Are Your Requests Your True Needs? Checking Excessive Data Collection in VPA Apps

Fuman Xie
The University of Queensland
Brisbane, Australia

Chuan Yan
The University of Queensland
Brisbane, Australia

Mark Huasong Meng*
Institute for Infocomm Research, A*STAR
Singapore

Shaoming Teng
The University of Queensland
Brisbane, Australia

Yanjun Zhang
Deakin University
Melbourne, Australia

Guangdong Bai†
The University of Queensland
Brisbane, Australia

ABSTRACT

Virtual personal assistants (VPA) services encompass a large number of third-party applications (or *apps*) to enrich their functionalities. These apps have been well examined to scrutinize their data collection behaviors against their declared privacy policies. Nonetheless, it is often overlooked that most users tend to ignore privacy policies at the installation time. Dishonest developers thus can exploit this situation by embedding excessive declarations to cover their data collection behaviors during compliance auditing.

In this work, we present Pico, a privacy inconsistency detector, which checks the VPA app's privacy compliance by analyzing (in)consistency between data requested and data essential for its functionality. Pico understands the app's functionality topics from its publicly available textual data, and leverages advanced GPT-based language models to address domain-specific challenges. Based on the counterparts with similar functionality, suspicious data collection can be detected through the lens of anomaly detection. We apply Pico to understand the *status quo* of data-functionality compliance among all 65,195 skills in the Alexa app store. Our study reveals that 21.7% of the analyzed skills exhibit suspicious data collection, including Top 10 popular Alexa skills that pose threats to 54,116 users. These findings should raise an alert to both developers and users, in the compliance with the *purpose limitation principle* in data regulations.

CCS CONCEPTS

• **Security and privacy → Web application security**.

KEYWORDS

Virtual Personal Assistant, privacy compliance, Alexa skills

*Also with National University of Singapore, Singapore.
†Corresponding Author.

ICSE '24, April 14–20, 2024, Lisbon, Portugal

ACM ISBN 979-8-4007-0217-4/24/04…$15.00
https://doi.org/10.1145/3597503.3639107

ACM Reference Format:
Fuman Xie, Chuan Yan, Mark Huasong Meng, Shaoming Teng, Yanjun Zhang, and Guangdong Bai. 2024. Are Your Requests Your True Needs? Checking Excessive Data Collection in VPA Apps. In *2024 IEEE/ACM 46th International Conference on Software Engineering (ICSE '24), April 14–20, 2024, Lisbon, Portugal.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3597503.3639107

1 INTRODUCTION

The rapid growth of the Internet of Things (IoT) is transforming the way we interact with technology. It has led to an increasing demand for IoT-based services for different purposes. Various AI (artificial intelligence)-backed virtual personal assistant (VPA) services, e.g., Amazon Alexa and Google Assistant, are prominent examples. They offer users an unprecedented level of convenience and efficiency in their daily lives. Users can effortlessly interact with these services to access functionalities from playing music, making a phone call, to controlling IoT devices. As reported by Statista [35], VPA services are now accessible on billions of devices worldwide.

Centered around the VPA services, an ecosystem similar to the successful model seen in mobile applications is growing rapidly. VPA services enable third-party developers to create VPA apps, e.g., *skills* in Amazon Alexa and *actions* in Google Assistant, and distribute them through app stores. These apps are designed to be easily accessible and user-friendly, responding to specific wake-up commands (i.e., the so-called *utterances*) such as "Alexa, open <*skill name*>" and "Alexa, tell me the weather today". Once activated, an app engages in conversation-based interactions with users and provides user-tailored services. During the conversations, VPA apps may request users' personal information, such as full name and location, raising concerns about user privacy protection.

Over the past few years, the research community has made significant efforts to scrutinize VPA apps' data collection behaviors against their declared privacy policies [20, 38, 40, 42] or requested data access permissions [13, 14]. However, existing studies still fall short in adequately addressing privacy concerns in the context of VPA apps, as users are known to often overlook the privacy policies and permission requests [25, 27, 28]. Dishonest developers can exploit this oversight by exaggerating their true needs in their declared privacy policies, and excessively collecting users' data during at runtime. This put users in an unfair position whenever a data breach occurs, as all data collection behaviors are perceived as being under users' (unintentional) consent and in accordance

with apps' privacy policies. Therefore, we emphasize that *data collection must align with the data required to fulfill the intended functionality*. To address privacy concerns, it is crucial to check the consistency between the personal data needed for the provided functionality (referred to as *data needed*) and the personal data requested by developers (referred to as *data requested*).

Our work. In this work, we conduct the first systematic study on the (in)consistency between data requested and data needed among VPA apps. We propose Pico (Privacy Inconsistency Detector), which focuses on Alexa skills, apps of the most popular VPA service. To avoid the laborious and error-prone task of crafting explicit functionality-data mapping, Pico leverages a key insight that *skills providing the similar functionalities tend to exhibit a similar pattern of data collection*. Pico is designed as a two-phase approach, including *topic clustering* and *consistency checking*. The first phase groups skills into clusters such that those skills in a cluster share similar functionality topics. In the second phase, Pico adopts an unsupervised anomaly detection method. It calculates the anomaly score of the skill based on its behavioral profile, and based on the scores of skills in the same cluster, it finds the appropriate threshold to determine whether the skill is an outlier. In these two phases, Pico mainly addresses the following three key obstacles caused by the uniqueness of the VPA ecosystem.

Obstacle #1: lack of standardized functionality topics. There is an absence of a comprehensive list of functionality topics that formally or informally define the functionalities skills provide. The developer may brief them in the relevant documents of the skill, such as its title, description, and category, but these are often determined at the developer's discretion. Many skills are released without essential user instructions, and some even end up in totally inappropriate categories, such as a real-world example in Fig. 1. Consequently, Pico has to navigate the functionalities.

Obstacle #2: lack of comprehensive documents. In contrast to mobile app stores that are relatively mature, the skill store lacks a stringent and comprehensive governance process for releasing skills' documents. Those documents are often written in natural language with varying lengths, formats, and quality of the information provided. For instance, our preliminary study indicates that around 34.4% skills come with a description of fewer than 30 words.

Obstacle #3: partial availability. Skills are black-box with their source code and backend not accessible to the public [13, 20, 38, 40], rendering it impossible to infer their functionality through program analysis techniques that have been adopted for analyzing mobile or desktop apps [9, 34].

To alleviate **Obstacle #1**, we resort to available textual data of skills to infer their functionality topics. Besides the descriptions available on their homepages, Pico takes into account other domain-unique elements, such as utterances and other functional documents that are specifically targeted to the VPA context, and then applies natural language processing (NLP) techniques to infer the functionality of skills. For each skill topic cluster, Pico provides typical commands and descriptive keywords to comprehensively reflect the functionality in the VPA context (detailed in Section 4.5).

To mitigate **Obstacle #2**, since traditional NLP parsers are limited in their ability to deal with functional documents that vary greatly in length and quality, Pico leverages advanced GPT-based language models [1] to understand the semantics of documents,

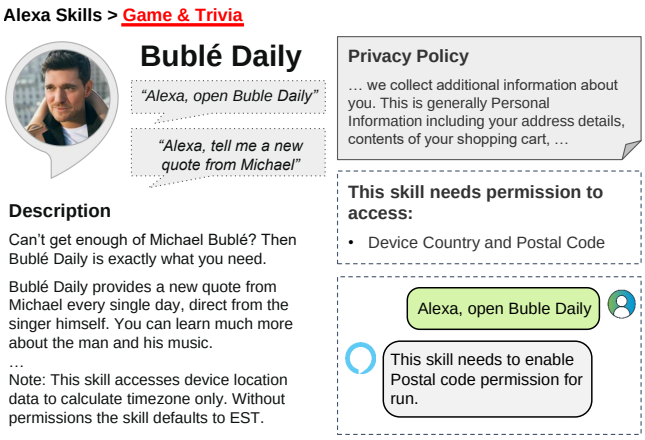


Figure 1: A running example of an Alexa skill page (wrongly released in "Game & Trivia" category by the skill store)

considering its training on a large corpus and its superior ability in natural language understanding [29] (detailed in Section 4.2). For **Obstacle #3**, we obtain the comprehensive personal data requested by Alexa skills from three sources, namely requested permissions, privacy policies, and runtime data collection, to ensure the comprehensiveness and completeness of our analysis.

We conduct a large-scale evaluation of Pico with all 65,195 Alexa skills available in the Amazon skill store, to explore the landscape of the inconsistency issues in real-world VPA apps. Specifically, Pico first performs topic clustering over the collected skills, leading to skill groups of 30 functionality topics. Based on the inferred functionality of skill clusters, Pico manages to find 2,463 out of 11,338 (21.7%) skills that contain behavioral information are suspicious of inconsistency issue between their data needed and data requested. Among the identified issues, users' locations, email addresses, and names are the top three types of personal data involved in over-request behaviors, all of which are highly sensitive. Our study reveals that even among the Top 10 popular Alexa skills [37], there are functionality inconsistent issues found, posing threats to 54,116 users. All these findings unveil the worrying statues of failing to comply with of *purpose limitation principle* of data regulations in the VPA ecosystem.

Contributions. The main contributions of this work are summarized as follows.

- **Investigating the functionality topic inference problem in the Alexa skill ecosystem.** To the best of our knowledge, we are the first work to investigate the functionality topic inference problem in Alexa skills. We group skills by their functional documents and leverage the latest language models into traditional topic modeling techniques.
- **Understanding the over-request issues on a large scale.** We conduct the first comprehensive study on the over-request issues of skills. Our work determines the personal data needed for a skill's functionality by functionality inference, and the personal data requested through multiple sources (i.e., permission, privacy policy, and runtime data collection). Thereby, we detect

inconsistencies between the data needed and data collected of Alexa skills on a large scale.
- **Revealing the *status quo* of VPA apps.** We present the landscape of functional inconsistency issues among Amazon Alexa skills, the apps of the most popular VPA service. Our findings reveal that 21.7% of skills with data collection are suspicious of inconsistency issues. Our findings should raise an alert to both developers and users, and are expected to encourage the VPA service providers to reflect on the *purpose limitation principle* [15] of data regulations like GDPR and in-corporate corresponding regulations into their official developer documentation.

Availability. The source code of Pico and relevant artifacts are available online [2].

2 BACKGROUND

In this section, we introduce the functional documents of an Alexa skill that reflects its data collection. We then use a skill example shown in Figure 1 to demonstrate the inconsistency issue regarding data collection.

2.1 Data Collection of Alexa Skills

Amazon Alexa provides a development ecosystem that is close to the mobile apps market. It allows third-party developers to upload a skill to the Alexa skill store. When developers publish their skills, they are also given an option to declare the permissions needed to access certain personal data, and upload privacy policies of their skills. Developers can disclose their data collection in these two types of documents, although in a voluntary-based manner. Apart from documents, we can also reason a skill's data collection from its runtime behaviors. Therefore, we identify the *documented permissions*, *privacy policy*, and *runtime behaviors* as the three sources of information that reflect a skill's requested data.

Documented Permissions. Alexa skills can require users' personal information to complete its functionality. According to the official requirements of Amazon Alexa [5], developers can configure their skills to request a specific type of permission from users, and to do so, the requested permission information will be listed on the skill page for users to review, e.g., the section with the bold text "This skill needs permission to access" in Figure 1. When running the skill for the first time, users will be asked to go to the Alexa app to grant permissions, and in this way, developers obtain the specific information they want. The full list of 13 subtype permissions including *device address, device country and postal code, email address, Alexa notifications, location services, mobile number, reminders, lists read access, lists write access, first name, full name*, and *Amazon pay*.

Runtime Data Collection. Users can use the example utterances to start the skill, e.g., "Alexa, start Buble daily". Then, skill and user can have verbal communication. During this process, the skill can ask for the user's personal information and record it. We obtain a set of runtime data collection behaviors of Alexa skills from a recent work [38], which defines the data types of runtime data collection types including *location, name, phone, email, birthday, age*, and *postcode*.

Privacy Policy. Since skills may request users' personal information during conversations, due to privacy concerns, Alexa requires developers to release their privacy policy when publishing the skills to the public. In the privacy policy document, skills developers need to honestly declare their data handling practices. Existing research [38] defines the 57 data collection types that appeared in VPA privacy policy documents.

2.2 A Running Example

To better understand our goal, we provide a skill example to demonstrate the inconsistency between the functionality-required data and the actual data collected, which is what Pico aims to accurately detect from a large number of Alexa skills.

As shown in Figure 1, the skill is categorized as a "Game & Trivia" skill (highlighted in red color in the figure). However, by reading its description, we learned that the functionality of the example skill is to provide a quote from a celebrity every day. This mistake, although possibly made by the skill developer, impedes us to analyze the *data needed* of an Alexa skill by merely referring to the categorization of the Alexa skill store. Similar Alexa skills that periodically provide different quotes or customized services tend to be released in the category named "Education & Reference".

Next, we explore the *data requested* of the skill. As shown in Figure 1, its developer mentions in the document that it requires to access location data to calculate the timezone. However, the document cannot convincingly show the necessity to know *where the user locates* to complete its providing-quote functionality. The skill can simply read the system clock to determine the users' local time rather than collecting users' personal data. In this case, we treat its location data collection as suspicious and unnecessary, and therefore, we determine an inconsistency issue exists between the data needed for the skill's functionality and its actual data collection (i.e., data requested). Similarly, any skill like "Buble Daily" that offers similar functionality and asks for the user's location information without a proper justification will be also determined to be suspicious of an inconsistency issue.

3 APPROACH OVERVIEW

The core idea of Pico is to check the inconsistency between the skill's data needed and data requested. Pico first clusters the skills by their functionality-related documents, and then, it checks the inconsistency issues by detecting the outliers in their requested data within each skill cluster. To this end, we propose a two-phase process, namely *topic clustering* and *consistency checking*. The details are shown in Figure 2. We briefly explain the two phases below.

Phase 1: Topic Clustering. This phase aims to cluster the skills with similar topics. To this end, Pico first leverages the latest language model to transform the non-structured natural language inputs into machine-understandable encoding. It then applies a density-based clustering algorithm to group them based on text similarity. To infer the functionality, Pico resorts to the named-entity recognition (NER) technique [21] to extract the typical voice command patterns from skill utterances and adopts the KeyBERT [17] technique to extract the representative keywords from the skills' documents in each topic cluster. This phase is detailed in Section 4.

Phase 2: Consistency Checking. This phase aims to check the functionality consistency between a skill's data needed and data requested. Pico defines benign behaviors of skills according to their

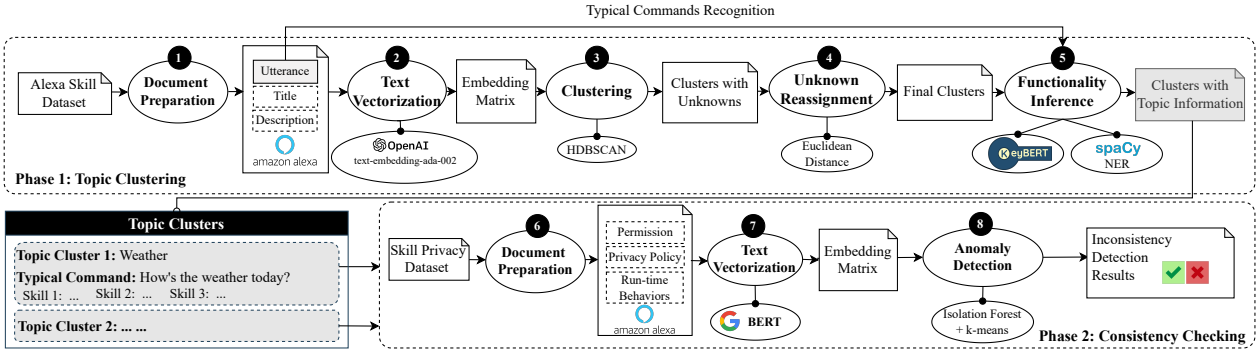


Figure 2: The two-phase workflow of Pico

belonging clusters and determines a skill is suspicious of an inconsistency issue if abnormal behaviors are observed. Specifically, it takes as input the behavioral information, i.e., documented permissions and run-time data collection behaviors, and leverages a state-of-the-art anomaly detection algorithm to detect suspicious skill behaviors. We detail this phase in Section 5.

4 TOPIC CLUSTERING

Pico aims to analyze thousands of skills' functional documents with varying quality. We refer to the existing literature [16, 41] that performs clustering over natural language contents in designing our approach. As a result, Pico adopts a strategy of topic modeling to process skills' documents. It takes the raw documents as input and embeds the documents into vector space. Then, it applies clustering techniques to group the vectors and generates the topic representations by extracting the command patterns and keywords within the documents of each topic.

We propose a five-step approach in the topic clustering phase, which is shown in Figure 3. First, we prepare the contents to be used for the topic clustering from the collected skill documents (Section 4.1). Next, we use a pre-trained language model to convert the natural language contents into embedding vectors (Section 4.2) and then, we leverage an unsupervised clustering algorithm named HDBSCAN to group the skills by clustering the vector representation of the selected documents (Section 4.3). To deal with the skills that cannot be clustered in HDBSCAN, we reassign them into their nearest clusters according to the vector distance (Section 4.4). Last, we adopt NER and KeyBERT techniques to extract the typical commands and representative keywords from documents in each topic (Section 4.5). We detail each step of our approach below.

4.1 Document Preparation

A recent dataset related to Amazon Alexa [37] offers comprehensive details of skills including the skill titles, sample utterances, descriptions, terms and conditions, and download links. Not all contents provided in the dataset are useful for Pico's topic clustering. Thus, we need to identify the useful contents and prepare the *functional documents* for the upcoming clustering.

Existing research on Alexa skills [23] extracts skill functionality phrases from the description text of the skill documents for model-based testing purposes, and they specifically focus on the short phrases in quote marks (e.g., "Alexa, open Buble Daily" in description section in Figure 1), based on the insight that the skill developers tend to use the quote marks to illustrate the valid input in the description section. However, we find that the example quotes are not common for developers to provide in their description text and therefore, cannot sufficiently represent the functionality of a skill in many cases. Instead of only extracting quotations, we take the *entire description section* into account to form the functional documents of a skill. In order to increase the coverage, we also enclose the skill titles and sample utterances into the functional documents and feed it into the clustering task.

4.2 Text Vectorization

In this step, we aim to convert the skills' functional documents written in human natural language into numerical vectors that machines can understand. As described in Section 1, *how to deal with the skill documents written in natural language with varying quality* is one of the challenges we are facing in designing Pico (**Obstacle #2**). Since there is a lack of official standards, functional documents of different Alexa skills vary in length, structure, and content. This led to traditional NLP approaches not performing well with skill documents [8, 16]. The results of document preparation show that the functional documents (i.e., including titles, utterances and descriptions) vary from 41 to 4,193 in length. However, most of clustering algorithms assume that the input data points are in the same dimensions. The heterogeneity of functional documents hinders our application of existing clustering algorithms. To tackle this challenge, we adopt an NLP embedding model called *text-embedding-ada-002*[1] to convert the documents to a fix-length vector and thereby, transform the natural language into text embedding.

The *text-embedding-ada-002* is a pre-trained GPT-based model that has been widely used in a line of research works for understanding natural language [10, 12, 19, 32]. As one of the state-of-the-art large language models, it does not request input text preprocessing such as stop word removal, lemmatization, and stemming, and therefore, can preserve the coherence and semantic relationship

[1]We access the model through the OpenAI API https://openai.com/blog/new-and-improved-embedding-model.

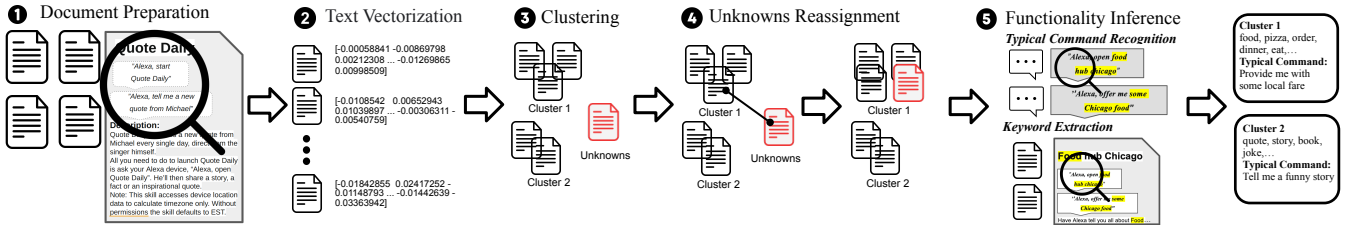


Figure 3: Topic Clustering Phase

within the text to the maximum extent. In this work, the functional document of each skill is converted into a 1536-dimensional numerical vector.

4.3 Clustering

This step aims to group skills with similar functional documents, based on our assumption that Alexa skills with similar semantics in their functional documents have similar functionality in design. Conventional clustering algorithms like k-means are not applicable to this work because they request the number of clusters to be provided in advance, while in this work, Pico aims to explore the number of functional topic clusters and groups the Alexa skills in an unsupervised way. To this end, we leverage the HDBSCAN algorithm [26] to perform the clustering task. By applying the HDBSCAN, Pico can automatically find the optimal number of clusters without manual effort. Besides, HDBSCAN is a density-based algorithm and therefore, is capable to produce clusters with different shapes and densities, which suits our task since the available Alexa skills may not be uniformly distributed by their functionality.

Configuring the Clustering. The adoption of unsupervised clustering enables us to cluster Alexa skills without specifying the number of clusters in advance. However, we still configure the HDBSCAN algorithm to minimize the number of "unknowns" (i.e., skills that failed to be clustered). We configure HDBSCAN with two parameters, namely `min_cluster_size` and `min_samples`, which specify the minimum reasonable number of clusters to be produced, and the minimum number of data points to constitute a cluster, respectively. To set the former, we refer back to the Amazon Alexa skill store [4] and count the number of skill categories officially defined by Amazon (e.g., Food & Drink, Games & Trivia, Health & Wellness). We find 23 categories from the skill store and as a result, we set the minimum number of clusters as 23. For the latter, we assign a comparably small number 2 to the `min_samples` to help the HDBSCAN algorithm find its optimal number of clusters and meanwhile, avoid producing lonely "unknown" data points without sharing any semantic similarity.

4.4 Unknowns Reassignment

As mentioned in the previous step, the skill clustering has been configured to minimize the number of unknowns. However, those unknowns cannot be completely eliminated when using the HDBSCAN algorithm. To obtain a complete topic clustering result, we reassign those unknowns to their nearest clusters based on their semantic context, which is represented by the embedding vectors.

We resort to calculating the Euclidean distance between the unknowns and the remaining skill vectors that have been successfully clustered in the last step. We then re-assign each unknown data point to the cluster where its nearest neighbor vector belongs to. Recall that the embedding vector of each skill document is produced by a language model, a small distance between a pair of vectors implies a strong relevance between the semantic contexts. Our reassignment strategy is accordingly proposed based on an insight that an unknown-labeled skill, if necessary, shall be categorized into the group with the most relevant functionality description. As a result, every skill has been assigned to a topic cluster.

4.5 Functionality Inference

The clustering of the vectors considers the relevance of both the embedding tokens (i.e., features of the vectors) and the latent semantic contexts. However, we still need a human-readable representation to explain the clustering outcomes and facilitate the subsequent consistency checking.

In this step, we aim to infer the functionality for each skill cluster. To get the comprehensive and complete functionality inference information from skill functional documents, Pico extracts the key information from two aspects, namely typical commands recognition and keyword extraction.

Typical Commands Recognition. Sample utterances are voice command patterns specifically made for skills. These utterances show users how to use voice to start a skill, and often match what this skill can do. For example, as shown in Fig. 1, utterances showed in dialog boxes (e.g., "Alexa, tell me a new quote from Michael") can partially unveil the functionality of the skill "Buble Daily".

To find the functional voice command patterns from skill utterances, Pico resorts to named-entity recognition (NER) to label entities within utterances. Since the existing NER models are not designed to be domain-specific for skill utterances, they may not perform optimally in that context. Therefore, we enhance the spaCy's NER engine [21] and fine-tune it with our own training data.

As shown in Table 1, we first summarize five common utterance patterns that are frequently found in Alexa skills in the labeling process, where the key entity (shown as $[E]$) contains the key command patterns we want. Then, for each pattern, we randomly select 30 utterances that matched the specific features from our dataset as training data. Multiple rounds of training are performed until the loss rate begins to converge. This approach has achieved promising results in other domains [6].

Table 1: Skill utterance patterns and example commands

#	Utterance Patterns†	Example Commands
1	<Alexa, [E]>	Alexa, open the door
2	<Alexa, [V] {$Title$}>	Alexa, open math practice
3	<Alexa, [V] {$Title$} [E]>	Alexa, ask Unofficial Ripple the current price
4	<Alexa, [V] my [E]>	Alexa, play my Flash Briefing
5	<Alexa, [IW] [V] [E]>	Alexa, when is water bill due?

† (V) stands for verbs, including modal verbs and 'be' verbs, (E) stands for key entity of skill utterances, and (IW) stands for interrogative words.

Keyword Extraction. In addition to the typical commands recognition, Pico also leverages a state-of-the-art lightweight NLP technique called KeyBERT to extract keywords from the functional documents for each skill cluster (i.e., functionality topic). To ensure the produced keywords properly reflect the functional uniqueness, Pico needs to sanitize the functional documents of skills in the same cluster. Specifically, it cleans the documents by removing prepositions, conjunctions, and auxiliary verbs that are commonly written in the documents but irrelevant to the functional keywords extraction. Moreover, we also maintain a set of irrelevant words that high-frequently appear in skill documents, such as "Alexa", "Amazon", "next", "newest" and "skill". Then, Pico resorts to KeyBERT with its default model named *all-MiniLM-L6-v2* to generate a keyword list. In the end, Pico normalizes the extracted keywords by lemmatization. The extracted keywords, together with the recognized typical commands, will be associated with the corresponding cluster for the upcoming consistency checking.

5 CONSISTENCY CHECKING

Through Phase 1 (Section 4), we have received the clustering results of the 65,195 skills from the Alexa skill store. In this phase, Pico evaluates the functionality consistency of an Alexa skill through the lens of anomaly detection, based on the assumption that a skill is suspicious of inconsistency issues if its requested data is detected as outliers when compared with other counterparts within the same cluster. We design the consistency checking as a two-step process. First, we pre-process the skill documents to get a clean text set only containing the data collection and relevant behaviors, and use NLP techniques to convert the text data to embedding vectors. We detail this step in Section 5.1. Next, we leverage an unsupervised anomaly detection algorithm named *isolation forest* [24] to assess the *anomaly score* of each skill that reflects the anomalous degree of its requested data. In the end, we determine the threshold based on the anomaly scores and accordingly assess if a skill is anomalous in the requested data. This step is detailed in Section 5.2.

5.1 Document Preparation & Text Vectorization

This step aims to extract the data requested of Alexa skills and convert them into a machine-understandable format to facilitate the upcoming anomaly detection. As mentioned in Section 1, one of the main challenges that Pico is facing is that *skills are black-box with their source code not accessible to the public* (**Obstacle #3**). To deal with this challenge and ensure the completeness and comprehensiveness of the analysis, we consider three types of data requested information of an Alexa skill, namely the *documented permissions*, *privacy policy*, and *runtime collection behaviors*.

Documented Permissions. The documented permissions refer to the list of permission declared by developers in their skill documents. Not all permissions of Alexa skills are related to personally sensitive data collection, so we resort to a recent study [40] that summarizes 22 personally identifiable information (PII) in the context of VPA apps, and accordingly tighten the scope of the documented permissions to permissions of eight pre-identified data types.[2] Next, we collect the list of declared permissions from the permission section of each skill document in the dataset [37] and intersect the list of claimed data with the eight pre-identified data types to obtain the documented permissions of each skill.

Privacy Policy. We consider the privacy policy of skills as another source of documents that implies what type of personal data to be collected. For that reason, Pico also evaluates the consistency of a skill's potential requested data between the functional documents and what is stated in the privacy policy. We resort to a recent Alexa skill privacy dataset [38], which conducts large-scale privacy compliance testing on Alexa skills and extracts 57 types of requested data from the skills' privacy policy documents. Not all of them are related to sensitive personal data collection. Again, by taking the intersection set with PII, we narrow down the range of personal sensitive data collection types that appeared in skill's privacy policy to 14 types.[3] We note that at the moment of this work being carried out, releasing the privacy policy of a skill is voluntary-based to its developer. As a result, our consistency checking based on privacy policy only covers Alexa skills that are released together with a developer's privacy policy.

Runtime Data Collection. Compared with documented permission, the runtime data collection can better reflect the actual data collection of Alexa skills. We again resort to the skill privacy dataset [38] to collect the seven types of runtime collection behaviors from large-scale dynamic testing.

After obtaining three types of requested data, Pico leverages the BERT [11] model to decode each skill's behavioral data to numerical vectors. The reason why we consider using NLP techniques here is that Pico needs to understand the short phrases in skill permission data with similar semantics (e.g., *device address* and *location services*). Also, converting behavioral information into an embedding vector allows for a consistent representation of data, making it easier for our consistency checking tasks.

5.2 Anomaly Detection

Pico uses the *isolation forest* [33] algorithm to detect the skill with suspicious data collection compared to its counterpart skills with similar functionality. Isolation forest is a widely-used unsupervised anomaly detection method. It can provide an anomaly score for each skill that reflects the degree of anomaly. Conventional isolation forest tasks specify the data points with a negative score as outliers, and the remaining data points are considered normal [33]. Pico adopts the isolation forest algorithm because it scales well

[2]Including device address, device country and postal code, email address, location services, mobile number, first name, full name, and Amazon pay.
[3]Including name, email, phone number, birth date, age, gender, location, phonebook, income, social security number, credit card, postcode, occupation, and Amazon pay.

Algorithm 1 Find Heuristic Threshold by K-means

```
   Input: anomaly_scores
   Output: heuristic_threshold
 1: function HEURISTIC_THRESHOLD(anomaly_score, n_clusters = 3)
 2:     n_clusters ← MIN(n_clusters, LEN(anomaly_scores))
 3:     k_means ← KMEANS(n_clusters)
 4:     cluster_label ← k_means.fit_predict(anomaly_scores)
 5:     cluser_avg_scores ← [ ]
 6:     for each i in range(n_clusters) do
 7:         cluster_scores ← [ ]
 8:         for each j, label in enumerate(cluster_labels) do
 9:             if label = i then
10:                 ADD anomaly_scores[j] in cluster_scores
11:             end if
12:         end for
13:         ADD AVG(cluster_scores) in cluser_avg_scores
14:     end for
15:     heuristic_threshold ← MIN(cluser_avg_scores)
16:     return heuristic_threshold
17: end function
```

with different data sizes, which means it can effectively detect the anomaly in both small and large datasets.

Fine-grained Anomaly Detection. The adoption of the isolation forest brings us ease to quantitatively assess the anomalous degree of a data point within a cluster. However, we notice that determining a skill's requested data as an anomaly merely because of a negative anomaly score creates a lot of false positive predictions. For that reason, we need to find the *heuristic threshold* from the given anomaly scores of skills of each cluster to determine the anomalies. To this end, Pico adopts a fine-grained anomaly detection strategy by performing an additional *k-means* clustering over the anomaly scores. The refinement process aims to further cluster the anomaly scores of the analyzed skills[4] and precisely identify the anomalous requested data from the outliers produced by isolation forest (i.e., negative scores). We briefly present this process in Algorithm 1.

Finding a proper k value of the k-means method is the main challenge in this step. Since we aim to shrink the scope of anomaly skills from the outliers produced by isolation forest, a k value greater than 2 is necessary (otherwise there is no clustering at all). We also consider that a too large k may lead to the growth of false negative cases, undermining the overall performance of Pico. Therefore, we adopt a conservative approach and set $k = 3$ for the refinement. We begin with clustering anomaly scores by k-means (lines 2-4 of Algorithm 1). Next, we calculate the average anomaly scores in each cluster (lines 6-14), i.e., the centroid of each cluster, and then choose the minimum average score, i.e., the centroid of the most anomalous skill cluster, as the heuristic threshold (line 15).

After we find the heuristic threshold in each cluster, we assign the label to each skill. If the anomaly score is less than or equal to the threshold, Pico assigns a label of 1 (outlier); otherwise, Pico assigns a label of 0 (benign).

6 EVALUATION AND LANDSCAPE

We implement Pico and evaluate it on Amazon Alexa skills. Our evaluation aims to study the topic clustering performance and overall performance of Pico. We are also interested in understanding the landscape of functionality consistency in real-world VPA apps. We target to answer the following three research questions (RQs).

RQ1. What is the Pico's performance in functional topic clustering?

RQ2. What is the Pico's performance in identifying functionality inconsistency issues?

RQ3. Based on Pico's findings, what is the status quo of functionality inconsistency in existing skills?

6.1 RQ1: Topic Clustering Performance

To answer RQ1, we first evaluate the ability of Pico to accurately cluster topics from functional documents in the context of VPA. We benchmark Pico's performance and compare it with different combinations of embedding models and clustering approaches that have been adopted in existing studies.

Baseline. For the embedding model, in addition to the *text-embedding-ada-002* model from OpenAI (shortly written as OpenAI), we also evaluate a pre-trained BERT model named *all-mpnet-base-v2* as the baseline because it is applied in relevant literature [18] for the topic modeling purpose. For the clustering, we compare the proposed HDBSCAN with the k-means algorithm as it is adopted in recent research such as [16, 41]. As a result, we set up 3 baseline methods for topic clustering, which are the combinations of embedding and clustering approaches other than Pico adopts, namely BERT+HDBSCAN, BERT+k-means, and OpenAI+k-means.

Experimental Setup. Our evaluation is carried out based on the UQ-AAS21 dataset [37] that contains comprehensive details of 65,195 skills crawled from the Alexa skills store. When performing text vectorization, our approach leverages an OpenAI model and therefore the embedding task is performed online through the OpenAI server. Our evaluation of the BERT baseline is performed by reproducing the Bertopic framework project on our local machine. For the clustering step, we use the same parameters as described in Section 4.3 to evaluate HDBSCAN algorithm. In the evaluation of the k-means baseline, we refer to the existing literature that performs clustering on Android apps [16, 41] to find the optimal k value. Considering VPA apps share similar functionality topics with mobile apps, we use the same configuration, i.e., $k = 30$, in our evaluation. All experiments are running on an AMD Ryzen Threadripper PRO 5965WX PC with 250GB memory.

We involve two volunteers from our research lab to annotate the benchmark skills. They have never been introduced to the internal structure of Pico. For each skill in the clusters, they are asked to read its functional documents, and determine whether this skill is correctly clustered. In the end, we organize a discussion to resolve the conflicts that exist in their annotation results. We evaluate the accuracy of topic clustering by calculating the percentage of correctly clustered skills, i.e. *the number of skills that correctly clustered / total number of skills in the cluster*, in the five largest clusters.

Results. After receiving the clustering results, we rank the clusters by the number of skills they contain and output the five largest clusters. We construct the evaluation dataset by randomly picking 5% of skills from the five clusters.

Table 2 presents Pico's topic clustering performance. We highlight the best performance in each column that represents accuracy. Our findings show that Pico outperforms the three baselines in almost all clusters. It brings improvement of 43.73%, 28.40%, and 3.12%

[4]We remark that the clustering task in Phase 2 is different from the topic clustering in Phase 1, which is implemented to categorize Alexa skills according to their functional documents

Table 2: Topic clustering accuracy of Pico and the three baselines (including the five largest clusters produced by each approach and overall accuracy)

Approaches	C1		C2		C3		C4		C5		Average Accuracy
	Wrongly clustered # (out of total #)	Accuracy	Wrongly clustered # (out of total #)	Accuracy	Wrongly clustered # (out of total #)	Accuracy	Wrongly clustered # (out of total #)	Accuracy	Wrongly clustered # (out of total #)	Accuracy	
BERT+k-means	269 (540)	50.19%	199 (520)	61.73%	323 (430)	24.88%	242 (396)	38.89%	95 (292)	67.47%	48.21%
OpenAI+k-means	98 (259)	62.16%	116 (238)	51.26%	86 (234)	63.25%	86 (226)	61.95%	38 (206)	81.55%	63.54%
BERT+HDBSCAN	14 (243)	94.24%	19 (182)	**89.56%**	24 (170)	84.17%	19 (120)	85.47%	17 (117)	85.88%	88.82%
OpenAI+HDBSCAN (Pico)	4 (117)	**96.58%**	15 (112)	86.61%	8 (102)	**92.16%**	3 (98)	**96.94%**	12 (92)	**86.96%**	**91.94%**

Table 3: Confusion matrix of consistency checking of the 119 sampled skills with data requested

		Truth	
		Inconsistent	Consistent
Prediction	Outlier	27 (TP)	5 (FP)
	Benign	18 (FN)	69 (TN)

in the average accuracy to the three baselines, namely BERT+k-means, OpenAI+k-means, and BERT+HDBSCAN, respectively.

6.2 RQ2: Inconsistency Detection Performance

Experimental Setup. Since there is not a benchmark available in the existing research, we proceed with constructing one for our study. To explore the landscape of Alexa skills inconsistency issues, we randomly sample 5% of skills from a total of 65,195 skills and analyze their data requested. As a result, we preliminary benchmark 3,260 skills, and among them, 119 skills contain data requested behavioral information. We then label these skills to create ground truth. To avoid bias, we again invite two volunteers to label the data. For each analyzed skill, both volunteers are asked to read its functional documents and the behavioral information to independently determine whether it is suspicious of inconsistency issues. We then organize a discussion to resolve their disagreements, and thus, obtain the ground truth about the inconsistency issues from the sampled skills.

Results. We evaluate the Pico's inconsistency detection of the 119 sampled skills containing data requested, and compare the detection outcomes with the human annotations. We present our comparison in a confusion matrix in Table 3. The results show that Pico successfully detects 27 skills with inconsistency issues and achieves 84% (27/32) on precision, which means Pico can identify the positive instances (outliers) accurately with few false positives.

Next, we look into the 18 missed detections (false negatives) to reason the cause. After reviewing skill details, we find 3 false negatives are missed in Pico's detection because they are wrongly clustered in Phase 1. The remaining 15 are due to the prevalence of over-request issues in the majority of skills within the corresponding cluster, misleading Pico into classifying them as normal data collection. Considering the 15 missed detections are because of the prevalence of over-request issues rather than the technical fault of Pico, we calculate the performance metric of Pico once again by excluding them from false negatives, and then record a 90% (27/30) recall and 93% (96/104) accuracy in detecting inconsistency issues.

Prevalence of Over-request Issues. Our benchmark results unveil the existence of over-request issues in one skill cluster, which greatly worries us because it violates the *purpose limitation principle* of GDPR. We then investigate skills in all clusters and find the over-request issues are prevalent in multiple skill clusters. For example, we find 30 out of 32 skills with data requested in one cluster are published by one developer. The involved cluster is about the sports topic indexed with keywords "nba", "tonight", "conference", and "season" and typical command "Alexa, ask the *<skill name>*, when is the next game". The 30 skills' functionality is to provide NBA-related information.[5] All these skills request personal sensitive data "*Device Country and Postal Code*", although this location information is not necessary according to the skill documents, nor in the remaining skills in the same functional topic. We will discuss more this issue in Section 8.

6.3 RQ3: Status quo of Amazon Alexa skills

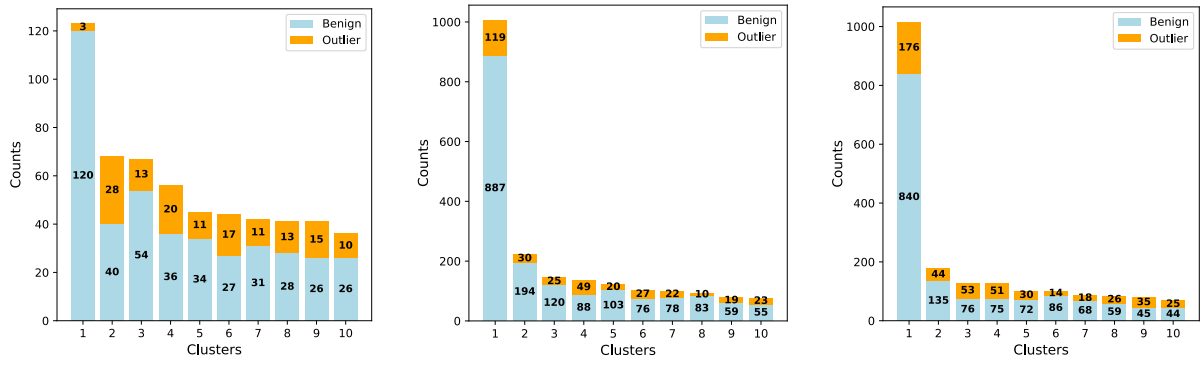
In RQ3, we apply Pico to the large-scale study of Alexa skills. We first follow Section 5.1 to create the detection dataset, and then filter out the skills that do not have data requested information.

To check the permission inconsistency, we select 2,254 skills that contain permission information from all 65,195 skills collected. We then exclude the skill that only declares the permissions irrelevant to the personal information collection, and eventually identify 1,498 skills. To check the runtime inconsistency, we refer to the skills with runtime data collection from the dataset [38] and obtain 5,377 skills' data. For privacy policy checking, we obtain 4,463 skills that claim their data handling practices with detailed data types in privacy policy documents.

Landscape of (In)consistency. Pico manages to detect 376 skills (25.1% of skills that provide permission information) are suspicious of permission inconsistency issues, 1,000 skills (18.6% of skills with runtime data collection behaviors) are suspicious of runtime behaviors inconsistency issues, and 1,087 skills (24.4% of skills that disclose collected data type in their privacy policy) are suspicious of privacy policy inconsistency issues. We present the consistency detection results of the ten largest clusters in Figure 4, in which the three sub-figures show the consistency landscape of skills from the perspectives of documented permissions, runtime data collection, and privacy policy.

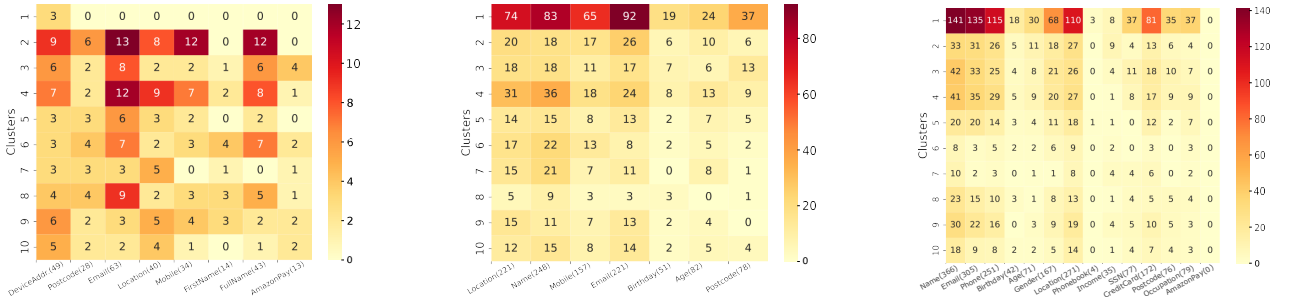
For skills with documented permissions, Table 4 displays more details about the ten largest clusters. Combining with Figure 4(a), we observe that the largest topic cluster "radio" (cluster #1), which is made up of 123 skills, shows the best compliance in data collection as only 2.4% (3 out of 123) skills demonstrate inconsistency

[5]One example skill named "Houston Rockets" provides utterances like: "Alexa, open the Rockets; Alexa, ask the Rockets who's winning tonight, etc". The rest of the similar skills only change the NBA team name in their functional documents including skill title, utterances, and description.



(a) Consistency landscape in documented permission (b) Consistency landscape in runtime data collection (c) Consistency landscape in privacy policy

Figure 4: Consistency landscape of Alexa skills (skill count vs topic cluster number)



(a) Inconsistency issues by data types in documented permission (b) Inconsistency issues by data types in runtime data collection (c) Inconsistency issues by data types in privacy policy

Figure 5: Distribution of inconsistency data types of Alexa skills (topic cluster number vs data type)

Table 4: Functionality inference from the ten largest topic clusters within skills containing documented permission

#	Topic	Representative keywords	Key entities extracted from typical commands
1	Radio	play, listen, station, radio, start	recommend a station, play radio
2	Property	home, estate, house, agent, buyer	check my loans, house worth, book a home showing
3	Podcast	podcast, voice, launch, forecast, interview	play a podcast, get show
4	Food	restaurant, search, finder, order, location	find me a snack, for a place to eat, pick a restaurant
5	Weather	weather, forecast, wear, wind, temperature	the weather, wear today, clothes
6	Quiz	quiz, question, voice, learn, policy	start quiz, the question of day
7	Smart Home	voice, light, temperature, alert, air	indoor air quality, turn on lights, turn off
8	Contact	conference, contact, location, appointment, community	give contact information, nearby location
9	Vehicles	vehicle, car, price, remotely, station	pick up vehicle, start my car
10	Event	event, ticket, calendar, weekend, tomorrow	list the events, find tickets

Table 5: Functionality inference from the ten largest topic clusters within skills containing runtime data collection

#	Topic	Representative keywords	Key entities extracted from typical commands
1	Smart Home	device, light, enable, control, switch	turn off, turn on, the temperature
2	Business	briefing, flash, news, podcast, business	flash briefing, market update
3	Radio	listen, voice, playlist, radio, play	play forecast, play info
4	Information	briefing, news, daily, headline, technology	flash briefing, in the news
5	Entertainment	quiz, movie, open, game, dialogue	play movie, start quiz
6	Camera	camera, view, door, lock, tv	start recording, show the room
7	Quote	life, inspiration, daily, quote, motivation	flash briefing, give quotes
8	Local	news, latest, weather, radio, sport	flash briefing
9	Payment	account, credit, balance, payment, loan	summarize finance, bills due
10	Fitness	workout, exercise, routine, yoga, fitness	log exercise, suggest a workout

issues. The majority of permission inconsistency issues occur in topic cluster #2, which focuses on property indexed with keywords "home", "estate" and typical command "Alexa, what is my house worth". 41.1% (28 out of 68) of the skills are found suspicious of inconsistency issues.

Table 5 displays more details about the ten largest clusters within skills containing runtime data collection behaviors. Together with 4(b), we find that topic cluster #8 (assigned topic "local") has the lowest ratio of inconsistency issues, which is 10.8% (10/93). Cluster

Table 6: Functionality inference from the ten largest topic clusters within skills mentioning requested data in privacy policies

#	Topic	Representative keywords	Key entities extracted from typical commands
1	Smart Home	device, light, enable, control, switch	turn on, turn off discover devices
2	Business	briefing, flash, news, podcast, business	flash briefing, full report
3	Radio	podcast, radio, playlist, audio, station	play, pause forecast
4	Daily	briefing, flash, news, daily, update	flash briefing, in the news
5	Quiz	quiz, question, game, open, play	start quiz, play game
6	Camera	camera, video, door, view, lock	turn on camera, show my front door
7	Bank	account, financial, credit, payment, banking	get balance, expense report
8	Local	briefing, flash, news, weather, radio	flash briefing, in the news
9	Marketing	marketing, podcast, business, news, tip	flash briefing, in the news
10	Music	music, radio, play, station, stream	play radio

#4 with a topic of "information" has the highest ratio of inconsistency issues at 37.5% (49/137), followed by cluster #10 (assigned topic "fitness") with an inconsistency ratio of 29.5% (23/78).

Table 6 displays more details about the ten largest clusters within skills containing requested data in the privacy policy. Cluster #9 with a topic of "marketing" results in the highest inconsistency ratio, which is 43.8% (35/80). Cluster #6 (topic "camera") has the lowest skill number with inconsistency issues, which is 14% (14/100).

Characterization of Inconsistency. We further conduct an investigation into the identified inconsistency cases to explore their characteristics. Figure 5 displays the distribution of inconsistency issues by data types in the ten largest clusters in documented permission, runtime data collection, and privacy policy.

For documented permissions (see Figure 5(a)), Pico detects the highest number of inconsistency issues in collecting users' email addresses (63), followed by device addresses (49) and full names (43). The fewest inconsistency issues are found in the collection of Amazon Pay details (13). Among the inconsistency issues involving email data collection, the skill group focused on "property" (cluster #2) records the highest number (13), and the "food" cluster (#4) ranks second place (12).

For runtime behaviors (Figure 5(b)), most of the inconsistency cases are caught in name collection (248), followed by email collection (221) and location collection (221). The fewest runtime inconsistencies are relevant to the birthday information collection (51). Among the inconsistencies involving name collection, the skill cluster "smart home" (#1) records the highest number at 83.

For privacy policy inconsistency issues (Figure 5(c)), most of the inconsistency cases fall in name collection (366), followed by email (305). Cluster #1 (topic "Smart Home") has the most number of inconsistency issues, focusing on the collection of name, email, phone and location.

Overall, we can conclude that users' locations, emails and names are the data that third-party developers prefer to over-request. In addition, users' phone numbers and location information are also heavily suffered from the over-request issues. Our research reminds users to pay attention to protecting their privacy and carefully determine whether such personal data is truly needed for the skill's functionality. We also advise developers to restrain their data collection and avoid over-requesting users' private data beyond the functionality needs.

7 DISCUSSION

Our work reveals that the current situation of Alexa skills on the consistency between the skill's data needed and data requested is not satisfying, with 2,463 inconsistency issues (21.7%) found within 11,338 skills that have personal data collection. Most of the inconsistency issues happened because of the short and unclear functional documents, especially the skill descriptions. During the investigating process, we find that the quality of skill description is worrisome with around 34.4% skills provide a description consisting of less than 30 words, among them 4,448 skills even come with a description of less than 10 words. VPA service providers should pay attention to this issue and encourage developers to provide well-formatted and meaningful descriptions. To comply with the *purpose limitation principle* of GDPR, we also suggest service providers update their developer documentation, ensuring that third-party developers have the awareness of only collecting necessary data in their VPA apps. They should also enforce this in the vetting process and introduce the document quality assessment mechanism to regulate skill release process.

Our evaluation also reveals the homogenization of the sample utterances provided by skill developers. Some skills' utterances are insufficient to reflect their functionality, and as a result, impedes our functionality inference. For example, in Table. 4, Table. 5 and Table. 6, we observe multiple functional clusters are featured by a command with key entity "flash briefing", although their representative keywords show that these skills discuss different topics. We then notice that "Alexa, what's my Flash Briefing?" is a template utterance used when creating *flash briefing skills*, which are a special type of skills that allow users to customize the multiple news sources for short updates [3]. Considering there is a large number of flash briefing skills available in the skill store, we recommend that Alexa skill developers should focus on diversifying the utterances to enhance the usability of the skills.

8 LIMITATIONS

Pico focuses on the inconsistency issues between VPA app's data needed and data requested. To the best of our knowledge, we propose the first study that performs a comprehensive end-to-end analysis in the VPA domain. However, as an initial effort in this field, the current work of Pico has a few limitations that could be addressed and enhanced in the future.

First, Pico uses similar skill groups to find the skill's necessary required data. As shown in Section 6.2, the method could fail when the majority of similar skill groups have data over-request behaviors. In this situation, over-request skills can escape from Pico's detection, putting the normal skill as a false positive.

Second, the adoption of unsupervised clustering and anomaly detection algorithms in Pico can result in unstable outcomes due to the inherent nature of these techniques. This variability can also lead to inaccurate results in the detection phase. We plan to investigate potential improvement techniques in future work.

Third, the app's conversation log may be incomplete. Further enhancement such as fuzzing [30] can be applied to comprehensively capture the app's actual behaviors.

9 RELATED WORK

Checking App Behaviors Against App Documents. A line of research [7, 8, 34, 36] has found inconsistency issues between privacy policies and mobile (i.e, Android) app behaviors. POLICYCOMP [43] studies whether personal data collection practices in an app's privacy policy are over claimed from the perspective of counterpart comparison. Gorla et al. [16, 41] detect suspicious behaviors in mobile apps on the Android market by grouping the apps according to their descriptions. They use the traditional topic modeling strategy, LDA, to extract the topics. Following this, Peddinti et al. [31] propose an algorithmic mechanism to find the required permissions for a specific mobile app by computing the permissions from a set of similar apps. The key differences between Android and VPA apps are two-fold. First, Android apps have standardized documents and rich permissions, while the quality of skill documents can vary. Second, multiple sources exist for us to obtain Android app packages and analyze their data collection, while the implementation of Alexa skills remains a black box. These factors limit the applicability of previous Android-based methods to be reused on skills.

Analysis of Skill Data Handling Practices. Edu et al. [13, 14] traced the changes in privacy policy and documented permissions of Alexa skills over three years. Yan et al. [39] conducted the systematic study on the quality of privacy policies in the Alexa skill domain. A line of research [14, 20, 22, 23, 38, 40] uses chat-bot-like testing to explore the skill data collection behaviors and check privacy compliance issues. Li et al. [23] employed model-based testing to extract the functionality from skill descriptions using the TF-IDF algorithm. In Pico, we use the latest NLP models to extract functionality-related keywords after topic clustering, instead of directly extracting them from a single document. Existing literature addresses privacy compliance issues in Alexa ecosystem. However, there is still lacking of research to explore the inconsistency issues between skills' data needed and data requested.

10 CONCLUSION

Pico aims to automatically detect the inconsistency issues between skills' data needed and data requested. The main idea of Pico is to cluster the skills by their semantics of functional documents, and then detect the suspicious data collection from each cluster. Our work reveals the *status quo* of the functional inconsistency issues in these emerging voice-based VPA apps. We remark that Pico is a preliminary work in the direction of functionality consistency, and more research is desirable in the future to address the challenges we identified.

ACKNOWLEDGEMENT

We thank the anonymous reviewers for their insightful comments to improve this manuscript. This work is partially supported by Australian Research Council Discovery Projects (DP230101196, DP240103068).

REFERENCES

[1] 2023. *OpenAI/Models.* Retrieved April 7, 2023 from https://platform.openai.com/docs/models/gpt-3-5
[2] 2024. *PICO.* Retrieved Janurary 9, 2024 from https://github.com/UQ-TrustLab/PICO
[3] Amazon Alexa. 2023. *Set Up News and Flash Briefings for Alexa.* https://www.amazon.com/gp/help/customer/display.html?nodeId=GXMFWZJ8FKRGLFFU
[4] Amazon. 2023. *Alexa skills.* Retrieved January 7, 2023 from https://www.amazon.com.au/Alexa-Skills/b?node=4931595051
[5] Amazon Developer Documentation. 2022. *Configure Permissions for Customer Information in Your Skill.* https://developer.amazon.com/en-US/docs/alexa/custom-skills/configure-permissions-for-customer-information-in-your-skill.html
[6] Benjamin Andow, Samin Yaseer Mahmud, Wenyu Wang, Justin Whitaker, William Enck, Bradley Reaves, Kapil Singh, and Tao Xie. 2019. PolicyLint: investigating internal privacy policy contradictions on google play. In *28th USENIX security symposium (USENIX security)*. 585–602.
[7] Benjamin Andow, Samin Yaseer Mahmud, Justin Whitaker, William Enck, Bradley Reaves, Kapil Singh, and Serge Egelman. 2020. Actions Speak Louder than Words: Entity-Sensitive Privacy Policy and Data Flow Analysis with POLICHECK. In *29th USENIX Security Symposium (USENIX Security)*.
[8] Duc Bui, Yuan Yao, Kang G. Shin, Jong-Min Choi, and Junbum Shin. 2021. Consistency Analysis of Data-Usage Purposes in Mobile Apps. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security* (Virtual Event, Republic of Korea) *(CCS)*. Association for Computing Machinery, New York, NY, USA, 2824–2843.
[9] Yunang Chen, Mohannad Alhanahnah, Andrei Sabelfeld, Rahul Chatterjee, and Earlence Fernandes. 2022. Practical Data Access Minimization in Trigger-Action Platforms. In *31st USENIX Security Symposium (USENIX Security)*. Boston, MA, 2929–2945.
[10] Ioana Ciuca and Yuan-Sen Ting. 2023. Galactic ChitChat: Using Large Language Models to Converse with Astronomy Literature. *Research Notes of the AAS* 7 (09 2023), 193.
[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *North American Chapter of the Association for Computational Linguistics*.
[12] Jan Digutsch and Michal Kosinski. 2023. Overlap in meaning is a stronger predictor of semantic activation in GPT-3 than in humans. *Scientific Reports* 13, 1 (2023), 5035.
[13] Jide Edu, Xavi Ferrer Aran, Jose Such, and Guillermo Suarez-Tangil. 2021. SkillVet: Automated Traceability Analysis of Amazon Alexa Skills. *IEEE Transactions on Dependable and Secure Computing* (2021).
[14] Jide Edu, Xavier Ferrer-Aran, Jose Such, and Guillermo Suarez-Tangil. 2022. Measuring Alexa skill privacy practices across three years. In *Proceedings of the ACM Web Conference 2022*. 670–680.
[15] European Parliament. 2020. *General Data Protection Regulation (GDPR).* Retrieved July 26, 2023 from https://gdpr-info.eu/
[16] Alessandra Gorla, Ilaria Tavecchia, Florian Gross, and Andreas Zeller. 2014. Checking app behavior against app descriptions. In *Proceedings of the 36th international conference on software engineering (ICSE)*. 1025–1035.
[17] Maarten Grootendorst. 2020. KeyBERT: Minimal keyword extraction with BERT. https://doi.org/10.5281/zenodo.4461265
[18] Maarten R. Grootendorst. 2022. BERTopic: Neural topic modeling with a class-based TF-IDF procedure. *ArXiv* abs/2203.05794 (2022).
[19] Edward Guo, Mehul Gupta, Sarthak Sinha, Karl Rössler, Marcos Tatagiba, Ryojo Akagami, Ossama Al-Mefty, Taku Sugiyama, Phillip E Stieg, Gwynedd E Pickett, et al. 2023. neuroGPT-X: Towards an Accountable Expert Opinion Tool for Vestibular Schwannoma. *medRxiv* (2023), 2023–02.
[20] Zhixiu Guo, Zijin Lin, Pan Li, and Kai Chen. 2020. Skillexplorer: Understanding the behavior of skills in large scale. In *29th USENIX Security Symposium (USENIX Security)*. 2649–2666.
[21] Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. *To appear* 7, 1 (2017), 411–420.
[22] Tu Le, Danny Yuxing Huang, Noah Apthorpe, and Yuan Tian. 2022. SkillBot: Identifying Risky Content for Children in Alexa Skills. *ACM Trans. Internet Technol.* 22, 3, Article 79 (July 2022), 31 pages.
[23] Suwan Li, Lei Bu, Guangdong Bai, Zhixiu Guo, Kai Chen, and Hanlin Wei. 2022. VITAS: Guided Model-based VUI Testing of VPA Apps. In *37th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. 1–12.
[24] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation forest. In *8th IEEE international conference on data mining*. IEEE, 413–422.
[25] Aleecia M McDonald and Lorrie Faith Cranor. 2008. The cost of reading privacy policies. *I/S: A Journal of Law and Policy for the Information Society (ISJLP)* 4 (2008), 543.
[26] Leland McInnes, John Healy, and Steve Astels. 2017. hdbscan: Hierarchical density based clustering. *J. Open Source Softw.* 2, 11 (2017), 205.

[27] Anca Micheti, Jacquelyn Burkell, and Valerie Steeves. 2010. Fixing broken doors: Strategies for drafting privacy policies young people can understand. *Bulletin of Science, Technology & Society* 30, 2 (2010), 130–143.

[28] Jonathan A Obar and Anne Oeldorf-Hirsch. 2020. The biggest lie on the internet: Ignoring the privacy policies and terms of service policies of social networking services. *Information, Communication & Society* 23, 1 (2020), 128–147.

[29] OpenAI. 2021. *GPT-3 powers the next generation of apps.* https://openai.com/blog/gpt-3-apps

[30] Lianglu Pan, Shaanan Cohney, Toby Murray, and Van-Thuan Pham. 2023. EDE-Fuzz: A Web API Fuzzer for Excessive Data Exposures. In *2024 IEEE/ACM 46th International Conference on Software Engineering (ICSE)*. IEEE Computer Society, 519–530.

[31] Sai Teja Peddinti, Igor Bilogrevic, Nina Taft, Martin Pelikan, Úlfar Erlingsson, Pauline Anthonysamy, and Giles Hogben. 2019. Reducing permission requests in mobile apps. In *Proceedings of the internet measurement conference (IMC)*. 259–266.

[32] Mayk Caldas Ramos, Shane S Michtavy, Marc D Porosoff, and Andrew D White. 2023. Bayesian Optimization of Catalysts With In-context Learning. (2023).

[33] scikit-learn. 2023. *sklearn.ensemble.IsolationForest.* Retrieved July 26, 2023 from https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html

[34] Rocky Slavin, Xiaoyin Wang, Mitra Bokaei Hosseini, James Hester, Ram Krishnan, Jaspreet Bhatia, Travis D. Breaux, and Jianwei Niu. 2016. Toward a Framework for Detecting Privacy Policy Violations in Android Application Code. In *2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE)*. 25–36.

[35] Bergur Thormundsson. 2023. *Virtual Assistant Technology - statistics & facts.* Retrieved July 27, 2023 from https://www.statista.com/topics/5572/virtual-assistants

[36] Xiaoyin Wang, Xue Qin, Mitra Bokaei Hosseini, Rocky Slavin, Travis D Breaux, and Jianwei Niu. 2018. Guileak: Tracing privacy policy claims on user input data for android applications. In *Proceedings of the 40th International Conference on Software Engineering (ICSE)*. 37–47.

[37] Fuman Xie, Yanjun Zhang, Hanlin Wei, and Guangdong Bai. 2022. UQ-AAS21: a comprehensive dataset of Amazon Alexa skills. In *International Conference on Advanced Data Mining and Applications (ADMA)*. Springer, 159–173.

[38] Fuman Xie, Yanjun Zhang, Chuan Yan, Suwan Li, Lei Bu, Kai Chen, Zi Huang, and Guangdong Bai. 2022. Scrutinizing Privacy Policy Compliance of Virtual Personal Assistant Apps. In *37th IEEE/ACM International Conference on Automated Software Engineering (ASE)*.

[39] Chuan Yan, Fuman Xie, Mark Huasong Meng, Yanjun Zhang, and Guangdong Bai. 2024. On the Quality of Privacy Policy Documents of Virtual Personal Assistant Applications. In *24th Privacy Enhancing Technologies Symposium (PETS)*.

[40] Jeffrey Young, Song Liao, Long Cheng, Hongxin Hu, and Huixing Deng. 2022. SkillDetective: Automated Policy-Violation Detection of Voice Assistant Applications in the Wild. In *31st USENIX Security Symposium (USENIX Security)*.

[41] Chengpeng Zhang, Haoyu Wang, Ran Wang, Yao Guo, and Guoai Xu. 2018. Rechecking App Behavior against App Description in the Context of Third-party Libraries. In *International Conference on Software Engineering and Knowledge Engineering*.

[42] Nan Zhang, Xianghang Mi, Xuan Feng, XiaoFeng Wang, Yuan Tian, and Feng Qian. 2019. 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)*. IEEE, 1381–1396.

[43] Lu Zhou, Chengyongxiao Wei, Tong Zhu, Guoxing Chen, Xiaokuan Zhang, Suguo Du, Hui Cao, and Haojin Zhu. 2023. POLICYCOMP: Counterpart Comparison of Privacy Policies Uncovers Overbroad Personal Data Collection Practices. In *32nd USENIX Security Symposium (USENIX Security 23)*. 1073–1090.