DetBERT: Enhancing Detection of Policy Violations for Voice Assistant Applications using BERT

Rawan Baalous, Joud Alzahrani, Mariam Ali, Rana Asiri, Eman Nooli
Cybersecurity Department, University of Jeddah, Jeddah, Saudi Arabia

***Abstract*—Voice Assistants, also known as VAs, have gained popularity in the last few years. They make our daily tasks easier via simple voice instructions. VAs platforms allow third-party developers to develop voice applications and publish them on the VAs platforms. However, VAs applications may collect users' personal information for different purposes. To maintain the security and privacy of users, VAs platforms have specified a set of policies that must be adhered to by VAs applications' developers. This paper aims to automatically detect voice apps that do not comply with the VA's platforms policies. To this end, DetBERT, a comprehensive testing tool, was built. DetBERT evaluates voice apps' compliance with the policies using BERT model by analyzing the apps' behaviors and detecting violations. With DetBERT, a total of 50,000 voice assistant apps from Amazon Alexa and Google Assistant platforms were tested. The paper demonstrates that DetBERT can accurately identify whether a voice assistant application has violated the platform's policy or not.**

Keywords—Alexa; Google assistant; BERT; policy violation detector; voice assistant; user privacy; security

I. Introduction

Voice assistants (VAs) have become widespread and integrated into billions of people's daily lives due to their ease of everyday tasks and the comfortable services they provide. Amazon Alexa and Google Assistant are ones of the most popular VAs platforms, which allow third-party developers to publish their voice applications1[1] in the stores [1]. Many users' activities can be accomplished through these skills, including placing orders, obtaining information like weather and news, and making phone calls. This attracts tens of millions of users and, in turn, more developers.

As skills grow rapidly, dangerous skills also appear. Since third-party developers can share their skills, the privacy and security concerns of VAs users arise regarding the skills developers' intents [2]. Recent studies have revealed that developers are capable of redirecting users' requests to malicious skills without their knowledge. This can achieved by naming their skills similarly to legitimate ones [3][4]. In fact, malicious skills could eavesdrop on users' conversations or even monitor them, which affect users' privacy [5]. In order to maintain the security and privacy of users, VAs platforms have defined a set of policy requirements and enforced them to be adhered to by third-party developers. Nonetheless, some VAs platforms use a weak vetting system [1], which allows several skills that violate policies to bypass the VAs platform's verification process and get certified.

The main challenge obstructing authoritative skill certification is the VAs platforms' distributed architecture. Using static code analysis to investigate a skill's behavior is not an option for current VAs systems. This is because the skill's code is hosted externally in the developer's servers, making it not accessible. As a result, the only way to comprehend a skill's actual behavior is through dynamic analysis (by interacting with a skill) [6]. This motivated us to explore VA's skills that violate policy requirements by behaving suspiciously, such as asking for personal information when it is not supposed to request such details.

In this work, we aim to identify stealthy policy violations conducted by third-party developers in Amazon Alexa and Google Assistant by enhancing the robustness and accuracy of the policy-violation detection process. Prior works showed many limitations in the approaches used by the pre-crafted policy-violation detection tools [6][7]. The drawback of these approaches lies in the inability to handle a skill's textual speech in a contextual meaning for the entire sentence. This results in decreasing the accuracy of detecting violations among variant policies type. To this end, we created VA's policy-violation detection tool using Bidirectional Encoder Representations Transformers (BERT) model. BERT [8] works in a bidirectional way to figure out the ambiguous language in the text, hence increasing the accuracy of analyzing the speech context.

In summary, this paper contributes to the field of privacy compliance checking by enhancing VA's policy violation detection. We developed a dynamic policy violation detection tool, called DetBERT. Our tool utilizes BERT model to improve the process of detecting skills that violate the VA's platforms policies. We developed two BERT models: User Privacy BERT and Content Safety BERT. The accuracy of both models in identifying violations is 0.98 % and 0.93 respectively.

The reminder of this paper is organized as follows: The background survey is detailed in Section II. Section III summarizes the recent literature on policy violation detection of VAs as well as several attacks on them. The process of detecting skills that violated the platforms policies is discussed in Section IV. Sections V and VI present the results of violations detected and compare the results with previous works. Finally, Section VII presents the conclusion, limitations and future work.

[1]Voice-apps are known as skills on Amazon and actions on Google. In this paper, we refer to voice applications as skills unless there is a need to clearly distinguish between the two platforms.

II. Background

A. VAs Platforms and Skills Interaction

Fig. 1 presents an overview of the VA platform and skill interaction flow. A skill comprises a front-end interaction model and a back-end cloud service (skill code). The back-end is responsible for handling requests that come from users and directing a VA device's response. Similar to smartphone applications, most skills are created by third-party developers and made accessible through a skill store website. Skill's front-end and back-end are hosted separately due to the distributed architecture of VAs platforms. The back-end code is typically hosted on the developer's server (e.g., hosted by Amazon Web Services AWS Lambda under the developer's account or other third-party servers). On the other hand, the VAs platforms host the front-end interface [1].

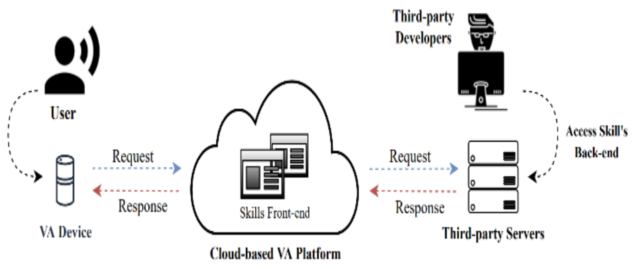

Fig. 1. Cloud-based alexa platform.

Amazon and Google provide an online repository of skills through their skills stores. Each skill is an individual product with a unique web page in the store. Skill's webpage includes the developer's information, skill description, skill identifier, privacy statement, users ratings, and users reviews [9]. In addition, it includes a sample of utterances (i.e., skill invocations) that enable the user to interact with the skill. A skill's privacy policy on the skill store should describe whatever data the skill may collect and how it will use that data in the future. Some skills may ask users to grant access to personal information to receive customized services. In this case, the user must provide permission through the VA companion app (Android/iOS) for the skill to get the required personal data [6]. VAs platforms provide a skill simulator for the testing needs of developers' skills [10]. The skill simulators provide a text-based interface that will receive text input, produce text output, and deliver external content to make testing more manageable. VA's simulators consist of a virtual VA device that can communicate with other skills available in the skill store [6].

B. VAs Platforms Policies

VAs platforms provide a set of policy and security requirements that skills must adhere to. Before publishing a skill to the store, the platforms conduct a test to verify if the skill complies with these requirements. This process is designed to restrict the amount of potentially exploitable content that may be found on the skill store. If the VAs platform determines the existence of a violation, it has the right to take disciplinary action. Amazon Alexa specified 14 policy requirements and 7 privacy requirements [11][12]. Under each requirement, there are several statements that describe illegal skill usage regarding the policy. On the other hand, Google Assistant has 10 main sections of policies [13]. Every section covers several policies related to it. To demonstrate, Google has a section called Content Restrictions that specifies allowed and disallowed content. This section involves 15 policies related to content. Another section is Privacy and Security which establishes requirements of what data is allowed to be collected, how skills must handle users' data, and maintain security among skills. There are common policies between Amazon and Google regarding specific skills categories (e.g., Health and Kids).

C. BERT

In late 2018, researchers at Google AI developed a state-of-art model that has been an inflection point for Natural Language Processing (NLP). **B**idirectional **E**ncoder **R**epresentations from **T**ransformers is a machine learning pre-trained model. BERT has been pre-trained on a large dataset of books (800 million words) and texts from English Wikipedia pages (2,500 million words). It combines the right-left and left-right contexts to create a complete picture of the text. As a result, it helps machines to understand the contextual relation between words in the sentence. BERT uses transfer learning which improves the fine-tuning-based approaches. This means training a model on a general task (pre-training), then taking advantage of the knowledge that the model gained to solve related tasks (fine-tuning). With just one output layer, the BERT model can be fine-tuned to produce state-of-the-art text representations for various tasks, such as question answering, semantic analysis, and text prediction [8].

There are two main versions of BERT model: BERT base and BERT large. The main difference between the two versions is the number of used encoders. The first version: BERT base consists of 12 encoders, whereas the second version: BERT large consists of 24 encoders that originate from the transformer model. The large version of BERT represents more robust than the BERT base version, therefore it requires more powerful resources [8].

BERT uses special tokens as part of its input representation. These special tokens serve specific functions in the BERT input format and help the model process the input text correctly. Positions and meanings of the special tokens are taken into consideration by the model when generating representations for the input tokens. The most commonly used special tokens in BERT are [CLS], and [SEP]. The [CLS] or Classification token is added at the beginning of the input sequence and is used to represent the entire input sequence for classification tasks. In other words, the output of the BERT model for the [CLS] token is used as a representation of the entire input text for classification tasks, such as sentiment analysis or named entity recognition. The second token is the [SEP] or (separation) token. It is added at the end of the first sentence and in between subsequent sentences in the input sequence. It is used to separate the different sentences in the input text. This allows BERT to treat each sentence independently and capture the relationship between the sentences, which is important for tasks like question answering [8].

III. Related Works

In this section, we summarize recent literature on policy violation detection of VAs as well as several attacks on them.

With regard to policy violation detection, Cheng et al. [1] conducted a study about the trustworthiness of skill certification in Amazon Alexa and Google Assistant platforms in terms of catching any policy violations in the third-party skills and whether policy-violating skills are published in the stores. The authors were interested in evaluating the level of difficulty in publishing skills that violate the policies in the stores. Intentionally, they submitted 234 Alexa skills and 381 Google Assistant actions that violated privacy and content policies. Surprisingly, all the violated skills got certified. At the end, the authors provided strategies in order to enhance the skills certification process. Similarly, Lentzsch et al. [14] identified flaws in the vetting process conducted by Amazon and tested only the skills that request permissions for data collection.

Dynamic testing tools [6][7][15][16] have been developed to enable automated skills analysis on a broad scale. SkillDetective [6] is a testing tool that explores voice apps' behaviors and identifies possible policy violations through live interaction with skills. The authors tested 54,055 Amazon Alexa skills and 5,583 Google Assistant actions and identified 6,079 skills and 175 actions violating at least one policy requirement. They utilized data-driven methodology to identify question types and used the Feedforward Neural Networks (FNN) and a bag-of-words (BoW) approach for answer prediction. However, implementing FNN and a BoW approach in the SkillDetective Chatbot slowed the policy-violation detection process. As a result, SkillDetective could not test every skill at full capacity. On the other hand, Guo et al. [7] developed SkillExplorer, an automated testing tool used to examine a skill's behavior through a grammar-rule based technique. The tool mainly focused on skills that collect private information from users. Authors found that 1,141 skills and 1,897 actions request personal information from users without specifying that in their privacy policies. As a consequence of using grammar-rule based technique, SkillExplorer was not able to properly answer some questions during the interaction with skills. This affected the accuracy of the violation detection results negatively.

Focusing on health VAs, Shezan et al. [15] proposed a static and dynamic machine learning-based solution. The dynamic part triggered and detected the violation through deep interaction with 813 health-related skills in Alexa. At the same time, the static part analyzed the web page of these skills. The study aimed to detect skills that provide life-saving assistance and lack disclaimers, which is prohibited by Amazon. They consulted medical school students regarding the correctness of the potential violation. In the end, VerHealth detected 244 out of 813 skills violating Amazon's health-related policies. In terms of kids related apps, authors in SkillBot [16] developed an automatic tool using natural language processing that interacts only with kids-related skills. They aimed to find out the risky skills that may ask for kids' personal information or contain inappropriate content. The tool analyzed 3,434 Alexa kids skills. Results showed that there are 28 risky child-directed skills. In addition, a user-study has been conducted with parents. The authors also identified a novel risk in VAs called, confounding utterance. They defined it as: voice commands that invoke a non-child skill over a child-directed skill. In the end, they found 4,487 confounding utterances which indicate the high risk surrounding the children of invoking a non-child skill by accident.

Considering attacks on VAs, Cheng et al. [17] proved that skills are features that provide an entry point for attackers. They analyzed multiple attacks on VAs. Also, there are many researches on hidden voice attacks [18]–[20] and their corresponding defenses [21][22]. Kumar et al. [23] validated skill squatting attacks. In this attack, the attacker gets the advantage of sentence ambiguities and similar pronunciation to redirect the VAs users into a malicious skill. There are many types of masquerading attack, including voice squatting, in which the attacker exploits how the user call the skill and alter this call either by imitating the skill call or reordering the sentence words. In voice masquerading, the malicious skill impersonates the VA service to gather users' personal information or eavesdrops on their private conversations. Richard et al. [24] showed man-in-the-middle attacks against benign skills. The attack utilizes a weakness presented in a skill interface to redirect a victim's voice when invoking the skill to a malicious skill, then hijack the conversation between Alexa and the victim.

IV. Methodology

This section provides an overview of the whole process of detecting skills that violated the platforms' policies (Section A). After that, the key modules of the process are discussed in detail. The data collection procedure is firstly presented (Section B). Then, the interaction with the skills procedure (Section C). Lastly, the violation detection procedure using the BERT model (Section D).

A. Overview

As illustrated in Fig. 2, the first step in the policy violation detection process is the interaction with a skill. To analyze a skill's potentiality of violating the policy, the skill's outputs to users must be collected. To this end, a chatbot that communicates with the VA's device simulator has been used. The chatbot automatically interacts with the targeted skill to collect its outputs, making the process faster and easier than manual interaction. The interaction starts when the first utterance (e.g., "Alexa, Open My Nutrition") is fed to the skill, resulting in activating it. When the skill receives an invocation word, it will pass back an output. During the communication, all skill's outputs will be stored to be examined later for policy violation. When the communication with the simulator ends, the violation detection process will be started in offline mode. The collected outputs are passed to the policy violation detector tool which analyzes the gathered outputs to identify any violation. To accomplish this process, the tool utilizes the BERT model that is trained on analyzing and classifying the outputs, searching for violation indications. Using BERT in the detector tool helps to understand the ambiguous violation in the sentences and phrases. As a result, it improves the accuracy of policy violation detection in the skill's outputs.

B. Data Collection

1) Skills sample: As a primary dataset, we used SkillDetective's dataset [6]. Each record in the dataset consists of the skill's data divided into six features. Table I describes each feature in the dataset. However, as the store releases new skills continuously, and to ensure violation detection of recently published skills, we have collected more skills using the Octuparse extraction tool [25]. Using this tool, we can automatically access web pages and extracts data from them. Once we have finished the extraction process, we combined our skills dataset and SkillDetective's dataset, with paying attention to remove duplicated skills. In total, the final skills sample is 69,843 skills and 16,003 actions.

2) Violations dataset: We had many challenges while looking for suitable datasets to fine-tune BERT Model on violation detection based on Amazon Alexa and Google Assistant policies. The reason is that BERT receives datasets in form of sentences and paragraphs (i.e., Tweets). The lack of User Privacy Violations Datasets led us to craft one from scratch. For other types of violations, we used publicly published datasets.

TABLE I. FEATURES DESCRIPTION IN SKILLDETECTIVE'S DATASETS

Feature	***Description***
Skill ID	Unique identifier of the skill, consists of 10 digits.
Name	Specifies skill name in the store.
Category	Specifies which categories the skill belongs to.
Invocation	Keywords used to start the interaction with skill.
Description	Specifies the skill functionalities.
Privacy Policy Link	Specifies the privacy policy that the skill adheres to.

The User Privacy Violation Dataset consists of sentences and questions mentioned during the conversation, with the intent of collecting information about users. For example, Are you alone at home? or Provide me with your graduation date. For help in creating this dataset, we used ChatGPT [25], which is an AI-powered language model that generates human-like responses to various questions and prompts. Based on the VAs platforms policies, we prompted ChatGPT to generate questions and sentences that violate user privacy. We specifically asked for queries related to collecting personal, sensitive, health, and kids' information, since the privacy policies of the VAs state different permissions regarding collecting user information based on skill’s category. In that manner, our dataset contains four classes: 1) Personal Data Collection, 2) Sensitive Data Collection, 3) Health Data Collection, and 4) Non-violation. Table II presents some examples of information involved in each class. Many of the generated questions were duplicated, so we had to remove duplication using python scripts. The total number of questions we got was 3745 unique questions. After collection, the second step was the annotation. We chose a human annotation approach because it is often more accurate compared to automated approaches. We manually checked for the mismatch between data contents and their corresponding labels. The process was iterative across the authors.

The Content Safety Violation Dataset consists of content identified as harmful and inappropriate in policies (e.g., sexual content). Such content is prohibited from occurring or being mentioned by the skill. We used two published datasets from Kaggle, Toxic Content [26] and Cyberbullying datasets [27]. They consist of two columns: Toxicity (toxic and non-toxic), and Content. We merged both datasets for fine-tuning, resulting in 207,266 records.

C. Interaction with the Skills

Once the interaction with the skill starts, the goal is to maintain the conversation continued between the chatbot and the skill as long as possible. By keeping the conversation ongoing, we can gather more skills’ outputs to be analyzed. As a result, more skills’ behaviors will be identified. To this end, we used the SkillDetective Chatbot [28], which was published for future research.

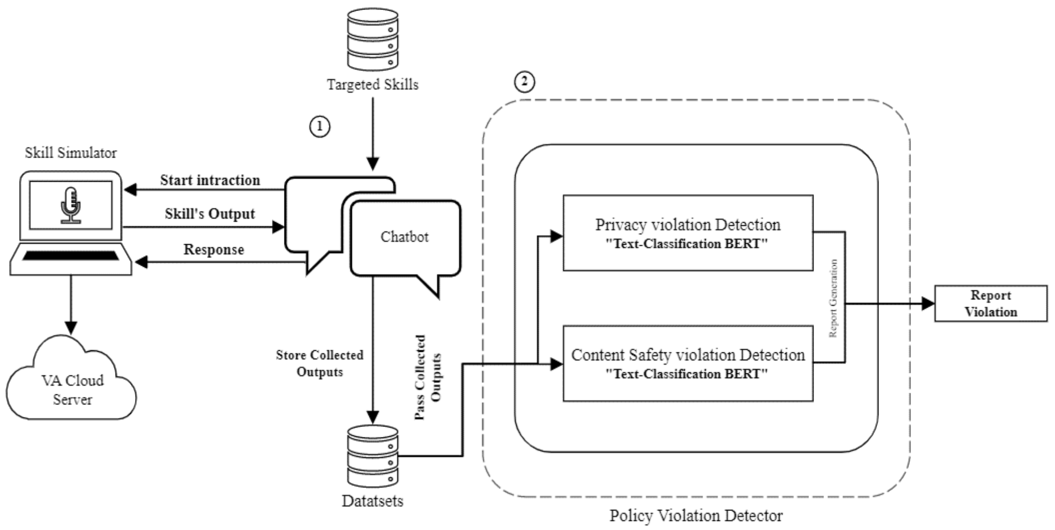

Fig. 2. DetBERT methodology.

Of all the published chatbots that previous works have used, we chose SkillDetective Chatbot for two reasons. First, the chatbot can handle five types of questions with high classification accuracy, as shown in Table III. The second reason is that it implements an approach called SkillTree. This approach is meant to build a dynamically growing tree to keep track of all branches (i.e., paths) that have been examined and those that have not, to ensure all the skill's possible behaviors have been explored. It is used in situations where questions with multiple answers exist. For example, the Yes/No questions have two answers, and the selection questions have two or more answers. Each answer is saved in the branch of the tree to be visited later. Using this approach helps to increase skill's coverage and reduces testing latency. One drawback of the chatbot, it utilizes FNN and BoW approaches, which make communication heavy and very slow. Besides, the continuous interruptions of the connection with the simulator which required human intervention. Lastly, by the end of every interaction, all outputs generated by the skill are recorded for later analysis.

D. *Violation Detection*

The policy violation detector tool focuses on detecting violations that happened during the interaction with skill. In this work, we mainly focus on two types of policies: 1) User Privacy Policies; 2) Content Safety Policies. These policies are described in details in the Appendix (Table VIII), which lists the policy statements as mentioned in the VAs platforms. To detect violations related to these policies, we used BERT base model to build our tool. We developed two different BERT models for the text classification task according to the different violations we looked for during the examination. The first model is a multi-classification model which was developed to check for User Privacy Violations. The second model is a binary-classification model trained on detecting Content Safety Violations. We mainly took pre-trained BERT models and then fine-tuned them on specific datasets to lower the cost of BERT training. Using BERT in violation detection allows the sentences (skills' outputs) to be processed in a contextualized meaning for the entire output sentences. Therefore, the accuracy of detecting violations increases.

Fig. 3 illustrates how the policy violation detector works. First, the skill's output is tokenized using the wordPiece tokenizer [29], which is the BERT tokenizer. The tokenizer breaks down the sentence into chunks of words. Depending on the vocabulary file utilized by the tokenizer, some words are split into a single word, and some are split into multiple words. After tokenization, BERT special tokens, [CLS] and [SEP], are added at the beginning and end of the sentence. Then, the tokens are padded with the "PAD" token to reach the maximum input size for BERT. After padding the tokens, they are converted to token IDs which are fed to the BERT model. In the following step, the input tokens are transformed into integer IDs based on the tokenizer vocabulary file. These integer IDs are input into the BERT model along with a matrix of ones and zeros called an Attention Mask. The matrix represents whether the token input is genuine or padded input. The attention mask elements corresponding to genuine inputs are set to 1, while those corresponding to padded inputs are set to 0. The BERT model converts each genuine input into a vector of a specific size known as the BERT hidden size. This vector is created by an encoder with an attention layer, which allows it to better understand the token's context. After all the vectors have been made, they are then used as input to a classification layer, which determines the class that the input belongs to.

TABLE II. EXAMPLES OF INFORMATION INVOLVED IN DATASET CLASSES

Class	***Involved Information***
Personal information	Name, Age, Birthday, Gender, Location
Sensitive data	Social Security Number, Visa Code, Passwords, Bank Account Numbers, Credit Card Numbers
Health data	Heart Rate, Mass Index, Blood Pressure, Blood Type
Non-violation	General Questions and Statements

TABLE III. SKILLDETECTVE CHATBOT'S QUESTIONS CLASSIFICATION ACCURACY [6]

Question Type	***Example***	***Identification Accuracy***
Binary	Are you in the car?	100%
Selection	Say your name.	99.3%
Instruction	Do you want to eat, run, or watch TV?	99%
Open-Ended	What is your mother's name?	98%
Mixed	There are A, B, and C to choose from. Which one do you want?	98%

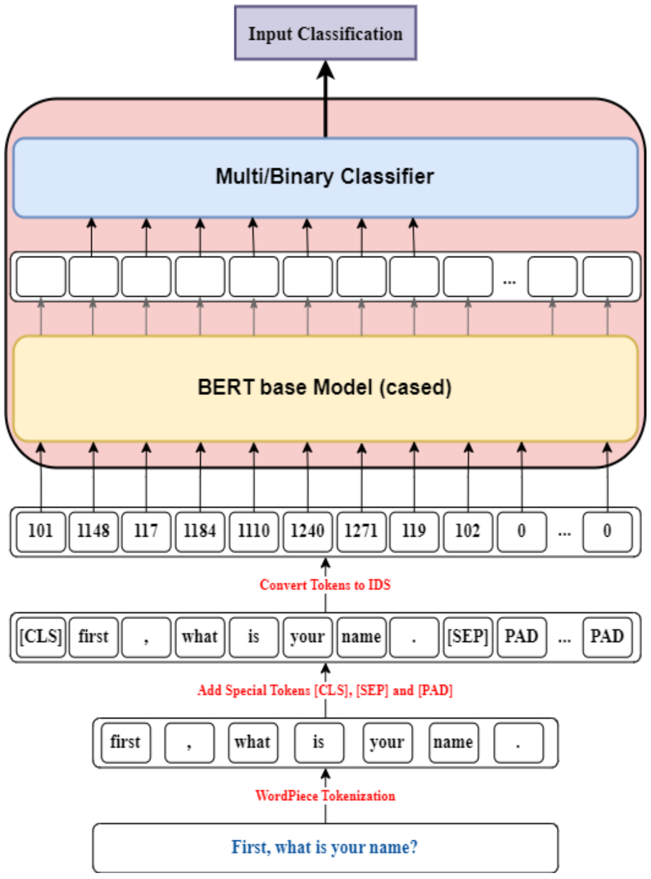

Fig. 3. How policy violation detector works.

The multi-classification BERT model which was developed to check for User Privacy Violations has been used to determine the violation type. It classifies each tweet (i.e. skill's output) entered the model into one of four classes, 1) Personal Data Collection; 2) Sensitive Data Collection; 3) Health Data Collection; 4) Non-violation. Note that skills are allowed to ask for personal data in order to perform some of their required tasks, with the condition of providing a privacy policy link outlining legal data usage. It is also important to highlight that skills related to health deal with sensitive data about users' medical conditions. These kinds of data cannot be collected or disclosed without user's permission. In a like manner, kids' skills are designed for kids who are targeted by potential threats more than adults. The tool detects any potential unauthorized data collection of users' personal or sensitive data. For personal data collection, the skill must attach a privacy notice (i.e., privacy policy link) to its page. The tool uses the User Privacy BERT model to determine whether the skill gathered user data during the conversation. If it does, the existence of privacy notices on the skill's page will be checked. If there is no link provided, the skill is considered violated.

On the other hand, Content Violations are hard to predict. In fact, the skill's behavior may differ based on the conversation user had with the skill or because of the skill's update. Both VAs platforms stated many policies related to content. Based on these policies, the policy violation detector used Content Safety BERT model to detect skills that violate content restrictions. All skills are not allowed to use inappropriate and harmful content like profanity or hate language and promoting or sale of illegal materials like drugs. For kids' skills, content must be appropriate for all ages. To this end, the binary BERT model is deployed to recognize and differentiate between harmful and legal content. It classifies each tweet entered in the model into one of two classes: 1) Toxic; 2) Non-Toxic.

To generate the final report, we summarized all the violations results after the detection process ends. The violations detected were recorded and saved into four files. Files were divided based on four categories of violations. The categories are as follows: 1) Kids Policy Violations; 2) Health Policy Violations; 3) Toxic Policy Violations; 4) General Policy Violations. We have divided the results into these files as they are the main types of violations we have focused on. Each file contains six primary columns: Order, Category, Violated_policy, Skill_id, Skill_name, and Skill_output.

V. RESULTS

The results of violations detected in terms of user privacy violations and content safety are summarized in Table IV.

Table V presents the results obtained by our models: User Privacy BERT and Content Safety BERT, in terms of precision, recall, F1-score, and accuracy. The results indicate that User Privacy BERT model is performing extremely well in detecting policy violations related to user privacy. With the achieved accuracy of %98, the model makes correct predictions about violations. Overall, the results shown by the User Privacy BERT model demonstrate its high performance. On the other hand, the results obtained by the Content Safety BERT model shows that the model is performing well, but there is still room for improvement. In fact, the model was trained on seven epochs, which is better to be increased especially for huge datasets like the content safety violations dataset.

TABLE IV. SUMMARY OF VIOLATIONS DETECTED BY DETBERT

Detected Violations Type		*# Of skills*	*# Of actions*
Violation of User Privacy Policies	*Collect health data*	147	0
	Collect personal data	52	14
	Collect sensitive data	0	0
	Lack of privacy policy	172	6
Violation of Content Safety	*Contain toxic content in general categories*	154	0
	Contain toxic content in kids category	1	0

TABLE V. PERFORMANCE EVALUATION OF DETBERT MODELS

Model	*Accuracy*	*Precision*	*Recall*	*F1*
User Privacy BERT	0.98	0.98	0.98	0.98
Content Safety BERT	0.93	0.93	0.93	0.93

Fig. 4 displays the User Privacy BERT model confusion matrix. The TP and TN show a high score, which in turn indicates the good performance of the model. The matrix reveals that the model has identified 1106 violations correctly out of 1125 actual values. Additionally, it accurately categorized 356 non-violations out of 375 actual values. The results of the confusion matrix of the Content Safety BERT model are shown in Fig. 5. The high score for TP and TN indicates also the model's good performance. The matrix shows that the model correctly identified 27,519 Toxic content violations out of 30,000. Additionally, it accurately categorized 28,018 Non-toxic contents out of 30,000.

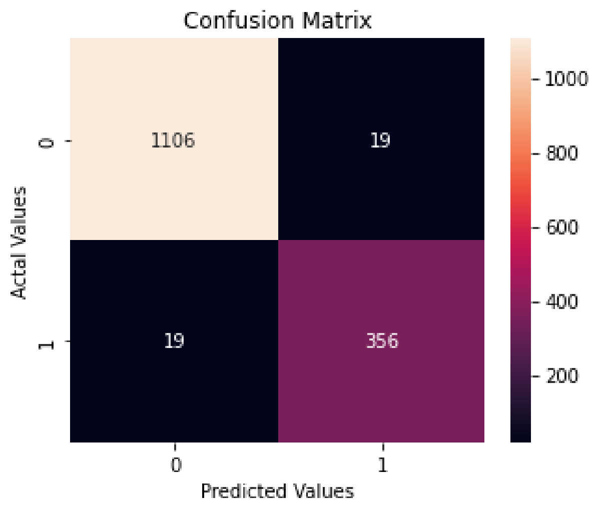

Fig. 4. Confusion matrix of user privacy violations dataset.

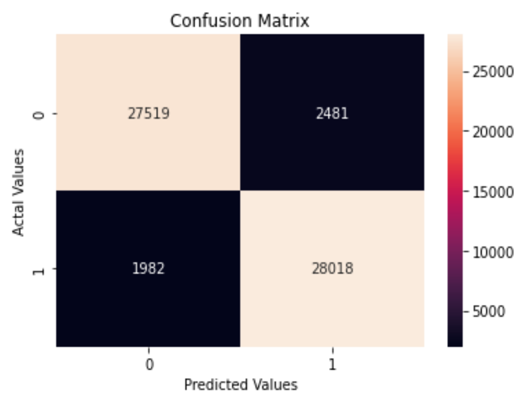

Fig. 5. Confusion matrix of content safety violations dataset.

In summary, we proved the high performance of our proposed DetBERT tool. This is done by utilizing confusion matrix analysis for BERT models. The results of both matrices showed successful TP and TN predictions by the models.

VI. Discussion

In order to compare our work with previous works, we conducted a performance comparison based on the results of the detected violations between DetBERT and SkillDetective[6], as shown in Table VI. The comparison was performed on the same sample for both models. The results of SkillDetective showed that 557 skills and 13 actions violated at least one policy. To ensure the correctness of the results, we conducted a manual revision of SkillDetective results. In fact, we have found some FP results in Content Safety. After excluding the FP, the final results obtained was 159 violations detected. On the other hand, The BERT-based approach of DetBERT has led to better performance in detecting policy violations as shown in the table. This comparison demonstrates the efficiency of DetBERT and its potential to provide valuable insights into policy compliance.

In terms of accuracy, Table VII shows the differences in accuracy between SkillDetective and DetBERT. Regarding user privacy policies, SkillDetective has developed two different methods to detect skills that collect user data. As summarized in Table VII, both methods achieved results less than what the User Privacy BERT model achieved. This comparison concludes that BERT-based model provided more accurate results in terms of detecting data collection policy violations. For content safety, SkillDetective did not reveal much details about accuracy results, and hence no comparison was provided.

TABLE VI. Number of Violations Detected in SkillDetective and DetBERT

Policy Violation Type	***SkillDetective***		***DetBERT***	
	# Of skills	***# Of actions***	***# Of skills***	***#Of actions***
User Privacy Policies	364	13	371	20
Content Safety Policies	159	0	171	0
Total Violated Skills	523	13	542	20

TABLE VII. Accuracy Comparison between SkillDetective and DetBERT

Tool	***Model/Method***	***Accuracy***
SkillDetective	Kids and Health Categories	92%
	Data Collection for general Categories	89%
DetBERT	User Privacy BERT	**98%**

VII. Conclusion

To conclude, the detection of policies' violations is an ongoing challenging process and an open area for researchers in the NLP field. In this paper, we have presented an improvement in examining voice assistants apps' compliance with stores' policies using the BERT model. We designed and implemented a violation detection tool called DetBERT. The results provide valuable insights about policy violations and can be used in the future for more accurate policy compliance checking.

This research has some limitations. Due to a limitation in the server we used to run DetBERT, we could only test a total of 50,000 skills. In addition, there was no publicly available dataset related to policy violations for the voice assistants platforms. As a result, we ended up creating our dataset with manual annotation. The dataset consists of 3745 records, which was considered small for fine-tuning the model properly. We believe there are various ways to improve and extend this study in the future. Future works can consider using larger dataset for fine tuning BERT. In addition, creating a chatbot using AI such as ChatGPT that can engage in a conversation with the skills, may enhances the question-answering precision and provides insights into new and ongoing risky behaviors.

References

[1] L. Cheng, C. Wilson, S. Liao, J. Young, D. Dong, and H. Hu, "Dangerous skills got certified: Measuring the trustworthiness of skill certification in voice personal assistant platforms," in Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security, 2020, pp. 1699–1716.

[2] D. Su, J. Liu, S. Zhu, X. Wang, and W. Wang, " 'Are you home alone?''' Yes" Disclosing Security and Privacy Vulnerabilities in Alexa Skills," arXiv Prepr. arXiv2010.10788, 2020.

[3] N. Zhang, X. Mi, X. Feng, X. Wang, Y. Tian, and F. Qian, "Dangerous skills: Understanding and mitigating security risks of voice-controlled third-party functions on virtual personal assistant systems," in 2019 IEEE Symposium on Security and Privacy (SP), 2019, pp. 1381–1396.

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

[5] D. Kumar et al., "Skill squatting attacks on Amazon Alexa," in 27th USENIX security symposium (USENIX Security 18), 2018, pp. 33–47.

[6] J. Young, S. Liao, L. Cheng, H. Hu, and H. Deng, "SkillDetective: Automated Policy-Violation detection of voice assistant applications in the wild," in USENIX Security Symposium, 2022.

[7] Z. Guo, Z. Lin, P. Li, and K. Chen, "{SkillExplorer}: Understanding the Behavior of Skills in Large Scale," in 29th USENIX Security Symposium (USENIX Security 20), 2020, pp. 2649–2666.

[8] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," arXiv Prepr. arXiv1810.04805, 2018.

[9] S. Liao, C. Wilson, L. Cheng, H. Hu, and H. Deng, "Measuring the effectiveness of privacy policies for voice assistant applications," in Annual Computer Security Applications Conference, 2020, pp. 856–869.

[10] "Test with the Alexa Simulator | Alexa Skills Kit." https://developer.amazon.com/en-US/docs/alexa/devconsole/alexa-simulator.html (accessed Oct. 01, 2022).

[11] "Policy Testing | Alexa Skills Kit." https://developer.amazon.com/fr-FR/docs/alexa/custom-skills/policy-testing-for-an-alexa-skill.html (accessed Sep. 20, 2022).

[12] "Security Testing for an Alexa Skill | Alexa Skills Kit." https://developer.amazon.com/en-US/docs/alexa/custom-skills/security-testing-for-an-alexa-skill.html#25-privacy-requirements (accessed Nov. 14, 2022).

[13] "Policies for Actions on Google | Actions console | Google Developers." https://developers.google.com/assistant/console/policies/general-policies (accessed Oct. 28, 2022).

[14] C. Lentzsch, S. J. Shah, B. Andow, M. Degeling, A. Das, and W. Enck, "Hey Alexa, is this skill safe?: Taking a closer look at the Alexa skill ecosystem," Netw. Distrib. Syst. Secur. Symp., 2021.

[15] F. H. Shezan, H. Hu, G. Wang, and Y. Tian, "Verhealth: Vetting medical voice applications through policy enforcement," Proc. ACM interactive, mobile, wearable ubiquitous Technol., vol. 4, no. 4, pp. 1–21, 2020.

[16] T. Le, D. Y. Huang, N. Apthorpe, and Y. Tian, "Skillbot: Identifying risky content for children in alexa skills," ACM Trans. Internet Technol., vol. 22, no. 3, pp. 1–31, 2022.

[17] P. Cheng and U. Roedig, "Personal Voice Assistant Security and Privacy—A Survey," Proc. IEEE, vol. 110, no. 4, pp. 476–507, 2022.

[18] N. Carlini et al., "Hidden voice commands," in 25th USENIX security symposium (USENIX security 16), 2016, pp. 513–530.

[19] Y. Wu et al., "HVAC: Evading Classifier-based Defenses in Hidden Voice Attacks," in Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security, 2021, pp. 82–94.

[20] G. Zhang, C. Yan, X. Ji, T. Zhang, T. Zhang, and W. Xu, "Dolphinattack: Inaudible voice commands," in Proceedings of the 2017 ACM SIGSAC conference on computer and communications security, 2017, pp. 103–117.

[21] I.-Y. Kwak, J. H. Huh, S. T. Han, I. Kim, and J. Yoon, "Voice presentation attack detection through text-converted voice command analysis," in Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, 2019, pp. 1–12.

[22] C. Wang, S. A. Anand, J. Liu, P. Walker, Y. Chen, and N. Saxena, "Defeating hidden audio channel attacks on voice assistants via audio-induced surface vibrations," in Proceedings of the 35th Annual Computer Security Applications Conference, 2019, pp. 42–56.

[23] N. Zhang, X. Mi, X. Feng, X. Wang, Y. Tian, and F. Qian, "Understanding and mitigating the security risks of voice-controlled third-party skills on amazon alexa and google home," arXiv Prepr. arXiv1805.01525, 2018.

[24] R. Mitev, M. Miettinen, and A.-R. Sadeghi, "Alexa lied to me: Skill-based man-in-the-middle attacks on virtual assistants," in Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security, 2019, pp. 465–478.

[25] "ChatGPT: Optimizing Language Models for Dialogue." https://openai.com/blog/chatgpt/ (accessed Feb. 13, 2023).

[26] "Toxic Comment Classification Challenge | Kaggle." https://www.kaggle.com/competitions/jigsaw-toxic-comment-classification-challenge/data (accessed Feb. 13, 2023).

[27] "Cyberbullying Classification | Kaggle." https://www.kaggle.com/datasets/andrewmvd/cyberbullying-classification (accessed Feb. 14, 2023).

[28] "skilldetective/ChatBot at master • clemsonsec/skilldetective." https://github.com/clemsonsec/skilldetective/tree/master/ChatBot (accessed Feb. 13, 2023).

[29] Y. Wu et al., "Google's neural machine translation system: Bridging the gap between human and machine translation," arXiv Prepr. arXiv1609.08144, 2016.

APPENDIX

TABLE VIII. POLICIES CONSIDERED BY DETBERT

Category	Policy Violation Type	Policy defined by VAs Platforms
User Privacy	**P1:** Collecting health data	Collects information related to any person's physical or mental health or condition, the provision of health care to a person, or payment for the same
	P2: Collecting kids' data	Collects any personal information from end users
	P3: Collects any sensitive personal information from end users	Collect sensitive personally identifiable information, including, passport number, social security number, national identity number, full bank account number, or full credit/debit card number
	P4: Lacking a privacy policy	Collect personal information from end users without providing a privacy notice that displayed in skill's detail page
Content Safety	**P5:** Toxic content	It includes content not suitable for all ages
	P6: Kids' safety	Promotes any content, or services, or directs end users to engage with content outside of Alexa