Help Them Understand: Testing and Improving Voice User Interfaces

EMANUELA GUGLIELMI, STAKE Lab, University of Molise, Italy
GIOVANNI ROSA, STAKE Lab, University of Molise, Italy
SIMONE SCALABRINO, STAKE Lab, University of Molise, Italy
GABRIELE BAVOTA, Università della Svizzera italiana, Switzerland
ROCCO OLIVETO, STAKE Lab, University of Molise, Italy

Voice-based virtual assistants are becoming increasingly popular. Such systems provide frameworks to developers for building custom apps. End-users can interact with such apps through a Voice User Interface (VUI), which allows the user to use natural language commands to perform actions. Testing such apps is not trivial: The same command can be expressed in different semantically equivalent ways. In this paper, we introduce VUI-UPSET, an approach that adapts chatbot-testing approaches to VUI-testing. We conducted an empirical study to understand how VUI-UPSET compares to two state-of-the-art approaches (*i.e.,* a chatbot testing technique and ChatGPT) in terms of (i) correctness of the generated paraphrases, and (ii) capability of revealing bugs. To this aim, we analyzed 14,898 generated paraphrases for 40 Alexa Skills. Our results show that VUI-UPSET generates more bug-revealing paraphrases than the two baselines with, however, ChatGPT being the approach generating the highest percentage of correct paraphrases. We also tried to use the generated paraphrases to improve the skills. We tried to include in the *voice interaction models* of the skills (i) only the bug-revealing paraphrases, (ii) all the valid paraphrases. We observed that including only bug-revealing paraphrases is sometimes not sufficient to make all the tests pass.

CCS Concepts: • **Software and its engineering** → **Software evolution**; **Maintaining software**; *Software defect analysis.*

Additional Key Words and Phrases: voice user interfaces, software testing, NLP

ACM Reference Format:
Emanuela Guglielmi, Giovanni Rosa, Simone Scalabrino, Gabriele Bavota, and Rocco Oliveto. 2023. Help Them Understand: Testing and Improving Voice User Interfaces. In *Proceedings of Transactions on Software Engineering and Methodology (TOSEM).* ACM, New York, NY, USA, 32 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Voice holds significant power as an input method in human-computer interaction, serving as a fundamental mode of human communication [32]. Voice-based virtual assistants like Google Assistant, Alexa, and Siri have gained rapid popularity among users due to their extensive integration with external services, such as home automation devices, and their user-friendly nature. It is projected that the number of voice-based virtual assistants in use will reach a staggering 8.4 billion by the year 2024 [31]. These virtual assistants enable users to perform various basic tasks, such as checking the weather or setting timers. Moreover, they provide developers with frameworks that allow for the expansion of supported actions through the development of specific applications. An example of such applications is the TikTok Alexa skill. This application enables users to search and discover content on TikTok, check messages, and even start recordings. What distinguishes these voice-based applications from traditional ones is the mode of user interaction. Instead of relying on a Graphic User Interface (GUI), where users interact with visual elements on a screen, these applications employ a Voice User Interface (VUI). The VUI allows users to engage with the application using natural language commands and queries spoken aloud, making the interaction more intuitive and conversational [15]. To enable smooth and accurate interactions, the developers of these applications need to define a Voice Interaction Model (VIM). The VIM serves as a framework that establishes a connection between the different states or contexts in which the application can be, and the example utterances or phrases that users may speak to trigger specific actions. In other words, it maps the various possible ways users can express their intentions to the corresponding actions or functionalities that the application should execute. By defining a comprehensive VIM, developers ensure that the voice-based application can effectively understand and respond to user commands and queries, providing a seamless and user-friendly experience in the absence of a traditional graphical interface. More specifically, the VIM provides an explicit association between *seed sentences* and the *intents* they trigger.

The underlying Artificial Intelligence model of voice-based virtual assistants grants them a significant degree of tolerance towards slight differences between the spoken utterance and the anticipated one. For example, if the expected utterance is "*what are the news?*" and the actual utterance is "*read me the news*", the desired action will likely be triggered. However, if the actual utterance is too far from the expected one (*e.g.,* "*what happened today?*"), the app might not behave properly. The fact that users can adopt different utterances to express the same command/query makes the testing of VUIs far from trivial.

Previous research introduced approaches for automatically testing GUIs [39, 42, 48]. However, research in VUI-testing is still at its early stages, and only a few solutions exist. One of the state-of-the-practice solutions was experimentally provided in the Alexa Developer Console (ADC), the environment used to implement and publish Alexa skills. ADC integrated a Deep Learning (DL)-based tool that allowed developers to generate, given a seed utterance s, a set of paraphrases P_s. As of February 2023, however, the *Automated Test Sets* tool has been deprecated and is no longer available on the developer console. Such a tool, anyways, generated few paraphrases [25] and, therefore, it was unlikely to cover several corner cases.

Testing VUI-based apps poses the same conceptual problems as testing chatbots. Chatbots provide an interface in which users *write* commands, thus using the same type of interaction of VUI-based apps (*i.e.,* based on natural language) but through different means (text-based *vs* voice-based). Previous research introduced a rigorous approach for automatically generating paraphrases for testing chatbots [26], which consists in (i) replacing words and phrases with synonyms, thus generating a set of candidates, and (ii) filtering candidates based on their similarity with the input utterance through a combination of metrics. Besides, Large Language Models, like ChatGPT, have recently proven to be very effective in handling text-related tasks such as summarization, amending, and rephrasing. It is unclear, however, to what extent both approaches work with VUI-based apps, given that they are based on a different mean of communication, which results in using expressions of different type and length.

In this paper, we extend our previous work [25] in which we introduce VUI-UPSET (Voice User Interfaces Utterance ParaphraSe gEneraTor), an approach that adapts chatbot-testing techniques to VUI-testing. Similarly to the approach proposed by Guichard *et al.* [26], *i.e.,* the chatbot-testing technique we took inspiration from (GRSBV, from now on), VUI-UPSET (i) generates a large set of candidate paraphrases and (ii) filters the most promising ones. As for the first step, while GRSBV replaces a word/phrase at a time with a synonym leaving the others unchanged, VUI-UPSET replaces also more than one word, thus generating a much larger set of candidate paraphrases. The main point of novelty lies, however, in the second step: VUI-UPSET uses a DL model for checking the equivalence between the seed sentence and each generated paraphrase, while GRSBV relies on several manually crafted metrics. We run an empirical study with 40 open-source Alexa skills to understand how VUI-UPSET compares to both GRSBV [26] and a more sophisticated approach that relies on Large Language Models, based on ChatGPT. We checked to what extent (i) the generated paraphrases are valid (*i.e.,* imperatively equivalent to the seed sentence) and (ii) able to reveal bugs in the VUIs. To this aim, we manually analyzed 14,898 generated paraphrases, totaling 33,835 evaluations. Our results show that an approach based on ChatGPT generates the highest percentage of valid paraphrases (~59%) as compared to the ~45% achieved by VUI-UPSET and ~21% by GRSBV. However, VUI-UPSET generates a higher absolute number of valid paraphrases for most of the skills. We obtained a similar results in terms of percentage of bug-revealing paraphrases generated by the approaches, which is higher for ChatGPT (27.4%) than for VUI-UPSET. Still, since VUI-UPSET generates a higher number of valid paraphrases, it also generates a significantly higher number of bug-revealing paraphrases (713 more than ChatGPT, in total). This means that VUI-UPSET allows developers to test more thoroughly their apps at the cost of discarding a higher number of irrelevant paraphrases.

To understand whether the generated paraphrases can be used to improve the VIM and, thus, possibly the users' experience, without introducing bugs when default commands are used, we tried to integrate the generated paraphrases into the original VIMs. More specifically, we tried to improve the VIMs in three ways: (i) We only added the valid bug-revealing paraphrases, assuming that they are the ones that might contribute the most to the skill improvement; (ii) we included all the valid paraphrases, to understand the best improvement achievable by the techniques; (iii) we tried to modify the VIMs by adding subsets of paraphrases while using the remainder for testing them in a 4-fold cross validation for each technique, to understand to what extent the generated paraphrases might allow to improve the skill when unseen utterances are provided . We tested the three improved types of VIM with the whole test suites (which includes both seed sentences and valid generated paraphrases). We found that, while including only bug-revealing paraphrases is safe (*i.e.,* the seed sentences still work as intended), it is not sufficient for some skills. On the other hand, as expected, including all the paraphrases allows to make all the tests pass (except for a single case). Finally, we observed that improving the VIMs with generated paraphrases almost always results in more robust skills, which is proven by the fact that the percentage of failing tests is lower for most of the skills when tested on unseen paraphrases generated by both VUI-UPSET and ChatGPT.

Summarizing, the novel contributions of this paper as compared to our previous work [25] are the following:

- We added comparison of VUI-UPSET with a state-of-the-art approach that relies on Large Language Models and, specifically, on ChatGPT.
- We doubled the sample of skills taken into account for the empirical study to improve the generalizability of our results. Specifically, we extended the initial dataset by adding 20 skills for validation from the 20 skills considered in our previous study. Since for 2 skills we considered in our previous study the VIM had been updated, we re-ran our experiments also for them.
- We assessed the extent to which the paraphrases generated by VUI-UPSET and ChatGPT can be integrated into the VIM of the skills to reduce the number of issues.
- We checked the extent to which the sets of paraphrases generated by VUI-UPSET and ChatGPT might realistically allow to cover unseen new paraphrases by running a 4-fold cross-validation for both the approaches.

The rest of the paper is organized as follows. Section 2 provides background information and discusses the related literature. Section 3 provides a motivating analysis in which we report the differences between typical Chatbot and VUI inputs. Section 4 presents in details VUI-UPSET. Section 5 describes the empirical study that we conducted to evaluate the capability of VUI-UPSET in generating imperatively equivalent and bug-revealing paraphrases. We present the results of the study in Section 6. Section 8 concludes the paper after a discussion of the threats that could affect the validity (Section 7).

2 BACKGROUND AND RELATED WORK

In this section we present an overview of the basic concepts behind VUIs and VUI-testing. Then, since we instantiate the approaches on the framework provided by Amazon Alexa, we explain how a VUI-based app for Alexa (*Alexa skill*) works. Finally, we describe in detail the baseline technique we use in our study: The chatbot-testing approach defined by Guichard *et al.* [26].

2.1 Voice User Interfaces

A Voice User Interface (VUI) is a type of interface through which users interact with a computer system by providing spoken commands or queries and receiving primarily spoken responses in return. Although VUIs have been in existence for quite some time, their widespread adoption has been facilitated by advancements in natural language processing (NLP) and voice recognition technologies. Typically, a VUI consists of two main components: a speech recognition component that converts spoken words into text, and an NLP component that matches the natural language input to a predefined set of commands and triggers the corresponding action. We mainly focus on the latter component. When users adopt the exact commands defined by the developers, matching the command is easy. However, users might not be willing to have tight constraints when using VUIs, which are supposed to feel natural as talking to another human being. Thus, a good VUI should work even when users adopt variations of the commands intended by developers. For example, the command "list all the recipes" is equivalent to the command "read me all the recipes," and users should be allowed to use either of them with the same result. In this context, the problem of testing VUIs is quite hard to address: It requires to define a set of variations (*i.e.,* paraphrases) of the commands defined by developers and check whether the same action is triggered.

The field of testing VUIs (and chatbots) is still in its infancy, and only a few recent approaches exist in the literature. Cabot *et al.* [11] conducted a study aimed at identifying the relevant testing properties and techniques and their adaptation for NLP-based bots. Bird *et al.* [7] tackled the problem of training DL models for tasks related to chatbot interaction. To this aim, they introduced the "Chatbot Interaction with Artificial Intelligence" (CI-AI) framework, which augments user-generated data with paraphrases, similar to what we do in VUI-UPSET. Bozic *et al.* [10] proposed an approach based on metamorphic testing that automatically verifies the functionality of a chatbot.

As the field of chatbot testing research continues to evolve, new studies are being conducted to explore the usability testing aspect of chatbots. These studies aim to provide valuable insights and understanding into how chatbots can be effectively tested for usability purposes [46]. Zhang *et al.* [57] propose a Panel of Experts (PoE), which is a multitask network comprising a shared transformer encoder and lightweight adapters. The work aims to develop a model-based automatic dialogue evaluation metric (ADEM) that can effectively measure the quality of conversational agents across various domains. Coca *et al.* [14] present a framework for generating synthetic schemas which uses tree-based ranking to jointly optimise lexical diversity and semantic faithfulness.

According to the literature, the development of a good chatbot requires a continuous process that relies on various high-quality data. Therefore, previous work [44] introduced approaches for generating paraphrases through the use of different techniques. Automatically generating a test case in this context means creating natural language sentences,

```
"intents": [
  "name": "RegisterBirthdayIntent",
  "samples": [
    "register my birthday",
    "record my birthday",
    "remember my birthday",
    "register my date of birth"
  ]
]
```

Fig. 1. Example of voice interaction model.

given as input to the VUI (*test data*), and checking if the response behavior (which usually includes the target state of the app) is correct (*test oracle*).

Generally, the response behavior consists in (i) changing the state of the app and, optionally, (ii) providing a spoken response. A natural way of generating test cases in this context is by using an approach inspired by metamorphic testing [49]: Given an utterance u for which the response behavior b_u is expected, it is possible to change the utterance to an equivalent one (*i.e.,* a paraphrase), u', and assert that the executed behavior is still b_u.

Traditional approaches for utterance paraphrasing, such as hiring experts or crowdsourcing, are costly, time consuming, and with their own trade-offs in terms of quality [29, 53]. For example, J.Bird *et al.* [8] present the transformer-based Chatbot Interaction with Artificial Intelligence (CI-AI) framework for task classification. The intelligent system enhances human-generated data by utilizing artificial paraphrasing techniques to create a substantial training dataset. Automatic paraphrasing is becoming increasingly appealing as a viable alternative, offering the potential for a rapid, scalable, and cost-effective testing process. A recent work was introduced by Li *et al.* [33] in specifically presented a study on voice interface testing, introducing VITAS, a framework for testing VUIs in VPA applications by simulating a multi-state interaction. On the other hand, VUI-UPSET addresses a very specific problem (*i.e.,* the generation of valid paraphrases) regardless of how the skill reaches the state from which the desired activation phrase is to be used.

2.2 Alexa Skills and Developer Console

Amazon's cloud-based voice assistant, *Alexa* [2], serves as the "intelligence" behind devices like Amazon Echo. The applications developed for Alexa are referred to as *skills*, and developers utilize the Amazon Developer Console (ADC)[4], an integrated environment, for the development, testing, and distribution of these skills. A skill consists of two main components: the *voice interaction model* and the *programming logic*. The voice interaction model defines the user utterances that can activate various functionalities within the programming logic. Specifically, it associates *seed utterances* with specific intents, which in turn trigger corresponding *handlers* in the programming logic to generate a response for the user. Fig. 1 reports an example of the structure of a voice interaction model of the skill "Happy Birthday". The *name* field refers to the name of the intent to invoke when one of the seed sentences in the *samples* array is matched.

Alexa is tolerant to small variations between the utterance pronounced by the user and the expected seed utterance: For example, if the seed utterance is "*register birthday*" it is most likely that Alexa will still trigger the same intent for the pronounced utterance "*register my birthday*". This is not necessarily true when the equivalence of the utterances is less trivial (*e.g.,* when the user pronounces "*take note of my birthday*"). Therefore, the larger the set of seed utterances provided by the developer, the higher the chance that Alexa recognizes the users' intent and triggers the desired action, thus improving the perceived quality of the app. Seed utterances might contain *slots*, *i.e.,* placeholders that allow users to give inputs [3]. For example, slots can be used to recognize the date of birth of the user pronounced in the sentence

Table 1. Example of Alexa Skill stateful

Happy Birthday Skill

User	**VUI**
take note on my birthday	
	when is your birthday?
the 8th	
	of which month?
December	
	...

"*I was born on January 1st, 2000*". The ADC can be used to test a given skill by providing a set of utterances associated with the expected intent that will be triggered (*annotation sets*). This feature is available in the NLU-evaluation tool of the ADC [6]. Such a tool associates to each utterance in the annotation set a binary value (*PASSED* or *FAILED*) based on the fact that the actual intent equals or not the expected one.

There are some tools that facilitate the generation of annotation sets to test the skills. KayLearch [30] relies on using a grammar to generate variants of the given utterances. However, such a tool requires developers to manually define the grammar. ADC provides a feature that allows developers to automatically generate paraphrases for seed utterances to either test their skill or to enlarge the set of supported seed utterances. Also, it is important to note that some Alexa skills are state-based therefore the user can interact with VUIs by going through multiple interaction states for a single request. We report an example of the use of the Alexa *Happy Birthday* Skill in Table 1.

The ADC tool for generating paraphrases allows developers to choose among three alternatives: (i) *Interaction model*, which simply creates the tests containing the sample utterances from the voice interaction model; (ii) *Frequent Utterances*, which creates tests based on the utterances frequently used in the *simulation* step by the users, and (iii) *Utterances Recommendation Engine*, which automatically generates paraphrases by varying the seed utterances in the voice interaction model. Since the first two alternatives are semi-automatic, in our study we focus on the fully-automated alternative, *i.e.,* the *Utterances Recommendation Engine*. From now on, when we refer to the *ADC tool* we are specifically referring to the *Utterance Recommendation Engine*. While it is not entirely clear how such a feature works, the official documentation mentions that it is based on machine-learning [5]. There are several papers presented by researchers in the Amazon Alexa team [43, 50, 51] describing possible solutions that might have been adopted in the recommendation engine. If this is the case, the approach behind the Utterances Recommendation Engine is based on encoder-decoder deep recurrent neural networks.

2.3 Generating Paraphrases for Testing Chatbots

While VUIs have their own peculiarities, they are conceptually similar to the textual interface used in chatbots [9]. A *chatbot* is a software system designed to answer human questions in text format, according to the specifications for which it was designed. Over time, chatbots have embedded increasingly sophisticated algorithms to create more natural and complex dialogues. Therefore, we describe below in details the only approach involving both (i) a paraphrase generation and (ii) a filtering phase (to the best of our knowledge) defined in the literature to generate paraphrases for testing chatbots, since we use it a starting point for our approach and as a baseline in our comparison [26].

Guichard *et al.* [26] introduce a rule-based approach (as opposed to the data-driven paradigm) that generates paraphrases aimed at evaluating the robustness of chatbots. For simplicity, from now on we call such an approach

GRSBV (from the initials of the authors' surnames). GRSBV involves, as a first step, the selection of the original utterances present within the intent of the conversational agent. The authors assume that all input utterances are grammatically and syntactically correct with no spelling mistakes. The workflow of GRSBV support the generation of candidate paraphrases through lexical substitution, and the filtering of non-equivalent paraphrases through a set of metrics. We describe such steps in details below.

2.3.1 Generation of Candidate Paraphrases. First, GRSBV transforms the input utterances to lower-case as a preprocessing step to avoid case sensitivity issues. The utterance is then tokenized and each token is tagged with its respective Part-of-Speech (PoS). GRSBV performs dependency parsing among tokens to identify phrasal verbs. Tokens that appear in the stop word list are left as they are. For each of the remaining ones, GRSBV runs lemmatization and it uses the lemmas for finding synonyms in a lexical database, *i.e.,* WordNet [41]. The synonyms obtained are then inflected (*i.e.,* pluralization, singularization, and conjugation), based on the inflection of the respective original token before lemmatization. To increase the likelihood to generate semantically equivalent candidate paraphrases, GRSBV changes a token at a time with each of its respective synonyms. Hence, it does not consider the simultaneous substitution of multiple tokens in the input utterance.

2.3.2 Filtering Paraphrases. To further ensure the quality of the generated paraphrases, GRSBV combines some of the strategies proposed by Hassan *et al.* [27] to score the candidate paraphrases to filter out poor ones. Many words have different meanings depending on the context in which they are found. Given a candidate paraphrase p generated from the input utterance i, and the word in the original utterance w_i which was replaced with a synonym w_p in the paraphrase, the following metrics are computed.

Language Model (LM). A 5-gram language model is built using an English corpus. Then, it is used to predict the probability of every token in p from the 6th to the last one, given the previous five tokens. The LM score is computed as the average probability obtained for such tokens. If the candidate paraphrase contains 5 or less tokens, the assigned score is 0. *Rationale*: The higher the naturalness of p, the higher the likelihood that p is syntactically correct.

Translation Pivoting (TP). The candidate paraphrase p is translated into a foreign language (*i.e.,* French) and it becomes $p_{E \to F}$. Then, $p_{E \to F}$ is translated into the original language (*i.e.,* English) again, and it becomes $p_{E \to F \to E}$. If w_p is still present in $p_{E \to F \to E}$, the TP score is 1; otherwise, it is 0. *Rationale*: If w_p makes sense in the context, the translation in English of the equivalent sentence in the foreign language still contain w_p.

Word Vectors (WV). The Word2Vec model [40] is used to compute the lexical similarity between w_p and all the tokens in i which are not stop-words. WV is equal to the average of such values. *Rationale*: If w_p fits the context, the WV score will be higher (*i.e.,* it will be similar to the other tokens in i).

Web Search (WS). A Web Search engine (specifically, Microsoft Bing) is used to search p (exact match of the whole sentence). The WS score is equal to the total number of web pages retrieved. *Rationale*: The higher the number of pages retrieved, the higher the likelihood that p is semantically equivalent to the input sentence.

Word-Sense Disambiguation (WSD). The Simplified Lesk algorithm implemented in Pywsd [52] is used to retrieve the most likely sense of w_i. If the synset of the most common sense of w_i, identified by using WordNet [41], contains w_p, the WSD score is 1; otherwise, it is 0. *Rationale*: If a synonym related to the most common meaning of a word is used, it is more likely that p is semantically equivalent to the input sentence.

Lexical Frequency (LF). The synsets related to the all the senses of w_i are identified through WordNet [41]. The LF score is equal to the number of synsets for that contain w_p. *Rationale*: The higher the number of senses for which w_p is a valid alternative of w_i, the higher the likelihood that w_p is a synonym that preserves semantic equivalence, regardless the sense of w_i.

The metrics are then normalized and linearly combined using weights identified through a genetic algorithm. Finally, a threshold (0.59) is used to filter out paraphrases not sufficiently good. It is worth noting that GRSBV requires the execution of many different models and tools, some of which require training (*e.g.,* LM) or cannot be used for free (*e.g.,* WS and TP). This makes the approach hard to reproduce and slow in the execution.

2.4 Large Language Models for Paraphrase Generation

In recent years, large-scale language models such as GPT-4 have revolutionized natural language processing (NLP). These models are designed to process and generate human language and have a wide range of applications, including natural language understanding, text generation, language translation and more [47, 54]. One of the best known large language models recently introduced is GPT-4, which stands for "Generative Pre-trained Transformer 4." GPT-4 is a product of OpenAI and is known for its ability to understand and generate human-like texts in various languages and domains [18]. Paraphrase generation and identification tasks are the most important in the field of natural language processing. The use of techniques such as language models are critical for recognizing or creating sentences that convey similar meanings [45]. C. Hegde and S. Patil exploit the generation capability of GPT-2 to generate unsupervised paraphrases from labeled data [28]. Witteveen and Andrews introduced a valuable technique that exploits the capabilities of these large language models for generating paraphrases in different textual domains. Their approach is not only effective in generating sentence-level paraphrases, but also extends its usefulness to longer portions of text, such as paragraphs, without requiring text segmentation [55].

3 ARE VUI INPUTS DIFFERENT FROM CHATBOT ONES?

The need for this study and for a new approach is mainly motivated by our assumption that *chatbot inputs are intrinsically different from VUI inputs*. To support our hypothesis, we run a preliminary analysis to compare the two populations of input sentences. We considered all the utterances extracted from the Voice Interaction Models (VIMs) of the 40 skills we use in our main study (reported in Section 5) as examples of VUI inputs. Instead, we used a set of conversational datasets for chatbots [19, 21–23] as examples of chatbot inputs. We randomly sampled 100 instances from both and compared them in terms of length (median number of characters and mode number of words) and readability (using the Coleman-Liau Index [34]).

The results of this analysis show that typical chatbot input sentences are longer than VUI input sentences (46 median number of characters against 19, *i.e.,* ~59% less). This result is confirmed if we look at the mode number of words, which is 9 for chatbot inputs and only 4 for VUI inputs (~56% less). Finally, when we take readability into account, we can observe that chatbot input texts are much more readable than VUI input texts (4.6 and 1.6, respectively). This suggests that the two sets of sentences are intrinsically different.

To qualitatively show how chatbot and VUI inputs are different, we report some examples. Examples of chatbot inputs are *"I often watch movies with my girlfriend, and I want to make sure whatever we watch, it's something we can both enjoy and and relax while watching"*. Instead, examples of VUI inputs are *"watch video"*. As it can be noticed, the VUI inputs are much more direct and imperative than chatbot inputs.

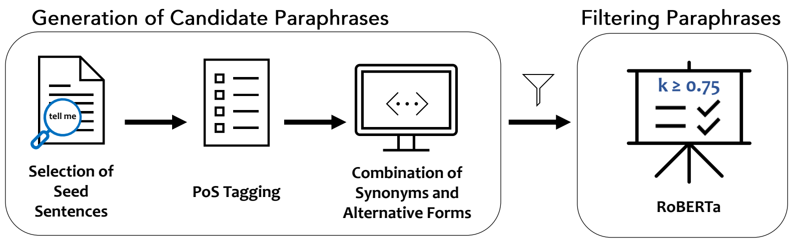
Generation of Candidate Paraphrases
Filtering Paraphrases
tell me
Selection of Seed Sentences
PoS Tagging
Combination of Synonyms and Alternative Forms
k ≥ 0.75
RoBERTa

Fig. 2. The workflow of VUI-UPSET.

4 GENERATING PARAPHRASES FOR ROBUST TESTING VUIS

VUI-UPSET (Voice User Interfaces Utterance ParaphraSe gEneraTor) is an approach that, given a seed sentence s generates several *imperatively equivalent* paraphrases of s, P_s. 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. VUI-UPSET uses the same workflow defined in the work by Guichard *et al.* [26]. First, it generates a set of candidate paraphrases, and then it filters them. Compared to the approach by Guichard *et al.* [26] there are three main variations. First, we use Part-of-Speech (PoS) tagging on the seed sentence and define strategies to handle words based on their PoS. Second, we change more than one synonym at a time to generate a higher number of candidate paraphrases. Third, we replace the convoluted and expensive filtering technique used in [26], which requires the computation of six metrics, with a simpler approach that relies on a Deep Learning model trained to check the semantic equivalence of two sentences. Fig. 2 depicts the workflow of VUI-UPSET.

4.1 An example of Paraphrase Generation

In Fig. 3, we present a concrete example of the use of VUI-UPSET in the context of the skill "Happy Birthday." Specifically, during the candidate paraphrase generation phase, the initial step involves the selection of seed sentences from the voice interaction model. In this case, one of the selected seeds is "record my birthday." In the next step, we perform Part-of-Speech (PoS) labeling each word in the sentence. As a result, the words "register", "my", and "birthday" are tagged with *VB* (verb, base form), *PRP* (possessive pronoun), and *NN* (common noun), respectively. Next, we extract the synonyms for "register" (*e.g.,* "record" and "file") and "birthday" (*e.g.,* natal day). We do not look for synonyms of the word "my" because of its PoS (*i.e.,* pronoun): instead, we use our pre-defined rules (detailed below) to generate alternative forms for such a word (*e.g.,* we omit the word or we replace it with "our"). At this point, we generate all the combinations of the alternative forms of each word extracted for the three original words in the seed sentence and generate a set of candidate paraphrases (*e.g.,* "record birthday" and "file our natal day"). Finally, in the "Filtering Paraphrases" stage, we use the RoBERTa model to decide which generated paraphrases we should keep. Specifically, we give each generated paraphrase (*e.g.,* "record birthday") and the original seed sentence ("register my birthday") as input to the model, and it returns a value between 0 and 1. We discard all the paraphrases for which such a value is lower than a threshold (we provide more details on this aspect and on the tuning of the threshold later on). For example, we keep "record birthday" because the model computes a high similarity value (*i.e.,* 0.82), while we discard "file our natal day" because its similarity value is lower, according to the model (*i.e.,* 0.67).

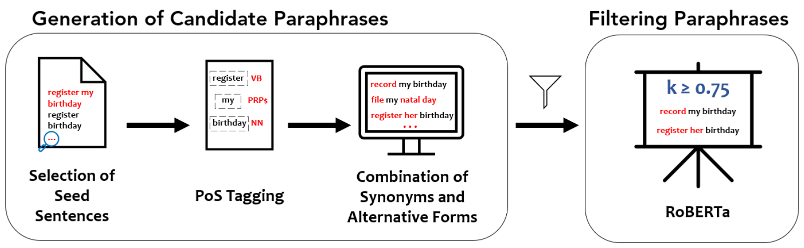

Fig. 3. Example “Happy birthday Skill” VUI-UPSET.

4.2 Generation of Candidate Paraphrases

First, the developer selects the seed sentences of interest. Then, for each of them, we use a PoS tagger to assign each word with a PoS. We extract the set of synonyms and PoS-sensitive variations for each word (or phrase) in the seed sentence, and we combine such sets to generate the candidates (using the cartesian product). We describe below in details all the steps.

4.2.1 Selection of Seed Sentences. To ensure the generation of good quality utterances, the developer manually selects the seed utterances for which paraphrases will be generated. We do not generate paraphrases for all the seed sentences for two reasons. First, we want to select only grammatically correct sentences. This is also required by GRSBV [26], as otherwise grammatically incorrect paraphrases would be consequently generated. An example of grammatically incorrect seed sentence is “*I not sure*”. Second, we want to exclude sentences that are minor variations of other sentences. We do this because our approach would generate, as a consequence, very similar (or identical) paraphrases starting from them. Such near-clones would not be useful in practice since Alexa (like other virtual assistants) is able to automatically compensate for such small variations. For example, if both utterances “*where is my order*” and “*where’s my order*” are present, we only keep one of them (*e.g.,* the first one). Finally, we exclude seed sentences (i) containing slots (*i.e.,* variable parts of the sentence), as handling them goes beyond the scope of VUI-UPSET, and (ii) appearing in the default skill intent (*e.g.,* AMAZON.HelpIntent) because, even though they can be expanded, they are already handled by the framework.

4.2.2 PoS Tagging. The idea behind the generation of the new utterances is to generate paraphrases that are imperatively equivalent to the set of seed sentences manually selected in the previous step. At this point, in this second step, it is necessary to analyze the syntactic structure of the seed sentence. This can be achieved by performing a grammatical analysis of the sentence, *i.e.,* the PoS tagging. PoS tagging is a NLP task in which the goal is to find the part of speech corresponding to each word contained in a sentence. For this purpose, VUI-UPSET relies on the open source *Stanford CoreNLP* tool [38]. Based on the grammatical analysis performed, *PoS* tagging associates each word with a tag. We consider all 36 tags supported by *Stanford CoreNLP.* We defined a set of rules for each type of PoS that will be used in the paraphrase generation phase, as described in the next section.

4.2.3 Combination of Synonyms and Alternative Forms. First, we run dependency parsing using *Stanford CoreNLP* to identify and retrieve dependencies of interest among words. This is particularly important for finding and correctly handling phrasal verbs [26]. Then, for each token, we verify that it is not a stop word by using a predefined stop word list [1]. In case it is classified as a stop word it remains unchanged in all the generated paraphrases. At this point, based

Table 2. Rules for the utterance generation based on the Part-of-Speech Tags.

Tag	Name	Action	Tag	Name	Action
CC	Coordinating Conjunctions	Unchanged	**PRP**	Possessive Pronouns	{my | our } or Omitted
CD	Cardinal Numbers	Unchanged	**RB**	Adverbs	Synonyms or Omitted
DT	Determiners	Unchanged	**RBR**	Comparative Adverbs	Unchanged
EX	Existence There	Unchanged	**RBS**	Superlative Adverbs	Unchanged
FW	Foreign Words	Unchanged	**RP**	Particles	Unchanged
IN	Prepositions and Sub. Conjunctions	Unchanged	**SYM**	Symbols	Unchanged
JJ	Adjectives	Synonyms or Omitted	**TO**	to	Unchanged
JJR	Comparative Adjectives	Synonyms or Omitted	**UH**	Interjection	Unchanged or Omitted
JJS	Superlative Adjectives	Synonyms or Omitted	**VB**	Verbs (base form)	Synonyms
LS	List Item Markers	Unchanged	**VBD**	Verbs (past tense)	Synonyms
MD	Modal Verbs	Unchanged	**VBG**	Verbs (gerund or present participle)	Synonyms
NN	Common Nouns	Synonyms	**VBN**	Verbs (past participle)	Synonyms
NNS	Common Nouns (Plural)	Synonyms	**VBP**	Verbs (non 3rd person singular present)	Synonyms
NNP	Proper Nouns (Singular)	Synonyms	**VBZ**	Verbs (3rd person singular present)	Synonyms
NNPS	Proper Nouns (Plural)	Unchanged	**WDT**	Wh-determiner	Unchanged
PDT	Predeterminers	Unchanged or Omitted	**WP**	Wh-pronoun	Unchanged
POS	Possessive Endings ('s)	Unchanged	**WP**	Possessive wh-pronoun	Unchanged
PRP	Personal Pronouns	{me | her | him | us } or Omitted	**WRB**	Wh-adverb	Unchanged
PRP	Personal Pronouns	{I | you | she | he | we | they } or Omitted			

on the PoS tagged in the previous step, we apply a consequent strategy, as described in Table 2. *Unchanged* means that the word must be left in the sentence as is in all the generated paraphrases. Some words, indeed, should/can not be modified. For example, prepositions or foreign words are never changed. *Omitted* means that the word can be completely removed without altering the meaning of the sentence. Therefore, some paraphrases will contain such a word, while others will not. *Synonyms* means that the word can be replaced with a synonym. For some specific words, such as pronouns, we manually identified and determined a set of alternatives. For example, personal pronouns can be replaced with "my" or "our". All the phrases for which VUI-UPSET decided to use a "Synonyms" strategy are lemmatized and we use WordNet [41] for finding, for each of them, the complete list of synonyms. At this point, each phrase is represented as a set of alternative forms, based on the previously described strategies. We use the cartesian product to generate candidate paraphrases. Finally, similarly to Guichard *et al.* [26], we inflect the replaced words as the original words.

For example, if a word was plural in the seed sentence, its variation is pluralized as well in the paraphrase. We use the *Pattern* approach [16], available as a Python library, to achieve this goal. For example, let us consider the seed sentence *"register my birthday"*. From the PoS analysis we get (register, VB) (my, PRP) (birthday, NN). Then, in the paraphrase generation step we are going to consider the following combinations: {remember|file|record|...} {|my|our} {birthday|natal day} and finally we get a set of candidate paraphrases (*e.g., record my birthday*, *record our birthday*, *file our birthday*, *record our natal day*, *remember birthday*).

4.3 Filtering Paraphrases

It is possible that some paraphrases generated in the previous step are not fully equivalent to the seed sentence since synonyms might refer to different meanings of a word. For example, the candidate paraphrase "time mark" might be generated from the seed sentence "time check". While "mark" is a synonym of "check", the two words have different senses in this context (*i.e.,* "control" in the seed sentence and "symbol of ticking off" in the paraphrase). Therefore, it is necessary to discard paraphrases that are not imperatively equivalent to the seed sentence. To that end, VUI-UPSET integrates an approach based on DL to filter out the generated sentences that are not valid. Specifically, we use a semantic similarity model, RoBERTa [35], which is a state-of-the-art NLP model, extension of the BERT model. RoBERTa has the same architecture as BERT [17], but uses a Byte Pair Encoding (BPE) as a tokenizer and a different pre-training objective. We used a pre-trained model available on HuggingFace [20]. In particular, we used the "large" version of RoBERTa, trained on the STS Dataset [12], containing five English-language corpora of varying size and domain, totaling over 160GB of uncompressed text, for 500k steps. Given a pair of sentences, the model returns a score between 0 and 1, which indicates the likelihood that the two sentences are imperatively equivalent. We run RoBERTa for each pair $\langle$*seed*, *paraphrase*$\rangle$; we discard paraphrases for which the returned score is below a given threshold k, which indicates the minimum similarity over which two sentences should be considered semantically equivalent. We discuss in detail how we tuned such a parameter in Section 5.

5 EMPIRICAL STUDY DESIGN

The *goal* of our study is to evaluate the effectiveness of VUI-UPSET in the automatic generation of paraphrases for testing VUI-based apps. In particular, we want to assess: (i) the semantic equivalence between the input seeds and the utterances generated by VUI-UPSET (ii) the extent to which our approach allows developers to identify bugs in the behavior of an *Alexa* skill and (iii) the extent to which VUI-UPSET allows developers to reduce the number of bugs by including imperatively equivalent generated sentences in their skills. As baseline, we use the approach by Guichard *et al.* [26].

We answer the following research questions (RQs):

- RQ$_1$: *To what extent does VUI-UPSET generate valid paraphrases?* This RQ aims at evaluating the quality of the paraphrases generated by VUI-UPSET in terms of imperatively equivalence compared to the original (seed) sentence.
- RQ$_2$:*To what extent do the paraphrases generated by VUI-UPSET reveal bugs?* With this RQ we want to understand if VUI-UPSET allows developers to generate a higher percentage and absolute number of paraphrases that reveal bugs compared to the baseline.
- RQ$_3$:*To what extent do the paraphrases generated by VUI-UPSET and ChatGPT reduce bugs?* With this RQ we want to understand if VUI-UPSET allows developers to reduce the percentage and absolute number of bugs in their Alexa skill by adding the imperatively equivalent paraphrases generated by VUI-UPSET.

5.1 Study Context

To answer our RQs, we have performed the evaluation on a dataset of 40 skills. These skills have been manually selected starting from the list of most popular open source skills on GitHub (with popularity based on number of stars). Several skills have been discarded since not properly working, while for the ones working we checked if both the voice interaction model and the programming logic were available. We only considered interaction models defined for the English language (en-US) and focused on skills written in *JavaScript*, as it is commonly used to implement Alexa skills.

We excluded skills with multi-step dialogues between *Alexa* and the user. The reason for this choice is that the management of skills that involve an interaction based on a continuous dialogue must be done considering the states that allow to predict the possible responses (*e.g.,* through the use of a graph of states). This goes beyond the scope of VUI-UPSET. Moreover, each selected *Alexa* skill was imported into the *Amazon Developer Console* to be verified. We executed some basic samples defined in the interaction model of the skill, excluding again the ones for which we did not observe a correct functioning (*i.e.,* no correspondence between the Voice Interaction Model and the Programming Logic). Table 3 describes all the *Alexa* skills selected for our study.

Table 3. Skills used for the empirical evaluation and their characteristics.

		# seeds		# generated paraphrases	
Skill	**Description**	**total**	**selected**	**GRSBV**	**VUI-UPSET**
Ilama facts	provides facts and trivia	8	8	15	35
Happy birthday	stores birthday information	6	6	103	211
City guide	recommends activities in the city	7	6	59	23
Quiz game	quiz game about a topic	6	5	66	268
Scheduling	schedules calls and appointments	9	9	193	102
Name the show	guess game about TV shows	18	15	204	167
Berry bash	quiz game about a topic	25	20	371	769
History facts	provides interesting historical facts	7	4	12	13
Particle cloud	provides access to Particle devices	19	14	298	391
Greeting sender	order management and sharing	17	16	382	319
Button trivia	single- and multi-player quiz game	24	17	373	523
Video app	allows to play a video	8	8	112	39
Salesforce	identifies opportunities on Salesforce	12	8	99	1203
Store Amazon pay	VUI app for the Amazon Store	45	24	557	1168
Timers	manages custom timers	16	11	407	473
Pocket	manages lists of items	21	11	178	1421
Piano player	teaches the piano	9	7	172	201
Week calendar	allows to plan the vacation	12	7	462	1342
Crash notification	demonstrates messaging for different errors	14	9	231	78
Memory	memory-matching game	4	4	74	171
Radio	allows the radio to be managed	15	11	246	41
Trading teacher	teaches options trading to end users	31	23	90	78
College finder	query the public dataset of colleges	102	61	1004	6115
Feed reader	allows you to play your feed	108	51	1177	4693
Premium fact	allows you to add new categories	16	14	36	35
Humour me	provides funny stories	12	10	176	994
Multistream audio	provides multiple audio, event and control managers	15	5	152	836
Premium hello world	adds greetings in several languages	23	21	317	329
Voice developer	allows the voice to be edited and provides news	48	36	848	3826
Tasty beverage	allows you to create a list	18	13	271	52
Weather wear	offers climate-appropriate clothing options	13	11	236	791
Next prayer	shows prayer times for Japan	4	3	35	47
Math facts	provides arithmetic exercises	7	4	49	44
Forget me not	VUI app for last minute reminders	11	4	55	84
Charity roster	allows to get to know charities and make donations	12	8	60	159
Cyclist assistant	provides info to cyclist	5	5	117	155
Internet archive	VUI app of Internet Archive	19	12	202	263
My barkeeper	store information on cocktails	14	110	1127	12385
Girls volleyball	info on volleyball match	14	9	168	15
Boys basketball	info on basketball match	5	5	77	6
Total		7 79	631	10,811	39,865

5.2 Experimental Procedure

We compare VUI-UPSET to two baselines: GRSBV, described in Section 2, and a new baseline based on Large Language Models and, specifically, ChatGPT. We needed to re-implement GRSBV since it is not publicly available. We carefully followed the indications provided in the paper to achieve this goal. As for the other baseline, we used ChatGPT [13] and, specifically, we employed the GPT-3.5-turbo model via the official APIs. Given a seed sentence, we used the following prompt to generate paraphrases through such a model: "Given the following input sentence 'seed-sentence', generate all possible paraphrases that you consider semantically equivalent to it". Note that we did not explicitly specify a maximum number of paraphrases but let the model autonomously decide based on its confidence for the seed sentence at hand. This means that for some sentences it might generate many paraphrases, while for others just a few of them.

To answer RQ$_1$, we evaluate the semantic equivalence between the sentences generated by VUI-UPSET and the two baselines. First, we manually selected the seed sentence for each *Amazon Alexa* skill in our dataset. In the selection of the utterances to consider, we chose the ones that (i) allowed us to cover all the intents of the skill, (ii) were well-constructed to satisfy the requirements of both VUI-UPSET and GRSBV (see Section 4), and (iii) did not contain slots. Second, we executed VUI-UPSET and GRSBV providing as input the selected seed sentences, obtaining for each skill a set of test utterances filtered by the semantic equivalence model.

The number of paraphrases generated by VUI-UPSET, GRSBV and ChatGPT was very high (39,865, 10,811 and 4,814, respectively), and manually evaluating their equivalence with the corresponding seed sentence would have been infeasible. Therefore we randomly selected a sample of the utterances generated with each approach for manual evaluation. We choose the size of the sample for each skill/approach so that it allows us to have a 5% margin of error with 95% confidence level. In total, we selected 6,291 instances for VUI-UPSET, 5,132 for GRSBV and, 3,475 for ChatGPT. As a result, in total, we evaluated 14,898 paraphrases.

Each generated paraphrase was independently evaluated by at least two authors, with three authors involved overall (*i.e.,* each paraphrase was assigned to two of the three authors involved in the manual validation). More specifically, each author checked whether the generated paraphrase and the respective seed sentence were imperatively equivalent.

Conflicts that arose after the comparison of the evaluations were resolved through a discussion among the three evaluators aiming at reaching consensus. The two evaluators originally assigned to each paraphrase had disagreements in 1,938 cases for VUI-UPSET (*i.e.,* ~30%), 1,146 cases in total for GRSBV (*i.e.,* 22%) and 955 in total for ChatGPT (*i.e.,* ~28%) while independently evaluating the imperatively equivalent of the paraphrases. After discussion, consensus was reached for all the paraphrases. In total, we manually performed 33,835 comparisons. Most of the conflicts arose in connection with borderline situations in which the paraphrases were interpreted differently by the evaluators. Furthermore, the high number of conflicts often relate to many paraphrases (*e.g.,* when an unusual synonym was used for a given word in the seed sentence or the use of the specific slang). For example, from the seed sentence "register my birthday" a generated paraphrase is "file my date of birth". The use of the verb "file" instead of "register" is imperatively equivalent, but unusual in this context. One of the evaluators reported the whole paraphrase as equivalent, while the other did not. The same happened for all the paraphrases that used such a synonym.

For each skill we measure the percentage of imperatively equivalent paraphrases. Then, given such percentages, we compare the three approaches by using the Wilcoxon Signed-Rank test [56]. The null hypothesis is that there is no difference between VUI-UPSET and the baseline in terms of percentage of imperatively equivalent generated paraphrases. 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 [37] 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 $|\delta| \geq 0.474$. Finally, we estimate the absolute number of imperatively equivalent generated paraphrases over the whole population for each skill s and approach a. To do this, we consider the percentage of imperatively equivalent generated paraphrases C_a^s computed on the sample, the total number of paraphrases generated with a for s, G_a^s, and the 5% margin of error given by the considered sample size (95% confidence level). Specifically, we compute the confidence interval of the number of imperatively equivalent generated paraphrases for s as $P_a^s = (C_a^s \pm 0.05) \times G_a^s$.

To compare the absolute number of imperatively equivalent generated paraphrases, we cannot check the difference in terms of absolute number of imperatively equivalent paraphrases found in the manually evaluated samples: Such numbers, indeed, strongly depend on the sample size, and we will have different sample sizes for the different approaches.

For example, if we have two samples of 10 and 100 paraphrases for the two approaches and we find that 9 out of 10 are imperatively equivalent in the first sample while 11 out of 100 are imperatively equivalent for the second sample, it is most likely that the first one would have achieved better result than the second one if we evaluated the whole population. Therefore, we compare the confidence intervals identified on the estimated number of imperatively equivalent paraphrases over the whole populations for each skill independently. 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). If there is an overlap, instead, we cannot exclude that the difference is due to the chance.

To answer RQ$_2$, based on the results obtained in RQ$_1$, we define and execute test cases containing the generated paraphrases considering only the valid paraphrases we manually identified to answer RQ$_1$. The execution of the generated test cases is performed through the *NLU evaluation* function from the developer console. Test cases are marked as *PASSED* when the actual intent activated for a the related paraphrase in the virtual version of Alexa matches the expected intent indicated in the VIM for the seed sentence from which the paraphrase originates. Otherwise, they are marked as *FAILED*.

For each skill, we compute the percentage of generated bug-revealing paraphrases by computing the number of *FAILED* test cases divided by the total number of valid paraphrases. We do this to understand if any of the approaches is able to outperform the other in terms of quality of the generated paraphrases. Then, we estimate the absolute number of bug-revealing paraphrases for each approach in the original population of paraphrases. To do this, we use a similar process used for RQ$_1$. We start from the number of imperatively equivalent paraphrases (both the lower and the upper bound of the confidence interval) to estimate the number of bug-revealing paraphrases. Also in this case, we consider 5% margin of error for both such values. We make sure that such an interval is lower-bounded by the number of actually found bug-revealing paraphrases in the sample (we are certain that at least such a number of bug-revealing paraphrases exist) and upper-bounded by the number of generated paraphrases. We use the Wilcoxon signed-rank test [56] 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 [37]. To compare the absolute number of bug-revealing paraphrases, we use an analogous 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 other for such a skill (with 95% confidence level).

To answer RQ$_3$, we conducted three separate assessments where we add the paraphrases generated with VUI-UPSET and ChatGPT in the voice interaction model for the respective skills, to check whether they help in fixing the bugs detected using process described for RQ$_2$. We focused on the paraphrases that we manually judged as imperatively equivalent in RQ_1, since those are the only ones which would make sense adding to the interaction model. As a first analysis, we added to the VIM only the paraphrases that detected a bug in RQ_2 (*i.e.,* those for which the skill was not properly working) and tested that (i) the original utterances present in the skill (*i.e.,* the ones implemented by the original developers) were still properly working as expected, with no "damage" done by the added paraphrases; and (ii) the complete set of original utterances plus the added paraphrases were now handled correctly by the voice interaction model (thus, the paraphrases fixed the bugs previously observed in RQ_2). As a second analysis, we added to the VIM all the imperatively equivalent paraphrases (not only the bug-revealing ones) to check if it would make sense for a developer to just add all valid paraphrases as a bulk without worrying to discriminate whether the paraphrase was bug-revealing or not. As a third analysis, we performed a four-fold cross validation, to understand how the skill behaves when tested on utterances completely different from the ones in the which it has been trained (in the VIM). This analysis allows us to estimate the actual improvement of the skill in terms of robustness. Specifically, we performed four evaluations for each skill by dividing the entire set of imperatively equivalent paraphrases in four folds. We used, in turn, three of these folds (75% of the instances) for improving the voice interaction model of the skill (we always keep the seed sentences in the VIM), and the remainder fold (25%) for testing this new version of the skill. For all the combinations previously described, *i.e.,* testing original utterances only or all added paraphrases with two different sets of paraphrases (failing only or all imperatively equivalent), we compute and report, for each skill, the absolute number of failed test cases, the percentage of failed test cases, and the estimated number of bugs that could be revealed by using the complete set of paraphrases generated by VUI-UPSET (similarly to RQ_2).

5.3 Tuning the Semantic Threshold for VUI-UPSET

VUI-UPSET requires the tuning of a threshold, k. We empirically determine such a threshold by running VUI-UPSET on a set of 10 *Alexa* skills different from the set we use for the empirical study. To do this, we randomly selected skills matching the same criteria we used for the empirical study, but also having a voice interaction model with at least 10 sample utterances after the manual selection. We ran VUI-UPSET without the filtering step, *i.e.,* we only generated the candidate paraphrases. Then, we computed the similarity score between the seed sentences and the generated ones. To calculate the similarity score, we used cosine similarity: Given two sentences s_1 and s_2, we first extract two numerical vectors representing them by using [12], v_1 and v_2; then, we compute the cosine between such vectors using the formula $\frac{v_1 \cdot v_2}{\|v_1\| \|v_2\|}$. As a result, we obtained a set of pairs of sentences composed of the source seed and the generated paraphrase, plus the similarity score between them. Two of the authors independently performed a manual analysis on a sample selected among all the sentences pairs, with 5% margin of error (95% confidence level). The similarity score of each pair was not shown to the evaluators to avoid any influence. In our evaluation, we analyzed a total of 371 sentence pairs, indicating *"not imperatively equivalent"* with 0 and *"imperatively equivalent"* with 1. After comparing the two assessments, we discussed the conflicts found to motivate the final choice (0 or 1). In total, the two evaluators disagreed in 94 paraphrases generated by ADC tool, 884 paraphrases generated by VUI-UPSET and 406 paraphrases generated by GRSBV and in the end they reached consensus on all of them. As a result of this process, each pair was assigned with a imperatively equivalent value and a similarity score.

We tested different threshold values, between 0.05 and 0.95 with a step of 0.05. We did not test the extremes because it would have implied to include (0) or discard (1) all the paraphrases. For each threshold, we computed well-known metrics in information retrieval, namely *precision* and *recall*. Given a candidate threshold k, we have a set of *retrieved paraphrases* (*i.e.,* paraphrases for which the similarity is higher than k). The set of *correctly retrieved paraphrases* is the subset of *retrieved paraphrases* for which, in our manual validation, we marked the paraphrases as imperatively

Table 4. Metrics computed for the similarity threshold evaluation.

k	**Precision**	**Recall**	**$F_{0.5}$ Score**	**Accuracy**
0.05	0.24	1.00	0.28	0.24
0.10	0.24	1.00	0.29	0.26
0.15	0.25	1.00	0.29	0.28
0.20	0.27	1.00	0.31	0.34
0.25	0.28	0.97	0.33	0.40
0.30	0.30	0.92	0.34	0.46
0.35	0.32	0.91	0.37	0.52
0.40	0.33	0.86	0.38	0.56
0.45	0.34	0.81	0.38	0.58
0.50	0.38	0.80	0.42	0.64
0.55	0.41	0.73	0.44	0.68
0.60	0.45	0.69	0.48	0.72
0.65	0.45	0.58	0.47	0.73
0.70	0.47	0.48	0.47	0.74
0.75	**0.51**	**0.40**	**0.49**	**0.77**
0.80	0.55	0.31	0.48	0.78
0.85	0.62	0.18	0.42	0.78
0.90	0.87	0.15	0.44	0.79
0.95	1.00	0.03	0.15	0.77

equivalent to the respective seed. Finally, the set of (imperatively) *equivalent paraphrases* is the set of all the paraphrases that we manually marked as imperatively equivalent (regardless of the fact that they were retrieved or not). Given such sets $precision_k$ is computed as $\frac{|correctly\ retrieved\ paraphrases_k|}{|retrieved\ paraphrases_k|}$, while $recall_k$ is computed as $\frac{|correctly\ retrieved\ paraphrases_k|}{|equivalent\ paraphrases|}$. Additionally, we computed the F_β *score*, *i.e.,* the generalization of the more commonly used F_1 *score*. While the latter gives equal weight to precision and recall, with the F_β *score* it is possible to give more weight to precision ($\beta < 1$) or to recall ($\beta > 1$). The F_β score is computed as:

$$F_{\beta,k} = (1 + \beta^2) \cdot \frac{precision_k \cdot precision_k}{(\beta^2 \cdot precision_k) + recall_k}$$

In our context, we set $\beta = 0.5$, to give more importance to *precision*: Given the high number of paraphrases generated by the first step of VUI-UPSET, we are more interested in discarding non-equivalent paraphrases than in including *all* the valid paraphrases. We set k as the threshold that allows to achieve the best $F_{0.5}$ *score*.

We report in Table 4 the results of the tuning. The best $F_{0.5}$ *score* can be achieved using as a threshold $k = 0.75$. With it, we achieve 0.51 precision (*i.e.,* half of the generated paraphrases are imperatively equivalent) and 0.4 recall (*i.e.,* we discard 60% of the valid generated paraphrases). The identified threshold is in line with the threshold used for determining the semantic similarity between texts in traceability link recovery [36]. It is worth noting that it would be possible to select other values of k depending on the context in which VUI-UPSET is executed and on how many false-positives practitioners are willing to manually discard. Based on the results of accuracy report in Table 4 we can see that as the threshold increases the accuracy also increases. Specifically, increasing the threshold allows us to discard a greater number of sentences that are not imperatively equivalent.

5.4 Replication Package

We publicly release the implementation of VUI-UPSET, our re-implementation of GRSBV, the command prompt used to interact with ChatGPT via API and the data used for answering RQ_1, RQ_2, and RQ_3 in our replication package [24].

Table 5. Comparison among VUI-UPSET and GRSBV in terms of imperatively equivalent of generated paraphrases. ◦ 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.

	RQ$_1$: Imperative Equivalence of the Generated Paraphrases											
	VUI-UPSET				**GRSBV**				**ChatGPT**			
Skill	**#generated◦**	**#equiv◦**	**%equiv**	**#equiv★**	**#generated◦**	**#equiv◦**	**%equiv**	**#equiv★**	**#generated◦**	**#equiv◦**	**%equiv**	**#equiv★**
llama facts	32	32	**100.00%**	* **34 ± 1**	14	9	64.29%	10 ± 1	61	26	42.62%	* **30 ± 4**
Happy birthday	136	47	34.55%	**73 ± 11**	79	21	26.58%	27 ± 5	48	23	**47.92%**	26 ± 3
City guide	22	17	**77.27%**	18 ± 1	51	7	13.72%	9 ± 2	57	26	45.61%	**31 ± 3**
Quiz game	158	64	40.50%	**109 ± 13**	54	32	59.26%	39 ± 3	43	26	**60.47%**	29 ± 2
Scheduling	81	54	66.66%	68 ± 5	129	46	35.66%	69 ± 10	69	53	**76.81%**	65 ± 4
Name the show	117	84	**71.79%**	**120 ± 8**	133	28	21.05%	43 ± 10	104	52	50.00%	72 ± 7
Berry bash	256	97	37.89%	**291 ± 38**	188	36	19.14%	71 ± 19	126	99	**78.57%**	148 ± 9
History facts	13	12	**92.31%**	12 ± 0	12	2	16.66%	2 ± 0	31	27	87.10%	**30 ± 2**
Particle cloud	194	53	27.32%	* **107 ± 20**	168	27	16.07%	48 ± 15	96	79	**82.29%**	***105 ± 6**
Greeting sender	175	32	18.28%	* **58 ± 16**	192	14	7.29%	30 ± 16	110	40	**36.36%**	* **56 ± 8**
Button trivia	222	103	**46.39%**	* **243 ± 26**	188	44	23.40%	87 ± 19	114	51	44.74%	74 ± 8
Video app	35	23	**65.71%**	26 ± 2	87	17	19.54%	22 ± 5	64	29	45.31%	* **35 ± 4**
Salesforce	291	134	46.05%	**554 ± 60**	79	15	18.99%	19 ± 4	64	40	**62.50%**	49 ± 4
Store Amazon pay	289	61	21.11%	**247 ± 58**	228	29	12.72%	71 ± 28	146	90	**61.64%**	145 ± 12
Timers	205	95	46.34%	**219 ± 24**	198	26	13.13%	53 ± 20	65	36	**55.38%**	43 ± 4
Pocket	303	86	28.38%	**403 ± 71**	122	44	36.07%	**64 ± 9**	80	47	**58.75%**	59 ± 5
Piano player	132	50	37.87%	**76 ± 10**	119	10	8.40%	17 ± 7	56	29	**51.79%**	34 ± 3
Week calendar	299	117	39.13%	**525 ± 27**	210	61	29.05%	134 ± 23	54	42	**77.78%**	34 ± 3
Crash notification	65	46	**70.77%**	55 ± 4	139	37	26.62%	61 ± 12	66	40	60.61%	48 ± 4
Memory	119	64	**53.78%**	**92 ± 9**	61	26	42.62%	32 ± 4	30	15	50.00%	17 ± 2
Radio	37	18	48.65%	20 ± 2	150	21	14.00%	34 ± 12	80	41	**51.25%**	**52 ± 5**
Trading teacher	65	47	**72.31%**	56 ± 4	73	22	30.14%	27 ± 5	136	77	56.62%	**120 ± 11**
College finder	362	138	38.12%	**2331 ± 306**	285	65	22.81%	229 ± 50	230	99	**43.04%**	248 ± 29
Feed reader	355	108	30.42%	**1428 ± 235**	290	30	10.34%	122 ± 59	197	76	**38.58%**	155 ± 20
Premium fact	32	23	**71.88%**	25 ± 2	33	5	15.15%	6 ± 1	92	51	55.43%	67 ± 6
Humour me	277	185	**66.79%**	**664 ± 50**	121	24	19.83%	35 ± 9	75	26	34.67%	32 ± 5
Multistream audio	263	40	15.21%	**127 ± 42**	109	17	15.60%	24 ± 7	40	20	**50.00%**	23 ± 2
Premium hello word	177	70	39.55%	* **130 ± 16**	174	32	18.39%	58 ± 16	129	79	**61.24%**	***118 ± 10**
Voice developer	349	106	30.37%	**1162 ± 191**	265	58	21.89%	186 ± 42	182	121	**66.48%**	230 ± 17
Tasty beverage	46	20	43.48%	23 ± 3	159	28	17.61%	* **48 ± 14**	89	47	**52.81%**	* **61 ± 6**
Weather wear	259	219	**84.56%**	**669 ± 40**	146	49	33.56%	79 ± 12	83	57	68.67%	73 ± 5
Next prayer	42	21	50.00%	* **24 ± 2**	32	4	12.50%	5 ± 1	28	21	**75.00%**	* **23 ± 2**
Math facts	40	32	**80.00%**	**35 ± 5**	44	14	31.82%	16 ± 2	33	20	60.61%	22 ± 2
Forget me not	69	25	36.23%	30 ± 4	48	19	39.58%	22 ± 3	33	36	**91.67%**	**37 ± 2**
Charity roster	112	81	**72.32%**	**115 ± 8**	52	18	34.62%	21 ± 3	74	52	70.27%	64 ± 5
Cyclist assistant	111	91	81.98%	**127 ± 8**	90	28	31.11%	36 ± 6	44	39	**88.64%**	44 ± 3
Internet archive	156	66	42.31%	**111 ± 13**	133	30	22.56%	46 ± 10	84	62	**73.81%**	78 ± 5
My barkeeper	373	178	47.72%	**5910 ± 619**	287	106	36.93%	416 ± 56	252	165	**65.48%**	111 ± 54
Girls volleyball	14	10	71.43%	11 ± 1	117	6	5.13%	12 ± 16	67	51	**76.12%**	**62 ± 4**
Boys basketball	6	3	50.00%	3 ± 2	64	5	7.81%	7 ± 2	40	32	**80.00%**	**35 ± 2**
Overall	6289	2752	44.7%	**16331 ± 1957**	5123	1112	21.1%	2337 ± 538	3475	2039	**58.7%**	2785 ± 292

6 EMPIRICAL STUDY RESULTS

This section reports the analysis of the results for the three research questions of our study.

6.1 RQ$_1$: Paraphrase Equivalence

The results of the analysis conducted for RQ$_1$ are described in Table 5. It is worth noting that since VUI-UPSET generates a higher number of paraphrases, the size of the evaluated sample is naturally larger because of the methodology we used to select it. The first clear result we obtained is that VUI-UPSET generates a higher percentage of imperatively equivalent paraphrases (44.7%) compared to GRSBV (21.1%), while ChatGPT (58.7%) generates the highest percentage of imperatively equivalent paraphrases. When using statistical hypothesis tests, we observe a significant difference between VUI-UPSET and GRSBV: In this case, the *p-value* is lower than 0.001 with a *large* effect size (δ = 0.77). The boxplots in Fig. 4 visually confirm the difference. Despite the greater number of imperatively equivalent paraphrases generated there are some very common paraphrases that GRSBV is unable to reproduce. For example, they do not

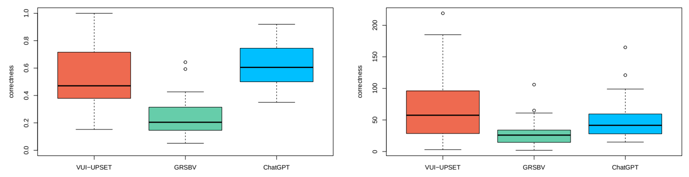

Fig. 4. Distribution of percentages (left) and absolute numbers (right) of imperatively equivalent paraphrases over the 40 skills analyzed.

generate paraphrases with personal pronoun variations (*e.g., "give me a space fact"* $\longrightarrow$ *"give her a space fact"*). Given the higher number of generated paraphrases and the acceptable percentage of imperatively equivalent paraphrases, VUI-UPSET achieves the best results for most of the skills (36 out of 40, 4 of which non-significantly different from the results achieved by GRSBV). It is important to note that the number of the seed sentences can affect the performance of VUI-UPSET in terms of percentage of imperatively equivalent generated paraphrases.

When comparing VUI-UPSET and ChatGPT by analyzing the percentage of imperatively equivalent paraphrases generated we obtain an adjusted *p-value* of 0.032, with a *small* effect size (δ = -0.28). The absolute number of paraphrases generated by VUI-UPSET is 6,289, while ChatGPT generates 3,475 paraphrases, of which we obtain 2,752 and 2,039 imperatively equivalent paraphrases, respectively. We observed that ChatGPT behaves in a very different way compared to the other approaches. It often modifies the sentence structure from direct commands to more complex sentences, while the others keep (by design) the sentence structure unaltered. For example, from the seed sentence "don't want humour me" we obtain "Being funny with me is not something I want." In addition, based on the manual validation of the sample of ChatGPT generated paraphrases, we observed that all paraphrases are grammatically correct and relevant to the original sentence, differently from the other approaches. However, when transforming a direct command into a more discursive adding a specific detail, imperatively equivalence is often lost, as observed in the results. For example based on the seed sentence "tell me a fact" (used to ask a fact) we obtain "Do you have any fascinating facts to share?".

Another important point to highlight concerns the number of paraphrases generated by ChatGPT. The number of generated paraphrases is a result of the initial prompt, in which we intentionally refrained from specifying a fixed number of expected paraphrases. Instead, our prompt aimed to generate all paraphrases equivalent to the initial seed according to ChatGPT itself. With careful engineering of the prompt and more refined post-processing of the generated paraphrases, it is plausible to assume that ChatGPT would be able to generate a higher number of imperatively equivalent paraphrases. However, deciding how to engineer the prompt involves the evaluation of a trade-off, as over-extension may introduce non-equivalent paraphrases. Consequently, we consider this as part of our future work.

Answer to RQ$_1$. VUI-UPSET achieves better results than GRSBV. ChatGPT achieves a higher percentage of imperatively equivalent paraphrases compared to VUI-UPSET, but VUI-UPSET generates a higher absolute number of imperatively equivalent paraphrases than one ChatGPT invocation.

6.2 RQ$_2$: Paraphrase Capability of Finding Bugs

We report the results of the analysis conducted for RQ$_2$ in Table 6. VUI-UPSET generates the highest absolute number of bug-revealing paraphrases compared to the two baselines for 15 skills out of 40. GRSBV, generates the highest number of bug-revealing paraphrases for 3 skills out of 40. Finally, ChatGPT obtains the best results for 13 skills out of 40. It

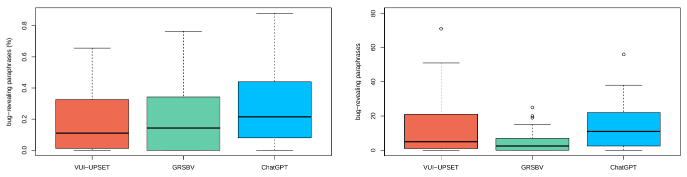

Fig. 5. Distribution of percentages (left) and absolute numbers (right) of bug-revealing paraphrases over the 40 skills analyzed.

Table 6. Comparison among VUI-UPSET and GRSBV in terms their bug-revealing capability. ∘ 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.

	RQ$_2$: Paraphrase Capability of Finding Bugs											
	VUI-UPSET				**GRSBV**				**ChatGPT**			
Skill	**#equiv∘**	**#bugs∘**	**%bugs**	**#bugs★**	**#equiv∘**	**#bugs∘**	**%bugs**	**#bugs★**	**#equiv∘**	**# bugs∘**	**%bugs**	**#bugs★**
llama facts	32	21	65.63%	23 ± 2	9	6	66.67%	7 ± 1	26	23	**88.46%**	**29 ± 2**
Happy birthday	47	13	**27.66%**	**23 ± 4**	21	3	14.29%	5 ± 2	23	2	8.70%	3 ± 1
City guide	17	0	0.00%	0 ± 0	7	1	14.28%	2 ± 1	26	18	**69.23%**	**24 ± 2**
Quiz game	64	0	0.00%	3 ± 3	32	0	0.00%	1 ± 1	26	0	0.00%	1 ± 1
Scheduling	54	5	**9.26%**	8 ± 3	46	4	8.70%	7 ± 3	53	4	7.55%	6 ± 2
Name the show	84	37	44.08%	**56 ± 6**	28	10	35.71%	19 ± 3	52	26	**50.00%**	39 ± 4
Berry bash	97	0	0.00%	8 ± 8	36	3	8.33%	7 ± 4	99	16	**16.16%**	**25 ± 8**
History facts	12	1	8.33%	1 ± 0	2	0	0.00%	0 ± 0	27	11	**40.74%**	**13 ± 2**
Particle cloud	53	3	5.66%	8 ± 5	27	2	7.41%	5 ± 3	79	6	**7.59%**	10 ± 4
Greeting sender	32	5	15.63%	12 ± 4	14	6	**42.86%**	**20 ± 2**	40	8	20.00%	13 ± 3
Button trivia	103	14	13.59%	* **37 ± 13**	44	0	0.00%	3 ± 3	51	17	**33.33%**	* **27 ± 4**
Video app	23	1	4.35%	2 ± 1	17	13	**76.47%**	**21 ± 1**	29	0	0.00%	1 ± 1
Salesforce	134	50	37.31%	**229 ± 31**	15	6	**40.00%**	9 ± 1	40	3	7.50%	5 ± 2
Store Amazon pay	61	25	**40.98%**	**125 ± 15**	29	0	0.00%	2 ± 2	90	33	36.67%	57 ± 8
Timers	95	35	**36.84%**	**89 ± 12**	26	0	0.00%	2 ± 2	36	2	5.56%	3 ± 1
Pocket	86	1	**1.16%**	15 ± 14	44	0	0.00%	2 ± 2	47	0	0.00%	1 ± 1
Piano player	50	0	0.00%	2 ± 2	10	0	0.00%	1 ± 1	29	2	**6.90%**	3 ± 1
Week calendar	117	0	0.00%	15 ± 15	61	0	0.00%	4 ± 4	42	0	0.00%	1 ± 1
Crash notification	46	21	45.65%	27 ± 3	37	15	40.54%	* **30 ± 4**	40	26	**65.00%**	* **34 ± 3**
Memory	64	39	**60.94%**	**61 ± 5**	26	6	23.07%	8 ± 2	15	8	53.33%	10 ± 1
Radio	18	1	5.56%	2 ± 1	12	1	4.76%	3 ± 2	41	27	**65.85%**	**37 ± 3**
Trading teacher	47	7	14.89%	9 ± 2	22	7	31.82%	10 ± 2	77	37	**48.05%**	**63 ± 7**
College finder	138	51	36.96%	**975 ± 132**	65	20	30.77%	86 ± 14	99	38	**38.38%**	106 ± 14
Feed reader	108	18	16.67%	**277 ± 83**	30	7	**23.33%**	42 ± 9	76	11	14.47%	25 ± 9
Premium fact	23	3	13.04%	4 ± 1	5	1	20.00%	1 ± 0	51	24	**47.06%**	**34 ± 4**
Humour me	185	5	2.70%	**30 ± 25**	24	1	**4.17%**	3 ± 2	32	26	3.85%	2 ± 1
Multistream audio	40	7	**17.50%**	**30 ± 8**	14	2	11.76%	4 ± 2	20	0	0.00%	1 ± 1
Premium hello word	70	6	8.57%	13 ± 7	24	6	18.75%	14 ± 4	79	56	**70.89%**	**91 ± 6**
Voice developer	106	30	28.30%	**383 ± 68**	58	19	**32.76%**	75 ± 11	121	12	9.92%	25 ± 12
Tasty beverage	20	2	10.00%	3 ± 1	24	12	**42.86%**	* **26 ± 3**	47	19	40.43%	* **27 ± 3**
Weather wear	219	18	8.22%	**58 ± 35**	49	9	**18.37%**	17 ± 5	57	9	15.79%	13 ± 4
Next prayer	21	0	0.00%	1 ± 1	4	2	**50.00%**	3 ± 0	21	5	23.81%	**6 ± 1**
Math facts	32	7	21.88%	9 ± 2	14	6	**42.86%**	8 ± 1	20	7	35.00%	8 ± 1
Forget me not	25	3	**12.00%**	**4 ± 1**	19	0	0.00%	1 ± 1	33	0	0.00%	1 ± 1
Charity roster	81	1	1.23%	4 ± 3	18	0	0.00%	1 ± 1	52	21	**40.38%**	**28 ± 3**
Cyclist assistant	91	0	0.00%	3 ± 3	28	0	0.00%	1 ± 1	39	7	**17.95%**	**9 ± 2**
Internet archive	66	26	39.39%	**49 ± 6**	25	14	**46.67%**	26 ± 3	62	14	22.58%	19 ± 4
My barkeeper	178	71	**39.89%**	**2604 ± 326**	200	25	23.58%	111 ± 24	165	30	18.18%	22 ± 8
Girls volleyball	3	0	0.00%	0 ± 0	6	0	0.00%	0 ± 0	51	21	**41.18%**	**27 ± 3**
Boys basketball	3	0	0.00%	0 ± 0	5	0	0.00%	0 ± 0	32	15	**46.88%**	**18 ± 2**
Overall	2752	527	19.2%	**5202 ± 851**	1112	207	18.6%	587 ± 127	2039	559	**27.4%**	867 ± 141

is worth noting, however, that ChatGPT is significantly better than VUI-UPSET for two additional skills (*i.e.,* Crash notification and Tasty beverage), for which it is not the best approach overall (it is not better than GRSBV). Thus, in this regard, the two approaches achieve comparable results. If we sum the results obtained over the 40 skills, we find that

VUI-UPSET generates (over the whole population) between 4,351 and 6,053 bug-revealing paraphrases, GRSBV between 460 and 714, and ChatGPT between 726 and 1,008. This shows that the absolute number of bug-revealing paraphrases generated by VUI-UPSET at least four times higher than the number of bug-revealing paraphrases generated by the best baseline (ChatGPT).

VUI-UPSET and GRSBV achieve similar results in terms of percentage of bugs found (overall, 19.2% and 18.6%, respectively), while ChatGPT achieves much better results (overall, 27.4%). However, when running statistical hypothesis tests, we can not reject any null hypothesis (as also confirmed by the boxplots in Fig. 5). Indeed, in this case, the p-values obtained is higher than 0.05 (never lower than 0.55) when comparing VUI-UPSET with GRSBV and it is 0.06 when comparing VUI-UPSET with ChatGPT. Similarly, the effect size is negligible in the comparison with GRSBV, while in the comparison with ChatGPT it is *small* (δ = -0.27). Therefore, a first conclusion is that none of the compared approaches inherently generates better paraphrases for identifying bugs, even if ChatGPT generally achieves slightly better results. Thus, intuitively, the larger the absolute number of generated paraphrases, the better (*i.e.,* the higher the number of generated bug-revealing paraphrases).

There are skills for which VUI-UPSET works particularly well. For example, it generates 125 ± 15 bug-revealing paraphrases for the skill Store Amazon pay, which is remarkable if we compare it to the GRSBV where we obtain 2 ± 2. ChatGPT, instead, works reasonably well for the same skill, even if with worse results (57 ± 8 bug-revealing paraphrases). At the same time, we can observe that, for some skills (*e.g.,* Quiz game, Piano player, and Week calendar), we could not find any bug-revealing paraphrases generated by the any approach, despite the high number of imperatively equivalent paraphrases generated (*e.g.,* VUI-UPSET generated 76 ± 10 imperatively equivalent paraphrases). As for VUI-UPSET and GRSBV, this probably happens because the synonyms selected, regardless of their number, are clearly similar to the words used in the original seeds. For example, in the "Piano player" skill, the word "scale" has "musical scale" as a valid synonym, and the word "lesson" is often replaced with "object lesson". Every time such substitutions happen, Alexa is still able to compensate for such small variations. Remarkably, this happens also for ChatGPT which works in a completely different way. In this case, we can observe that ChatGPT does not always act only on synonyms but generates paraphrases with minor changes from the seed. For example, in the skill Week calendar the generated paraphrases retain the main words of the seed that trigger the appropriate intent by negligible changes in the sentence structure (*e.g.*,, it adds 'please' to the end of the sentence or transforms the direct command into a question). Again, we can see that Alexa is able to handle such small variations.

An example of bug found through VUI-UPSET is the following. For the *Particle cloud* skill, given the seed sentence *"register my date of birth,"* VUI-UPSET generates, among the other paraphrases, *"register our date of birth."* Such a paraphrase is imperatively equivalent to the seed sentence but, when tested in the Alexa Developer Console, it does not allow to obtain the same result (*i.e.,* the test case fails).

In summary, while VUI-UPSET achieves better results in terms of absolute number of bug-revealing paraphrases generated, it is worth highlighting that ChatGPT is able to generate a higher percentage of bug-revealing paraphrases, thus making the execution of such tests more cost-effective than the execution of the tests generated with VUI-UPSET. Also, ChatGPT could be further improved in several ways (*e.g.,* through prompt engineering or by asking to generate a larger set of paraphrases): We relied on a simple version of the ChatGPT baseline which deserves more fine-tuning and investigation in the future. Indeed, it showed to be competitive (and potentially better) than a technique specifically designed for this task, namely VUI-UPSET. We discuss more the strengths and weaknesses of the two approaches in Section 6.4.

Answer to RQ$_2$. ChatGPT generates a higher percentage of bug-revealing paraphrases than VUI-UPSET and GRSBV, while VUI-UPSET generates the highest total absolute number of such paraphrases. The two approaches generate a significantly higher number of paraphrases than the other for different skills, showing a strong complementarity.

6.3 RQ$_3$: Bug Reduction

We report the results of the analyses conducted for answering RQ$_3$ in Table 7. In detail, we can observe the results of the three evaluations conducted regarding the inclusion in the vocal interaction model of the skill of (i) imperatively equivalent paraphrases that resulted in bugs in RQ$_2$, (ii) all imperatively equivalent paraphrases, and (iii) by performing a four-fold cross validation. For all the evaluations, we first verified that the inclusion of new paraphrases within the skill did not alter the initial behavior. We observed that in all the scenarios never alters the correct functioning of the original utterances included in the skill (*i.e.,* 0 failed original utterances). We report below the results of the three evaluation separately. Note that the results obtained by the two approaches for this evaluation should not be used to compare them since the test sets are different.

Including only bug-revealing paraphrases. In the first evaluation, despite the inclusion of bug-revealing paraphrases in the voice interaction models, we can bring the number of failed tests to 0 for 33 out of 40 skills. We can still find failing test cases in 7 skills when using VUI-UPSET. The *"Button trivia"* and *"Voice developer"* skills have a higher percentage of failing tests (32.0% and 32.1%, respectively) compared to the base skill (13.6% and 28.3%, respectively, as shown in the results of RQ$_2$), while the opposite is true for all the other skills. An example of failing test is the paraphrase "let's initiate the gamey" (generated for the seed "let's start the game"). While the correct intent is activated when using the original set of utterances defined by the developers, after editing the voice interaction model by inserting our bug-revealing paraphrases, the expected intent is no longer activated. We obtain a similar result when using ChatGPT. In this case, we can bring the number of failed tests to 0 for 30 skills out of 40, while we can still find failing test cases in 10 skills. The *"Particle Cloud"* and *"Premium Hello Word"* skills have a higher percentage of failing tests (17.7% and 13.2%, respectively) compared to the other skill. For *"Premium Hello Word"*, we found a significantly lower percentage of bugs compared to the original skill (13.2% instead of 70.89%).

For both the approaches, we can observe, in general, that most of the skills that have a high percentage of bugs in their original version tend now to have an extremely low percentage of bugs when augmented with bug-revealing paraphrases. For example, for the skill *"Crash notification"*, 21 paraphrases generated by VUI-UPSET failed when using the original skill (40.54%), while the improved version we observed only one failure. This happens for the paraphrase *"missing replay,"* which is supposed to activate the intent *TimeoutIntent*, but it activates the intent *InvalidResponseIntent*, probably because it is very similar to the utterance "missing time-out response" that activates such an intent in the VIM. For the skill *"Trading teacher"* the only bug is encountered with the phrase *"I want a return"*. In this case, the activation intent fails and no response is output because the skill does not understand the request and is unable to associate it with any intent (not even a similar intent). After conducting a thorough analysis of failed inputs across all skills, we have noticed that including certain paraphrases in a skill can lead to ambiguity in activating the intended action. Based on this observation, we conclude that defining similar intents within the same skill can pose problems. Therefore, when creating a skill, developers should clearly and explicitly define the various functionalities that the skill is capable of offering and the scope of each of them, by precisely delimiting the boundaries.

Including all the imperatively equivalent paraphrases. When including all the imperatively equivalent utterances in the voice interaction models of the skills (central part of Table 7), we observed that almost all the tests

Table 7. Results of RQ_3 for the two analyses conducted (absolute number of failing tests, percentage of failing tests, and estimated absolute number of failing tests on the whole population). We only report the results of the tests augmented with the generated paraphrases since the tests conducted with the seed utterances always pass.

	RQ_3: Skill Improvement											
	Including bug-revealing paraphrases						**Including all the equivalent paraphrases**					
	VUI-UPSET			**ChatGPT**			**VUI-UPSET**			**ChatGPT**		
Skill	**Abs.**	**Rel.**	**Est.**	**Abs.**	**Rel.**	**Est.**	**Abs.**	**Rel.**	**Est.**	**Abs.**	**Rel.**	**Est.**
llama f.	0	0.00%	1 ± 1	0	0.00%	1 ± 1	0	0.00%	1 ± 1	0	0.00%	1 ± 1
Happy b.	0	0.00%	2 ± 2	0	0.00%	1 ± 1	0	0.00%	2 ± 2	0	0.00%	1 ± 1
City g.	0	0.00%	0 ± 0	0	0.00%	1 ± 1	0	0.00%	0 ± 0	0	0.00%	1 ± 1
Quiz g.	0	0.00%	3 ± 3	0	0.00%	1 ± 1	0	0.00%	3 ± 3	0	0.00%	1 ± 1
Sched.	0	0.00%	2 ± 2	0	0.00%	2 ± 2	0	0.00%	2 ± 2	0	0.00%	2 ± 2
Name the s.	0	0.00%	3 ± 3	0	0.00%	2 ± 2	0	0.00%	3 ± 3	0	0.00%	4 ± 4
Berry b.	0	0.00%	8 ± 8	0	0.00%	4 ± 4	0	0.00%	8 ± 8	0	0.00%	1 ± 1
History f.	0	0.00%	0 ± 0	0	0.00%	1 ± 1	0	0.00%	0 ± 0	0	0.00%	3 ± 3
Particle c.	0	0.00%	3 ± 3	**17**	**17.71%**	**21 ± 4**	0	0.00%	3 ± 3	0	0.00%	2 ± 2
Greet. s.	**2**	**6.25%**	**5 ± 3**	**4**	**3.64%**	**5 ± 1**	0	0.00%	2 ± 2	0	0.00%	2 ± 2
Button t.	**33**	**32.04%**	**86 ± 13**	**4**	**3.51%**	**5 ± 1**	0	0.00%	7 ± 7	0	0.00%	1 ± 1
Video app	0	0.00%	1 ± 1	0	0.00%	1 ± 1	0	0.00%	1 ± 1	0	0.00%	1 ± 1
Salesf.	0	0.00%	15 ± 15	**4**	**6.25%**	**5 ± 1**	0	0.00%	15 ± 15	0	0.00%	4 ± 4
St. Amaz. p.	0	0.00%	8 ± 8	0	0.00%	4 ± 4	0	0.00%	8 ± 8	0	0.00%	1 ± 1
Timers	0	0.00%	6 ± 6	0	0.00%	1 ± 1	0	0.00%	6 ± 6	0	0.00%	1 ± 1
Pocket	0	0.00%	12 ± 12	0	0.00%	2 ± 2	0	0.00%	12 ± 12	0	0.00%	2 ± 2
Piano p.	0	0.00%	2 ± 2	0	0.00%	1 ± 1	0	0.00%	2 ± 2	0	0.00%	1 ± 1
Week c.	0	0.00%	15 ± 15	0	0.00%	1 ± 1	0	0.00%	15 ± 15	0	0.00%	1 ± 1
Crash n.	**1**	**2.17%**	**3 ± 2**	**1**	**1.52%**	2 ± 1	0	0.00%	1 ± 1	0	0.00%	1 ± 1
Memory	0	0.00%	3 ± 3	0	0.00%	0 ± 0	0	0.00%	3 ± 3	0	0.00%	0 ± 0
Radio	0	0.00%	1 ± 1	0	0.00%	1 ± 1	0	0.00%	1 ± 1	0	0.00%	1 ± 1
Trading t.	**1**	**2.13%**	**3 ± 2**	**1**	**0.74%**	**4 ± 3**	0	0.00%	2 ± 2	0	0.00%	3 ± 3
College f.	0	0.00%	66 ± 66	**2**	**0.87%**	**9 ± 7**	0	0.00%	66 ± 66	**2**	**0.87%**	**9 ± 7**
Feed read.	0	0.00%	42 ± 42	0	0.00%	4 ± 4	0	0.00%	42 ± 42	0	0.00%	4 ± 4
Prem. fact	0	0.00%	1 ± 1	0	0.00%	2 ± 2	0	0.00%	1 ± 1	0	0.00%	2 ± 2
Humour me	0	0.00%	18 ± 18	**1**	**1.33%**	**2 ± 1**	0	0.00%	18 ± 18	**1**	**1.33%**	**2 ± 1**
M. audio	0	0.00%	4 ± 4	0	0.00%	1 ± 1	0	0.00%	4 ± 4	0	0.00%	1 ± 1
P. Hello w.	**4**	**5.71%**	**10 ± 6**	**17**	**13.18%**	**20 ± 3**	0	0.00%	4 ± 4	0	0.00%	3 ± 3
Voice dev.	**34**	**32.08%**	**434 ± 68**	**12**	**6.59%**	20 ± 8	0	0.00%	34 ± 34	**12**	**6.59%**	20 ± 8
Tasty bev.	0	0.00%	1 ± 1	0	0.00%	2 ± 2	0	0.00%	1 ± 1	0	0.00%	2 ± 2
Weath. wear	0	0.00%	18 ± 18	0	0.00%	2 ± 2	0	0.00%	18 ± 18	0	0.00%	2 ± 2
Next pray.	0	0.00%	1 ± 1	0	0.00%	1 ± 1	0	0.00%	1 ± 1	0	0.00%	1 ± 1
Math facts	0	0.00%	1 ± 1	0	0.00%	1 ± 1	0	0.00%	1 ± 1	0	0.00%	1 ± 1
Forg. me not	0	0.00%	1 ± 1	0	0.00%	1 ± 1	0	0.00%	1 ± 1	0	0.00%	1 ± 1
Charity r.	0	0.00%	3 ± 3	0	0.00%	2 ± 2	0	0.00%	3 ± 3	0	0.00%	2 ± 2
Cyc. as.	0	0.00%	3 ± 3	0	0.00%	1 ± 1	0	0.00%	3 ± 3	0	0.00%	1 ± 1
Internet a.	0	0.00%	3 ± 3	0	0.00%	2 ± 2	0	0.00%	3 ± 3	0	0.00%	2 ± 2
My bark.	**2**	**1.12%**	**201 ± 199**	0	0.00%	1 ± 1	**1**	**0.56%**	**182 ± 181**	0	0.00%	1 ± 1
Girls voll.	0	0.00%	0 ± 0	0	0.00%	2 ± 2	0	0.00%	0 ± 0	0	0.00%	2 ± 2
Boys bask.	0	0.00%	0 ± 0	0	0.00%	1 ± 1	0	0.00%	0 ± 0			
Overall	77	2.80%	989 ± 540	63	1.80%	140 ± 77	1	0.04%	479 ± 478	15	4.30%	92 ± 77

generated with VUI-UPSET pass, except for one in the "My barkeeper" skill. In this case, the paraphrase *"please add to favourite"* is supposed to activate the intent *"AddToFavoritesIntent"*, while it activates the *"IngredientIntent"*. The analysis of the voice interaction model shows that the two intents are very similar, so the addition of this paraphrase creates ambiguity in the activation of the correct intent. In detail, in the *"IngredientIntent"* it creates ambiguity with seed "please add to list ingredient". Also, considering the example given in RQ_2, we can observe that there are no more failing tests in the skill *"button trivia"* in this second evaluation. As for ChatGPT, surprisingly, we observe that some tests still fail for three skills. The one for which this is more evident is *Voice developer*, for which including all the imperatively equivalent paraphrases does not provide any benefit in terms of additional tests passed compared to the previous analysis.This occurs because the paraphrases generated in this context by ChatGPT are ambiguous for the intents in the skill. Specifically different intents have similar seeds and this creates problems in the activation request. Ambiguity problems occur when similar paraphrases are added in different intents, and this may cause the intent to be activated incorrectly. However, this issue might affect the skill from its design: If the activation phrases for the intents identified by the developer are very similar to each other, users might have a hard time at selecting one or the other when they do not use the exact sentence meant by the developers. Having similar paraphrases for two different seed sentences should raise a warning for developers, who might use this information to more clearly delimit the scope of the ambiguous sentences.

For example, the paraphrase "Switch to a dissimilar vocal sound" should activate the *PollyVoiceIntent* from the seed "use a different voice". What happens instead is that: it activates the *SSMLIntent* where we have different seeds that can create ambiguity for example "play a sound effect". We conjecture that this is due to the fact that ChatGPT tends to change more deeply the structure of the paraphrases: Some of the imperatively equivalent paraphrases might be very similar to others or even imperatively equivalent to them, thus the confusion. For example from the seed "what are the latest updates" ChatGPT generates "Please share the newest updates with me" which is very similar to the seed "share the results" present in another intent in the same skill.

Four-fold cross validation. We conducted four evaluations for each skill by grouping the entire collection of imperatively equivalent paraphrases into four folds. Then, we used three of these folds (constituting 75% of the instances) to improve the voice interaction model of the skill. The remaining folds (25%) was used to evaluate the performance of the newly updated version of the skill. We report the results of each iteration of the four-fold cross validation in Table 8. In detail, the table features columns corresponding to each assessed fold, considering both the VUI-UPSET and ChatGPT. Beyond the individual evaluations for each fold, we also provide a comprehensive summary of the four-fold cross-validation, which is derived from aggregating the results obtained across the four individual evaluations. We can observe that when we do not include the same paraphrases both in the VIM and in the test set, most of the skills still improve in terms of absolute number and percentage of observed bugs. As for VUI-UPSET, we observed that 25 out of 40 skills have a lower percentage of bugs compared to the original skills, 11 do not change (for most of them we did not find any bug in both the cases) while only 4 out of 40 have a higher percentage of bugs. The improvement is particularly high for some skills. For example, *Ilama facts* improves by ~65 percentage points (we found no bugs when using the improved versions). We also have remarkably good results for College finder, for which we reduce the percentage of failed tests by ~17 percentage points but, most importantly, we estimate that the improved version has a reduced absolute number of bugs on the whole population of generated paraphrases between 195 and 723 bugs. On the other hand, one of the skills (*Charity roster*) achieves remarkably worse results (percentage of failing tests increased by ~42 percentage points). This happens because the inclusion of specific paraphrases within the voice interaction model (which differs from those tested) creates ambiguity in the activation of the expected intent. For example, the paraphrase "i want to discover more about this charity" whose seed is "I want to learn more about this charity" expects the activation of the intent *LearnMoreIntent*. However it activates the intent DonateIntent where we have the sentence "I want info to contribute to this charity". Therefore we assume it is a skill construction problem. We obtain a very

Table 8. Results of RQ$_3$ for the analyses conducted estimated absolute number of failing tests on single iteration in 4-fold cross validation.

	RQ$_3$: Skill Improvement Four-fold cross validation																					
	VUI-UPSET											ChatGPT										
	Fold 1		Fold 2		Fold 3		Fold 4		Overall			Fold 1		Fold 2		Fold 3		Fold 4		Overall		
Skill	**Abs.**	**Rel.**	**Abs.**	**Rel.**	**Abs.**	**Rel.**	**Abs.**	**Rel.**	**Abs.**	**Rel.**	**Est.**	**Abs.**	**Rel.**	**Abs.**	**Rel.**	**Abs.**	**Rel.**	**Abs.**	**Rel.**	**Abs.**	**Rel.**	**Est.**
llama facts	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	1 ± 1	0	0,00%	0	0,00%	1	3,85%	0	0,00%	1	3,85%	2 ± 1
Happy birthday	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	2 ± 2	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	1 ± 1
City guide	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0 ± 0	0	0,00%	1	3,85%	1	3,85%	1	3,85%	3	11,54%	4 ± 1
Quiz game	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	3 ± 3	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	1 ± 1
Scheduling	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	2 ± 2	0	0,00%	1	1,89%	0	0,00%	0	0,00%	1	1,89%	3 ± 2
Name the show	2	2,38%	0	0,00%	0	0,00%	1	1,19%	3	3,57%	7 ± 4	3	5,77%	5	9,62%	2	3,85%	0	0,00%	10	19,23%	19 ± 0
Berry bash	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	8 ± 8	1	1,01%	1	1,01%	0	0,00%	0	0,00%	2	2,02%	7 ± 5
History facts	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0 ± 0	0	0,00%	1	3,70%	0	0,00%	0	0,00%	1	3,70%	2 ± 1
Patricle cloud	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	3 ± 3	1	1,27%	1	1,27%	2	2,53%	0	0,00%	4	5,06%	8 ± 4
Greeting sender	1	3,13%	1	3,13%	0	0,00%	0	0,00%	2	6,25%	5 ± 3	1	2,50%	0	0,00%	0	0,00%	0	0,00%	1	2,50%	3 ± 2
Button trivia	8	7,77%	6	5,83%	0	0,00%	0	0,00%	14	13,59%	37 ± 13	0	0,00%	3	5,88%	2	3,92%	0	0,00%	5	9,80%	13 ± 0
Video app	0	0,00%	1	4,35%	0	0,00%	0	0,00%	1	4,35%	2 ± 1	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	1 ± 1
Salesforce	2	1,49%	0	0,00%	0	0,00%	0	0,00%	2	1,49%	21 ± 19	2	5,00%	1	2,50%	2	5,00%	0	0,00%	5	12,50%	28 ± 18
Store Amazon pay	2	3,28%	0	0,00%	1	1,64%	1	1,64%	4	6,56%	20 ± 15	1	1,11%	1	1,11%	0	0,00%	2	2,22%	4	4,44%	9 ± 5
Timers	2	2,11%	0	0,00%	0	0,00%	0	0,00%	2	2,11%	10 ± 8	0	0,00%	4	11,11%	1	2,78%	1	2,78%	6	16,67%	19 ± 9
Pocket	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	12 ± 12	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	2 ± 2
Piano player	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	2 ± 2	0	0,00%	1	3,45%	0	0,00%	0	0,00%	1	3,45%	2 ± 1
Week calendar	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	15 ± 15	0	0,00%	0	0,00%	0	0,00%	1	2,38%	1	2,38%	2 ± 1
Crash notification	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	1 ± 1	0	0,00%	0	0,00%	1	2,50%	3	7,50%	4	10,00%	6 ± 2
Memory	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	3 ± 3	0	0,00%	2	13,33%	1	6,67%	0	0,00%	3	20,00%	10 ± 5
Radio	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	1 ± 1	4	9,76%	0	0,00%	2	4,88%	1	2,44%	7	17,07%	10 ± 3
Trading	2	4,26%	0	0,00%	0	0,00%	0	0,00%	2	4,26%	4 ± 2	7	9,09%	6	7,79%	0	0,00%	7	9,09%	20	25,97%	30 ± 10
Collage	5	3,62%	12	8,70%	4	2,90%	6	4,35%	27	19,57%	516 ± 132	6	6,06%	9	9,09%	12	12,12%	11	11,11%	38	38,38%	500 ± 380
Feed-reader	13	12,04%	1	0,93%	0	0,00%	0	0,00%	14	12,96%	215 ± 83	0	0,00%	0	0,00%	4	5,26%	2	2,63%	6	7,89%	35 ± 13
Premium fact	0	0,00%	0	0,00%	2	8,70%	2	8,70%	4	17,39%	5 ± 1	2	3,92%	0	0,00%	1	1,96%	1	1,96%	4	7,84%	7 ± 3
Humor me	1	0,54%	0	0,00%	0	0,00%	0	0,00%	1	0,54%	20 ± 19	0	0,00%	0	0,00%	2	7,69%	0	0,00%	2	7,69%	12 ± 7
Multistream audio	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	4 ± 4	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	1 ± 1
Premium hello world	4	5,71%	1	1,43%	2	2,86%	1	1,43%	8	11,43%	17 ± 7	4	5,06%	9	11,39%	3	3,80%	6	7,59%	22	27,85%	38 ± 4
Voice developer	8	7,55%	13	12,26%	6	5,66%	10	9,43%	37	34,91%	472 ± 68	1	0,83%	2	1,65%	7	5,79%	2	1,65%	12	9,92%	52 ± 15
Tasty beverage	0	0,00%	0	0,00%	1	5,00%	0	0,00%	1	5,00%	2 ± 1	4	8,51%	3	6,38%	0	0,00%	4	8,51%	11	23,40%	15 ± 4
Weather wear	4	1,83%	0	0,00%	0	0,00%	0	0,00%	4	1,83%	26 ± 22	1	1,75%	3	5,26%	0	0,00%	1	1,75%	5	8,77%	19 ± 8
Next prayer	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	1 ± 1	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	1 ± 1
Math facts	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	1 ± 1	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	1 ± 1
Forget me not	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	1 ± 1	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	1 ± 1
Charity roster	5	6,17%	15	18,52%	9	11,11%	6	7,41%	35	43,21%	53 ± 6	5	9,62%	0	0,00%	2	3,85%	1	1,92%	8	15,38%	13 ± 1
Cyclist assistant	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	3 ± 3	1	2,56%	0	0,00%	0	0,00%	0	0,00%	1	2,56%	2 ± 1
Internet archive	0	0,00%	0	0,00%	0	0,00%	4	6,06%	4	6,06%	9 ± 5	0	0,00%	0	0,00%	1	1,61%	0	0,00%	1	1,61%	3 ± 2
My barkeeper	11	6,18%	5	2,81%	12	6,74%	24	13,48%	52	29,21%	1907 ± 326	6	3,64%	7	4,24%	0	0,00%	0	0,00%	13	7,88%	98 ± 90
Girls volleyball	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0 ± 0	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	2 ± 2
Boys basketball	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0	0,00%	0 ± 0	1	3,13%	0	0,00%	8	25,00%	0	0,00%	9	28,13%	11 ± 2

similar result for ChatGPT: 29 out of 40 skills have a lower percentage of bugs compared to the original skills, 7 do not change, and only 4 out of 40 have a higher percentage of bugs. Also in this case, we observed big improvements in some skills: For example, using some of the paraphrases for improving the VIM of the *Ilama facts* allows to achieve, also here, an almost bug-free version of the skill (~-85 percentage points of failing tests, with a single bug found in the improved version). Differently from what we observed for VUI-UPSET, the four cases in which the percentage of failing tests increases are milder. Still, for one of the skills (*Store Amazon pay*) we observed an increased percentage of failing test quite high (~11 percentage points).

Answer to RQ$_3$. Adding the paraphrases generated by VUI-UPSET and ChatGPT to the VIMs is always safe (*i.e.,* the skill still works with the intended basic utterances). However, adding only bug-revealing paraphrases is not sufficient for several skills, for which including *all* the valid paraphrases is preferable. Finally, adding paraphrases almost always improves the robustness of the skill when faced with utterances never seen before.

6.4 Discussion

Based on the achieved results, we highlight the differences between VUI-UPSET and ChatGPT. In addition, we identify possible future research directions in this field, and guidelines for practitioners.

6.4.1 Strengths and Weaknesses of VUI-UPSET and ChatGPT. Our results clearly show that, overall, VUI-UPSET and ChatGPT emerge as the best approaches, while GRSBV generally achieves sub-optimal results compared to them. For this reason, we report an overview of the strengths and weaknesses of VUI-UPSET and ChatGPT to provide a broader overview of our results. The first clear message coming from our study is that VUI-UPSET excels in the *absolute number* of imperatively equivalent paraphrases generated , while ChatGPT excels in the *relative number* (percentage). This result is also reflected in the number of bug-revealing paraphrases (RQ$_2$), where, again, VUI-UPSET wins in terms of absolute number while ChatGPT wins in terms of percentage. In the context of VUI testing, both the aspects are crucial. A higher absolute number of generated/bug-revealing paraphrases is good to thoroughly test the software. On the other hand, a higher percentage of generated/bug-revealing paraphrases means that the adoption of such paraphrases for testing will require less effort since developers will need to manually discard fewer paraphrases. We argue that, depending on the skill at hand, developers might prefer one quality over the other. An advantage of VUI-UPSET over ChatGPT is that the former is white-box, while the latter is black-box. VUI-UPSET can be further improved by refining its steps. This aspect is fundamental in facilitating continuous improvement, as researchers can iteratively optimize the approach to address specific weaknesses. In contrast, ChatGPT, as a Large Language Model, can only be improved by engineering the prompt used to generate the paraphrases or by updating the model. However, it must be noticed that VUI-UPSET has been "fine-tuned" via trial-and-error refinements (*e.g.,* for the similarity threshold), while we used a simple prompt for ChatGPT, which achieved remarkable results "out of the box." This suggests that further potential can be unleashed from ChatGPT (*e.g.,* via prompt engineering), potentially making it the state-of-the-art approach for this task, though more challenging to understand and control.

6.4.2 Lessons Learned for Developers. While for RQ$_1$ we can observe clear trends, *i.e.,* the results are almost the same for all the skills, this was not the case for RQ$_2$, with substantial variation among the subject skills. To analyze this phenomenon more in depth, we tried to understand how different the paraphrases generated by the two approaches (VUI-UPSET and ChatGPT) are. To do this, we compute the overlap between them by using the formula $\frac{|equivalent_{v_i} \cap equivalent_{v_j}|}{|equivalent_{v_i} \cup equivalent_{v_j}|}$. The divergence of results between the two approaches is confirmed by the observation that they do not generate shared paraphrases. In other words, the two compared approaches are highly complementary. It might be worth exploring to what extent combining such approaches allows to generate more interesting bug-revealing paraphrases.

Research Direction 1. Combining VUI-UPSET and ChatGPT might allow generating a larger number of bug-revealing paraphrases.

We also observed that a clear weak point of VUI-UPSET and GRSBV is that they sometimes find non-equivalent synomyms in the context in which they are used. This is partially due to the fact that the filtering step fails at discarding incorrect paraphrases. Despite we used a state-of-the-art approach for this step, it still proved mostly insufficient. We believe this is due to the fact that the sentences at hand, in our context, are very short, while the available models (including the one we use) work better for larger sentences.

Research Direction 2. Training models and defining approaches for checking the semantic equivalence of short sentences might be useful for supporting VUI test generation.

In other cases, such a step fails because, for words with a given PoS, synonyms with a different PoS are used. For example, the "start story" input utterance for the skill "Memory" is wrongly paraphrased into "first tale". This happens because "first" is a synonym of the noun "start." However, the verb "start" is used in this context. Filtering out the synonyms of the noun "start" would not have allowed VUI-UPSET to generate this paraphrase in the first place. We believe that this limitation, due to the wrong PoS tagging, mostly comes from the shortness of the sentences at hand.

This drawback, however, highlights another key limitation of both VUI-UPSET and GRSBV: They are constrained to the syntactic structure of the original sentence to generate paraphrases. This limitation, indeed, is not shared with ChatGPT. Large Language Models (LLMs), indeed, have proven particularly effective for handling natural language text (*e.g.,* for rephrasing text) [58]. Given the very promising results achieved by ChatGPT in our experiments, we believe it is important to further study their applicability for the generation of VUI tests.

Research Direction 3. The use of ChatGPT and LLMs in general for this task should be further explored, particularly for their capability of changing the syntactic structure of the seed sentences.

Generating paraphrases for testing VUIs is still at an early research stage, and no tool is available to practitioners. In this paper, we have shown that an advanced tool like VUI-UPSET might help them to generate bug-revealing sentences. However, due to the hard work required to manually check the semantic equivalence of the generated paraphrases, we believe VUI-UPSET might not be the best option for all the developers yet. VUI-UPSET might be particularly useful for safety- and security-critical apps (*e.g.,* for home automation or home banking, respectively) that might benefit from a thorough testing phase, even if the cost of doing so is high.

Guideline for Developers 1. We suggest using VUI-UPSET mostly for skills that require thorough testing since it might cost too much for non-critical skills. On the other hand, we recommend using ChatGPT for such skills since it generates fewer and more useful paraphrases (in relative terms).

Finally, we observed that introducing the generated utterances in the VIM of the skills we tested is always safe and including all the valid paraphrases allows achieving the best results. On the other hand, including too many paraphrases increases the maintenance costs of the app: If developers decide to change an utterance, they need to carefully check whether this interferes with other utterances, and the more the utterances the harder this job. Removing unnecessary utterances for the VIM and still make the skill pass all the tests would be the best option. Therefore, the developer must ensure that ambiguity problems related to intent activation within the skill do not occur. Specifically, problems related to ambiguity occur when similar paraphrases are added in different intents, and this can cause incorrect activation of the expected intent. However, this happens if already at the skill design stage the intents identified by the developer are very similar to each other, which is why it would be appropriate to expect activation of intents that refer to very different application content.

Specifically, obvious examples include paraphrases that vary seed pronouns by generating paraphrases with different pronouns that signal the presence of bugs within the skill. For example, starting with the seed sentence: "take note on my birthday" we get the following paraphrase: "take note on his birthday" VUI-UPSET generates the following paraphrase : "take note on his birthday" which identifies a bug.

Research Direction 4. Future research could focus on defining approaches for automatically looking for the minimal set of utterances that allows a skill to pass all the tests.

Guideline for Developers 2. Practitioners can safely add the generated sentences to the VIM, but they should remove the larger number of unnecessary utterances they can to reduce the maintenance costs of the skill.

It is important to highlight that VUIs do not have visual feedback and correction mechanisms. Humans, on the other hand, are not necessarily good at pronouncing perfect sentences and sometimes correct themselves. Realistic sentences pronounced to a VUI might include a correction (*e.g.,* "tell me the recipe for cooking a beef steak, oh, sorry, I meant ribs"). This behavior could create ambiguity for VUIs during and result in errors. In our work we did not explicitly address this problem, which should be investigated in future work.

Research Direction 5. Future research should focus on generating paraphrases for testing self-correcting input sentences.

It might be argued that ChatGPT could be improved by simply asking it to generate more paraphrases after the initial generation. From some preliminary tests we conducted, however, we observed that ChatGPT would never stop generating paraphrases when asked for more. In other words, it would be necessary to define a methodology for deciding when to stop. This is not trivial, however, and this is the reason why, in our experiments, we decided to ask ChatGPT to generate paraphrases only once, with ChatGPT in charge of deciding the number of paraphrases to generate for a given sentence. Of course, this aspect can be improved. Finding the best way to generate the optimal number of paraphrases with ChatGPT is an interesting future research direction that should be explored.

Research Direction 6. Future research should aim at finding a methodology for deciding when ChatGPT (or LLMs in general) have generated the optimal number of paraphrases.

7 THREATS TO VALIDITY

Threats to construct validity. We manually selected a sample of the seed utterances to use for generating paraphrases from the voice interaction models of the skills we considered based on the requirements of VUI-UPSET and GRSBV. Choosing different utterances might have lead to different results. Also, there is a possible subjectivity introduced during manual analysis for the results of both the RQs. This threat was mitigated by using a rigorous qualitative analysis process, as described in Section 5.

Threats to internal validity. We did not directly use the VUI provided by the skills through Alexa to run the tests, but we deployed the skills in the test environment of the ADC, and simulated the execution of paraphrases through the NLU-evaluation tool. Directly using the VUI might have allowed finding other issues (*e.g.,* related to failures in the speech recognition). However, this goes beyond the scope of our approach and of the approach we compared. Another threat is related to the fact that we needed to re-implement GRSBV since it was not available. Also, the description of some of the metrics of the filtering step in the original paper [26] were slightly ambiguous. We implemented the metrics based on our interpretation, as explained in Section 2. To foster the verifiability and the replicability of VUI-UPSET, we publicly release our implementation. Also, it is important to highlight that in VUI-UPSET, we calculated the similarity threshold, whereas here we used the default threshold (0.59). To mitigate this threat, we also tuned this threshold as in the VUI-UPSET.

We report in Table 9 the results of the tuning. The best $F_{0.5}$ *score* can be achieved using as a threshold $k = 1.3$. With it, we achieve 0.21 precision (*i.e.,* half of the generated paraphrases are imperatively equivalent) and 0.52 recall. Based on the results obtained in terms of accuracy we can see that as the threshold increases the accuracy also increases. Specifically, increasing the threshold allows us to discard a greater number of sentences that are not imperatively equivalent. Although the implementation of the default threshold 0.6 has incentivized the generation of more paraphrases, it clearly fails to produce an adequate number of them compared to VUI-UPSET. As a result, VUI-UPSET proves superior even when compared with the optimal baseline threshold.

Threats to external validity. Our results are based on 40 *Alexa* skills. Such a sample might not be representative of all the Alexa skills. We mitigated this risk by choosing different skills in terms of application domain. In our experiment,

Table 9. Metrics computed for the similarity threshold evaluation.

k	Precision	Recall	$F_{0.5}$ Score	Accuracy
0.2	0.19	1.00	0.23	0.19
0.3	0.19	1.00	0.23	0.19
0.4	0.19	0.98	0.23	0.21
0.5	0.19	0.96	0.23	0.21
0.6	0.18	0.83	0.22	0.27
0.7	0.19	0.78	0.22	0.31
0.8	0.19	0.76	0.22	0.34
0.9	0.19	0.70	0.23	0.39
1.0	0.19	0.63	0.22	0.43
1.1	0.20	0.59	0.23	0.48
1.2	0.21	0.58	0.24	0.52
1.3	**0.21**	**0.52**	**0.24**	**0.54**
1.4	0.19	0.41	0.21	0.58
1.5	0.20	0.39	0.22	0.58
1.6	0.18	0.32	0.19	0.20
1.7	0.18	0.29	0.20	0.62
1.8	0.16	0.23	0.17	0.63
1.9	0.16	0.19	0.16	0.65
2.0	0.15	0.17	0.15	0.66
2.1	0.14	0.15	0.15	0.67
2.2	0.13	0.13	0.13	0.67
2.3	0.11	0.10	0.10	0.67
2.4	0.11	0.10	0.10	0.68
2.5	0.10	0.09	0.10	0.68
2.6	0.10	0.08	0.09	0.68
2.7	0.10	0.07	0.09	0.69
2.8	0.10	0.07	0.09	0.69
2.9	0.09	0.06	0.08	0.70
3.0	0.08	0.05	0.08	0.71
3.1	0.08	0.05	0.07	0.71
3.2	0.07	0.04	0.06	0.72
3.3	0.05	0.03	0.04	0.73
3.4	0.05	0.03	0.05	0.73

we selected *Alexa* skills developed in JavaScript. However, we expect no significant differences in skills developed in other programming languages since our approach focuses on the VIM rather than on the programming logic of the skills. An additional point to note is that our results are only related to skills for English-speaking users. VUI-UPSET is implemented to work only on English sentences as well. However, it is possible to generalize VUI-UPSET for other languages as well through the use of other synonym libraries by changing the specific rules defined in the PoS analysis (*e.g.*, pronouns) and DL models specific to the desired language. Finally, it is possible that the results obtained do not generalize to other technologies, *e.g.*, Actions for Google Assistant. Indeed, the results of RQ_2 depend on how tolerant the framework is with variations of the pre-defined seed utterances, and this might change.

8 CONCLUSION

The interest in *Voice User Interface*-based apps has grown over the last years. However, the research on automated testing for VUIs is still in its infancy. In this paper, we presented VUI-UPSET, an approach for automatically generating paraphrases that allow developers to test VUIs, built upon previous research in chatbot-testing. We run a large empirical study on 40 open-source Alexa skills where we compared VUI-UPSET with an existing state-of-the-art approach for chatbot-testing, *i.e.,* GRSBV, and an LLM-based approach, *i.e.,* ChatGPT. Our results show that VUI-UPSET generates the highest absolute total number of both imperatively equivalent paraphrases and bug-revealing paraphrases, while ChatGPT is better in terms of percentage for both. When counting the number of skills for which each approach generates the highest number of bug-revealing paraphrases, we observe that VUI-UPSET and ChatGPT obtain comparable results. Finally, we checked if the generated paraphrases can be added to the VIM to improve the skills. We found that doing so is always *safe* (*i.e.,* the original utterances always work as intended). It is important to point out the promising results obtained by ChatGPT. Although VUI-UPSET outperforms ChatGPT in terms of absolute total number of bug-revealing paraphrases, we used ChatGPT "out of the box," without performing prompt tuning and without addressing other challenges it poses, such as the determination of a criterion for stopping generating paraphrases. Improving ChatGPT appears to be a promising and cost-effective research direction. Besides that, we provide several other research directions in this field and guidelines to developers who plan to use VUI-UPSET for their skills.

REFERENCES

[1] Stop Word List. https://countwordsfree.com/stopwords.

[2] "Amazon". Alexa. https://developer.amazon.com/en-US/alexa.

[3] "Amazon". Alexa Slots. https://developer.amazon.com/en-US/docs/alexa/custom-skills/slot-type-reference.html.

[4] "Amazon". Amazon Developer. https://developer.amazon.com/en/.

[5] "Amazon". Amazon official documentation. https://developer.amazon.com/en-US/docs/alexa/custom-skills/get-utterance-recommendations.html.

[6] "Amazon". NLU-evaluation tool. https://developer.amazon.com/it-IT/docs/alexa/smapi/nlu-evaluation-tool-api.html.

[7] Jordan J Bird, Anikó Ekárt, and Diego R Faria. Chatbot Interaction with Artificial Intelligence: human data augmentation with T5 and language transformer ensemble for text classification. *Journal of Ambient Intelligence and Humanized Computing* (2021), 1–16.

[8] Jordan J Bird, Anikó Ekárt, and Diego R Faria. Chatbot Interaction with Artificial Intelligence: human data augmentation with T5 and language transformer ensemble for text classification. *Journal of Ambient Intelligence and Humanized Computing* 14, 4 (2023), 3129–3144.

[9] Josip Bozic, Oliver A Tazl, and Franz Wotawa. Chatbot testing using AI planning. In *2019 IEEE International Conference On Artificial Intelligence Testing (AITest)*. IEEE, 37–44.

[10] Josip Bozic and Franz Wotawa. Testing chatbots using metamorphic relations. In *IFIP International Conference on Testing Software and Systems*. Springer, 41–55.

[11] Jordi Cabot, Loli Burgueno, Robert Clarisó, Gwendal Daniel, Jorge Perianez-Pascual, and Roberto Rodriguez-Echeverria. Testing challenges for NLP-intensive bots. In *2021 IEEE/ACM Third International Workshop on Bots in Software Engineering (BotSE)*. IEEE, 31–34.

[12] Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Crosslingual Focused Evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics, Vancouver, Canada, 1–14. https://doi.org/10.18653/v1/S17-2001

[13] "ChatGPT". ChatGpt. https://chat.openai.com.

[14] Alexandru Coca, Bo-Hsiang Tseng, Weizhe Lin, and Bill Byrne. More Robust Schema-Guided Dialogue State Tracking via Tree-Based Paraphrase Ranking. *arXiv preprint arXiv:2303.09905* (2023).

[15] Michael H Cohen, Michael Harris Cohen, James P Giangola, and Jennifer Balogh. *Voice user interface design.* Addison-Wesley Professional.

[16] Tom De Smedt and Walter Daelemans. Pattern for python. *The Journal of Machine Learning Research* 13, 1 (2012), 2063–2067.

[17] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[18] Adrian Egli. ChatGPT, GPT-4, and Other Large Language Models: The Next Revolution for Clinical Microbiology? *Clinical Infectious Diseases* (2023), ciad407.

[19] "Hugging Face". Hugging Face squad_v2. https://huggingface.co/datasets/squad_v2/viewer/squad_v2/train?p=4&row=440.

[20] "Hugging Face". Hugging Face. https://huggingface.co/cross-encoder/stsb-roberta-large.

[21] "Hugging Face". Hugging Face ambig_qa. https://huggingface.co/datasets/ambig_qa/viewer/full/train.

[22] "Hugging Face". Hugging Face break_data. https://huggingface.co/datasets/break_data/viewer/logical-forms/test?row=1.

[23] "Hugging Face". Hugging Face conv_ai_3. https://huggingface.co/datasets/conv_ai_3/viewer/conv_ai_3/train?row=36.

[24] Emanuela Guglielmi, Giovanni Rosa, Simone Scalabrino, Gabriele Bavota, and Rocco Oliveto. Replication Package of "Help Them Understand: Testing and Improving Voice User Interfaces". https://figshare.com/s/36c3475659710714175d.

[25] Emanuela Guglielmi, Giovanni Rosa, Simone Scalabrino, Gabriele Bavota, and Rocco Oliveto. Sorry, I don't Understand: Improving Voice User Interface Testing. In *37th IEEE/ACM International Conference on Automated Software Engineering*. 1–12.

[26] Jonathan Guichard, Elayne Ruane, Ross Smith, Dan Bean, and Anthony Ventresque. Assessing the robustness of conversational agents using paraphrases. In *2019 IEEE International Conference On Artificial Intelligence Testing (AITest)*. IEEE, 55–62.

[27] Samer Hassan, Andras Csomai, Carmen Banea, Ravi Sinha, and Rada Mihalcea. Unt: Subfinder: Combining knowledge sources for automatic lexical substitution. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*. 410–413.

[28] Chaitra Hegde and Shrikumar Patil. Unsupervised paraphrase generation using pre-trained language models. *arXiv preprint arXiv:2006.05477* (2020).

[29] Kuan-Hao Huang and Kai-Wei Chang. Generating syntactically controlled paraphrases without using annotated parallel pairs. *arXiv preprint arXiv:2101.10579* (2021).

[30] "KayLearch". KayLearch. https://github.com/KayLerch/alexa-utterance-generator/.

[31] Federica Laricchia. Number of digital voice assistants in use worldwide from 2019 to 2024. https://www.statista.com/statistics/973815/worldwide-digital-voice-assistant-in-use/.

[32] Kwang B Lee and Roger A Grice. The design and development of user interfaces for voice application in mobile devices. In *2006 IEEE International Professional Communication Conference*. IEEE, 308–320.

[33] Suwan Li, Lei Bu, Guangdong Bai, Zhixiu Guo, Kai Chen, and Hanlin Wei. VITAS: Guided Model-based VUI Testing of VPA Apps. In *37th IEEE/ACM International Conference on Automated Software Engineering*. 1–12.

[34] Ta Lin Liau, Carolyn B Bassin, Clessen J Martin, and Edmund B Coleman. Modification of the Coleman readability formulas. *Journal of Reading Behavior* 8, 4 (1976), 381–386.

[35] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[36] Andrea De Lucia, Fausto Fasano, Rocco Oliveto, and Genoveffa Tortora. Recovering traceability links in software artifact management systems using information retrieval methods. *ACM Transactions on Software Engineering and Methodology (TOSEM)* 16, 4 (2007), 13–es.

[37] Guillermo Macbeth, Eugenia Razumiejczyk, and Rubén Daniel Ledesma. Cliff's Delta Calculator: A non-parametric effect size program for two groups of observations. *Universitas Psychologica* 10, 2 (2011), 545–555.

[38] Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*. 55–60.

[39] Ke Mao, Mark Harman, and Yue Jia. Sapienz: Multi-objective automated testing for android applications. In *Proceedings of the 25th International Symposium on Software Testing and Analysis*. 94–105.

[40] Tomáš Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*. 746–751.

[41] George A Miller. WordNet: a lexical database for English. *Commun. ACM* 38, 11 (1995), 39–41.

[42] Kevin Moran, Mario Linares Vásquez, and Denys Poshyvanyk. Automated GUI testing of Android apps: from research to practice. In *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*. IEEE, 505–506.

[43] Leah Nicolich-Henkin, Taichi Nakatani, Zach Trozenski, Joel Whiteman, and Nathan Susanj. Comparing Data Augmentation and Annotation Standardization to Improve End-to-end Spoken Language Understanding Models. In *Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS)*. 1–6.

[44] Octavany Octavany and Arya Wicaksana. Cleveree: an artificially intelligent web service for Jacob voice chatbot. *TELKOMNIKA (Telecommunication Computing Electronics and Control)* 18, 3 (2020), 1422–1432.

[45] Hemant Palivela. Optimization of paraphrase generation and identification using language models in natural language processing. *International Journal of Information Management Data Insights* 1, 2 (2021), 100025.

[46] Ranci Ren, Mireya Zapata, John W. Castro, Oscar Dieste, and Silvia T. Acuña. Experimentation for Chatbot Usability Evaluation: A Secondary Study. *IEEE Access* 10 (2022), 12430–12464. https://doi.org/10.1109/ACCESS.2022.3145323

[47] Konstantinos I Roumeliotis and Nikolaos D Tselikas. ChatGPT and Open-AI Models: A Preliminary Review. *Future Internet* 15, 6 (2023), 192.

[48] Kabir S Said, Liming Nie, Adekunle A Ajibode, and Xueyi Zhou. GUI testing for mobile applications: objectives, approaches and challenges. In *12th Asia-Pacific Symposium on Internetware*. 51–60.

[49] Sergio Segura, Gordon Fraser, Ana B Sanchez, and Antonio Ruiz-Cortés. A survey on metamorphic testing. *IEEE Transactions on software engineering* 42, 9 (2016), 805–824.

[50] Siamak Shakeri and Abhinav Sethy. Label dependent deep variational paraphrase generation. *arXiv preprint arXiv:1911.11952* (2019).

[51] Alex Sokolov and Denis Filimonov. Neural machine translation for paraphrase generation. *arXiv preprint arXiv:2006.14223* (2020).

[52] "Liling Tan. Pywsd: Python implementations of word sense disambiguation (wsd) technologies [software]. https://github.com/alvations/pywsd.

[53] Yushi Wang, Jonathan Berant, and Percy Liang. Building a semantic parser overnight. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 1332–1342.

[54] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682* (2022).

[55] Sam Witteveen and Martin Andrews. Paraphrasing with large language models. *arXiv preprint arXiv:1911.09661* (2019).

[56] Robert F Woolson. Wilcoxon signed-rank test. *Wiley encyclopedia of clinical trials* (2007), 1–3.

[57] Chen Zhang, Luis Fernando D'Haro, Qiquan Zhang, Thomas Friedrichs, and Haizhou Li. PoE: A Panel of Experts for Generalized Automatic Dialogue Assessment. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 31 (2023), 1234–1250.

[58] Jianing Zhou and Suma Bhat. Paraphrase Generation: A Survey of the State of the Art. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 5075–5086. https://doi.org/10.18653/v1/2021.emnlp-main.414