Read Between the Lines: An Empirical Measurement of Sensitive Applications of Voice Personal Assistant Systems

Faysal Hossain Shezan
University of Virginia
fs5ve@virginia.edu

Hang Hu
Virginia Tech
hanghu@vt.edu

Jiamin Wang
Virginia Tech
jiamin@vt.edu

Gang Wang
University of Illinois at Urbana-Champaign
gangw@illinois.edu

Yuan Tian
University of Virginia
yt2e@virginia.edu

ABSTRACT

Voice Personal Assistant (VPA) systems such as Amazon Alexa and Google Home have been used by tens of millions of households. Recent work demonstrated proof-of-concept attacks against their voice interface to invoke unintended applications or operations. However, there is still a lack of empirical understanding of what type of third-party applications that VPA systems support, and what consequences these attacks may cause. In this paper, we perform an empirical analysis of the third-party applications of Amazon Alexa and Google Home to systematically assess the attack surfaces. A key methodology is to characterize a given application by classifying the sensitive voice commands it accepts. We develop a natural language processing tool that classifies a given voice command from two dimensions: (1) whether the voice command is designed to insert action or retrieve information; (2) whether the command is sensitive or nonsensitive. The tool combines a deep neural network and a keyword-based model, and uses Active Learning to reduce the manual labeling effort. The sensitivity classification is based on a user study (N=404) where we measure the perceived sensitivity of voice commands. A ground-truth evaluation shows that our tool achieves over 95% of accuracy for both types of classifications. We apply this tool to analyze 77,957 Amazon Alexa applications and 4,813 Google Home applications (198,199 voice commands from Amazon Alexa, 13,644 voice commands from Google Home) over two years (2018-2019). In total, we identify 19,263 sensitive "action injection" commands and 5,352 sensitive "information retrieval" commands. These commands are from 4,596 applications (5.55% out of all applications), most of which belong to the "smart home" category. While the percentage of sensitive applications is small, we show the percentage is increasing over time from 2018 to 2019.

CCS CONCEPTS

• **Security and privacy** → **Software security engineering**; • **General and reference** → *Measurement*.

WWW '20, April 20–24, 2020, Taipei, Taiwan

ACM ISBN 978-1-4503-7023-3/20/04.
https://doi.org/10.1145/3366423.3380179

KEYWORDS

Alexa, Google-Home, Skill, Voice-applications, Sensitive-commands, Sensitive-keyword, Malicious-command, Active-learning.

ACM Reference Format:
Faysal Hossain Shezan, Hang Hu, Jiamin Wang, Gang Wang, and Yuan Tian. 2020. Read Between the Lines: An Empirical Measurement of Sensitive Applications of Voice Personal Assistant Systems. In *Proceedings of The Web Conference 2020 (WWW '20), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3366423.3380179

1 INTRODUCTION

The ubiquitous usage of the Internet of Things (IoT) devices has proliferated the number of Voice Personal Assistant (VPA) systems in our home. As of Jan 2019, over 66.5 million households in the US [14] have one or more VPAs such as Amazon Alexa [7], Google Home [13], and Homepod [9]. The two dominating manufactures Amazon and Google introduce the voice assistant applications called "skills"[1]. Third-party developers have built and published more than 84,000 skills worldwide in the application markets in 2019 [4, 17]. Users can "talk" to these applications to complete various tasks including opening a smart lock, starting their car, placing shopping orders, and transferring money to a friend. Although these applications bring convenience, they also introduce new attack surfaces. Recent research shows that remote attackers can craft hidden voice commands to trigger the VPAs to launch malicious actions without user knowledge [63, 73, 76]. More recent work shows that attackers can publish malicious skills with similar pronunciations to fool the VPA to invoke the wrong application [47, 78]. Existing works have focused on proof-of-concept attacks by pointing out the potential ways of launching the attacks. However, there is a lack of empirical understanding of what functionality the third-party applications provide, and thus makes it difficult to systematically assess the consequences of these attacks.

In this paper, we perform the first large-scale measurement on the third-party applications of Amazon Alexa and Google Home to systematically assess the attack surfaces. More specifically, given a voice assistant application, we seek to characterize its risk by detecting and analyzing the sensitive voice commands that are subject to potential attacks. Based on the recent proof-of-concept attacks [47, 63, 73, 76, 78], there are two main types of attack consequences: (1) controlling the system to perform an action, and

[1]Google calls the applications as "actions". For consistency, we also call them as skills.

(2) obtaining sensitive information. As such, we develop a natural language processing tool that classifies a given voice command from two dimensions. First, we examine whether a voice command is designed to insert an action (*e.g.*, controlling a smart device) or retrieve information (*e.g.*, obtaining user bank balance). Second, we classify whether the command is sensitive or nonsensitive. These two dimensions help to provide a more comprehensive view of the voice assistant skills, and their susceptibility to the existing attacks.

Challenges. There are four key challenges to automatically analyze the functionality of VPA skills. *First*, unlike smartphone apps whose binaries (or source code) are available for analysis, voice applications are essentially web programs that are hidden behind the cloud (*e.g.*, Amazon/Google cloud). Thus, we cannot characterize the voice skills using traditional API analysis but need to design new tools to analyze its natural language interface (or voice commands). *Second*, the voice commands supported by VPA skills are very short, which provides little information to run typical Natural Language Processing tools. *Third*, there are already a large number of VPA skills in the current markets, and labeling their data (for model training) requires expensive manual efforts. *Fourth*, the perceived sensitivity of a voice command could vary from person to person, the measurement of which requires user participation.

System Design. To automatically analyze the voice commands, we design two classification models to characterize the *capability* and the *sensitivity* of the voice commands respectively.

First, regarding the voice command's capability, we train a model to classify *action injection* commands that control smart devices and services, *information retrieval* commands that retrieve information from the application. For example, "*Alexa, ask Watch-Man to open the red door*" is an injection command, while "*Alexa, ask Macys where is the nearest store to me*" is a retrieval command. Our model is based on a Convolutional Neural Network (CNN) [44]. To overcome the short length of each command, we append the skill category information to provide contexts. In addition, we design an active learning-based workflow so that we can minimize manual labeling efforts to train an accurate model. The ground-truth evaluation shows that our model achieves an accuracy of 95%.

Second, regarding the voice command's sensitivity, we build a model to classify *sensitive* commands from *nonsensitive* ones. For example, "*Alexa, unlock my front door*" is a sensitive command while "*Alexa, play Harry Potter Quiz*" is nonsensitive. The challenge is that sensitivity classification is rather subjective, and conventional user studies have limited scalability. As such, we use automated algorithms for sensitive keyword extraction and then perform a user study (N=404) for keyword pruning. Instead of using complex machine learning models (whose results are difficult to interpret during post-analysis), we use a keyword-based model that achieves an accuracy of 95.6% in finding the sensitive voice commands.

Measurement Results. We apply this tool to analyze 77,957 Amazon Alexa skills and 4,813 Google Home skills over two years (2018-2019). We identify 19,263 sensitive "action injection" commands and 5,352 sensitive "information retrieval" commands. We find these sensitive voice commands are from a small set of 4,596 skills (4,203 Alexa skills and 393 Google Home skills), which only take 5.55% of all the available skills. Note that there are some duplicated skills and voice commands for 2018 and 2019. After removing the duplicates (6,058 sensitive commands and 1,216 sensitive skills) between Alexa 2018 & Alexa 2019, and duplicates (40 sensitive commands and 165 sensitive skills) between Google 2018 & Google 2019, we identify 18,517 unique sensitive voice commands (16,844 from Amazon Alexa and 1,673 from Google), and 3,215 unique sensitive skills (2,987 from Amazon Alexa and 228 from Google). 90.46% of these sensitive commands are from skills that are used to communicate with smart-home devices. Surprisingly, categories that are traditionally perceived to be sensitive (*e.g.*, "Health" and "Kid") rarely have sensitive voice commands and skills. Even the "Shopping" category only contributed 146 sensitive commands across the two platforms. We also find that the sensitive voice commands are highly concentrated on a few sets of "actions". The top 30 sensitive keywords effectively cover 98.7% and 99.3% sensitive commands in Amazon Alexa and Google Home respectively. Overall, the results show that despite a large number of available skills (over 82,770), only a small portion of skills are for security and privacy-sensitive tasks that deserve further attention from researchers for security analysis. However, the number of sensitive skills and voice commands increase from 2018 to 2019 (907 new sensitive skills, and 6,088 new sensitive commands).

Summary of Contributions. Our key contributions are:

- *First*, we perform the first large-scale empirical measurement on two dominating Voice Personal Assistant application markets, covering 82,770 skills and 211,843 voice commands.
- *Second*, our results provide new understandings of the *capability* and the *sensitivity* of the third-party applications. We identify a small set of sensitive applications (5.55%) that contributed to the vast majority of sensitive voice commands.
- *Third*, we design and implement automated tools to classify VPA skills and their voice commands [18].
- *Fourth*, we perform a user survey with 400+ participants to measure the perceived sensitivity of voice commands.

2 BACKGROUND OF VPA

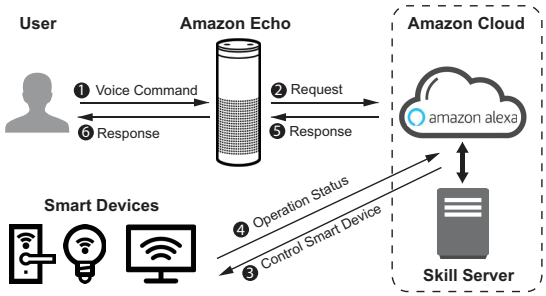


Figure 1: System model of how Amazon Alexa process a voice command.

Voice Personal Assistant (VPA) is a software agent, which provides aids for individuals, like setting calendar events, making reservations, controlling smart home devices. Most VPAs such as Amazon Alexa and Google Home use a cloud-based model to host skills and interact with users. The workflow is shown in Figure 1. After the user talks to the VPA device (❶), the voice command is first sent to the Amazon/Google Cloud (❷). The cloud needs to translate the natural language command into an API call, and then sends the

API call to the corresponding skill servers). To develop a skill, third-party developers can either host their skill servers directly within the cloud, or they can run the server independently. Regardless of which way the developers choose, they typically rely on the cloud to parse and interpret the voice command and route the API calls. The natural language process system in the cloud is the main target (or weak link) to launch attacks. We will introduce specific threat models in the next section. Some of the skills are used to control other smart home devices. In this case, either the Amazon/Google cloud or the skill server will send the request to the smart devices to perform tasks or configure the settings (❸–❹). After processing, the operation status or response will be sent back to users (❺–❻).

Some of the skills require the user to create an account (*e.g.*, shopping services, banks, smart device services). Both Amazon Alexa and Google Home use OAuth to link the user's skill-specific account to the user's voice assistant device so that users can interact with the skill service through the VPA system. This mechanism is called *account linking*. Many simple skills such as games, quizzes, and question-answering typically do not require account linking.

Google Home and Amazon Alexa maintain their "app stores" where each application (skill) has its own web page. For each skill, the web page shows the voice commands associated with that particular skill. Note that Amazon Alexa only allows up to three voice commands listed as example commands (five commands for Google Home skills). As such, developers would list the most important commands in the application descriptions in the introduction page.

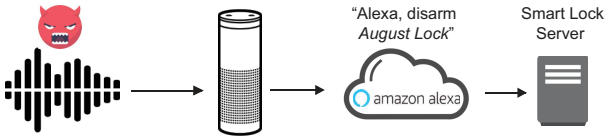
3 PROBLEM DEFINITION AND GOALS

In this section, we present the threat model and our research questions regarding the attack surfaces of VPA systems.

3.1 Threat Model

Researchers show different proof-of-concept attacks that can exploit the VPA ecosystem. Recent papers demonstrate that remote attackers can send malicious voice commands to control the devices stealthily [73]. Attackers can also create malicious skills whose names have similar pronunciations with those of popular skills, as a way to trick users to invoke malicious skills without their knowledge [47, 78]. Recent incidents [1, 2] also show that background noise or TV commercials can trigger unwanted actions in the VPA system. Despite the proof-of-concept, it is not yet clear what consequences these attacks can cause. The reason is that we still lack the understanding of what existing skills are capable of doing, and how sensitive their tasks are. As such, in this paper, we seek to measure the capability and sensitivity of the voice commands and the skills. In our threat model, the VPAs are trusted. We focus on two types of attacks from external attackers:

Hidden Voice Command Attack. Remote attackers can send malicious voice commands to trigger the VPAs for malicious actions. The voice commands can be sent through compromised local speakers or embedded in TV commercials and popular music. By utilizing the feature that humans cannot hear the high-frequency sound, an attacker can trigger the malicious voice commands without the user's notice [63, 76]. As illustrated in Figure 2a using inaudible voice command, an attacker can trigger malicious events in smart home devices such as unlock the smart door lock. Moreover, an attacker can record sensitive information (such as account balance, PIN code) if they also compromised microphones [5] in the home.



(a) Hidden Voice Command Attack: Unauthorized user (*e.g.*, high-frequency sounds), unauthorized members of the house (*e.g.*, kids), or even background music that accidentally triggered unintended actions (*e.g.*, unlock the door).

(b) Skill Squatting Attack: Attacker registering a malicious skill whose name sounds like that of the "PayPal" skill. The cloud could misinterpret the user voice command to invoke the malicious skill.

Figure 2: Two types of attack against the VPA system.

Skill Squatting Attack. Attackers can develop and publish a malicious skill to collect sensitive user information. The key idea of skill squatting is to register the skill with a name that sounds similar to the target skill. As is shown in Figure 2b, the Cloud may misinterpret the voice commands and invoke the malicious skills instead of the legitimate skills since their names sound similar. Then attackers can collect sensitive information (such as PIN code, PII) as users think they are interacting with the legitimate PayPal skill.

3.2 Analysis Goals

Given a skill and its voice commands, we seek to understand the *capability* and *sensitivity* of each voice command, to better understand the potential consequences caused by the above attacks. Note that this analysis is different from analyzing the data collection controlled by permissions. Alexa and Google Home also have permissions similar to smartphones, but these permissions are limited, and only protect information from Amazon or Google account (*e.g.*, user's zipcode). We focus on a different and more common way of data collection, where skills get the information from users directly via voice interfaces, instead of via Amazon or Google's account.

Capability Measurement. We classify the voice command's capability based on whether it is used to *inject actions*, or *retrieve information*. On one hand, an action injection command can be directly exploited by hidden voice attacks to insert an action (*e.g.*, unlocking doors, placing shopping orders) without user knowledge. On the other hand, an information retrieval voice command can be used for collecting sensitive information or return users' wrong information (*e.g.*, fake news). For commands that get information from the skill, the skill squatting attacker can pretend to be a benign skill and give fake information. Since users are interacting with the malicious skill (without knowing the skill is the wrong one), they would trust the fake information from the malicious skill. Besides,

an attacker can launch the hidden voice command attack to invoke a benign skill and send an information retrieval command secretly to the VPA, for example, "what is my PIN code?", When the VPA reply with sensitive information, the attacker can record the information by a compromised speaker.

Sensitivity Measurement. We seek to investigate whether the voice command is sensitive. Regardless of the capability (action injection or information retrieval), certain voice commands do not carry real risks, especially for simple skills without account linking, *e.g.*, games and quizzes. Our goal is to differentiate the sensitive commands (*e.g.*, unlock the front door) with the nonsensitive ones (*e.g.*, tell me some jokes) by considering user perceptions. In this work, we identify those voice commands as *sensitive* if exploited they will bring damage to the user by either leaking private information (*e.g.*, bank balance, inbox message) or violating security (*e.g.*, unlock the front door, stop recording camera) via hacking IoT devices. In contrast, according to our definition, *nonsensitive* voice commands do not pose security or privacy threat if exploited. Voice commands that give general information (*e.g.*, restaurant information, weather information) or use to operate third party applications that have no security implication generally fall into nonsensitive class (*e.g.*, tell a joke, play rock music).

3.3 Motivating Examples

Skill Name	Command Line	Type
Blink SmartHome	*"Stop the camera"*	Injection
	"Show me the last activity from front door"	Retrieval
Schlage Sense	*"Lock the door"*	Injection
FordPass	*"Ask FordPass to start my car"*	Injection
	"Tell FordPass to list all cars on the account"	Retrieval

Table 1: Example skills and their sensitive voice commands.

Table 1 lists three example skills and their sensitive commands. `Blink` is a home security skill that controls the cameras and alarms. For example, the user can change the home security mode by saying an injection command "Alexa, ask Blink to arm/disarm my home system", and check the camera feeds by a retrieval command "Alexa, show me the last activity from the front door". The underlying danger is that the security mode can be changed by attackers and it might release the user's recorded video information. `Schlage Sense` controls the smart locks on the doors. The user can use this skill to lock/unlock the door by the injection command "Alexa, lock/unlock the front door" and check the door status by the retrieval command "Alexa, is the front door locked?". The possible threat is that this skill gives incorrect door status to the user, leaving the user's home in a dangerous situation. `FordPass` is a skill to control network-connected cars. Users can control the car by injection commands "Alexa, ask FordPass to start/stop my car", and obtain vehicle information by the retrieval command "Alexa, ask FordPass my tire pressure". Our goal is to identify and characterize these sensitive voice commands that are likely subject to attacks. More examples of sensitive and nonsensitive voice commands are shared via the following link [19].

4 EXPERIMENT DESIGN

In the following section, we will describe the details of our data collection, system design, and experiment methodology.

	Alexa US 2019	Alexa UK 2019	Alexa US 2018	Google 2019	Google 2018
Skill	31,413	20,213	26,331	3,148	1,665
Command	80,129	51,922	66,148	9,096	4,548

Table 2: Our dataset.

4.1 Data Collection

For our analysis, we collected data for existing applications in the Alexa store [6] and Google Home store [12]. The goal is to cover a large number of applications with a priority for the popular ones. First, given an app store, we started from the homepage to identify all the application categories (23 categories in the Alexa store, and 18 categories in the Google Home store). Then, we crawled all the indexed skills and their introduction page under each category. Each skill's introduction page contains the skill name, skill description, category information, developer information, user rating and reviews, privacy policies, and the supported voice commands.

We crawled five datasets during 2018 and 2019 (Table 2). More specifically, we crawled the Alexa US store and Google Home store in June 2018, and later again in May 2019. Since Alexa has region-based skill markets, to compare the difference of skills in different regions, in August 2019, we also crawled a snapshot of Alexa UK (United Kingdom) store for comparison purposes (Google Home only has one universal app store). Note that even for the same store, the different snapshots do not necessarily contain the same set of skills. For example, comparing Alexa US 2019 and Alexa US 2018, there were only 18,309 (58.28%) skills in both snapshots as new skills are entering the stores while old skills disappearing. Even for skills in both snapshots, we found developers may *update* the skill description and example voice commands. In total, there are 38,093 (47.5%) overlapping voice commands in 2018 and 2019 Alexa US data. Moreover, we observed 1,060 skills and 1,441 voice commands from Google 2018 also appeared in Google 2019 data.

In total, there were 82,770 skills (77,957 from Amazon Alexa and 4,813 from Google Home) in the five datasets. By referencing the publicly reported statistics on Amazon Alexa and Google Home stores [3, 65], we believe our datasets are rather complete. In the following, for the ease of presentation, we will mainly use the datasets in the US store 2019 to present our findings. To show the evolution of the skills from 2018-2019 and the skills in different regions, we will present these results in the Section 5.3 evolution analysis, and Section 5.3 region-based analysis.

Extracting Voice Commands. As previously mentioned, each skill page displays several example voice commands, which are easy to extract. However, Amazon Alexa only allows showing up to three voice commands on the page [6] and Google Home allows showing up to five voice commands [16]. Due to this limit, we find that developers often include additional voice commands in the skill description as a *list* or in a *double quote*. To extract voice commands from the skill description, we follow the steps below.

(1) We convert the text to the lowercase format, convert Unicode objects to ASCII strings, and remove special characters.
(2) We divide the description into different chunks, based on a line break, newline, and double-quote ("").
(3) If the text chunk starts with the *wake word* (*i.e.*, "alexa," "google,"), then we mark that text chunk as a voice command.

Table 2 shows the number of voice commands extracted from each dataset (combining example commands and commands in the description). In total, we extracted 211,843 voice commands. While we cannot guarantee the voice command extraction is exhaustive, we argue that developers are motivated to put the most essential commands on the skill page to showcase the skill's functionality and teach users how to use the skill. Later in Section 4.6, we will run a dynamic interaction experiment with skills, and show that our voice command extraction is already rather complete.

4.2 Data Pre-processing

Before building the classification models, we first pre-process the voice command datasets to produce a clear format. We use the Amazon Alexa US 2019 and Google Home 2019 as the primary datasets to explain the following steps.

1. Removing voice commands used for enabling skills. These voice commands are used to turn on (or invoke) the skill on the user device (*e.g.*, open *Amex*, start *Song Quiz*). We remove them since they don't indicate the function of the skill or perform any actions. As a result, we remove 29,784 voice commands for Amazon Alexa, and 2,949 voice commands for Google Home.

2. Extracting action from a voice command. The action of a voice command refers to a user request. According to the developer guide [15, 21], voice commands must follow the general patterns (defined by Amazon and Google) as follows-

- <action> <connecting word> <invocation name>
- Ask <invocation name> <connecting word> <action>
- Ask <invocation name> < action>
- Ask <invocation name> <question beginning with a supported question word such as 'what', 'how', etc.>

Based on the above rules, we extract actions from voice commands. For example, for command "Ask *Mastermind* to text Kelly Miller", we first tag "Mastermind" as the invocation name, and then identify the connecting word "to". After that, we find the action word, which is "text Kelly Miller".

3. Data structure formatting. In this step, we remove punctuation, convert all the characters to the lowercase, and convert numeric value to the corresponding alphabetic value (*e.g.* 1 to "one", 2 to "two"). We also replace all the invocation name with a general name. For example, "ask *mastermind* to text Kelly Miller" will be converted to "ask *invk_name* to text Kelly Miller". This step is to remove potential distractions for the classification models.

4. Adding category name. Voice commands are usually too short and lack the necessary context. To mitigate it, we concatenate the skill category to the voice command to provide the context.

5. Removing redundancy. For example, "CNN Flash Briefing", "ABC News Update" and "Fox News" are three news skills who have the voice command "what's in the news?". We remove such identical voice commands to avoid biases of the trained models. Note that after the pre-processing steps above, certain previously non-identical commands become identical now. For example, "ask *Doctor Who Facts* for a fact" and "ask *Unofficial Stargate Facts* for a fact" become the same command after replacing invocation name with a common term (*invk_name*) in step-3. In total, we remove 1,141 duplicate voice commands for Amazon Alexa and 296 duplicate voice commands for Google Home.

4.3 Data Labeling

Our model training and evaluation require "ground-truth" data, and we create a small ground-truth dataset by manually labeling voice commands. Our model training is primarily based on the Alexa US 2019 data. Then, we evaluate the trained model on both Alexa US 2019 and Google Home 2019 data.

For training the *capability analysis model*, we randomly select 862 Amazon Alexa skills covering all 23 categories, and label the 1,810 voice commands as "action injection" or "information retrieval". To validate model performance, from Amazon Alexa we randomly select another 247 skills and label 475 voice commands. From Google Home, we randomly select 87 skills and label 200 commands.

Similarly, for training the *sensitivity analysis model*, we randomly select 721 skills from Alexa and label 1,652 voice commands into "sensitive" and "nonsensitive". For model validation, we select another 99 skills from Alexa and label 275 voice commands. From Google Home, we randomly select 83 skills and label 200 commands.

We have three researchers to label the voice commands. Each researcher works *independently* on the labeling tasks. If the three researchers label a command differently, we use the majority voting to resolve the conflict. For the labels on "action injection" and "information retrieval", we have very consistent labels across voice commands (agreement rate = 97.33%, Fleiss' kappa = 0.903) [34]. For the labels regarding sensitivity, the researchers have slightly bigger differences (agreement rate = 94.46%, Fleiss' kappa = 0.93). Because the sensitivity label is rather subjective, we conduct a user study to further validate and calibrate the sensitivity assessment (details are explained in Section 4.5).

4.4 Experiment 1. Capability Analysis Model

To classify if a voice command is action injection or information retrieval, we apply *active learning* [36, 53, 67, 72] to achieve our targeted accuracy with limited labeled data. Transfer learning might be another option for dealing with limited labeled data [39, 60, 61, 74], but in our case, active learning turns out to be a better solution because it is challenging to find a relevant source domain.

We use Convolutional Neural Network (CNN) for building the core machine learning model in our active learning approach. Applying CNN to natural language processing, especially text classification has been proven effective compared to the other DNN (Deep Neural Network) models [44, 51]. We build our embedding layer from word2vec [59] model using 80,129 unlabeled voice commands from Alexa US 2019 data. Our model classifies the outcome of the inputted voice command using a softmax dense layer, predicting its capability category.

Algorithm-1 shows the procedure of model training. Let us first illustrate some preliminaries and notations. $\mathcal{L} = \{x_i, y_i\}_{i=1}^{m}$ denotes the training dataset which contains m number of labeled instances, whereas $\mathcal{U} = \{x_i\}_{i=m+1}^{n}$ indicates the set of unlabeled instances. Here, $x_i \in \mathbb{R}^d$ is a d-dimensional feature vector and the class label, $y_i \in C = \{0, 1\}$, where zero and one represents action injection, and information retrieval class respectively. We consider achieving a targeted accuracy as a stopping criterion for active learning [50, 79]. For each round $\mathcal{R}$, we calculate the uncertainty value of instances

Algorithm 1 Retraining-based Active Learning algorithm for Capability Analysis Model

1: **Input:** Labeled data set $\mathcal{L}$, unlabeled data set $\mathcal{U}$, validation set, $\mathcal{V}$
2: **Output:** Trained Model, $\mathcal{M}_{\mathcal{R}}$ and Model's Accuracy, $\mathcal{A}$
3: **procedure** TrainCapabilityAnalysisModel
4: Train the classifier on $\mathcal{L}$
5: $\mathcal{R} \leftarrow$ initialize to $zero$
6: **repeat**
7: $\mathcal{R} \leftarrow \mathcal{R} + 1$
8: **for each** instance, $x_i \in \mathcal{U}$, $y_i \in C = \{0, 1\}$ **do**
9: calculate uncertainty, $\mathcal{P}_{\mathcal{L}}(y_i|x_i)$ using Equation 1
10: **end for**
11: Choose uncertainty set, $\{x_j^*\}_{j=1}^{100}$ using Equation 2
12: Construct newly labeled set, $\{x_j^*, y_j^*\}_{j=1}^{100}$
13: $\mathcal{L} \leftarrow \mathcal{L} \cup \{x_j^*, y_j^*\}_{j=1}^{100}$
14: $\mathcal{U} \leftarrow \mathcal{U} \setminus \{x_j^*\}_{j=1}^{100}$
15: Re-train the model using $\mathcal{L}$
16: Compute accuracy, $\mathcal{A}$ of the current model, $\mathcal{M}_{\mathcal{R}}$ on $\mathcal{V}$
17: **until** $\mathcal{A} > 95\%$
18: Return Trained Model, $\mathcal{M}_{\mathcal{R}}$ and Model's Accuracy, $\mathcal{A}$
19: **end procedure**

from the unlabeled dataset according to Equation 1 in Line 7-10 of Algorithm 1.

$$\mathcal{P}_{\mathcal{L}}(y_i|x_i) = |p(0|x_i) - p(1|x_i)| \quad (1)$$

In Equation 1, $\mathcal{P}_{\mathcal{L}}$ indicates the uncertainty value of an instance, and p denotes the probability of an instance being in one class.

$$\{x^*\}_{\mathcal{R}} = \underset{100}{argmin}(\mathcal{P}_{\mathcal{L}}(y_i|x_i)) \quad (2)$$

We then select those instances for which the model is mostly uncertain and label these instances manually. At each round, we select 100 most uncertain data,$\{x^*\}_{\mathcal{R}}$ according to Equation 2. Then, we remove $\{x^*\}_{\mathcal{R}}$ from $\mathcal{U}$ and label $\{x^*\}_{\mathcal{R}}$ by human to get $\{x^*, y^*\}_{\mathcal{R}}$. And thus, we get additional labeled data $\mathcal{L} \leftarrow \mathcal{L} \cup \{x^*, y^*\}_{\mathcal{R}}$ to train the model in the next round. We keep iterating this process until we reach our targeted accuracy. We compute our model's performance on a fixed validation data, $\mathcal{V} = \{x_j, y_j\}_{j=1}^{475}$. We stop this data augmentation process as soon as our model reach our expected performance. Since active learning is an iterative process, we need to decide when to stop the process. The most popular techniques for active learning stopping criteria are - achieving targeted accuracy, number of total iterations, gradient-based stopping, confidence-based stopping, performance degrading point, achieving stable performance [50, 79]. We set the targeted accuracy as 95% to be the stopping criteria.

4.5 Experiment 2. Sensitivity Analysis Model

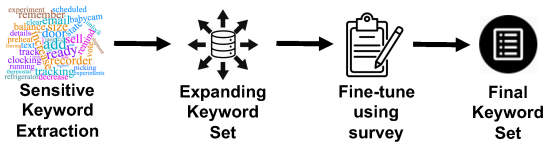


Figure 3: System Overview for the keyword-based approach of finding sensitive and nonsensitive voice commands.

We observe that it is intuitive to estimate the sensitivity of voice commands of account linking required skills based on the presence of certain keywords. Because such skills usually have more sensitive functionality (e.g., unlock the door, stop recording camera, etc.) compared to the non-account linking skills. With this observation, we design the keyword-based solution to identify sensitive voice commands. Figure 3 illustrates our complete keyword-based search technique. We can divide our whole keyword-based search technique into three different phases- (1) Sensitive keyword extraction, (2) Expanding sensitive keyword set, and (3) Fine-tune keyword set using user survey. The first phase is used for extracting sensitive keyword set from voice commands while the second phase is used for increasing that sensitive keyword set. As sensitiveness is subjective, our third phrase includes an online survey to collect feedback on the keywords that are selected by our tool. Note that, the online survey is approved by IRB (Institutional Review Board). Finally, we get the ultimate sensitive keyword set that we use for evaluating the sensitiveness of a voice command.

Sensitive Keyword Extraction. Initially, we have three sensitive keyword sets A_1, A_2, A_3 (identified by three annotators while labeling sensitive voice commands). After aggregating all the sensitive keywords, we have the initial keyword set, $K_I = A_1 \cap A_2 \cap A_3$. Then, we use RAKE (Rapid Automatic Keyword Extraction) [62] for extracting sensitive keywords, R_S from sensitive voice commands, S and nonsensitive keywords, R_N from nonsensitive voice commands, N. We remove those sensitive keywords from R_S that are also present in R_N. And thus, we compute keywords, $R'(= R_S - R_N)$ unique to sensitive voice commands. Finally, we get the initial sensitive keyword set (which contains 68 sensitive keywords).

Sensitive keyword set expansion. To decrease the false-negative ratio, we expand our sensitive keyword set by using word2vec [59]. To train the word2vec model, we use 80,129 unlabeled voice commands from Alexa US 2019 data to get expanded keyword set, R'_E. These keywords are semantically similar to our sensitive keywords, listed in R'. In this way, we get 38 new sensitive keywords. As a result, the size of our total keywords, $R''(= R' \cup R'_E)$ become 106.

Fine-tune keyword set using survey. We launch an online survey to get a fine-tuned keyword set. From the survey, we get our final keyword set that contains 57 sensitive keywords.

Survey Design. We launch our survey through Mechanical Turk [8] to collect data from a larger population. We collect three different categories of information: how often users use VPA, users' opinions on sensitive keywords, and users' demographic information.

From the first two steps of the keyword-based approach, we get 106 sensitive keywords. Listing this large set of keywords in the survey might introduce boredom to the participants. That's why we divide the keyword list into two subsets before running the survey. First, to find out whether user uses voice assistant or not, we begin with simple questions; e.g. do users' ever use any voice personal assistant (e.g., Google Home, Amazon Alexa) or, do they use voice assistants (e.g, SIRI, Cortana, Google Assistant) in their mobile devices, and also how long they have been using it. Then, we show them a list of voice commands highlighting the *functional keywords* which represent the functionality of the voice commands. For example, for the following command– 'Alexa, tell Virtual Keypad to "arm" my system away', we highlight functional keyword "arm". Next, we ask users to classify functional keywords into five different scales of sensitiveness, including - not sensitive, less sensitive, neutral, sensitive, and most sensitive. We also include

the "other" option for each of the voice commands where the user can list different functional keywords if she feels the highlighted one does not represent the functionality of the voice command. Finally, we collect users' demographic information, such as- age, gender, education level, and occupation. We have uploaded all of our survey questions via these links [22, 23].

$$\varphi(x) = \begin{cases} 1 & \text{if (\#most sens. + \#sens.) > (\#not sens. + \#less sens. + \#neutral),} \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

Survey Results and Analysis. In our survey, we also included an attention check question to identify invalid responses. After removing 221 invalid responses, we collected a total of 404 valid responses (199 for part-I and 205 for part-II). In the following, we focus on understanding users' responses to our sensitive keyword list. Participants can classify a keyword into five different scales (not sensitive, less sensitive and neutral are counted as a nonsensitive vote, whereas sensitive and most sensitive are counted as a sensitive vote). According to the Equation 3, we decide whether a keyword belongs to a sensitive class or nonsensitive class. If $\varphi(x)$ is equal to one for a particular keyword, then it is considered as a sensitive keyword, otherwise, that is categorized as a nonsensitive keyword. After this process, the size of our final keyword list becomes 57. Our survey results show that most participants are likely to agree with the sensitive keywords we generate in the first two stages (sensitive votes and most sensitive votes add up to 53.8%). We have uploaded all the sensitive keywords to this link [11].

4.6 Sanity Check for Undocumented Voice Commands

In this section, we conduct an extra step to uncover "undocumented" voice commands supported by skills. By "undocumented" we refer to those voice commands that are not listed in skill's recommended voice command list and description. We suspect that malicious developers may hide sensitive voice commands by not revealing to the user. In the future, by triggering those commands through a hidden voice command attack (Figure 2a), the attacker can steal user information or execute sensitive operations.

4.6.1 Tool for interacting with skill. Currently, we only focus on uncovering the undocumented voice commands for Amazon Alexa (US 2019) because it has 24 million more users than Google Home [14].

Challenges. An intuitive method is to use a speaker to play commands to an Alexa device and use a microphone to record audio responses from the Alexa device. However, this is time-consuming and requires a quiet room for better recognition of commands.

Virtual Alexa Client. Therefore, we introduce our lightweight testing tool that does not require any physical set up. The testing tool has two modules: a virtual Alexa client (which works the same as physical Alexa device) and a speech-to-text module. Alexa provides a testing interface [20] to help developers to test their skills and other skills on the market with text input. Our virtual Alexa client utilizes this feature to test skills with text input instead of voice input. After receiving the input, the virtual client returns a series of responses either in text format or in audio format. We use the Azure speech-to-text service [10] to transcribe audio responses.

4.6.2 Seed sensitive command for testing. Next, with the automated interaction tool, we test if skills support sensitive commands.

Categories and Commands. For doing the sanity check, we use our keyword-based model to identify sensitive voice commands. Unfortunately, 2,782 number of skills require account linking, and 1,657 of them are from the *smart home* category. As a result, we are unable to do the sanity check in this critical category due to the lacking of legitimate information. We select top 154 sensitive voice commands (based on the number of occurrences) from the following categories- shopping, communication, productivity, business & finance, and health & fitness. Because undocumented voice commands from these categories can damage a lot compared to others. We randomly choose 50 skills from those five categories, and investigate potential undocumented voice commands (250 skills in total). We run our experiment by taking each of the voice commands from each category, and test it with all the skills in that category.

5 MEASUREMENTS AND EVALUATIONS

We evaluate the performance of our method and find it effective for both the capability analysis (F1-score is 96.33%) and sensitivity analysis (F1-score is 91.78). We also run a systematic measurement of voice commands, including perspectives such as cross-platform analysis, category analysis, evolution analysis, and region-based analysis. With all the data we analyzed, we have found 5.55% (4,596 out of 82,770) skills are sensitive, and 11.62% (24,615 out of 211,843) voice commands are sensitive. We show the details of our evaluation and measurement results below. For ease of presentation, we report the results on US store data in 2019, and compare the results with 2018 data, and UK data in Section 5.3 and Section 5.3 respectively.

5.1 Capability Analysis Model

We achieved an accuracy of 95% for the capability analysis of commands with the active learning approach described in Section 4.4.

Capability	Action Injection	647 (28.32%)	2,285
	Information Retrieval	1,638 (71.68%)	
Sensitivity	Sensitive	515 (26.72%)	1,927
	Nonsensitive	1,412 (73.27%)	

Table 3: Ground-truth data of Amazon Alexa US 2019 for Capability and Sensitivity analysis model.

As described in Section 4.3, three annotators label 2,285 voice commands as action injection and information retrieval in Amazon Alexa. A complete overview of the labeled data for the capability analysis model is shown in Table 3. From the labeled data, we randomly select 247 skills (which include 475 voice commands) as the validation dataset, and use the rest of the 862 skills (1,810 voice commands, 501 as action injection, and 1309 as information retrieval) to train the model. We use the validation set for each round of the active learning process to evaluate our model's performance. Note that the validation set never overlapped with any of the training data. We reach our optimal point at round ten. And thus, we stop our active learning procedure at that point. Finally, we get our improved model $\mathcal{M}$ that has an accuracy of 95.16% (94.68% precision, 98.04% recall, and 96.33% F1-score) over the validation data.

To investigate if our model is biased towards this validation set, we extend our experiments to run more evaluations. We randomly

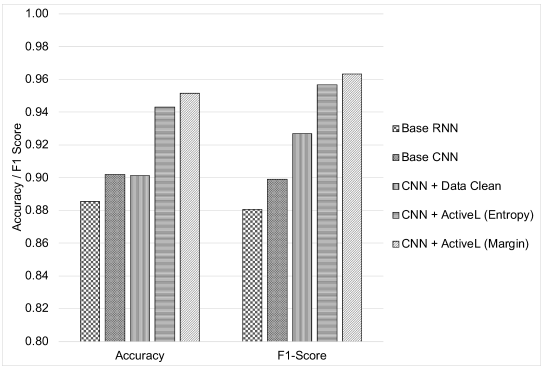


Figure 4: We compare our active learning using margin sampling model's performance with four different baseline approaches including-(1) Base RNN; where we use RNN network structure for building the machine learning model, (2) Base CNN; where we use CNN network structure, (3) CNN + data clean; where before training the model, we process the input data according to Section 4.2, (4) CNN + ActiveL (Entropy); where we use entropy metric to select unlabeled data to be labeled in each round of active learning approach, (5) CNN + ActiveL (Margin) is our proposed method; where we select the most uncertain unlabeled data to be labeled in each round of active learning approach (sorted based on F1-score).

pick 300 voice commands from the unlabeled dataset. Then, we make predictions using our model, $\mathcal{M}$ on those data. Finally, we verify those predictions by human, and find that our model's accuracy was 94.65% which is close to the validation accuracy.

We also evaluate our model's performance on Google data. To validate the performance, we label 200 voice commands from 87 voice applications (49 are action injection, 151 are information retrieval) from Google Home. Our active learning model achieves 95.99% Accuracy (whereas, 99.32% Precision, 95.38% Recall, and 97.31% F1-score) while running capability analysis on Google data. We compare our model's performance with four baselines. In Figure 4, we present these baseline models' performances along with our active learning using the margin sampling technique. The fifth bar is our proposed scheme, which outperforms all the baselines.

5.2 Sensitivity Analysis Model

Based on our keyword-based strategy (described in Section 4.5), we can classify the sensitivity of commands with an accuracy of 95.6%.

For sensitivity analysis, we label 1,927 data (247 Alexa skills) from different categories as illustrated in Table 3. Among them, we randomly select 1,652 data (sensitive data: 508, nonsensitive data: 1,144) for building the sensitive keyword set. And the rest of the data (e.g., 275 voice commands from 99 skill) are chosen for evaluating the model's performance. This validation set has never been considered while building the keyword list.

The performance of our keyword-based approach depends on correctly identifying the sensitive keywords. One major part of our keyword-based approach is expanding the sensitive keyword set using word2vec. While finding similar sensitive keywords, we set the cosine similarity value to 0.8 because of two reasons. First, by varying cosine similarity value from 0.5 to 0.95, we find that 0.8 works the best. Second, we perform a case study on the keyword pairs with different cosine similarity. For 0.8, we find several pairs such as (arm, disarm), (arm, activate), (increase, decrease), (dim, brighten). In a single tuple, the first word is the source word

Plat.		Sensitive		Nonsensitive	
		Inject.	Retriv.	Inject.	Retriv.
Alexa	US 2019	8,503(10.61%)	2,939(3.67%)	20,559(25.66%)	48,128(60.06%)
	UK 2019	3,397(6.54%)	757(1.46%)	13,538(26.07%)	34,230(65.93%)
	US 2018	6,126(9.26%)	1,180(1.78%)	17,695(26.75%)	41,147(62.2%)
Googl.	2019	671(7.38%)	341(3.74%)	3,018(33.18%)	5,066(55.7%)
	2018	566(12.45%)	135(2.97%)	1,327(29.18%)	2,520(55.41%)

Table 4: Overview of the total number of action injection-sensitive, action injection-non sensitive, information retrieval-sensitive, information retrieval-non sensitive voice commands in Amazon Alexa and Google Home.

while the second one is a similar word found by word2vec. We can observe that the similar keywords also represents the sensitive characteristics (according to Section 3.2), e.g.- *disarm* the alarm, *activate* the climate control. However, if we lower the threshold and set it to 0.75, it would give us similar keyword pairs, such as-(decrease, seventy-five), (dim, percent), which no longer represent the sensitive characteristics. Finally, using this keyword-based approach, we get an accuracy of 95.6%, precision of 95.71%, recall of 88.16% and F1-score of 91.78% over the Alexa validation dataset.

To evaluate our model's performance on Google data, we label 200 voice commands from 83 applications (57 are sensitive, 143 are nonsensitive). Our model achieves 96.5% Accuracy (whereas, 93.44% Precision, 95% Recall, and 94.21% F1-score) on Google data.

5.3 Measuring Security Implications of Skills.

Now, we put together the results from the capability and sensitivity analysis to identify the attack surface in the skills. We want to answer the following questions: (1) How many voice commands are sensitive in the current US store (2019)? (2) What kind of security impact it has? (3) Currently, which categories of skills are more sensitive? (4) Do Amazon Alexa and Google Home now perform differently in the security analysis? (5) In the US, how rapidly sensitive voice commands are increasing both in Amazon Alexa and Google Home compare to last year? (6) Do the sensitive skills in one region also appear in the other region?

Sensitive voice commands. To answer the first two questions, we combine our capability analysis (Section 4.4) and sensitivity analysis (Section 4.5). Through the capability analysis, in the 2019 US store, we find 29,062 action injection and 51,067 information retrieval voice commands from Amazon Alexa, and 3,689 action injection and 5,407 information retrieval voice commands from Google Home. Through our sensitivity analysis, we find that certain skills collect user personal information, banking information, and operate a smart device, smart car. We identified 12,454 security-sensitive voice commands in total (11,442 from Amazon Alexa and 1,012 from Google Home). Putting these analyses together, we show the detailed distributions of sensitive-injection and sensitive-retrieval voice commands of Amazon Alexa US 2019 and Google Home 2019 in Table 5 & 6 respectively.

Critical categories. We find that several categories contain security-sensitive voice commands than others. From Table 5 and 6, we can observe that *smart home* from Amazon Alexa, and *home control* category from Google Home are the most critical categories respectively. One counter-intuitive result is that several categories (*e.g.*, health & fitness, kids) that seem to be sensitive include less

Category	Sensitive		Nonsensitive	
	Inject.	Retriv.	Inject.	Retriv.
S. Home	8,095(75.4%)	2,218(20.66%)	157(1.46%)	265(2.47%)
C. Car	118(29.5%)	207(51.75%)	13(3.25%)	62(15.5%)
Product.	84(2.42%)	64(1.84%)	990(28.51%)	2,335(67.23%)
Lifestyle	45(0.96%)	60(1.28%)	1,648(35.03%)	2,952(62.74%)
Busin. & Fin.	35(0.64%)	146(2.68%)	1,422(26.12%)	3,842(70.56%)
Shopping	29(5%)	25(4.31%)	152(26.2%)	374(64.48%)
Home Serv.	18(4.1%)	15(3.4%)	164(37.19%)	244(55.33%)
Food & Drink	14(0.59%)	43(1.8%)	562(23.52%)	1,770(74.09%)
Music & Aud.	11(0.12%)	4(0.04%)	4,527(48.8%)	4,735(51.04%)
Games & Triv.	10(0.17%)	7(0.12%)	391(6.78%)	5,359(92.93%)
Utilities	9(2.36%)	37(9.71%)	64(16.8%)	271(71.13%)
Edu. & Ref.	8(0.12%)	9(0.14%)	1,948(29.37%)	4,667(70.37%)
Comm.	7(0.64%)	52(4.73%)	439(39.95%)	601(54.69%)
Trvl & Trans.	6(0.33%)	10(0.56%)	601(33.54%)	1,175(65.57%)
Health & Fit.	6(0.44%)	19(1.4%)	432(31.83%)	900(66.32%)
Social	4(0.46%)	7(0.81%)	227(26.33%)	624(72.39%)
Local	3(0.16%)	7(0.38%)	448(24.36%)	1,381(75.1%)
Sports	1(0.04%)	2(0.08%)	703(26.81%)	1,916(73.07%)
Movies & TV	-	3(0.28%)	303(28%)	776(71.72%)
Nvlty & Hum.	-	2(0.04%)	1,063(20.31%)	4,169(79.65%)
Weather	-	2(0.12%)	491(29.19%)	1,189(70.69%)
News	-	-	3,222(30.21%)	7,443(69.79%)
Kids	-	-	592(35.45%)	1,078(64.55%)

Table 5: We use our two models- active learning model & keyword-based model, to identify the total number of action injection-sensitive, action injection-nonsensitive, information retrieval-sensitive, information retrieval-nonsensitive voice commands in 80,129 voice commands from twenty-three different categories of Amazon Alexa US 2019. Inject. means action injection. Retriv. means information retrieval.

Category	Sensitive		Nonsensitive	
	Inject.	Retriv.	Inject.	Retriv.
Home Control	642(24.51%)	281(10.73%)	1,092(41.7%)	604(23.06%)
Productivity	10(3.45%)	14(4.83%)	119(41.03%)	147(50.69%)
Shopping	6(1.22%)	20(4.07%)	135(27.44%)	331(67.28%)
Health & Fit.	4(0.94%)	-	163(38.44%)	257(60.61%)
Communication	4(1.67%)	-	64(26.67%)	172(71.67%)
Movies & TV	3(1.29%)	-	99(42.67%)	130(56.03%)
Trvl & Trans.	1(0.23%)	6(1.4%)	96(22.33%)	327(76.05%)
Food & Drink	1(0.43%)	3(1.29%)	66(28.33%)	163(69.96%)
Busin. & Fin.	-	11(3.01%)	29(7.95%)	325(89.04%)
Edu. & Ref.	-	3(0.4%)	104(13.87%)	643(85.73%)
News	-	2(0.21%)	416(44.44%)	518(55.34%)
Local	-	1(0.34%)	26(8.78%)	269(90.88%)
Music & Aud.	-	-	246(80.13%)	61(19.87%)
Games & Triv.	-	-	208(25.74%)	600(74.26%)
Sports	-	-	57(32.39%)	119(67.61%)
Weather	-	-	50(30.3%)	115(69.7%)
Art & Life.	-	-	34(13.08%)	226(86.92%)
Kids	-	-	14(19.18%)	59(80.82%)

Table 6: Analysis results for 9,096 voice commands from eighteen different categories of Google Home 2019.

sensitive commands. Intuitively, the *health & fitness* category contains all the sensitive information such as- user health information, daily fitness activity. However, despite those skills, *health & fitness* category contains lots of skills that give users exercise guidelines or provide other nonsensitive functions. Therefore, the number of nonsensitive commands are higher in this category.

(a) Comparisons between sensitive skills in Alexa US store from 2018 and 2019

(b) Comparisons between sensitive skills in Google Home from 2018 and 2019

Figure 5: Comparisons of Sensitive skills and voice commands between 2018 and 2019 in Amazon Alexa and Google Home.

Cross-platform Analysis. We have identified three interesting findings regarding the skills and voice commands in both platforms (Amazon Alexa US 2019 & Google Home 2019). First, we find that there are 42 common sensitive voice commands (such as- arm my system, open door one, etc.) in Google Home 2019 and Amazon Alexa US 2019. Moreover, we find 62 vulnerable common skills (such as - K Smart, Mysa Thermostat, Lux Products, etc.). Second, we can observe the presence of certain sensitive keywords in most of the sensitive voice commands in both platforms, *e.g.*, Set, Camera, Arm, Add, Check, Lock, Send. Third, for the same category of voice commands, the two platforms have similar percentages of sensitive voice commands in both platforms. For example, *smart home* in Amazon Alexa has around 12.87% of sensitive voice commands whereas the number is 10.14% for Google home. Analysis on the rest of the categories is included in Table 5 & Table 6.

Evolution analysis. To understand the trend of voice commands and skills, we compare data between 2019 and 2018 for both platforms (Alexa US & Google Home). We list the detailed result in Table 4. First, we find that during 2018-2019, the number of sensitive skills grow from 1,592 (1,385 in Alexa, 207 in Google) to 2,330 (2,144 in Alexa, 186 in Google), and the number of sensitive voice commands grow from 8,007 (7,306 in Alexa, 701 in Google) to 12,454 (11,442 in Alexa, 1,012 in Google). Second, we find that Amazon Alexa has many more newly added sensitive skills and voice commands in 2019. Amazon Alexa has 881 new sensitive skills, and 5,116 new sensitive voice commands, whereas Google Home only has 26 new sensitive skills, and 972 new sensitive voice commands. More importantly, the ratio of sensitive voice commands in Amazon Alexa increases from 11.04% to 14.28%, but in Google Home, this ratio decreases from 15.41% to 11.13%. Third, we noticed 8,228 skills (7,789 from Alexa, and 439 from Google) were taken down. Interestingly, only 2.05% (169 out of 8,228) of the skills are sensitive, which is lower than the percentage of the sensitive skills that remain on the market (5.08%, 1,423 out of 27,996). However, the removed skills contain slightly more sensitive voice commands on average. On average, each removed skill has 5.42 sensitive commands, while the skills on the market have 5.3.

Region-based analysis. To figure out whether voice platforms have different sensitive applications in different regions, besides the US store, we also investigate skills from the UK store in 2019. Interestingly, we have not found any separate store for Google Home other than the US store. As a result, we only compare the UK and the US store for Amazon Alexa. As is shown in Table 4, the percentage of sensitive voice skills (3.33%, 674 out of 20,213) is slightly lower than the US. We also found that similar to the US

store, the *smart home* category contains the most sensitive voice commands (3,916 sensitive ones) compared to other categories in the UK store. In addition, we found 11,548 common skills in Alexa US 2019 and Alexa UK 2019 data, and 5.6% (646 out of 11,548) are sensitive skills. Finally, we noticed that 28 sensitive skills only from the UK store did not appear in the US store, while 1,498 sensitive skills only from the US store.

5.4 Sanity check evaluation

Only 2% skills (5 out of 250) that we investigate have hidden commands. Therefore, we believe undocumented voice commands are negligible. We identify 3 skills with hidden commands in the communication category, and 2 such skills in the shopping categories, and no skills with hidden commands in the following three categories: productivity, business & finance, and health.

6 DISCUSSION

Countermeasure. We find that currently in the US store, 6.74% skills and 13.95% voice commands from Amazon Alexa and Google Home are sensitive. As a result, manufacturers (i.e., Amazon or Google) can introduce an extra layer of protection by authenticating through PIN code or voice profiling before opening sensitive skills. They do not need to impose these restrictions for all the skills. As a result, it will also reduce the overhead of the user while using the nonsensitive skills and ensures a good user experience.

Result Summary and Implications. We perform a large-scale empirical measurement of 82,770 skills and 211,843 voice commands on two popular VPAs – Alexa, and Google Home. We only identify a small portion (5.55%) of the skills that contain sensitive voice commands. Among the sensitive voice commands, there are 19,263 sensitive "action injection" voice commands for controlling smart devices and setting or updating system values. In addition, there are 5,352 sensitive "information retrieval" voice commands for collecting sensitive information about users and the network-connected devices. Across the two VPA platforms (Alexa US 2019 & Google 2019), we only find 62 common sensitive applications available on both platforms and 42 common sensitive voice commands. The results indicate that only a small portion of skills are used for security and privacy-sensitive tasks, which deserves more research.

Limitations. This paper has a few limitations that need to be further discussed. First, for the *sensitivity analysis*, we use keyword-based methods, and focus on generating keywords from skills that require *account linking*. This could lead to false positive cases for the following two reasons: (1) voice commands might include keywords that we did not analyze; (2) the sensitive keywords might be biased because they might not be representative of skills without account linking. However, based on the evaluation of the 275 validation dataset, the false positive rate is only 1.09%. In addition, we manually analyze 270 additional voice commands from 100 randomly selected skills without account linking features, we only identify four sensitive voice commands (false positive rate is 1.48%). A second limitation is related to how we label *sensitive* voice commands, given that the definition of sensitivity can be quite subjective. In this paper, we have three people to label each voice command independently, and the agreement rate is reasonably high and acceptable [34] (agreement rate = 94.46%, Fleiss's Kappa = 0.93). Also, we fine-tuned our sensitive keyword set using a user survey, which would make it well representative. Third, our user study is done through MTurk, and the samples might not be representative of the general population. However, researchers have been using MTurk in prior security & privacy research [32, 38, 55]. Moreover, researchers identified Mechanical Turk as a valid source of high-quality human subject data [46]. Given that, the consistency of reported data, demographically-different samples, we believe our study provides important insights in sensitive voice commands.

7 RELATED WORK

Attacks and Defenses in VPAs. Recent research has proved the existence of vulnerabilities in voice interface both in voice controlled system [29, 42, 45, 54, 70, 71] and smartphones [31, 41, 43]. Roy et al. demonstrated an inaudible voice command attack to hijack a user's VPA [63]. Similarly, dolphin attack [76], Cocaine noodles [69], hidden voice command attack [28] also used inaudible or adversarial voice command to attack VPA. Recently, researchers showed that malicious voice command can also be embedded into audio signal [48, 73]. Apart from these voice commands attacks, attackers are making the VPA system fool by publishing semantically similar malicious skill. Kumar et al. and Zhang et al. demonstrated an innovative way of stealing important user information by skill squatting attack [47, 78]. These NLP level attacks demonstrate serious logic errors and indicate that the speech recognition system is still not mature that makes this system more vulnerable to attack.

Existing defenses include differentiating between human and machine voice, live user detection, voice authentication to protect VPA from attackers [24, 26, 33]. VoiceGesture [77] detected the presence of the live user by extracting user-specific features in the Doppler shift. Uzun et al. used captcha for authenticating the user whenever the system receives a voice command [68]. Researchers analyzed user interaction with the voice assistants [25, 30, 35, 37, 49], efficient way of controlling IoT devices [27, 40]. However, none of them measured the prevalence of sensitive voice commands.

NLP-based Security Analysis. Previous works used NLP techniques to conduct security analysis under various situations such as mobile apps [56–58], malware [66], privacy policy [52, 64]. For example, LipFuzzer to systematically study the problem of misinterpretation of voice command in VPA systems [75]. While they focused on the pronunciation and functionality of voice commands, our study focuses on capability and sensitivity of commands.

8 CONCLUSION

In this paper, we measure how a sensitive voice command can affect the security & privacy of VPA. We design an NLP-based tool to analyze sensitive voice command for their security and privacy implications. In our study, we demonstrate the presence of 12,454 sensitive voice commands in the current US store of Amazon Alexa and Google Home, and measure the evolution and region differences.

ACKNOWLEDGMENTS

We thank anonymous reviewers for their helpful feedback. This work was supported in part by National Science Foundation (NSF) grants CNS-1920462, CNS-1850479, CNS-1750101 and CNS-1717028.

REFERENCES

[1] 2017. Amazon Alexa ordered people dollhouses after hearing its name on TV. https://www.theverge.com/2017/1/7/14200210/amazon-alexa-tech-news-anchor-order-dollhouse.

[2] 2017. Burger King faces backlash after ad stunt backfires. https://wgntv.com/2017/04/13/burger-kings-whopper-ad-stunt/.

[3] 2018. Amazon Alexa Skill Count Surpasses 30,000 in the U.S. https://voicebot.ai/2018/03/22/amazon-alexa-skill-count-surpasses-30000-u-s/.

[4] 2018. Amazon Alexa Skill Count Surpasses 80,000 worldwide. https://voicebot.ai/2018/03/22/amazon-alexa-skill-count-surpasses-30000-u-s/.

[5] 2018. Are your phone camera and microphone spying on you? https://www.theguardian.com/commentisfree/2018/apr/06/phone-camera-microphone-spying.

[6] 2019. Alexa Skill Website. https://goo.gl/PvwL3T.

[7] 2019. Amazon Alexa. https://developer.amazon.com/alexa.

[8] 2019. Amazon Mechanical Turk. https://www.mturk.com/.

[9] 2019. Apple Homepod. https://www.apple.com/homepod/.

[10] 2019. Azure Speech to Text Service. https://azure.microsoft.com/en-au/services/cognitive-services/speech-to-text/.

[11] 2019. Complete List of Sensitive Keywords. https://github.com/faysalhossain2007/Read-Between-the-Lines-An-Empirical-Measurement-of-Sensitive-Applications-of-Voice-Personal-Assista/blob/master/sensitive_keyword_survey.json.

[12] 2019. Google Assistant. https://goo.gl/iM4H4L.

[13] 2019. Google Home. https://goo.gl/F8E7NH.

[14] 2019. Google Home Added 600,000 More U.S. Users in 2018 Than Amazon Echo, But Amazon Echo Dot is Still the Most Owned Smart Speaker. https://voicebot.ai/2019/03/07/google-home-added-600000-more-u-s-users-in-2018-than-amazon-echo-but-amazon-echo-dot-is-still-the-most-owned-smart-speaker/.

[15] 2019. Guideline on developing Google Action. https://developers.google.com/actions/overview.

[16] 2019. Guideline on publishing actions in Google. https://developers.google.com/assistant/console/publish.

[17] 2019. January 2019 Voice App Totals Per Voice Assistant. https://voicebot.ai/2019/02/15/google-assistant-actions-total-4253-in-january-2019-up-2-5x-in-past-year-but-7-5-the-total-number-alexa-skills-in-u-s/.

[18] 2019. Public link of our labeled dataset. https://github.com/faysalhossain2007/Read-Between-the-Lines-An-Empirical-Measurement-of-Sensitive-Applications-of-Voice-Personal-Assista/tree/master/Labeled%20Data.

[19] 2019. Sample example of Sensitive and Non-Sensitive Voice Commands. https://github.com/faysalhossain2007/Read-Between-the-Lines-An-Empirical-Measurement-of-Sensitive-Applications-of-Voice-Personal-Assista/blob/master/example-sensitive-non-sensitve-voice-commands.pdf.

[20] 2019. Steps to test Amazon Alexa skill. https://developer.amazon.com/docs/devconsole/test-your-skill.html#ways-to-test-an-alexa-skill.

[21] 2019. Understand How Users Invoke Custom Skills. https://developer.amazon.com/docs/custom-skills/understanding-how-users-invoke-custom-skills.html#cert-invoke-specific-request.

[22] 2019. User Survey of Sensitive Keyword, Part-1. https://github.com/faysalhossain2007/Read-Between-the-Lines-An-Empirical-Measurement-of-Sensitive-Applications-of-Voice-Personal-Assista/blob/master/Survey%20of%20Voice%20Assistant%20User-part-1.pdf.

[23] 2019. User Survey of Sensitive Keyword, Part-2. https://github.com/faysalhossain2007/Read-Between-the-Lines-An-Empirical-Measurement-of-Sensitive-Applications-of-Voice-Personal-Assista/blob/master/Survey%20of%20Voice%20Assistant%20User-part-2.pdf.

[24] Amr Alanwar, Bharathan Balaji, Yuan Tian, Shuo Yang, and Mani Srivastava. 2017. EchoSafe: Sonar-based Verifiable Interaction with Intelligent Digital Agents. In *Proc. of the 1st ACM Workshop on the Internet of Safe Things*.

[25] Frank Bentley, Chris Luvogt, Max Silverman, Rushani Wirasinghe, Brooke White, and Danielle Lottrjdge. 2018. Understanding the long-term use of smart speaker assistants. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 3 (2018), 91.

[26] Logan Blue, Luis Vargas, and Patrick Traynor. 2018. Hello, Is It Me You're Looking For?: Differentiating Between Human and Electronic Speakers for Voice Interface Security. In *Proc. of the 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks*.

[27] Giovanni Campagna, Rakesh Ramesh, Silei Xu, Michael Fischer, and Monica S Lam. 2017. Almond: The architecture of an open, crowdsourced, privacy-preserving, programmable virtual assistant. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 341–350.

[28] Nicholas Carlini, Pratyush Mishra, Tavish Vaidya, Yuankai Zhang, Micah Sherr, Clay Shields, David Wagner, and Wenchao Zhou. 2016. Hidden Voice Commands. In *Proc. of USENIX Security'16*.

[29] Nicholas Carlini and David Wagner. 2018. Audio adversarial examples: Targeted attacks on speech-to-text. *arXiv preprint arXiv:1801.01944* (2018).

[30] Eugene Cho. 2019. Hey Google, Can I Ask You Something in Private?. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. ACM, 258.

[31] Wenrui Diao, Xiangyu Liu, Zhe Zhou, and Kehuan Zhang. 2014. Your voice assistant is mine: How to abuse speakers to steal information and control your phone. In *Proc. of the 4th ACM Workshop on Security and Privacy in Smartphones & Mobile Devices*.

[32] Michael Fagan and Mohammad Maifi Hasan Khan. 2016. Why do they do what they do?: A study of what motivates users to (not) follow computer security advice. In *Twelfth Symposium on Usable Privacy and Security ({SOUPS} 2016)*. 59–75.

[33] Huan Feng, Kassem Fawaz, and Kang G Shin. 2017. Continuous authentication for voice assistants. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*. ACM, 343–355.

[34] Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin* 76, 5 (1971), 378.

[35] Mikhail Fomichev, Max Maass, Lars Almon, Alejandro Molina, and Matthias Hollick. 2019. Perils of Zero-Interaction Security in the Internet of Things. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 3, 1 (2019), 10.

[36] Yoav Freund, H Sebastian Seung, Eli Shamir, and Naftali Tishby. 1997. Selective Sampling Using the Query by Committee Algorithm. *Mach. Learn.* 28, 2 (Aug. 1997), 133–168.

[37] Radhika Garg and Christopher Moreno. 2019. Understanding Motivators, Constraints, and Practices of Sharing Internet of Things. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 3, 2 (2019), 44.

[38] Hana Habib, Jessica Colnago, Vidya Gopalakrishnan, Sarah Pearman, Jeremy Thomas, Alessandro Acquisti, Nicolas Christin, and Lorrie Faith Cranor. 2018. Away from prying eyes: analyzing usage and understanding of private browsing. In *Fourteenth Symposium on Usable Privacy and Security ({SOUPS} 2018)*. 159–175.

[39] Jeremy Howard and Sebastian Ruder. 2018. Universal Language Model Fine-tuning for Text Classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 328–339.

[40] Timo Jakobi, Gunnar Stevens, Nico Castelli, Corinna Ogonowski, Florian Schaub, Nils Vindice, Dave Randall, Peter Tolmie, and Volker Wulf. 2018. Evolving Needs in IoT Control and Accountability: A Longitudinal Study on Smart Home Intelligibility. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 4 (2018), 171.

[41] Yeongjin Jang, Chengyu Song, Simon P Chung, Tielei Wang, and Wenke Lee. 2014. A11y attacks: Exploiting accessibility in operating systems. In *Proc. of CCS'14*.

[42] Artur Janicki, Federico Alegre, and Nicholas Evans. 2016. An assessment of automatic speaker verification vulnerabilities to replay spoofing attacks. *Security and Communication Networks* 9, 15 (2016), 3030–3044.

[43] Chaouki Kasmi and Jose Lopes Esteves. 2015. IEMI threats for information security: Remote command injection on modern smartphones. *IEEE Transactions on Electromagnetic Compatibility* (2015).

[44] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proc. of EMNLP'14*.

[45] Tomi Kinnunen, Md Sahidullah, Héctor Delgado, Massimiliano Todisco, Nicholas Evans, Junichi Yamagishi, and Kong Aik Lee. 2017. The asvspoof 2017 challenge: Assessing the limits of replay spoofing attack detection. (2017).

[46] Aniket Kittur, Ed H Chi, and Bongwon Suh. 2008. Crowdsourcing user studies with Mechanical Turk. In *Proceedings of the SIGCHI conference on human factors in computing systems*. ACM, 453–456.

[47] Deepak Kumar, Riccardo Paccagnella, Paul Murley, Eric Hennenfent, Joshua Mason, Adam Bates, and Michael Bailey. 2018. Skill squatting attacks on amazon alexa. In *Proc. of USENIX Security'18*.

[48] Denis Foo Kune, John Backes, Shane S Clark, Daniel Kramer, Matthew Reynolds, Kevin Fu, Yongdae Kim, and Wenyuan Xu. 2013. Ghost talk: Mitigating EMI signal injection attacks against analog sensors. In *Proc. of IEEE S&P'13*.

[49] Josephine Lau, Benjamin Zimmerman, and Florian Schaub. 2018. Alexa, are you listening?: Privacy perceptions, concerns and privacy-seeking behaviors with smart speakers. *Proceedings of the ACM on Human-Computer Interaction* 2, CSCW (2018), 102.

[50] Florian Laws and Hinrich Schätze. 2008. Stopping criteria for active learning of named entity recognition. In *Proc. of the 22nd International Conference on Computational Linguistics-Volume 1*.

[51] Shen Li, Zhe Zhao, Tao Liu, Renfen Hu, and Xiaoyong Du. 2017. Initializing convolutional filters with semantic features for text classification. In *Proc. of EMNLP'17*.

[52] Fei Liu, Nicole Lee Fella, and Kexin Liao. 2016. Modeling language vagueness in privacy policies using deep neural networks. In *Proc. of AAAI'16*.

[53] Andrew McCallum and Kamal Nigam. 1998. Employing EM and Pool-Based Active Learning for Text Classification. In *Proc. of ICML'98*.

[54] Dibya Mukhopadhyay, Maliheh Shirvanian, and Nitesh Saxena. 2015. All your voices are belong to us: Stealing voices to fool humans and machines. In *Proc. of European Symposium on Research in Computer Security*.

[55] Pardis Emami Naeini, Sruti Bhagavatula, Hana Habib, Martin Degeling, Lujo Bauer, Lorrie Faith Cranor, and Norman Sadeh. 2017. Privacy expectations and preferences in an IoT world. In *Thirteenth Symposium on Usable Privacy and Security ({SOUPS} 2017)*. 399–412.

[56] Y Nan, Z Yang, M Yang, S Zhou, Y Zhang, G Gu, X Wang, and L Sun. 2017. Identifying User-Input Privacy in Mobile Applications at a Large Scale. *IEEE Trans. Inf. Forensics Secur.* (2017).

[57] Xiang Pan, Yinzhi Cao, Xuechao Du, Boyuan He, Gan Fang, Rui Shao, and Yan Chen. 2018. FlowCog: context-aware semantics extraction and analysis of information flow leaks in android apps. In *Proc. of USENIX Security'18*.

[58] Rahul Pandita, Xusheng Xiao, Wei Yang, William Enck, and Tao Xie. 2013. WHYPER: Towards Automating Risk Assessment of Mobile Applications. In *Proc. of USENIX Security'13*.

[59] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[60] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365* (2018).

[61] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y Ng. 2007. Self-taught learning: transfer learning from unlabeled data. In *Proceedings of the 24th international conference on Machine learning*. ACM, 759–766.

[62] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic keyword extraction from individual documents. *Text Mining: Applications and Theory* (2010).

[63] Nirupam Roy, Sheng Shen, Haitham Hassanieh, and Romit Roy Choudhury. 2018. Inaudible Voice Commands: The Long-Range Attack and Defense. In *Proc. of Networked Systems Design and Implementation (NSDI'18)*.

[64] Norman Sadeh, Alessandro Acquisti, Travis D Breaux, Lorrie Faith Cranor, Aleecia M McDonald, Joel R Reidenberg, Noah A Smith, Fei Liu, N Cameron Russell, Florian Schaub, et al. 2013. *The usable privacy policy project.* Technical Report. Technical report, Technical Report, CMU-ISR-13-119, Carnegie Mellon University.

[65] Sascha Segan. 2018. Google Assistant Now Has 1,830 Actions: Here They Are. https://au.pcmag.com/consumer-electronics-reviews-ratings/47619/feature/google-assistant-now-has-1830-actions-here-they-are.

[66] Bo Sun, Akinori Fujino, and Tatsuya Mori. 2016. POSTER: Toward Automating the Generation of Malware Analysis Reports Using the Sandbox Logs. In *Proc. of CCS'16*.

[67] Cynthia A Thompson, Mary Elaine Califf, and Raymond J Mooney. 1999. Active learning for natural language parsing and information extraction. In *ICML*. Citeseer, 406–414.

[68] Erkam Uzun, Simon Pak Ho Chung, Irfan Essa, and Wenke Lee. 2018. rtCaptcha: A Real-Time CAPTCHA Based Liveness Detection System. In *Proc. of NDSS'18*.

[69] Tavish Vaidya, Yuankai Zhang, Micah Sherr, and Clay Shields. 2015. Cocaine noodles: exploiting the gap between human and machine speech recognition. *WOOT* (2015).

[70] Zhizheng Wu, Sheng Gao, Eng Siong Cling, and Haizhou Li. 2014. A study on replay attack and anti-spoofing for text-dependent speaker verification.. In *APSIPA*.

[71] Zhizheng Wu, Tomi Kinnunen, Nicholas Evans, Junichi Yamagishi, Cemal Hanilçi, Md Sahidullah, and Aleksandr Sizov. 2015. ASVspoof 2015: the first automatic speaker verification spoofing and countermeasures challenge. In *16th Conference of the International Speech Communication Association*.

[72] Yazhou Yang and Marco Loog. 2016. Active learning using uncertainty information. In *Pattern Recognition (ICPR), 2016 23rd International Conference on*. IEEE, 2646–2651.

[73] Xuejing Yuan, Yuxuan Chen, Yue Zhao, Yunhui Long, Xiaokang Liu, Kai Chen, Shengzhi Zhang, Heqing Huang, Xiaofeng Wang, and Carl A Gunter. 2018. CommanderSong: A Systematic Approach for Practical Adversarial Voice Recognition. *arXiv preprint arXiv:1801.08535* (2018).

[74] Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. 2018. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3712–3722.

[75] GYangyong Zhang, Lei Xu, Abner Mendoza, Guangliang Yang, Phakpoom Chinprutthiwong, and Guofei Gu. 2019. Life after Speech Recognition: Fuzzing Semantic Misinterpretation for Voice Assistant Applications. In *Proc. of NDSS'19*.

[76] Guoming Zhang, Chen Yan, Xiaoyu Ji, Tianchen Zhang, Taimin Zhang, and Wenyuan Xu. 2017. DolphinAttack: Inaudible voice commands. In *Proc. of CCS'17*.

[77] Linghan Zhang, Sheng Tan, and Jie Yang. 2017. Hearing your voice is not enough: An articulatory gesture based liveness detection for voice authentication. In *Proc. of CCS'17*.

[78] Nan Zhang, Xianghang Mi, Xuan Feng, XiaoFeng Wang, Yuan Tian, and Feng Qian. 2018. Understanding and Mitigating the Security Risks of Voice-Controlled Third-Party Skills on Amazon Alexa and Google Home. *arXiv preprint arXiv:1805.01525* (2018).

[79] Jingbo Zhu, Huizhen Wang, Eduard Hovy, and Matthew Ma. 2010. Confidence-based stopping criteria for active learning for data annotation. *ACM Transactions on Speech and Language Processing (TSLP)* 6, 3 (2010), 3.