55%	**599 ± 82**	52	26	**50.00%**	39 ± 4	84	37	44.08%	56 ± 6
Next prayer	96	4	4.17%	***6 ± 6**	21	5	23.81%	***6 ± 1**	21	0	0.00%	1 ± 1
Notification	205	129	62.93%	**821 ± 65**	40	26	**65.00%**	34 ± 3	46	21	45.65%	27 ± 3
Particle cloud	281	24	**8.54%**	**293 ± 172**	79	6	7.59%	10 ± 4	53	3	5.66%	8 ± 5
Piano player	226	14	6.19%	**73 ± 59**	29	2	**6.90%**	3 ± 1	50	0	0.00%	2 ± 2
Pocket	227	1	0.44%	35 ± 35	47	0	0.00%	2 ± 2	86	1	**1.16%**	15 ± 14
Premium fact	187	137	**73.26%**	**903 ± 62**	51	24	47.06%	34 ± 4	23	3	13.04%	4 ± 1

Table 5 (continued)

	ChatGPT+				ChatGPT				VUI-UPSET			
Skill	#equ○	#bugs○	%bugs	#bugs⋆	#equ○	#bugs○	%bugs	#bugs⋆	#equ○	# bugs○	%bugs	#bugs⋆
Premium hello word	238	102	42.86%	**1,357 ± 158**	79	56	**70.89%**	91 ± 6	70	6	8.57%	13 ± 7
Quiz game	144	4	**2.78%**	**9 ± 9**	26	0	0.00%	1 ± 1	64	0	0.00%	3 ± 3
Radio	293	240	**81.91%**	**3,738 ± 228**	41	27	65.85%	37 ± 3	18	1	5.56%	2 ± 1
Salesforce	181	76	**41.99%**	**537 ± 64**	40	3	7.50%	5 ± 2	134	50	37.31%	229± 31
Scheduling	243	93	**38.27%**	**506 ± 66**	53	4	7.55%	6 ± 2	54	5	9.26%	8 ± 3
Store Amazon pay	301	123	40.86%	**1,499 ± 183**	90	33	36.67%	57 ± 8	61	25	**40.98%**	125± 15
Tasty beverage	252	80	31.75%	**1,629 ± 257**	47	19	**40.43%**	27 ± 3	20	2	10.00%	3 ± 1
Timers	246	39	15.85%	**227 ± 72**	36	2	5.56%	3 ± 1	95	35	**36.84%**	89 ± 12
Trading teacher	251	141	**56.18%**	**1,225 ± 109**	77	37	48.05%	63 ± 7	47	7	14.89%	9 ± 2
Video app	181	117	**64.64%**	**541 ± 42**	29	0	0.00%	1 ± 1	23	1	4.35%	2 ± 1
Voice developer	284	29	10.21%	**774 ± 379**	121	12	9.92%	25 ±12	106	30	**28.30%**	383± 68
Weather wear	328	169	**51.52%**	**3,839 ± 373**	57	9	15.7%	13 ± 4	219	18	8.22%	58 ± 35
Week calendar	225	3	**1.33%**	22 ± 22	42	0	0.00%	1 ± 1	117	0	0.00%	15 ± 15
Overall	9,232	3,464	**37.52%**	**44,765 ± 5,404**	2,039	559	27.42%	867 ± 141	2,752	527	19.15%	5,202 ± 851

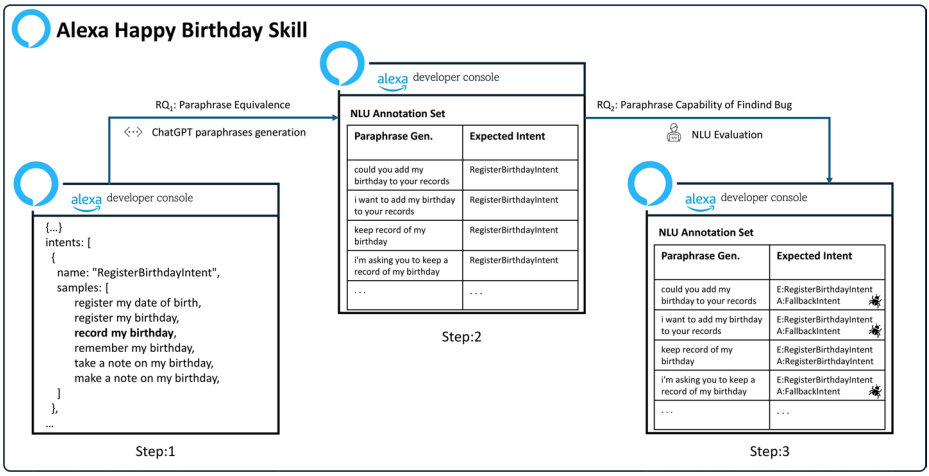

Fig. 2 Example of paraphrases capability of finding bug on *Alexa Happy Birthday Skill*

the VIM of such a skill, we found that this might be due to design issues of the VUI. For example, there are two intents named *PollyVoiceIntent* and *SSMLIntent*. Both of them aim to trigger commands related to managing voice effects, and their similarity might cause activation errors. A solution could be merging them and finding a way to handle the different functionalities (e.g., by asking a further question after activating the intent).

Separating VIM and Tests In the two-fold cross-validation, we naturally observed a higher number of bugs left since we (i) add a lower number of utterances, and, most importantly, (ii) we test the skill on different utterances than the ones used for improving the VIM. In such a more realistic evaluation scenario, we still observe that the number of bugs significantly decreases from 3,464 to 270. In particular, for 9 of the 40 skills, we can bring the number of bugs to zero.

It can be observed that for the skill *Internet Archive* the number of bugs observed in the two-fold cross-validation scenario is lower (1 bug) than the one achieved by adding all the paraphrases to the VIM (8 bugs). This suggests that incorporating all paraphrases is not always the best solution, as it may introduce ambiguity in the intent definitions of the skill.

Answer to $\mathbf{RQ_3}$. We confirm the results achieved by Guglielmi et al.: Including generated paraphrases to the VIM allows to significantly reduce the number of bugs. However, some paraphrases might introduce ambiguities.

5.5 RQ_4: Categories of Bugs

Since ChatGPT+ achieves the best results in terms of bug-revealing paraphrases (RQ_2), we use this approach to analyze the types of bugs identified. We defined the four categories using bottom-up coding based on all cases that revealed bugs. First, we performed open coding on the failures obtained in the bug analysis (RQ_2), examining: (i) the outputs of the

Table 6 Results of RQ_3 in both the considered scenarios. The columns report (i) the number of bugs in the sample, #B○, (ii) the percentage of bugs in the sample, %B, and the estimated absolute number of failing tests on the whole population, #B⋆

	2-fold cross validation			Complete evaluation		
Skill	#B○	%B	#B⋆	#B○	%B	#B⋆
Berry bash	10	3.3%	94 ± 94	0	0.0%	57 ± 57
Boys basketball	0	0.0%	43 ± 43	0	0.0%	43 ± 43
Button trivia	2	0.9%	62 ± 62	0	0.0%	53 ± 53
Charity roster	0	0.0%	37 ± 37	0	0.0%	37 ± 37
City guide	10	6.6%	85 ± 65	0	0.0%	32 ± 32
College finder	27	10.3%	1,406 ± 685	0	0.0%	342 ± 342
Cyclist assist.	2	0.9%	32 ± 32	0	0.0%	27 ± 27
Feed reader	9	3.5%	127 ± 127	0	0.0%	74 ± 74
Forget me not	0	0.0%	12 ± 12	0	0.0%	12 ± 12
Girls volleyball	1	0.3%	75 ± 75	0	0.0%	71 ± 71
Greeting sender	9	3.6%	116 ± 116	0	0.0%	67 ± 67
Happy birthday	0	0.0%	15 ± 15	0	0.0%	15 ± 15
History facts	0	0.0%	9 ± 9	0	0.0%	9 ± 9
Humour me	9	4.3%	45 ± 45	0	0.0%	24 ± 24
Internet archive	1	0.4%	41 ± 41	8	3.4%	63 ± 63
llama facts	0	0.0%	18 ± 18	0	0.0%	18 ± 18
Math facts	4	1.9%	22 ± 22	0	0.0%	16 ± 16
Memory	2	1.3%	10 ± 10	0	0.0%	8 ± 8
Multistr. audio	1	0.6%	9 ± 9	0	0.0%	8 ± 8
My barkeeper	36	10.9%	2,309 ± 1,055	0	0.0%	527 ± 527
Name the show	9	3.6%	71 ± 71	0	0.0%	41 ± 41
Next prayer	0	0.0%	4 ± 4	0	0.0%	4 ± 4
Notification	10	4.9%	64 ± 64	0	0.0%	33 ± 33
Particle cloud	2	0.7%	98 ± 98	0	0.0%	86 ± 86
Piano player	1	0.4%	32 ± 32	0	0.0%	30 ± 30
Pocket	0	0.0%	33 ± 33	0	0.0%	33 ± 33
Premium fact	10	5.4%	66 ± 62	0	0.0%	31 ± 31
Premium h.w.	22	9.2%	293 ± 158	0	0.0%	79 ± 79
Quiz game	1	0.7%	7 ± 7	0	0.0%	6 ± 6
Radio	12	4.1%	208 ± 208	0	0.0%	114 ± 114
Salesforce	1	0.6%	36 ± 36	0	0.0%	32 ± 32
Scheduling	11	4.5%	63 ± 63	0	0.0%	33 ± 33
Store Amazon p.	12	4.0%	165 ± 165	0	0.0%	92 ± 92
Tasty beverage	5	2.0%	179 ± 179	0	0.0%	128 ± 128
Timers	5	2.0%	50 ± 50	0	0.0%	36 ± 36
Trading teacher	6	2.4%	81 ± 81	0	0.0%	55 ± 55
Video app	8	4.4%	39 ± 39	0	0.0%	21 ± 21
Voice developer	29	10.2%	774 ± 379	29	10.2%	774 ± 379
Weather wear	3	0.9%	220 ± 220	0	0.0%	186 ± 186
Week calendar	0	0.0%	17 ± 17	0	0.0%	17 ± 17
Overall	270	2.9%	7,067 ± 4,538	37	0.4%	3,334 ± 2,940

Alexa developer console (intended intent vs. activated intent), (ii) each seed-paraphrase pair, and (iii) the set/structure of the skill's intents (other intents that could plausibly trigger the verbal expression). We manually analyzed 3,464 bugs observed before adding the paraphrases as described in RQ_3 plus additional 270 bugs we still found after adding the paraphrases. In total, 7,529 manual inspections were needed (3,735 $\times$ 2 evaluators + 59 further checks when conflicts arose). Most of the conflicts were due to borderline situations in which the bugs were interpreted differently by the evaluators. In details, the conflicts analyzed often involved cases where bugs had undefined behavior or were ambiguous (i.e., had similar behavior to the activated intent). We found four main categories of bugs, presented in Table 7 and discussed in the following.

Ambiguous Intent represent cases where the system fails to determine the correct intent due to multiple interpretations of the input. Overall, the number of ambiguous intent-related bugs is low (70), with only a few skills showing higher values. For example, *College Finder* has the highest number of ambiguous intent-related bugs (12), followed by *Premium Fact* (7) and *Voice Developer* (5). Some skills such as *Cyclist Assistant* and *Girls Volleyball* do not have any ambiguous intent-related bugs. This suggests that ambiguity may occur in cases where skills have overlapping intents or insufficient differentiation between some commands. After the improvement (i.e., the addition of paraphrases), the frequency of bugs in this category remains relatively low in most skills. However, a few skills show an increase in ambiguous bugs, particularly *College Finder* (13 ambiguous cases, up from 12) and *Premium Fact* (6 cases newly emerging). This indicates that paraphrases can introduce overlapping linguistic patterns that make intent recognition harder. Conversely, many skills that previously exhibited ambiguous cases (e.g., *Cyclist Assistant, Girls Volleyball*) show no ambiguous errors in this evaluation, suggesting that paraphrase expansion improves coverage rather than causing confusion in most cases.

Unexplained Intent refer to errors where no clear reason for failure is identified. The total number of unexplained bugs is 654, with some skills showing a high numbers of such issues. Specifically, *Premium Hello Word* (102) and *Video App* (117) stand out as the most problematic ones as for this category. This suggests that these abilities may suffer from design failures or incomplete handling of some user input. While after the improvement the total number of unexplained bugs dropped from 654 to 172, some skills still present these kinds of bugs. For example, *My Barkeeper, Voice Developer, and Premium Hello Word* continue to show high numbers of unexplained bugs, reinforcing the observation that some skills are intrinsically more prone to such unexpected behaviors, regardless of the addition of paraphrases. On a positive note, several skills that previously had a high number of unexplained errors (e.g., *Greeting Sender, Store Amazon Pay, Tasty Beverage*) now have significantly fewer problems, demonstrating that paraphrases can clarify system behavior for some skills, but not all.

Fallback Intent with 2,427 bugs represents the most significant bug category. The most affected skills are *Boys Basketball* (199), *Charity Roster* (133), and *llama Facts* (165), indicating that these skills often fail to recognize user input and trigger fallback responses expected by default in skill management. Such bugs have decreased dramatically after the improvement, dropping from 2,427 to just 52. This suggests that paraphrase introduction

Table 7 Results of RQ_4 both before and after the paraphrases additions as described in RQ_3. The bold values indicate the most frequent type of bug for each category

Skill	Ambiguous		Unexplained		Fallback		No-activation	
	Before	After	Before	After	Before	After	Before	After
Berry bash	6	0	**75**	**10**	0	0	0	0
Boys basketball	0	0	0	0	**199**	0	0	0
Button trivia	0	0	**35**	2	0	0	0	0
Charity roster	0	0	0	0	**133**	0	0	0
City guide	0	0	1	4	**83**	6	0	0
College finder	12	**13**	12	8	**104**	6	1	0
Cyclist assistant	0	0	0	0	0	0	127	2
Feed reader	5	4	11	**5**	0	0	0	0
Forget me not	0	0	6	0	0	0	0	0
Girls volleyball	0	0	0	0	169	1	0	0
Greeting sender	1	0	**88**	**8**	0	0	0	0
Happy birthday	0	0	0	0	**84**	0	0	0
History facts	0	0	0	0	**86**	0	0	0
Humour me	0	0	**42**	**9**	0	0	0	0
Internet archive	0	0	7	0	**98**	1	0	0
llama facts	0	0	0	0	**165**	0	0	0
Math facts	0	0	0	0	0	0	**12**	4
Memory	0	0	0	0	**82**	2	0	0
Multistream audio	0	0	0	0	**36**	1	0	0
My barkeeper	0	0	51	**31**	**107**	5	0	0
Name the show	1	1	1	2	**89**	**6**	0	0
Next prayer	0	0	0	0	**4**	0	0	0
Notification	0	0	6	**6**	**123**	4	0	0
Particle cloud	**14**	2	10	0	0	0	0	0
Piano player	**11**	1	3	0	0	0	0	0
Pocket	0	0	**1**	0	0	0	0	0
Premium fact	7	**6**	0	0	**130**	4	0	0
Premium hello word	0	0	102	**22**	0	0	0	0
Quiz game	0	0	**4**	1	0	0	0	0
Radio	1	2	3	**6**	**236**	4	0	0
Salesforce	0	0	14	**1**	0	0	**62**	0
Scheduling	4	4	0	1	**89**	**6**	0	0
Store Amazon pay	0	0	10	**10**	**133**	1	0	1
Tasty beverage	0	0	1	**4**	0	0	**79**	1
Timers	0	0	17	**5**	0	0	**22**	0
Trading teacher	3	0	5	1	**133**	**5**	0	0
Video app	0	0	**117**	**8**	0	0	0	0
Voice developer	5	3	**24**	**26**	0	0	0	0
Weather wear	0	0	5	**2**	164	0	0	0
Week calendar	0	0	**3**	0	0	0	0	0
Overall	70	32	654	**172**	**2,427**	52	311	8

plays a crucial role in reducing fallback errors by increasing input recognition accuracy. For some of the skills most affected by fallback intent-related bugs, introducing the generated paraphrases allowed us to eliminate such a category of bugs. This happened, for example, for *Boys Basketball*.

No-activation Intent represents cases where the system does not activate any response. A total of 311 no-activation intent were recorded, with *Bicyclist Assistant* (127) being the most affected. This suggests that some skills do not handle the activation of default responses via Fallback Intents. Similarly to fallback intent-related bugs, no-activation intent-related bugs are almost always eliminated after improvement, dropping from 311 cases to just 8. This confirms that introducing paraphrases significantly enhances intent coverage, reducing instances where user input is completely unrecognized.

We also tried to understand how introducing paraphrases in the VIM impacts the categories of bugs. We found that, out of the 270 inspected bugs left after the VIM improvement, 37 that were previously classified as *fallback intent* became *unexplained*. This has two side effects. First, the perceived quality of the VUI degrades. If the skill informs the user that it did not understand a command, the user can change the phrasing to activate the correct intent; on the other hand, the activation of the wrong intent can make the VUI much less usable (e.g., the user needs to go back to a previous state before retrying). Second, from the point of view of the developers, the fix of the issue becomes harder. We discuss the reasons why this might happen in Section 5.8.

Also, some utterances that previously activated the correct intent now fail. Let us consider, for example, the utterance *"Can you tell me about the premium subscription?"* from the skill *Premium Facts*. Before the improvement, this utterance successfully activated the intent *(TellMeMoreAboutPremiumSubscription)*. However, after the improvement, the same utterance triggers the wrong intent *(BuyPremiumSubscriptionIntent)*. Basically, the sentence which is supposed to retrieve information, now results in starting an action to purchase a subscription. This highlights that while paraphrases expand the recognition coverage, they may introduce problems, making it more difficult to distinguish between similar intents.

Answer to $\mathbf{RQ_4}$. In most of the cases, bugs are due to the fact that the sentence activates no intent or the fallback intent. The introduction of paraphrases in VIM leads to significant improvements such categories. However, introducing paraphrases does not always result in better behavior.

5.6 RQ_5: Ablation Study

Table 8 reports the results of the ablation study comparing the variants of ChatGPT+ related to the outcomes of (i) imperative equivalence between the seed utterance and the generated paraphrases (RQ_1), and (ii) bug-revealing paraphrases (RQ_2).

Although ChatGPT+ obtains a higher percentage of equivalent paraphrases in only two skills than the other variants, we can see that ChatGPT+ obtains for all skills the highest number of imperatively equivalent paraphrases.

Overall, enhancing the model (moving from GPT-3.5 to GPT-4) helps, but its not the dominant factor in the improvement. Indeed ChatGPT+ generates between 18,596 and 21,416 imperative equivalent paraphrases, much more than ChatGPT+ $_{\text{-I}}$ (between 485 and 643) and ChatGPT+ $_{\text{-PI}}$ (between 1,137 and 975) and ChatGPT+ $_{\text{-MPI}}$ (between 758 and 904).

ChatGPT+ achieves the best results in terms of percentage of bugs found (overall, 42.30%) as compared to the other variations. ChatGPT+ generates the highest absolute number of bug-revealing paraphrases compared to the other approaches for all the skills. Overall, ChatGPT+ achieves better results in terms of absolute number of bug-revealing paraphrases generated. Indeed ChatGPT+ generates between 9,934 and 12,752 bug-revealing paraphrases, much more than ChatGPT+ $_{\text{-I}}$ (between 485 and 643), ChatGPT+ $_{\text{-PI}}$ (between 332 and 480) and ChatGPT+ $_{\text{-MPI}}$ (between 371 and 511).

Answer to RQ$_5$. ChatGPT+ generates a higher absolute number of valid paraphrases as compared to its less sophisticated variants. In addition, ChatGPT+ achieves significantly better results than the other basic approaches in terms of percentage and absolute number of bug revealing paraphrases.

5.7 RQ$_6$: Open LLM Evaluation

Table 9 reports the results of the imperative equivalence between the seed utterance and the generated paraphrases. Overall, ChatGPT+ achieves a higher equivalence rate than GPT-oss-20B (70.95% and 69.33%, respectively). On the other hand, considering the absolute number of equivalent paraphrases estimated across the entire population, GPT-oss-20B produces more imperatively equivalent paraphrases overall (24,314 ± 1,753) than ChatGPT+ (20,006 ± 1,410). GPT-oss-20B achieves the highest absolute counts for most skills, with ChatGPT+ higher for the College finder skill and a statistical tie on Voice developer.

Table 10 shows the percentage and the absolute number of paraphrases that reveal bugs among paraphrases that are imperatively equivalent to the seeds. ChatGPT+ achieves a higher overall bug detection rate than GPT-oss-20B. In terms of skills, ChatGPT+ scores highest in 8 out of 10 skills, while GPT-oss-20B is superior two skills. For skill, GPT-oss-20B achieves higher absolute counts in six skills, while ChatGPT+ is superior in only three skills. This model reflects the fact that GPT-oss-20B often generates more paraphrases, increasing absolute performance, while ChatGPT+ offers greater accuracy (a higher percentage of its equivalents reveal bugs).

Table 11 the results in terms of bug-fixing capabilities under the two scenarios: (i) Complete Evaluation, and (ii) 2-Fold Cross-Validation.

Complete Evaluation When we add all the paraphrases generated by GPT-oss-20B to the VIMs and use the same paraphrases as tests, the number of residual bugs drops to 26 overall (1.24% in the evaluated sample; 1,500 ± 1,089 estimated failing tests over the whole population). Notably, 9 out of 10 skills reach zero residual bugs. The only remaining issues are concentrated in Voice developer (26 bugs), suggesting VUI design ambiguities specific to that skill.

Table 8 Results of RQ_5 comparing the different prompt templates and gpt model evaluated in terms of imperatively equivalent of generated paraphrases (RQ_1) and paraphrase capability of finding bugs (RQ_2). ○ indicates that the column refers to the evaluated sample; ⋆ indicates that it refers to the estimated value on the whole population.We report in boldface the significantly best results, while we mark with a * the cases in which the best result is shared by two approaches

	RQ_1: Paraphrase equivalence							
	ChatGPT+				ChatGPT+$_{-I}$			
Skill	#generated○	#equiv○	%equiv	#equiv⋆	#generated○	#equiv○	%equiv	#equiv⋆
City guide	294	151	51.36%	**663±65**	59	32	54.24%	37±3
College finder	374	263	70.32%	**9628±685**	247	182	**73.68%**	511±35
Forget me not	212	188	88.68%	**417±24**	29	27	**93.10%**	29±2
History Facts	191	169	88.48%	**334±19**	39	36	**92.31%**	40±2
Next Prayer	103	96	**93.20%**	**130±7**	29	27	93.10%	29±2
Notification	297	205	**69.02%**	**900±65**	63	43	68.25%	51±4
Premium fact	292	187	64.04%	**789±62**	98	82	83.67%	110±7
Salesforce	296	181	61.15%	**783±64**	69	46	**66.67%**	56±6
Video app	261	181	69.35%	**580±42**	70	54	**77.14%**	66±4
Voice developer	365	284	77.81%	**5894±379**	186	146	**78.49%**	282±18
Overall	2,685	1,905	70.95%	**20,006±1,410**	889	675	**75.93%**	1,217±80
	RQ_2: Paraphrase capability of finding bugs							
	ChatGPT+				ChatGPT+$_{-I}$			
Skill	#equiv○	#bugs○	%bugs	#bugs⋆	#equiv○	#bugs○	%bugs	#bugs⋆
City guide	151	84	55.63%	**718±65**	32	18	56.25%	39±3
College finder	263	131	**49.81%**	**6819±685**	182	69	37.91%	263±35
Forget me not	188	12	**6.38%**	**30±24**	27	1	3.70%	1±1
History Facts	167	86	51.50%	**195±19**	36	20	**55.56%**	24±2
Next Prayer	96	4	4.17%	**23±3**	27	0	0.00%	1±1
Notification	205	129	62.93%	**821±65**	43	35	**81.40%**	61±4
Premium fact	187	137	**73.26%**	**903±62**	82	50	60.98%	80±7
Salesforce	181	76	**41.99%**	**537±64**	46	19	41.30%	35±4
Video app	181	117	**64.64%**	**541±42**	54	19	35.19%	30±4
Voice developer	284	29	10.21%	**774±379**	146	12	8.22%	30±18
Overall	1,903	805	**42.30%**	**11,343±1,409**	675	243	36.00%	564±79

Table 8 (continued)

	RQ_1: Paraphrase equivalence							
	ChatGPT+$_{\text{-PI}}$				ChatGPT+$_{\text{-MPI}}$			
Skill	#generated○	#equiv○	%equiv	#equiv⋆	#generated○	#equiv○	%equiv	#equiv⋆
City guide	51	33	**63.71%**	38±3	57	26	45.61%	31±3
College finder	235	167	71.06%	430±30	230	99	43.04%	248±29
Forget me not	25	19	76.00%	21±1	36	33	91.67%	37±2
History Facts	36	32	88.89%	36±2	31	27	87.10%	30±2
Next Prayer	28	26	92.86%	28±2	28	21	75.00%	23±2
Notification	62	36	58.06%	43±4	66	40	60.61%	48±4
Premium fact	92	78	**84.78%**	103±6	92	51	55.43%	67±6
Salesforce	69	41	59.42%	50±4	64	40	62.50%	49±4
Video app	77	57	74.03%	71±5	64	29	45.31%	35±4
Voice developer	187	115	61.50%	224±18	182	121	66.48%	230±17
Overall	863	604	70.07%	1,050±75	850	487	57.29%	831±73
	RQ_2: Paraphrase capability of finding bugs							
	ChatGPT+$_{\text{-PI}}$				ChatGPT+$_{\text{-MPI}}$			
Skill	#equiv○	#bugs○	%bugs	#bugs⋆	#equiv○	#bugs○	%bugs	#bugs⋆
City guide	33	16	48.48%	28±3	26	18	**69.23%**	24±2
College finder	167	53	31.74%	**192±30**	99	38	**38.38%**	106±14
Forget me not	19	0	0.00%	1±1	33	0	0.00%	1±1
History Facts	32	14	43.75%	18±2	27	11	40.74%	13±2
Next Prayer	26	1	3.85%	1±1	21	5	**23.81%**	6±1
Notification	36	23	63.89%	47±4	40	26	65.00%	34±3
Premium fact	78	31	39.74%	48±6	51	24	47.06%	34±4
Salesforce	41	11	26.83%	23±4	40	3	7.50%	5±2
Video app	57	6	10.53%	10±5	29	0	0.00%	1±1
Voice developer	115	12	**10.43%**	38±18	121	12	9.92%	25±12
Overall	604	167	27.65%	406±74	487	137	28.13%	441±70

Table 9 Results of RQ_6 (imperative equivalence with GPT-oss-20B). ○ indicates that the column refers to the evaluated sample, while ⋆ indicates that it refers to the estimated value of the whole population (95% confidence level). We report in boldface the significantly best results, while we mark with a * the cases in which the best result is shared by both of the approaches

	Paraphrase equivalence							
	ChatGPT+				GPT-oss-20B			
Skill	#generated○	#equiv○	%equiv	#equiv⋆	#generated○	#equiv○	%equiv	#equiv⋆
City guide	294	151	51.36%	663 ± 65	322	213	**66.15%**	**1,441 ± 109**
College finder	374	263	**70.32%**	**9,628 ± 685**	373	232	62.20%	8,189 ± 659
Forget me not	212	188	**88.68%**	417 ± 24	245	188	76.73%	**519 ± 34**
History Facts	191	169	**88.48%**	334 ± 19	234	191	81.62%	**529 ± 35**
Next Prayer	103	96	**93.20%**	130 ± 7	192	173	90.10%	**391 ± 22**
Notification	297	205	**69.02%**	900 ± 65	334	159	47.60%	**1,260 ± 132**
Premium fact	292	187	64.04%	789 ± 62	310	252	**81.29%**	**1,450 ± 89**
Salesforce	296	181	61.15%	783 ± 64	315	201	**63.81%**	**1,155 ± 91**
Video app	261	181	**69.35%**	580 ± 42	343	232	67.64%	**2,194 ± 162**
Voice developer	365	284	**77.81%**	***5,894± 379**	368	264	71.74%	***6,082 ± 424**
Overall	2,685	1,905	**70.95%**	20,006 ± 1,410	3,036	2,105	69.33%	**24,314 ± 1,753**

Table 10 Results of RQ_6 (bug-finding capabilities with GPT-oss-20B). ○ indicates that the column refers to the evaluated sample, while ⋆ indicates that it refers to the estimated value on the whole population (95% confidence level). We report in boldface the significantly best results, while we mark with a * the cases in which the best result is shared by two approaches

	Bug-finding capabilities							
	ChatGPT+				GPT-oss-20B			
Skill	#equiv○	#bugs○	%bugs	#bugs⋆	#equiv○	#bugs○	%bugs	#bugs⋆
City guide	151	84	**55.63%**	718 ± 65	213	118	55.40%	**1,207 ± 109**
College finder	263	131	**49.81%**	**6,819 ± 685**	232	74	31.90%	4,200 ± 658
Forget me not	188	12	**6.38%**	**30 ± 24**	188	0	0.00%	17 ± 17
History Facts	169	86	**51.50%**	195 ± 19	191	95	49.74%	**322 ± 32**
Next Prayer	96	4	4.17%	6 ± 6	173	13	**7.51%**	**33 ± 22**
Notification	205	129	**62.93%**	821 ± 65	159	82	51.57%	**1,365 ± 132**
Premium fact	187	137	**73.26%**	***903 ± 62**	252	132	52.38%	***934 ± 89**
Salesforce	181	76	**41.99%**	**537 ± 64**	201	0	0.00%	45 ± 45
Video app	181	117	**64.64%**	541 ± 42	232	143	61.64%	**2,000 ± 162**
Voice developer	284	29	10.21%	774± 379	264	29	**10.98%**	**931 ± 4,244**
Overall	1,905	805	**42.30%**	***11,343 ± 1,409**	2,105	686	32.59%	***11,054 ± 1,691**

Table 11 Results of RQ_6 (bug-fixing capabilities). The columns report (i) the number of bugs in the sample, #B○, (ii) the percentage of bugs in the sample, %B, and the estimated absolute number of failing tests on the whole population, #B⋆

	Bug-finding capabilities											
	ChatGPT+						GPT-oss-20B					
	2-fold cross validation			Complete evaluation			2-fold cross validation			Complete evaluation		
Skill	#B○	%B	#B⋆	#B○	%B	#B⋆	#B○	%B	#B⋆	#B○	%B	#B⋆
City guide	10	6.60%	85 ± 65	0	0.00%	32 ± 32	1	0.47%	60 ± 60	0	0.00%	54 ± 54
College finder	27	10.30%	1,406 ± 685	0	0.00%	342 ± 342	21	9.05%	1,192 ± 658	0	0.00%	329 ± 329
Forget me not	0	0.00%	12 ± 12	0	0.00%	12 ± 12	0	0.00%	17 ± 17	0	0.00%	17 ± 17
History facts	0	0.00%	9 ± 9	0	0.00%	9 ± 9	2	1.05%	20 ± 20	0	0.00%	16 ± 16
Next prayer	0	0.00%	4 ± 4	0	0.00%	4 ± 4	0	0.00%	11 ± 11	0	0.00%	11 ± 11
Notification	10	4.90%	64 ± 64	0	0.00%	33 ± 33	9	5.66%	150 ± 132	0	0.00%	66 ± 66
Premium fact	10	5.40%	66 ± 62	0	0.00%	31 ± 31	2	0.79%	52 ± 52	0	0.00%	45 ± 45
Salesforce	1	0.60%	36 ± 36	0	0.00%	32 ± 32	1	0.50%	85 ± 65	0	0.00%	45 ± 45
Video app	8	4.40%	39 ± 39	0	0.00%	21 ± 21	0	0.00%	81 ± 81	0	0.00%	81 ± 81
Voice developer	29	10.20%	774 ± 379	29	10.20%	774 ± 379	6	2.27%	308 ± 308	26	9.85%	835 ± 424
Overall	270	2.90%	7,067 ± 4,538	37	0.40%	3,334 ± 2,940	42	2.00%	1,939 ± 1,388	26	1.24%	1,500 ± 1,089

2-Fold Cross-Validation In this setting where we augment the VIM with one fold and test with the other, GPT-oss-20B still yields a substantial reduction, with 42 residual bugs overall (2.00% in the sample; 1,939 ± 1,388 estimated over the whole population). With all paraphrases included, only one skill retains failures (Voice developer), likely due to intrinsic VUI design ambiguities; under cross-validation, the residual bug count remains low across skills, confirming the overall bug-reduction effect.

Answer to RQ6. Using a smaller, open-weights model is sufficient to obtain good-enough effectiveness both in terms of paraphrase equivalence (slightly higher) and bug-finding capabilities (slightly lower). Including the paraphrases generated with GPT-oss-20B allows to reduce the number of bugs more markedly than with ChatGPT+.

5.8 Discussion

Our results clearly show that the prompt engineering we performed by introducing few-shot learning and contextual information, and the iterative interaction with the LLM for paraphrase generation up to saturation provide significant benefits over the basic prompt preliminarily adopted in previous work (Guglielmi et al. 2023).

In our ablation study (Section 3), we observed that the two main improvements we brought (i.e., prompt tuning and iterative generation) contribute in improving the number of generated bug-revealing paraphrases. Interestingly, the most economically expensive step (i.e., using GPT-4 instead of GPT-3.5) does not provide substantial benefits when looking at the overall results, but only for some skills. Thus, we recommend developers to carefully evaluate whether it is worth the additional cost to use a more expensive model by also checking the results obtained on a sample of sentences before using it on all of them. We conjecture that this happens because generating paraphrases mostly requires language knowledge, that even more basic LLMs have. The advances in the more recent models (e.g., o1 or o3-mini (OpenAI 2024a, b)) mostly regard "reasoning" capabilities, which are not required for such a purely language-related task.

Lesson Learned 1. Simpler and smaller LLMs (e.g., GPT-3.5 or GPT-oss-20B instead of GPT-4) might be sufficient for VUI test case generation when combined with effective prompts and iterations. Adopting more advanced models might be unnecessary.

Some *unexplained* bugs we found while answering RQ_4 might be very hard to fix, and introducing paraphrases for them might not help. For example, in the skill *Berry Bash*, there is an intent called *QuizIntent* which should be activated when the user wants to start the game. However, the generated utterance "Let's take a quiz about berries" results in the activation of the wrong intent, i.e., *InformationIntent*. We manually tried to fix this issue. To pinpoint the culprit seed utterance, we tried to remove, one at a time, the utterances in the latter intent and found that one of them, alone, caused the bug, i.e., "teach me about berries." We further proceeded to remove a word at a time from such an utterances to find out if any of them was responsible. We found that simply removing the word "about" fixed the issue. Replacing such a word with "on" fixes bug disappear. This probably means that the AI integrated in the voice-based virtual assistants gives a higher weight to the word "about." A

possible explanation is that many skills contain specialized intents that explain the purpose of the skill (which are often activated by such a word). Note that manually finding the root-cause of this bug required about 1 hour because, at each step, it is necessary to re-build the skill on the ADC and re-run the tests. Devising approaches to automatically perform such a task could be of great help to developers.

Lesson Learned 2. Fixing some VUI bugs can be very tricky, above all when the skill confounds two intents for no apparent reason.

5.9 The Role of Role-Playing in the Prompt

We replaced the default assistant role ("You are a helpful assistant") with a task-specific role prompt. Specifically, we set the following role: "You are an Alexa Natural Language Understanding dataset engineer specialized in high-precision intent paraphrasing for activation sentences." We assess the extent to which paraphrases generated with this specific role are imperatively equivalent to their seed utterances. We run this comparison on 10 randomly selected skills from the 40 used in our main study. We randomly sample, per skill, a number of generated paraphrases that guarantees a 5% margin of error at 95% confidence, for a total of 2,650 paraphrases. Three of the authors were involved in the evaluation. Each generated paraphrase was assigned and independently evaluated by two authors (evaluators). Each evaluator checked whether the generated paraphrase and the respective seed utterance were imperatively equivalent. If the two original evaluators disagreed, the third one independently evaluated the paraphrase and the three authors discussed the case, aiming at reaching consensus. The two evaluators originally assigned to each paraphrase had disagreements in 570 cases (i.e., 21.50%). After discussion, consensus was reached for all the paraphrases. In total, 5,870 manual inspections were needed ($2,650 \times 2$ evaluators + 570 further checks). We compare the results about the original ChatGPT+ (neutral system prompt) with the ChatGPT+ w_{role} (domain role in the system prompt) in terms of: (i) percentage of imperatively equivalent paraphrases, and (ii) the absolute number of accepted paraphrases.

Table 12 compare the results of the imperative equivalent paraphrases generated by ChatGPT+ and ChatGPT+ w_{role}. In terms of percentage of imperatively equivalent paraphrases, the results show that ChatGPT+ achieves the best results in five skills (e.g., City guide, College finder, Forget me not, History Facts, Notification), whereas ChatGPT+ w_{role} is higher in the other five. The overall percentage is almost identical (70.95% ChatGPT+ and 70.26% ChatGPT+ w_{role}). In terms of absolute number of equivalent paraphrases, ChatGPT+ performs better in six skills and overall (20,006 ± 1,410 vs. 15,839 ± 1,127), indicating that role prompts (ChatGPT+ w_{role}) sometimes reduce effective coverage, typically by increasing verbosity/format violations or overly restricting style, so fewer candidates are validated as imperatively equivalent.

Lesson Learned 3. Role-playing has no substantial impact in this task and constraining the LLM to a role can even decrease coverage. The neutral system prompt in VUI paraphrase generation context allows for better results in terms of imperatively equivalent generated paraphrases.

Table 12 Results for role-playing on the generation of *imperatively equivalent* paraphrases. ○ indicates that the column refers to the evaluated sample; ⋆ indicates that it refers to the estimated value on the whole population. We report in boldface the significantly best results, while we mark with a * the cases in which the best result is shared by two approaches

	RQ_1: Paraphrase equivalence							
	ChatGPT+				ChatGPT+ w_{role}			
Skill	#generated○	#equiv○	%equiv	#equiv⋆	#generated○	#equiv○	%equiv	#equiv⋆
City guide	294	151	**51.36%**	***663 ± 65**	299	125	41.81%	***563 ± 67**
College finder	374	263	**70.32%**	**9,628 ± 685**	370	237	64.05%	7,187 ± 561
Forget me not	212	188	**88.68%**	**417 ± 24**	176	155	88.07%	285 ± 16
History Facts	191	169	**88.48%**	334 ± 19	208	180	86.54%	**395 ± 23**
Next Prayer	103	96	93.20%	130 ± 7	113	107	**94.69%**	**155 ± 8**
Notification	297	205	**69.02%**	***900 ± 65**	288	194	67.36%	***778 ± 58**
Premium fact	292	187	64.04%	***789 ± 62**	289	205	**70.93%**	***833 ± 59**
Salesforce	296	181	61.15%	***783 ± 64**	287	196	**68.29%**	***777 ± 57**
Video app	261	181	69.35%	***580 ± 42**	265	185	**69.81%**	***591 ± 42**
Voice developer	365	284	77.81%	**5,894± 379**	355	278	**78.31%**	3,693 ± 236
Overall	2,685	1,905	**70.95%**	**20,006 ± 1,410**	2,650	1,862	70.26%	15,839 ± 1,127

5.10 The Role of Activation Intent Examples in the Prompt

The few-shot prompt we designed only provides the LLM with *positive* examples of paraphrases and does not give context about the intent related to those paraphrases. It is possible that (i) including the intent related to the original utterance and (ii) adding *negative* examples (i.e., that do not trigger the same intent) help the model understand how it should not paraphrase the utterances. Therefore, we run an additional experiment in order to evaluate the extent to which paraphrases generated with this updated prompt ChatGPT+ $FewShot_{intent}$ are imperatively equivalent to their seed utterances. We run this comparison on the same 10 randomly selected skills from the 40 used in our main study. For each skill, we sampled enough paraphrases to ensure a 5% margin of error at 95% confidence, resulting in a total of 2,984 samples. The evaluation followed the same procedure described in R3.C04.

The two evaluators originally assigned to each paraphrase had disagreements in 466 cases (i.e., 15.62%). After discussion, consensus was reached for all the paraphrases. In total, 6,434 manual inspections were needed ($2,984 \times 2$ evaluators + 466 further checks). We compare the results about the original ChatGPT+ with the ChatGPT+ $FewShot_{intent}$ in terms of: (i) percentage of imperatively equivalent paraphrases, and (ii) the absolute number of accepted paraphrases.

Table 13 shows the results of this comparison. In terms of the percentage of imperatively equivalent paraphrases, ChatGPT+ performs best in 6 out of 10 skills and reaches a higher overall rate than ChatGPT+ $FewShot_{intent}$ (70.95% compared to 66.86%). However, when considering the absolute number of imperatively equivalent paraphrases, ChatGPT+ $FewShot_{intent}$ produces more valid outputs, with about 25,000 paraphrases compared to about 20,000 for ChatGPT+. The only exceptions are two skills, where the totals are very close and practically indistinguishable.

Lesson Learned 4. Adding few-shots information related to the intent outcome increases actual coverage. This allows to generate a greater number of imperatively equivalent paraphrases.

5.11 Fixing Unexplained Bugs

Given the importance and difficulty of fixing unexplained bugs, we manually tried to fix all the unexplained bugs in two skills: Particle Cloud and Piano Player. We report such case studies below.

Particle Cloud We identified a consistent pattern in intent misclassification: Whenever an utterance contained the word "which", the system incorrectly activated *NumberOfDevicesIntent* instead of the expected *ListDevicesIntent*. An example of failure due to "which" is *"Which devices are connected right now?"* where the expected intent is *ListDevicesIntent* and the activated intent is *NumberOfDevicesIntent*. As we can see in Table 14, regardless of the words "extra", when the word "which" appears the intent *NumberOfDevices* is triggered.

To correct this problem, we explicitly added a sentence in the skill pattern containing the word "which" in *ListDevicesIntent* to teach the AI behind the intent activation that such

Table 13 Results for activation intent based few shot for the generation of *imperatively equivalent* paraphrases. ○ indicates that the column refers to the evaluated sample; ⋆ indicates that it refers to the estimated value on the whole population. We report in boldface the significantly best results, while we mark with a * the cases in which the best result is shared by two approaches

	RQ_1: Paraphrase equivalence							
	ChatGPT+				ChatGPT+ $FewShot_{intent}$			
Skill	#generated○	#equiv○	%equiv	#equiv⋆	#generated○	#equiv○	%equiv	#equiv⋆
City guide	294	151	**51.36%**	663 ± 65	327	141	43.12%	**1,000 ± 116**
College finder	374	263	**70.32%**	***9,628 ± 685**	376	211	56.12%	***10,411 ± 928**
Forget me not	212	188	88.68%	417 ± 24	246	219	**89.02%**	**610 ± 34**
History Facts	191	169	**88.48%**	334 ± 19	249	212	85.14%	**598 ± 35**
Next Prayer	103	96	93.20%	130 ± 7	174	163	**93.68%**	**297 ± 16**
Notification	297	205	69.02%	900 ± 65	305	236	**77.38%**	**1,148 ± 74**
Premium fact	292	187	64.04%	789 ± 62	305	200	**65.57%**	**961 ± 73**
Salesforce	296	181	**61.15%**	783 ± 64	331	197	59.52%	**1,411 ± 119**
Video app	261	181	**69.35%**	580 ± 42	305	147	48.20%	**729 ± 76**
Voice developer	365	284	**77.81%**	***5,894± 379**	366	269	73.50%	***6,172 ± 420**
Overall	2,685	1,905	**70.95%**	20,006 ± 1,410	2,984	1,995	66.86%	**25,274 ± 1,890**

Table 14 Comparison of expected and activated intents for modified paraphrases

Modified paraphrase	Expected intent	Activated intent
~~can you~~ tell me what devices are connected right now	ListDevicesIntent	NumberOfDevicesIntent
~~can you~~ tell me **which** devices are connected right now	ListDevicesIntent	**NumberOfDevicesIntent**
~~tell me~~ what devices are connected right now	ListDevicesIntent	ListDevicesIntent
~~tell me~~ **which** devices are connected right now	ListDevicesIntent	**NumberOfDevicesIntent**
what devices are connected ~~right now~~	ListDevicesIntent	ListDevicesIntent
which devices are connected ~~right now~~	ListDevicesIntent	**NumberOfDevicesIntent**
~~can you~~ tell me what devices are connected right now	ListDevicesIntent	NumberOfDevicesIntent
~~can you~~ tell me **which** devices are connected right now	ListDevicesIntent	**NumberOfDevicesIntent**
~~can you~~ tell me what devices are connected	ListDevicesIntent	ListDevicesIntent
~~can you~~ tell me **which** devices are connected	ListDevicesIntent	**NumberOfDevicesIntent**

a word should not be mapped to *NumberOfDevicesIntent*. This reduced the bugs from 24 to 10. However, we observed that two new failures emerged, in which previously correct sentences started to fail. This suggests that, as seen before, the improvement of the skill may affect the behavior, reinforcing the need for balancing of intents rather than simply increasing coverage.

In addition, we analyzed bugs in which *ListDevicesIntent* was incorrectly activated instead of *NumberOfDevicesIntent*. By restructuring the statement from "connected device count" to "count of connected devices," we further reduced the bugs to only 4. Unfortunately, we could not fix the remaining four bugs.

Piano Player In this skill, a sentence that triggered unexplained bugs is *"Inform me about available song choices"*. The expected intent is *ListSongs*, while the activated intent is *BasicScale*. In this case, the failure was due to the fact that *ListSongs* did not contain enough examples of similar paraphrasing, which led the system to classify the utterance as a request to play a scale. To solve this problem, we added the sentence directly into *ListSongs*. However, this introduced a new classification error, increasing the number of bugs from 14 to 15, this highlights the importance of evaluating paraphrases added in the intent.

We then addressed another problem in which utterances for *ReverseScale* were being incorrectly classified as *BasicScale* due to insufficient paraphrase variety. The addition of three specific sentences reduced bugs from 15 to only 3. In detail, by adding the sentence "instruct me in a new lesson" we went from 15 to 8 bugs. Then, by adding the sentence "start a new lesson," we went from 8 to 3 bugs. In RQ_3 we obtained 0 bugs after including

226 paraphrases in the intents. This indicates that, while paraphrases offer an automated way to fix bugs, developers may in some cases obtain similar results by carefully selecting a few additional sentences to handle.

6 Threats to Validity

Threats to Construct Validity We manually selected a sample of the seed utterances to use for generating paraphrases from the VIMs of the skills we considered for the prompt engineering. Choosing different utterances might have led to different results.

Threats to Internal Validity There is a possible subjectivity introduced during manual analysis for the results of the prompt engineering evaluation, RQ_1 and RQ_4. This threat was mitigated by using a rigorous qualitative analysis process in which at least two independent evaluators have been involved. We computed the agreement between the two main evaluators by using the Cohen's kappa, which resulted to be 0.81 for the prompt engineering and 0.82 for the RQ_1.

Threats to External Validity Our results are based on 40 + 10 Alexa skills for the main empirical study and the prompt engineering step, respectively. Such a sample may not be representative of all Alexa skills. Such a set contains, however, different skills in terms of application domain (Guglielmi et al. 2023). Note that such a dataset only contains Alexa skills developed in JavaScript. We do not expect significant differences for skills developed in other programming languages, as our work focuses on the VIM rather than on the programming logic skills. All the available approaches have been only tested on VIMs containing utterances in English. To this end, it is worth noting that the two LLM-based approaches we tested could be easily adapted to work in other languages by simply translating the prompt, while the state-of-the-art VUI-UPSET is more strongly tied to the English language. Nevertheless, our study, similarly to the previous ones (Guglielmi et al. 2022, 2023), only focuses on skills for English-speaking users. Finally, it is possible that the results obtained are not generalizable to other technologies, such as actions for Google Assistant. Indeed, the results of RQ_2, RQ_3 and RQ_4 depend on the framework's tolerance to variations in the predefined starting statements, which may change.

7 Conclusion

The increasing use of voice interfaces integrated in everyday devices has led to a growing interest in ensuring their correct functionality. In this work, we tested to what extent introducing optimization on the usage of ChatGPT for paraphrase generation (i.e., prompt engineering and iterative interaction) increases their capability of generating VUI tests. This is exactly what we experimented in this paper by defining and tuning a more advanced usage of the LLM for VUI testing. An evaluation run on 40 open-source Alexa skills measured the ability of the experimented techniques to generate semantically equivalent and bug-revealing paraphrases. Our findings show the superiority of an LLM-based approach as compared to the specialized state-of-the-art techniques across all evaluations we run, confirming that

the potential of LLM-based solutions for VUI testing. Our results also suggest that big LLMs are not necessary: On-premise alternatives are good enough when used in the correct way. Also, we studied the extent to which including the generated paraphrases in the speech interaction model (VIM) can help in fixing bugs, and how introducing such paraphrases can help practitioners fix bugs. However, we also observe that introducing paraphrases can only fix some specific bug categories. This finding paves the way for the introduction of automated approaches for fixing two categories of bugs for which adding generated paraphrases to the VIM does not help: ambiguity-related and unexplained bugs.

Author Contributions Emanuela Guglielmi: Developed the research idea, designed the study, conducted experiments, performed statistical analysis, Writing original draft preparation. Angelica Spina: Conducted experiments, participated in the writing. Gabriele Bavota: Developed the research idea, designed the study, participated in the writing. Rocco Oliveto: Developed the research idea, designed the study, participated in the writing. Simone Scalabrino: Developed the research idea, designed the study, participated in the writing.

Funding Open access funding provided by Università degli Studi del Molise within the CRUI-CARE Agreement. This work has been supported by the European Union - NextGenerationEU through the Italian Ministry of University and Research, Projects PRIN 2022 "DevProDev: Profiling Software Developers for Developer-Centered Recommender Systems", grant n. 2022S49T4W, CUP: H53D23003610001. Bavota thank the Swiss National Science Foundation (SNSF) for the funding provided under the project "PARSED" (grant agreement No. 219294).

Data Availability We publicly release the command prompt used to interact with ChatGPT via API, the dataset of the skills used for prompt tuning, the evaluation data of prompt tuning (for all four prompt). We also release the data used answering RQ_0, RQ_1, RQ_2, RQ_3, RQ_4, RQ_5 and RQ_6. We report the results of the additional analysis and the dataset of the skills in our replication package (Guglielmi et al. 2024).

Declarations

Ethical approval Not applicable.

Informed Consent Not applicable.

References

Alexa A (2025) OUse LLMs to improve accuracy in your Interaction Model skill. https://developer.amazon.com/en-US/alexa/alexa-haus/LLM-utterances

Amazon (2018) Amazon official documentation. https://www.amazon.com/alexa-skills/b?ie=UTF8&node=13727921011

Chen J, Zhenhao L, Xing H, Xin X (2024) NLPerturbator: Studying the robustness of code LLMs to natural language variations. ACM Trans Softw Eng Methodol

Denny P, Kumar V, Giacaman N (2023) Conversing with copilot: Exploring prompt engineering for solving CS1 problems using natural language. In: Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1, pp 1136–1142

Egli A (2023) ChatGPT, GPT-4, and other large language models: The next revolution for clinical microbiology? Clinical Infectious Diseases p ciad407

Guglielmi E, Rosa G, Scalabrino S, Bavota G, Oliveto R (2022) Sorry, I Don't understand: Improving voice user interface testing. In: 37th IEEE/ACM International conference on automated software engineering, pp 1–12

Guglielmi E, Rosa G, Scalabrino S, Bavota G, Oliveto R (2023) Help them understand: Testing and improving voice user interfaces. ACM Trans Softw Eng Methodol

Guglielmi E, Spina A, Bavota G, Oliveto R, Scalabrino S (2024) Replication Package. https://figshare.com/s/1d7d555e54c466fd4d2e

Guichard J, Ruane E, Smith R, Bean D, Ventresque A (2019) Assessing the robustness of conversational agents using paraphrases. In: 2019 IEEE International Conference On Artificial Intelligence Testing (AITest), IEEE, pp 55–62

Guo Z, Lin Z, Li P, Chen K (2020) {SkillExplorer}: Understanding the behavior of skills in large scale. In: 29th USENIX Security Symposium (USENIX Security 20), pp 2649–2666

Hegde C, Patil S (2020) Unsupervised paraphrase generation using pre-trained language models. arXiv:2006.05477

Li S, Bu L, Bai G, Guo Z, Chen K, Wei H (2022) VITAS: Guided model-based VUI testing of VPA Apps. In: 37th IEEE/ACM International conference on automated software engineering, pp 1–12

Li S, Bu L, Bai G, Xie F, Chen K, Yue C (2024) Model-enhanced LLM-driven VUI testing of VPA Apps. arXiv:2407.02791

Liao S, Cheng L, Cai H, Guo L, Hu H (2023) SkillScanner: Detecting policy-violating voice applications through static analysis at the development phase. In: Proceedings of the 2023 ACM SIGSAC conference on computer and communications security, pp 2321–2335

Liu V, Chilton LB (2022) Design guidelines for prompt engineering text-to-image generative models. In: Proceedings of the 2022 CHI conference on human factors in computing systems, pp 1–23

Liu Y, Ott M, Goyal N, Du J, Joshi M, Chen D, Levy O, Lewis M, Zettlemoyer L, Stoyanov V (2019) RoBERTa: A robustly optimized BERT pretraining approach. arXiv:1907.11692

Lo LS (2023) The art and science of prompt engineering: a new literacy in the information age. Internet Ref Serv Q 27(4):203–210

Macbeth G, Razumiejczyk E, Ledesma RD (2011) Cliff's delta calculator: A non-parametric effect size program for two groups of observations. Universitas Psychologica 10(2):545–555

Meier D, Wahle JP, Ruas T, Gipp B (2024) Towards human understanding of paraphrase types in ChatGPT. arXiv:2407.02302

OpenAI (2024a) ntroducing OpenAI o1. https://openai.com/o1/

OpenAI (2024b) OpenAI o3-mini. https://openai.com/index/openai-o3-mini/

Reynolds L, McDonell K (2021) Prompt programming for large language models: Beyond the few-shot paradigm. In: Extended abstracts of the 2021 CHI conference on human factors in computing systems, pp 1–7

Roumeliotis KI, Tselikas ND (2023) ChatGPT and Open-AI models: A preliminary review. Future Internet 15(6):192

Segura S, Fraser G, Sanchez AB, Ruiz-Cortés A (2016) A survey on metamorphic testing. IEEE Trans Software Eng 42(9):805–824

Shezan FH, Hu H, Wang G, Tian Y (2020) VerHealth: Vetting medical voice applications through policy enforcement. Proceed ACM on Interact Mobile Wearable Ubiquit Technol 4(4):1–21

Wang S, Li Z, Qian H, Yang C, Wang Z, Shang M, Kumar V, Tan S, Ray B, Bhatia P et al (2022) ReCode: Robustness evaluation of code generation models. arXiv:2212.10264

Wei J, Tay Y, Bommasani R, Raffel C, Zoph B, Borgeaud S, Yogatama D, Bosma M, Zhou D, Metzler D et al (2022) Emergent abilities of large language models. arXiv:2206.07682

Witteveen S, Andrews M (2019) Paraphrasing with large language models. arXiv:1911.09661

Woolson RF (2007) Wilcoxon signed-rank test. Wiley encyclopedia of clinical trials pp 1–3

Xu Z, Wang C, Qiu M, Luo F, Xu R, Huang S, Huang J (2023) Making pre-trained language models end-to-end few-shot learners with contrastive prompt tuning. In: Proceedings of the Sixteenth ACM international conference on web search and data mining, pp 438–446

Yigit G, Amasyali MF (2024) From text to multimodal: a survey of adversarial example generation in question answering systems. Knowl Inf Syst 66(12):7165–7204

Young J, Liao S, Cheng L, Hu H, Deng H (2022) { SkillDetective }: Automated { Policy-Violation } detection of voice assistant applications in the wild. In: 31st USENIX security symposium (USENIX Security 22)

Zhang Y, Xu L, Mendoza A, Yang G, Chinprutthiwong P, Gu G (2019) Life after speech recognition: Fuzzing semantic misinterpretation for voice assistant applications. In: Proc. of the Network and Distributed System Security Symposium (NDSS'19)

Authors and Affiliations

Emanuela Guglielmi[1] · Angelica Spina[1] · Gabriele Bavota[2] · Rocco Oliveto[1] · Simone Scalabrino[1]

✉ Emanuela Guglielmi
emanuela.guglielmi@unimol.it

Angelica Spina
angelica.spina@unimol.it

Gabriele Bavota
gabriele.bavota@usi.ch

Rocco Oliveto
rocco.oliveto@unimol.it

Simone Scalabrino
simone.scalabrino@unimol.it

[1] STAKE Lab, University of Molise, da Fonte Lappone, 86090 Pesche (IS), Italy

[2] SEART @ Software Institute, Università della Svizzera italiana, Via Giuseppe Buffi 13, 6900 Lugano, Switzerland