

February 29, 2024

A Testbed for Voice Assistant Traffic Fingerprinting

Master Thesis

Milan van Zanten

milan.vanzanten@unibas.ch

Examiner

Prof. Dr. Isabel Wagner

Faculty of Science

Department of Mathematics and Computer Science

Privacy-Enhancing Technologies Group

<https://pet.dmi.unibas.ch>

ACKNOWLEDGEMENTS

Firstly, I would like to thank Prof. Dr. Isabel Wagner for the opportunity to write my master thesis in her research group. Her insightful perspectives during our progress meetings have been invaluable. Furthermore, I am especially grateful for the help and explanations provided by Diego, Shiva and Patrick, who have generously shared their knowledge with a machine learning novice like me. I also wish to express my heartfelt appreciation to Patricia and Heike for assisting me with navigating the bureaucratic process of delaying my thesis during a difficult time. Last, but certainly not least, I thank Michelle, my family, and my friends for supporting me throughout the work on this thesis. I would not have been able to complete it without them.

ABSTRACT

In this thesis, we investigate the viability of traffic fingerprinting on voice assistants like Apple’s Siri. Initially, we built a testbed that can house different kinds of smart speakers and autonomously interact with the voice assistants running on them. It collects network captures, audio recordings, timestamps and a transcript of the voice assistant’s response. With this system, we conducted an experiment over one month to collect data on approximately 70’000 interactions with an Apple HomePod. As a proof of concept, the collected traffic traces were used to train a machine learning model that can infer what was said to the voice assistant from only the network traffic. Our findings indicate that our system can correctly classify unseen network traffic with an accuracy of up to 86%, revealing a vulnerability in Siri’s security and privacy measures.

Contents

1. Introduction	5
1.1. Previous Work	5
1.2. Thesis Overview	5
2. Background	6
2.1. Traffic Fingerprinting	6
2.2. Threat Model	6
2.3. Voice Assistant Choice	7
3. System Design	8
3.1. Modularisation	8
3.2. Network	8
3.2.1. Monitoring	9
3.3. Audio	9
3.3.1. Text-to-Speech	9
3.3.2. Speech-to-Text	11
3.4. Data Storage	11
3.5. Interaction Process	12
3.6. Performance Considerations	13
3.6.1. Data Compression	14
3.6.2. Audio Transcription	14
4. Experiment	15
4.1. Setup	15
4.2. Data Analysis	16
4.2.1. Traffic Preprocessing	17
4.2.2. Adapting Network from Wang et al.	17
5. Results	19
5.1. Statistics	19
5.2. Transcription Accuracy	20
5.3. Traffic Fingerprinting	21
5.4. On-Device Recognition	25
6. Conclusion	26
6.1. Future Work	26
6.1.1. Recognising Different Speakers	26
6.1.2. Inferring Bounds of Traffic Trace	26
6.1.3. Classifier for Personal Requests	26
6.1.4. Voice Assistants Based on LLMs	26
6.1.5. Conversations	27
6.1.6. Tuning the Model Hyperparameters	27
6.1.7. Non-Uniform Sampling of Interactions According to Real World Data	28
6.1.8. Performance Improvements	28
6.1.9. Other Uses for varies	28
Glossary	29
Bibliography	30
Appendix A: SQL Queries	33
Appendix B: Query Datasets	35

1. Introduction

Smart Speakers have become ubiquitous in many countries. The global sale of Amazon Alexa devices surpassed half a billion in 2023 [1] and Apple sells over 10 million HomePods every year [2]. A study shows that 35% of adults in the U.S. own a smart speaker [3]. With half of the participants in the study reporting to have heard an advertisement on their smart speaker before, concerns about the privacy and security of these ever-listening devices are well founded. Especially since they are usually placed where private or confidential conversations take place.

To back claims about loss of privacy or decreased security, a thorough analysis of smart speakers becomes necessary. In this thesis, we build a system that allows us to collect a large amount of data on interactions with voice assistants.

1.1. Previous Work

There are several previous works analysing the privacy and security risks of voice assistants. Most of them go into topics concerning specifically Amazon Alexa since their adoption rate is significantly higher than Apple’s Siri or Google Assistant [4]. Some examples include experiments of smart speaker misactivations [5]–[7], analyses of voice assistant skills [8]–[12] and voice command fingerprinting [13], [14]. Apart from some old work on traffic patterns of Siri [15] and research by Apple [16], [17], we found little information about that voice assistant.

1.2. Thesis Overview

The primary bottleneck when collecting large amounts of data on voice assistants is the actual time speaking and waiting for the response. According to our tests, the majority of interactions take between 4 and 8 seconds plus any additional delay between the question and the response. This aligns with the results by G. Haas *et al.* [18], who found a median of 4.03s per typical sentence. With the goal of collecting a large number of samples for a list of voice commands (henceforth called “queries”), the system will therefore need to run for a substantial amount of time.

With this in mind, the primary aim of this thesis is building a testbed that can autonomously interact with a smart speaker. This system should take a list of queries and continuously go through it while collecting audio recordings, network packet traces, timestamps and other relevant metadata about the interaction. In Section 2, we establish why it is of interest to collect this data and the implementation of our system is explored in Section 3.

For the second part, documented in Sections 4 and 5, we run an experiment using our system, collecting interactions with an Apple HomePod mini¹ smart speaker. With this dataset of interactions, we train an ML (*machine learning*) model for traffic fingerprinting. The system is adapted from a similar implementation by C. Wang *et al.* [14] but will be applied to data collected from the Siri voice assistant instead of Alexa and Google Assistant.

¹<https://apple.com/homepod-mini>

2. Background

Having defined our unsupervised system for collecting interactions with smart speakers, it is necessary to explain the reasoning behind collecting that data. In the taxonomy in Diagram 1, adapted from a more detailed version in the survey by E. Papadogiannaki and S. Ioannidis [19], we can see that security and privacy are two of the main focus points for research on the analysis of encrypted network traffic.

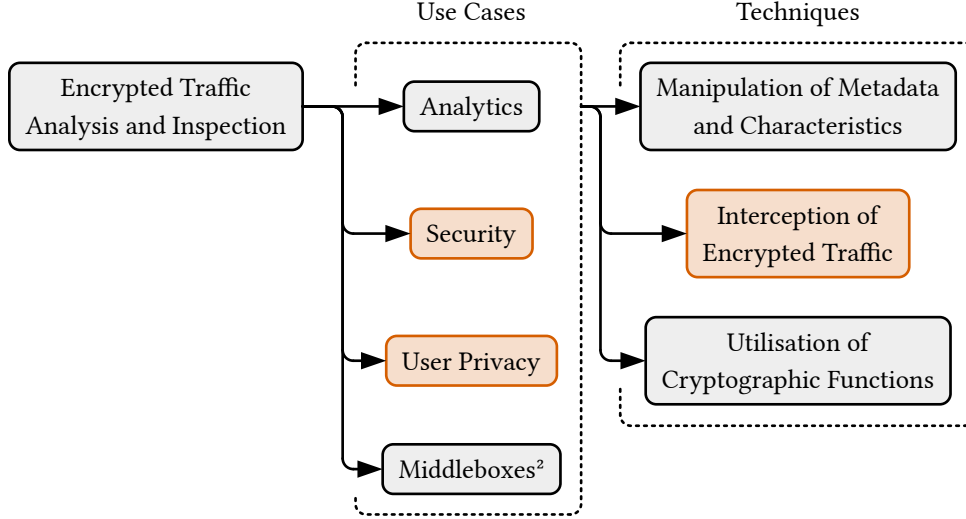


Diagram 1: A Taxonomy for research about encrypted network traffic analysis [19].

E. Papadogiannaki and S. Ioannidis write in their survey “In the second category *interception of encrypted traffic*, we encounter solutions that take advantage of behavioural patterns or signatures to report a specific event.” [19] The main focus for us lies on the encrypted network traffic being collected. This traffic cannot be decrypted, however, we can analyse it using a technique called traffic fingerprinting.

2.1. Traffic Fingerprinting

Traffic fingerprinting uses pattern recognition approaches on collected, encrypted traffic traces to infer user behaviour. This method is used successfully to analyse web traffic [20], traffic on the Tor network [21], IoT (*Internet of Things*) device traffic [13], [14], [22]–[24] and user location [25]. Traffic fingerprinting on smart speakers is generally used to infer the command given to the voice assistant by the user. To enable us to run traffic fingerprinting, our system stores the query and the network traffic for each interaction it has with the smart speaker.

We assume the concept of *open world* and *closed world* models from previous traffic fingerprinting works [14], [20]. Both models define an attacker who has a set of queries they are interested in. The interest in the open world scenario is whether or not a given command is in that set. The traffic analysed in this case can be from either known queries or queries not seen before. In the closed world scenario, the attacker wants to differentiate between the different queries in that set, but cannot tell whether a new query is in it or not. As an example, the open world model could be used to infer whether the smart speaker is being used to communicate with a person that an attacker is monitoring. In the closed world case, the attacker might want to differentiate between different people a user has contacted. In our thesis, we focus on the closed world model.

2.2. Threat Model

Our threat model is similar to the ones introduced by C. Wang *et al.* [14] and [13], specifically,

²Devices used for network function, e.g. NATs or firewalls.

- the attacker has access to the network the smart speaker is connected to;
- the MAC address of the smart speaker is known;
- the type of the voice assistant is known (e.g. Siri, Alexa or Google Assistant).

Previous work [22], [24] shows that it is feasible to extract this information from the encrypted network traffic. Typically unencrypted DNS queries can also be used to identify smart speakers on a network. Our own analysis shows that the Apple HomePod regularly sends DNS queries asking about the domain `gsp-ssl.ls-apple.com.akadns.net` and C. Wang *et al.* [14] found that the Amazon Echo smart speaker sends DNS queries about `unagi-na.amazon.com`. The specific domains queried may differ regionally (as indicated by the `-na` part in the Amazon subdomain), but should be simple to obtain.

Finally, we assume the traffic is encrypted, since the attack would be trivial otherwise.

2.3. Voice Assistant Choice

The most popular voice assistants are Apple’s Siri, Amazon Alexa and Google Assistant [4]. Support for Microsoft Cortana was dropped in August of 2023 [26], likely in favour of Copilot for Windows, which was announced one month later [27] (for more on this see Section 6.1.4).

Our system supports the Siri and Alexa voice assistants but can be easily extended with support for more. For the experiment in Section 4, we decided to use Siri as it is still less researched [4] than Alexa, where a host of security and privacy issues have already been found [14], [13], [5], [11], [6].

3. System Design

The primary product of this thesis is a testbed that can autonomously interact with a smart speaker for a long time. Our system, called `varys`, collects the following data on its interactions with the voice assistant:

- Audio recordings of the query and the response
- A text transcript of the query and the response
- Encrypted network traffic traces from and to the smart speaker
- The durations of different parts of the interaction

3.1. Modularisation

The wide range of data our system needs to collect makes it necessary to keep its complexity manageable. To simplify maintenance and increase the extensibility, the functionality of our system is separated into five modules:

`varys` The main executable combining all modules into the final system.

`varys-analysis` Analysis of data collected by `varys`.

`varys-audio` Recording audio and the TTS (*text-to-speech*) and STT (*speech-to-text*) systems.

`varys-database` Abstraction of the database system where interactions are stored.

`varys-network` Collection of network traffic, writing and parsing of `.pcap` files.

The modules `varys-audio`, `varys-database` and `varys-network` are isolated from the rest of `varys`. If the need arises (e.g. if a different database connector is required or a more performant TTS system is found), they can be easily swapped out. The dependencies between all modules are shown in Diagram 2.

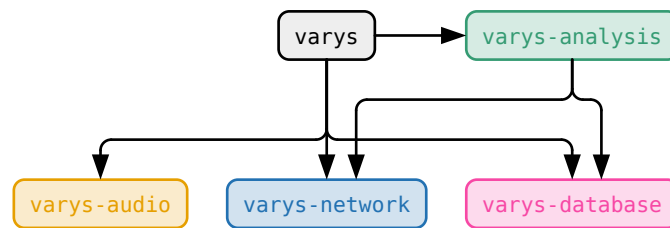


Diagram 2: Module dependencies.

The module `varys-analysis` does not currently do any analysis of the recorded audio and thus doesn't depend on that module.

In this section, we will lay out the design considerations behind the modules that make up `varys`. The colours shown in Diagram 2 will indicate which module each specific functionality belongs to.

3.2. Network

The system runs on a device that serves as a MITM (*man-in-the-middle*) between the Internet and an *internal network*, which it acts as a router for. The latter is reserved for the devices that are monitored. This includes the voice assistant and any devices required by it. Since the majority of smart speakers do not have any wired connectivity, the *internal network* is wireless.

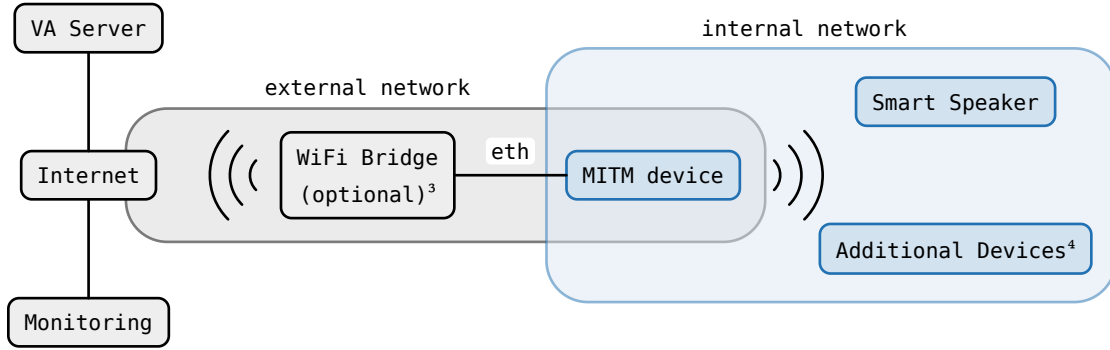


Diagram 3: Visualisation of the network layout.

This layout, visualised in Diagram 3, enables the MITM device to record all incoming and outgoing traffic to and from the voice assistant without any interference from other devices on the network. (We filter the network traffic by the MAC address of the smart speaker so this is not strictly necessary, but it helps to reduce the amount of garbage data collected.)

Currently, we assume that the beginning and end of an interaction with the voice assistant is known and we can collect the network capture from when the user started speaking to when the voice assistant stopped speaking. We discuss this restriction in more detail in Section 6.1.2.

The `varys-network` module provides functionality to start and stop capturing traffic on a specific network interface and write it to a `.pcap` file. Furthermore, it offers a wrapper structure around the network packets read from those files, which is used by `varys-analysis`. Each smart speaker has a different MAC address that is used to distinguish between incoming and outgoing traffic. The MAC address is therefore also stored in the database alongside the path to the `.pcap` file.

3.2.1. Monitoring

Since the system is running for long periods of time without supervision, we need to monitor potential outages. We added support to `varys` for *passive* monitoring. A high-uptime server receives a request from the system each time it completes an interaction. That server in turn notifies us if it hasn't received anything for a configurable amount of time.

3.3. Audio

The largest module of our system is `varys-audio`. Its main components are the TTS and STT systems, expanded upon in the next two subsections. Apart from that, it stores the recorded audio files encoded in the space-optimised OPUS [28] format and handles all audio processing. Converting it to mono; downsampling it for space efficiency; and trimming the silence from the beginning and end.

3.3.1. Text-to-Speech

The only interface that a voice assistant provides is via speaking to it. Therefore, we need a system to autonomously talk, for which there are two options:

- Prerecorded audio
- Speech synthesis

The former was dismissed early for several reasons. Firstly, audio recordings would not support conversations that last for more than one query and one response. Secondly, each change in our dataset would necessitate recording new queries. Lastly, given the scope of this thesis, recording and editing more than 100 sentences was not feasible.

³Only required if the MITM device cannot be connected directly while acting as router for the internal network.

⁴In the case of the HomePod, this is the iPhone required for personal requests.

Thus, we decided to integrate a TTS system into our **varys-audio** module. This system must provide high quality synthesis to minimise speech recognition mistakes on the smart speaker, and be able to generate speech without any noticeable performance overhead. The latter is typically measured by the f_{rt} (*real-time factor*), with $f_{rt} > 1$ if the system takes longer to synthesise text than to speak that text and $f_{rt} \leq 1$ otherwise. A good real-time factor is especially important if our system is to support conversations in the future; the voice assistant only listens for a certain amount of time before cancelling an interaction.

We explored three different options with $f_{rt} \leq 1$ on the machine we tested on:

- **espeak**⁵
- **Larynx**⁶
- **tts-rs** using the AVFoundation backend⁷

We primarily based our choice on how well Amazon Alexa⁸ would be able to understand different sentences. A test was performed manually by asking each of three questions 20 times with every TTS system. As a ground truth, the same was repeated verbally. The results, found in Figure 1, show that **tts-rs** with the AVFoundation backend performed approximately 20% better on average than both Larynx and espeak and came within 15% of the ground truth.

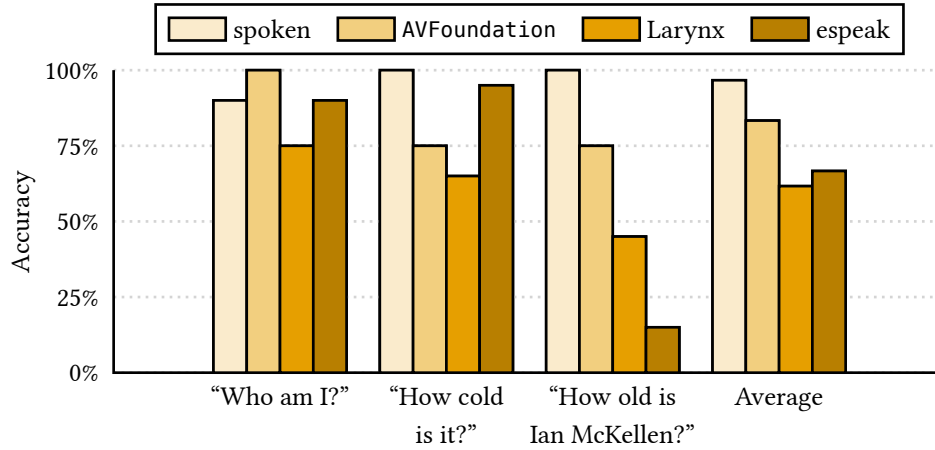


Figure 1: Recognition accuracy of sentences synthesised by different TTS systems and one human voice.

While performing this test, the difference in quality was very easily heard, as only the AVFoundation voices come close to sounding like a real voice. These were, apart from the ground truth, also the only voices that could be differentiated from each other by the voice assistants. Figure 2 shows how well Alexa was able to distinguish between two voices registered with voice recognition and one unregistered “guest” voice. The recognition of user voices works well while the guest voice was wrongly recognised as one of the others 65% of the time.

⁵<https://espeak.sourceforge.net>

⁶<https://github.com/rhasspy/larynx>

⁷<https://github.com/ndarilek/tts-rs> – the AVFoundation backend is only supported on macOS.

⁸We picked Alexa (over Siri) for these tests since its speech recognition performed worse in virtually every case.

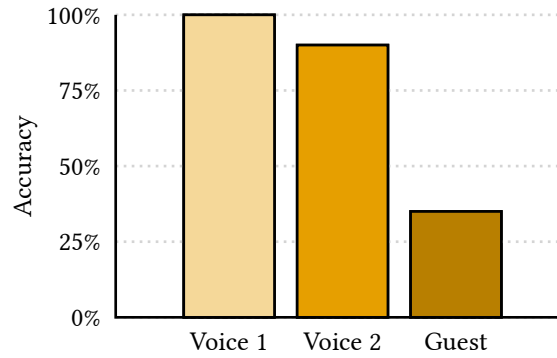


Figure 2: Recognition accuracy of different voices synthesised by AVFoundation.

With these results, we decided to use `tts-rs` on a macOS machine to get access to the AVFoundation backend.

Since running this test, Larynx has been succeeded by Piper⁹ during the development of our system. The new system promises good results and may be an option for running `varys` on non-macOS systems in the future.

3.3.2. Speech-to-Text

Not all voice assistants provide a history of interactions, which is why we use a STT system to transcribe the recorded answers.

In our module `varys-audio`, we utilise an ML-based speech recognition system called `whisper`. It provides transcription error rates comparable to those of human transcription [29] and outperforms other local speech recognition models [30] and commercial services [29]. In Section 5.2, we explore how well `whisper` worked for our use case.

3.4. Data Storage

The module `varys-database` acts as an abstraction of our database, where all metadata about interactions are stored. Audio files and `.pcap` traces are saved to disk by `varys-audio` and `varys-network` respectively and their paths are referenced in the database.

Since some changes were made to the system while it was running, we store the version of `varys` a session was ran with. This way, if there are any uncertainties about the collected data, the relevant code history and database migrations can be inspected.

The final schema of the database is shown in Table 1. To prevent data corruption if the project is moved, all paths are relative to a main data directory.

⁹<https://github.com/rhasspy/piper>

id	Sequential id of the interaction ¹⁰
session_id	Foreign key of the session
query	What was spoken
query_duration	Speaking duration in ms
response	Transcription of the response
response_duration	Response duration in ms
query_file	Path of the query audio file
response_file	Path of the response audio file
capture_file	Path of the network capture file
started	Timestamp when the interaction began
ended	Timestamp when the interaction ended ¹¹
query_category	The category of the query
assistant_mac	The MAC address of the smart speaker

id	Sequential id of the session
version	The version of varys
interactor_config_id	Foreign key of the configuration
data_dir	Path to where this session is stored
started	Timestamp when the session began
ended	Timestamp when the session ended

id	Sequential id of the configuration
interface	The network interface where traffic is collected
voice	The voice used for speaking
sensitivity	The amplitude of audio considered silence
model	The whisper model used for transcription

Table 1: Schema of the interaction, session and interactor_config tables.

3.5. Interaction Process

The main executable of varys combines all modules introduced in this section to run a series of interactions. When the system is run, it loads the queries from a TOML file formatted like the example in Listing 1. Until it is stopped, the system then runs interaction sessions, each with a shuffled version of the original query list.

```
conversion = [
    "What are 130 miles in yards?",
    "What's 2330 dollars in euros?",
]

mathematics = [
    "What's 9 plus 53?",
]

home = [
    "Open the garage",
]
```

Listing 1: Example of a very small dataset of four queries grouped in three categories.

The exact process is shown in Diagram 4. Resetting the voice assistant is implemented differently for each one supported. As an example, the exact procedure of resetting Siri is shown in Algorithm 1.

Algorithm 1: Resetting Siri after a recording timeout.

```
1 say "Hey Siri, stop."
2 wait for 2s of silence
3 say "Hey Siri, turn off the music."
4 wait for 2s of silence
5 say "Hey Siri, disable all alarms."
6 wait for 2s of silence
```

¹⁰Stays sequential across sessions for easier identification.

¹¹This includes the time taken to transcribe the response audio.

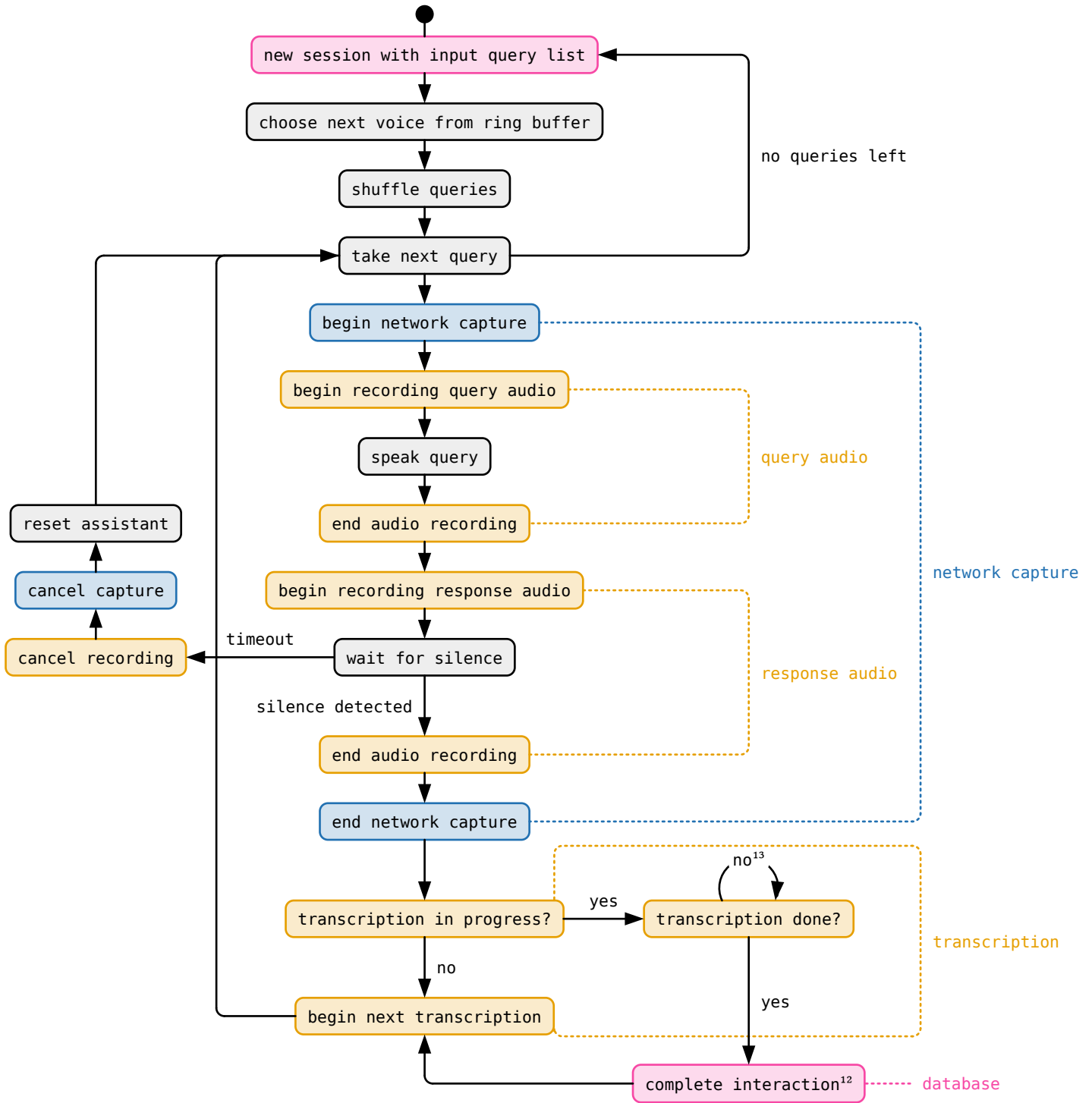


Diagram 4: Data collection flow chart.

3.6. Performance Considerations

The initial design of our system intended for it to be run on a Raspberry Pi, a small form factor computer with limited processing and storage capabilities. Mostly because we did not find a TTS system that ran with satisfactory performance on the Raspberry Pi, we now use a laptop computer.

¹²The previous interaction whose transcription was just completed.

¹³Sleep for a short amount of time before checking again.

3.6.1. Data Compression

Because of the initial plan of using a Raspberry Pi, we compress the recorded audio with the OPUS [28] codec and the captured network traffic with DEFLATE [31]. We are however now running the experiment on a laptop, where space is no longer a concern. The audio codec does not add a noticeable processing overhead, but the .pcap files are now stored uncompressed. Moreover, storing the network traces uncompressed simplifies our data analysis since they can be used without any preprocessing.

3.6.2. Audio Transcription

Apart from the actual time speaking (as mentioned in Section 1.2), the transcription of audio is the main bottleneck for how frequently our system can run interactions. The following optimisations allowed us to reduce the average time per interaction from ~ 120 s to ~ 30 s:

Model Choice. There are five different sizes available for the whisper model, shown in Table 2. The step from the medium to the large and large-v2 models does not significantly decrease WER (word error rate) but more than doubles the number of parameters. Since in our tests, this increase in network size far more than doubles processing time, we opted for the medium model.

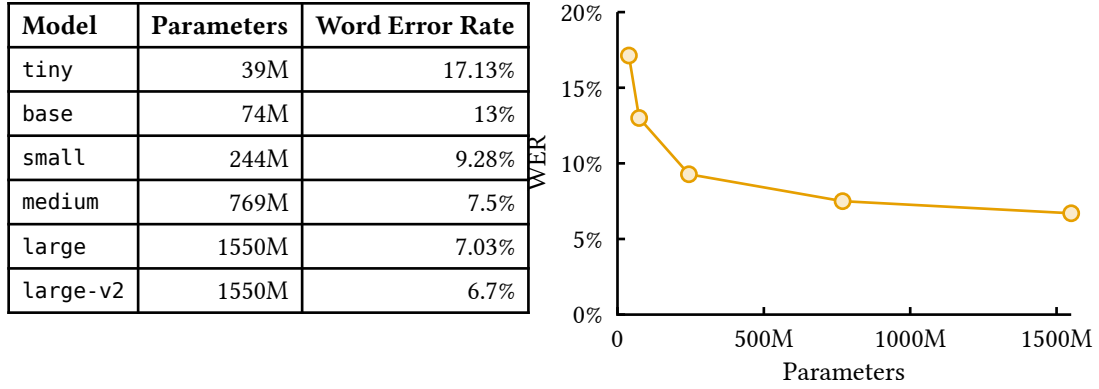


Table 2: Comparison of whisper models by WER on English datasets [29].

Transcription in Parallel. The bottleneck of waiting for whisper to transcribe the response could be completely eliminated with a transcription queue that is processed in parallel to the main system. However, if the system is run for weeks at a time, this can lead to arbitrarily long post-processing times if the transcription takes longer on average than an interaction.

Our solution is to process the previous response while the next interaction is already running, but to wait with further interactions until the transcription queue (of length 1) is empty again. This approach allows us to reduce the time per interaction to $\max(T_t, T_i)$, rather than $T_t + T_i$, where T_t represents the time taken for transcription and T_i the time taken for the rest of the interaction.

Audio Preprocessing. Another step we took to reduce the transcription time is to reduce the audio sample rate to 16 kHz, average both channels into one and trim the silence from the start and the end. This preprocessing does not noticeably reduce the quality of the transcription. Moreover, cutting off silent audio eliminates virtually all whisper hallucinations (see Section 5.2).

4. Experiment

To test our data collection system, we conducted several experiments where we let varys interact with an Apple HomePod mini.

Three different lists of queries were tested:

- A large dataset with 227 queries, split into 24 categories
- A small dataset with 13 queries, each from a different category
- A binary dataset with the two queries “Call John Doe” and “Call Mary Poppins”

Apple does not release an official list of Siri’s capabilities. A dataset of ~800 Siri commands was originally compiled from a user-collected list [32] and extended with queries from our own exploration. The HomePod does not support all Siri queries¹⁴. Therefore, during an initial test period, we manually removed commands from the list that did not result in a useful response. The number of redundant commands was further reduced to increase the number of samples that can be collected per query. Having less than 256 labels has the additional advantage of our query labels fitting in one byte.

The full lists of queries are shown in Tables 8, 7 and 6. Further analysis of our results can be found in Section 5. In this section we detail the experiment setup and our traffic fingerprinting system.

4.1. Setup

The experiment setup consists of a laptop running macOS¹⁵, connected to a pair of speakers and a microphone. The latter, together with the HomePod, are placed inside a sound-isolated box (shown in Figure 3). Additionally, the user’s iPhone is required to be on the *internal network* (see Section 3.2) for Siri to be able to respond to personal requests.

The iPhone is set up with a dummy user account for *Peter Pan* and two contacts, *John Doe* and *Mary Poppins*. Otherwise, the phone is kept at factory settings and the only changes come from interactions with the HomePod.

Finally, to minimise system downtime, an external server is used for monitoring as explained in Section 3.2.1.



Figure 3: Sound-isolation to improve transcription accuracy.

¹⁴Any queries that require Siri to show something to the user will be answered with something along the lines of “I can show you the results on your iPhone.”

¹⁵The choice of using macOS voices is explained in Section 3.3.1.

The experiment must be set up in a location with the following characteristics:

- Wired Internet connection¹⁶
- Power outlet available
- No persons speaking (the quieter the environment, the higher the voice recognition accuracy)
- Noise disturbance from experiment not bothering anyone
- Access to room during working hours (or if possible at any time)

The sound isolation of the box can reduce ambient noise in the range of 50 – 60 dB to a significantly lower 35 – 45 dB, which increases the voice recognition and transcription accuracy. This also helps finding an experiment location, since the noise disturbance is less significant.

When the hardware is set up, the command `varys listen - -calibrate` is used to find the ambient noise volume.

4.2. Data Analysis

We began data analysis by building a simple tool that takes a number of traffic traces and visualises them as shown in Figure 4. Red and blue represent incoming and outgoing packets respectively and the strength of the colour visualises the packet size. The x -axis does not necessarily correspond with time, but with the number of incoming and outgoing packets since the beginning of the trace. This method of analysing traffic traces aligns with previous work on traffic fingerprinting [14].



Figure 4: Example traffic trace visualisation of the query “What’s the temperature outside?”.

The example in Figure 4 already shows a clear correlation between the traces (excluding some outliers where the HomePod was likely downloading updates or doing other unrelated tasks). To test whether we can differentiate between multiple queries, we trained several ML classifiers on our data.

¹⁶A wireless connection is not possible, since macOS does not support connecting to WiFi and creating a wireless network at the same time. If no wired connection is available, a network bridge (e.g. a Raspberry Pi bridging WiFi to its Ethernet port) can be used to connect the Mac to the internet.

4.2.1. Traffic Preprocessing

We write a network traffic trace as $((s_1, d_1), (s_2, d_2), \dots, (s_n, d_n))$, where n is the number of packets in that trace, s_i is the size of a packet in bytes and $d_i \in \{0, 1\}$ is the direction of that packet.

To be used as input in our ML network, the traffic traces were preprocessed into normalised tensors of length 475. Our collected network traffic is converted like

$$((s_1, d_1), (s_2, d_2), \dots, (s_n, d_n)) \rightarrow (p_1, p_2, \dots, p_{475}),$$

where for all $i \in [1, 475]$ the components are calculated as

$$p_i = \begin{cases} (-1)^{d_i} \cdot \frac{s_i}{1514} & \text{if } i < n, \\ 0 & \text{else.} \end{cases}$$

This pads traces shorter than 475 packets long with zeroes, cuts off the ones with more and normalises all traces into the range $[-1, 1]$ ¹⁷.

4.2.2. Adapting Network from Wang et al.

Since there is similar previous work on traffic fingerprinting Amazon Alexa and Google Assistant interactions by C. Wang *et al.* [14], we based our network on their CNN (*Convolutional Neural Network*), shown in Diagram 5.

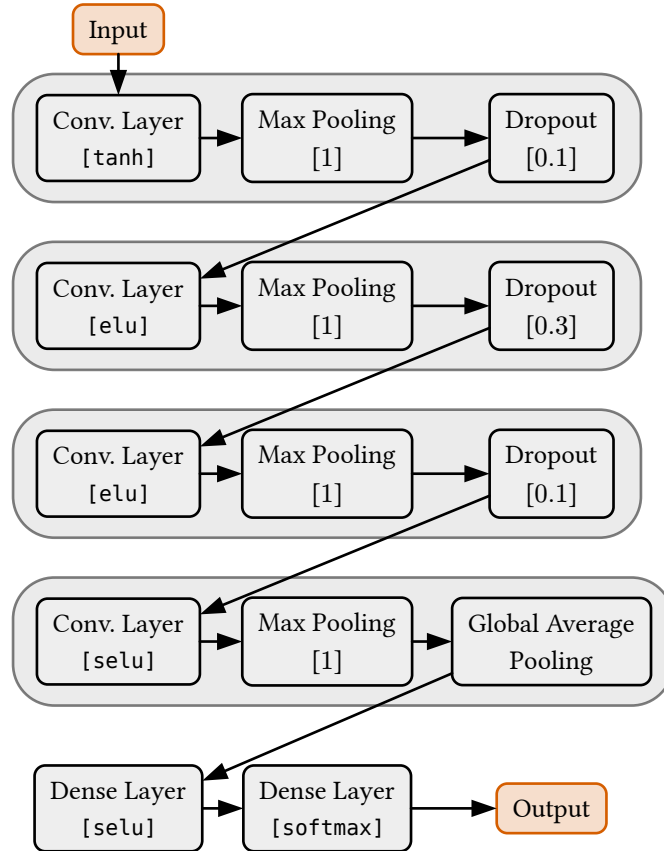


Diagram 5: The CNN design used in [14].

¹⁷The maximum size of the packets is 1514 bytes, which is why we can simply divide by this to normalise our data.

In addition to a CNN, they also tested an LSTM (*Long Short-Term Memory*), a SAE (*Stacked AutoEncoder*) and a combination of the networks in the form of ensemble learning. However, their best results for a single network came from their CNN, so we decided to use that.

Owing to limited computing resources, we reduced the size of our network to the one shown in Diagram 6. The network of C. Wang *et al.* used a pool size of 1, which is equivalent to a *no-op*. Thus, we removed all *Max Pooling* layers. Additionally, we reduced the four groups of convolutional layers to one.

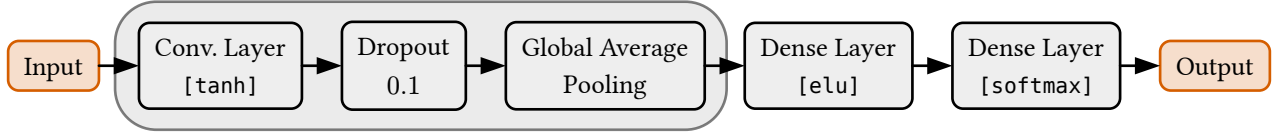


Diagram 6: The CNN implemented by us.

Since the hyperparameter search performed by C. Wang *et al.* found 180 to be the optimal dense network size when that was the maximum in their search space $\{100, 110, \dots, 170, 180\}$, we can assume that the true optimal size is higher than 180. In our case we used 475 – the size of our input.

To split out dataset into separate training, validation and testing parts, we used the same proportions as C. Wang *et al.* [14]; Of the full datasets, 64% were used for training, 16% for validation, and 20% for testing.

5. Results

Our system ran for approximately 800 hours and collected data on a total of 73'605 interactions. The results of the data collection for each dataset can be seen by using SQL listings 1, 2 and 3. The collected data is available at <https://gitlab.com/m-vz/varys-data>.

In this section we will go over our results, beginning with the statistics of our collected dataset.

5.1. Statistics

Of the 73'605 queries collected, 71'915 were successfully completed (meaning there was no time-out and the system was not stopped during the interaction). Our system varies is therefore able to interact with Siri with a 97.70% success rate¹⁸.

The average duration of interactions is shown in Figure 5 (data from SQL listings 4 and 5). As mentioned in Section 1.2, there are approximately 4 to 8 seconds of speaking during most interactions. The remaining duration (measured as the total duration minus the speaking duration) lies mostly between 25 and 30 seconds. This includes the time spent waiting for the voice assistant to answer and transcribing the response. It did not increase even for outliers with a significantly longer speaking duration.

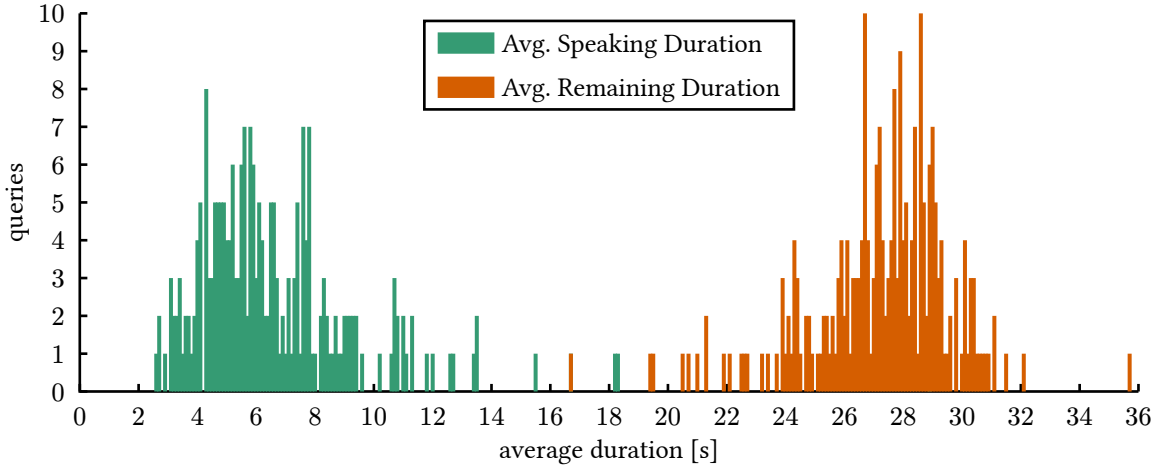


Figure 5: Distribution of average query duration. Outliers above 20s are shown in Table 3.

Query	Avg. Speaking Duration	Avg. Remaining Duration
Hey Siri. Tell me a bedtime story	79.53s	31.06s
Hey Siri. Tell me a story	68.86s	35.73s
Hey Siri. Tell me a poem	28.92s	21.31s

Table 3: Outlier queries with a speaking duration above 20s.

On average, a query took 2.47 seconds (± 0.64) to say, which is expected since all query sentences are of similar length. The responses lasted on average 4.40 seconds with a standard deviation of 5.63s.

In Figure 6 we show the uptime of our system in hours from when it was set up. During the first week, there were two bugs that resulted in several downtimes (the longest of which lasted ~ 42 h because we could not get physical access to the system during the weekend). After that, the system ran flawlessly except during one network outage at the university and during maintenance.

¹⁸See SQL listing 8.

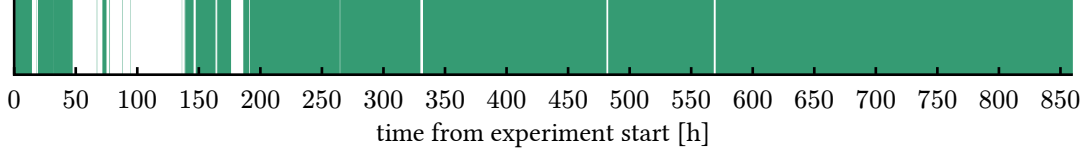


Figure 6: Uptime of our system.

5.2. Transcription Accuracy

Compared to the results by A. Radford *et al.* [29] and A. Seagraves [30], our own results show an even higher transcription accuracy apart from errors in URLs, certain names and slight formatting differences. This is likely due to the audio coming from a synthetic voice in a controlled environment. Table 4 shows the manually verified results for six randomly picked queries (see SQL listing 6). We filtered out any invalid queries where Siri was waiting for the answer to a follow-up question or could not answer due to a network issue.

Query	Correct	Incomplete	Incorrect	Invalid
“How far away is Boston?”	61.94% (83)	36.57% (49) ¹⁹	1.49% (2)	6
“What is the factorial of 6?”	99.08% (1823)	0% (0)	0.92% (17)	6
“Define airplane”	94.44% (136)	3.47% (5)	2.08% (3)	4
“Call back my last missed call”	99.28% (137)	0.72% (1)	0% (0)	8
“What’s 9 plus 53?”	99.28% (138)	0% (0)	0.72% (1)	5
“What’s 200 pounds in kilograms?”	99.3% (142)	0% (0)	0.7% (1)	2
“Flip a coin”	2.07% (3)	0.69% (1)	97.24% (141)	0

Table 4: whisper speech recognition accuracy samples.

The query “How far away is Boston?” was missing “as the crow flies” from the correct transcription “[...] it’s about 5,944 kilometers as the crow flies.” about 36% of the time. Together with the correctly transcribed cases, this makes up 98.51% of this query’s data. In the response to “What is the factorial of 6?”, Siri provided the source [solumaths.com](https://www.solumaths.com), which is text that whisper was unable to produce²⁰. Since mistakes in URLs are expected, we marked those transcriptions as correct.

There are some rare cases where the voice assistant responds with a single word. Since whisper does not support recognition of audio shorter than one second, these queries have an exceedingly high failure rate. One example is the query “Flip a coin”, which was only recognised correctly ~2% of the time. Out of the 140 incorrectly transcribed responses, 134 were cancelled due to the recorded audio being too short.

Another problem we identified with whisper is its tendency to hallucinate text when fed silent audio (one or more seconds of 0-samples). Some examples of text hallucinated from empty audio is shown in Table 5. Preventing this can be done by simply trimming off all silent audio in the preprocessing step before the transcription.

¹⁹These were all cases where the transcription of was missing

²⁰In 1823 cases the name *solumaths* was recognised as either *solomus*, *solomoths*, *solomons*, *solomuths*, *soloomiths*, *salumith* or *solemnus*, but never correctly.

Silent Audio [ms]	Transcription
3054	<i>When you find out about any of our videos, please like and subscribe, and we will see you all next time!</i>
744	<i>Please see review No.106243 on PissedConsumer.com</i>
2944	<i>Go to Beadaholique.com for all of your beading supply needs!</i> <i>Thank you for watching.</i> <i>Thank you for watching.</i>
1376	<i>It's super short.</i> <i>\$1.99."</i> <i>\$1.99.</i>

Table 5: *whisper* speech recognition hallucinations when transcribing empty audio.

5.3. Traffic Fingerprinting

With our implementation of a fingerprinting CNN, we were able to get promising results. Our network was trained on the binary, small and large datasets for 1000 epochs each. Detailed training statistics can be found in Figure 7. The accuracy axis of the training progress is bounded by the interval $[\frac{100}{n}, 100]$, where n is the number of labels. The bottom of the accuracy axis is therefore equivalent to random choice. (As an example, the accuracy of the binary classifier starts at 50%, which is choosing randomly between two options.) The loss axis is bounded to $[0, 3]$ to fit our data. The loss for training our large dataset started outside that range at 5.05.

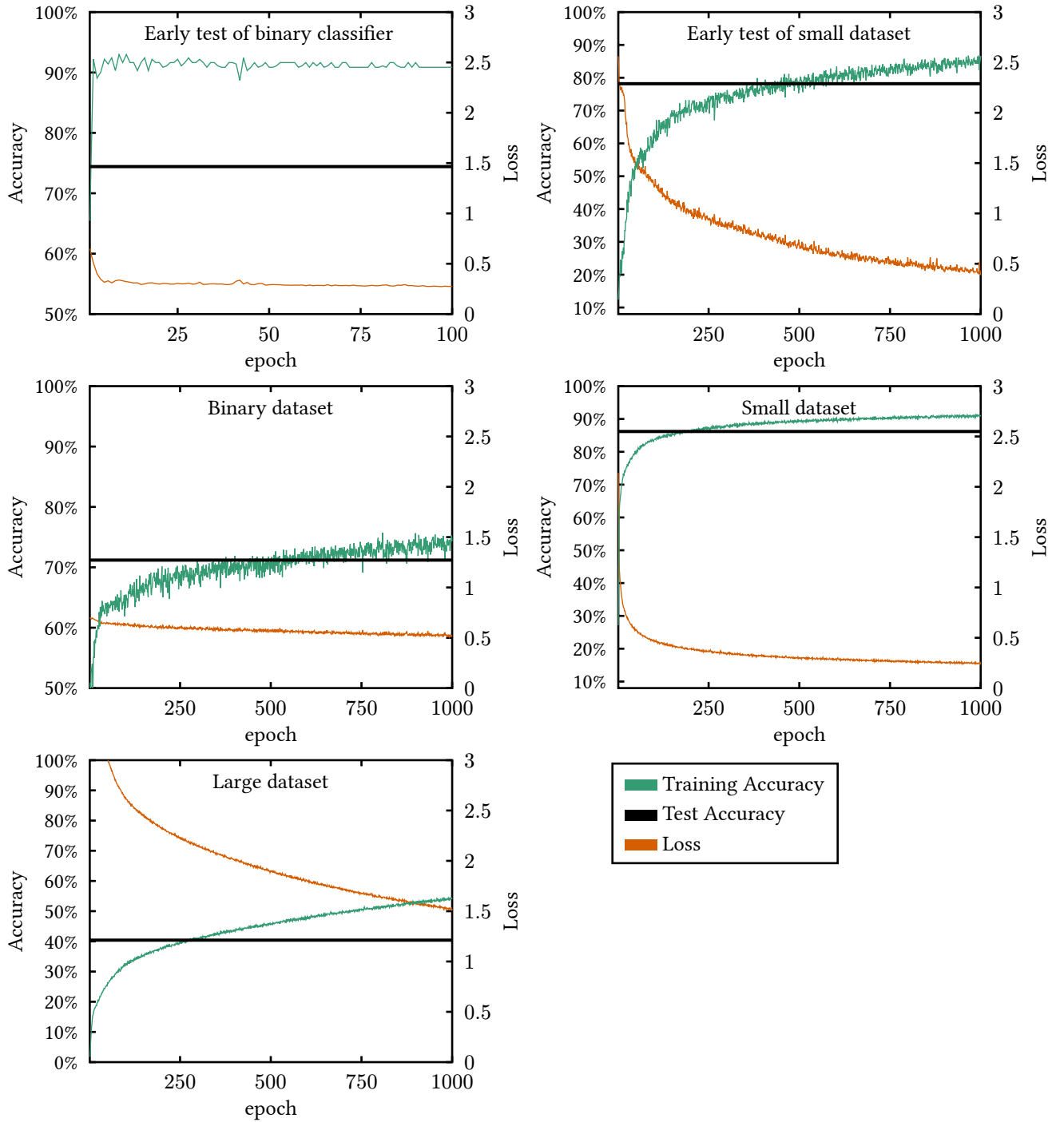


Figure 7: A selection of training and validation progress charts.

Most of the trends in these graphs show that with more training, the models could likely be improved. However, since this serves as a proof of concept and due to limited time and processing capacity, we did not train each network for more than 1000 epochs.

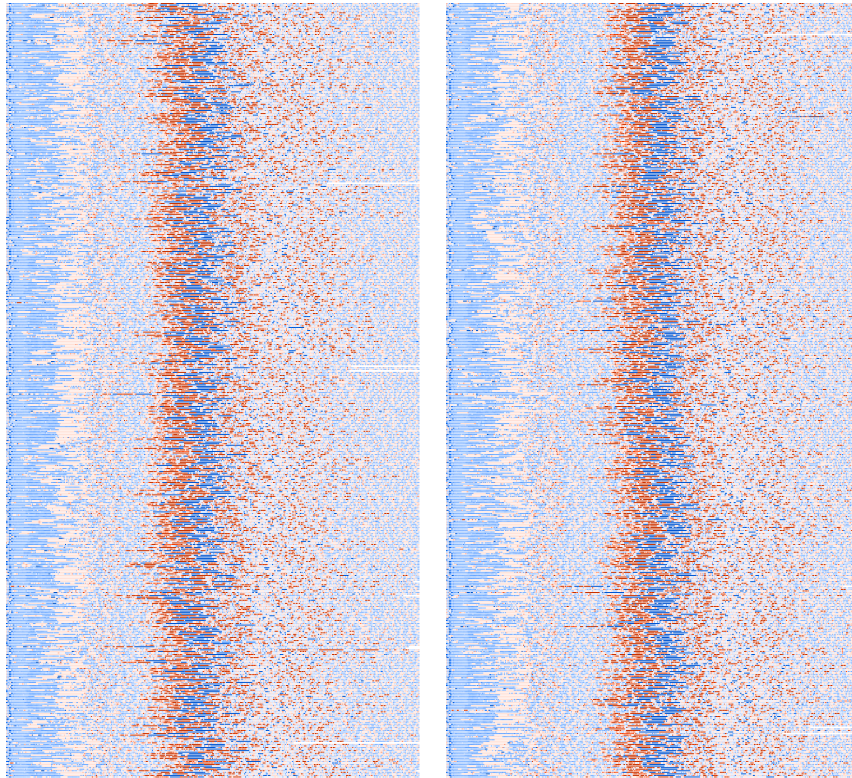
For binary classification between the queries “Call John Doe” and “Call Mary Poppins”, our final accuracy on the test set was $\sim 71.19\%$. This result is especially concerning since it suggests this method can differentiate between very similar queries. An attacker could train a network on names of people they want to monitor and tell from the encrypted network traffic whether the voice assistant was likely used to call that person.

Figure 8 shows 500 traces from each of the two binary classification queries. These two queries are visually clearly more similar to each other compared to the traces in Figure 9.

The second classification we attempted was between the 13 different queries from Table 7, each from a different category. The result for this dataset is an 86.19% accuracy on the test set. Compared to less than 8% for a random guess, this confirms our concerns about the privacy and security of the HomePod.

The large dataset with 227 different queries showed an accuracy of 40.40%. This result is likely limited by the size of our network and training capabilities. When compared to randomly guessing at 0.44% though, the loss of privacy becomes apparent.

With adjustments to our CNN, more training data and hardware resources, we estimate that highly accurate classification can be achieved even between hundreds of different queries.



*Figure 8: Traffic traces for the two queries in the binary dataset.
Each image contains 500 traces.*

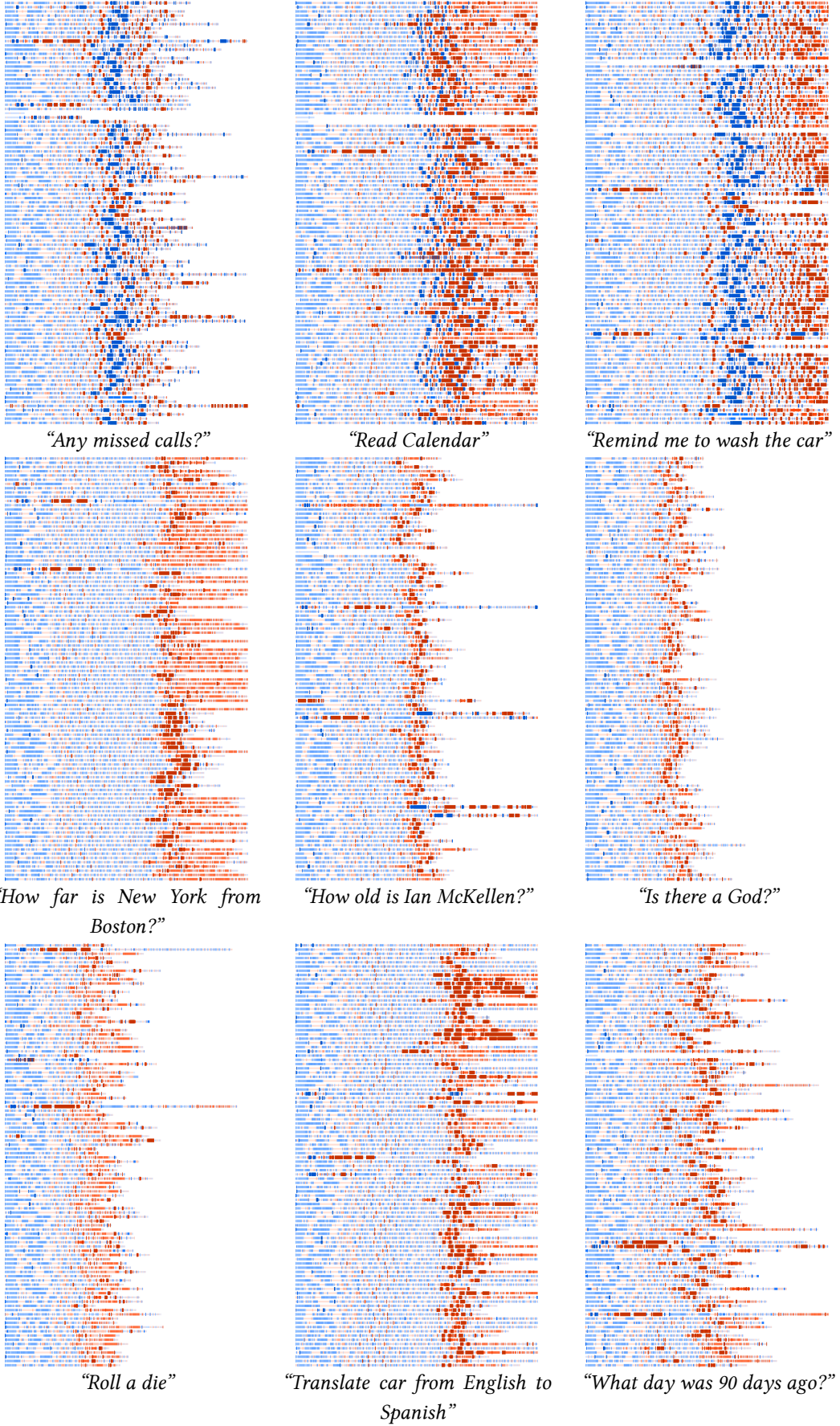


Figure 9: Traffic traces for nine different queries from the small dataset.
Each image contains 100 traces.

5.4. On-Device Recognition

According to HomePod marketing,

HomePod mini works with your iPhone for requests like hearing your messages or notes, so they are completed on device without revealing that information to Apple.

— <https://apple.com/homepod-mini>

Since our threat model from Section 2.2 assumes we collect data from the network that both the HomePod and iPhone are on, we still have access to that traffic. The difference in how traffic of normal queries and personal requests is handled can be seen in the collected traces. When looking at the visualisations of the traffic in Figure 8 and the traces for “*Any missed calls?*”, “*Read Calendar*” and “*Remind me to wash the car*” in Figure 9, (by relying on the sophistication of human pattern recognition,) a group of large packets being sent (in dark blue) after the first incoming packets can be seen across the vast majority of traces. This group of packets is missing from the normal queries seen in Figure 9. We speculate a binary classifier to distinguish those two types of requests should be trivial to build. However, due to time constraints, this was left for subsequent research to do.

6. Conclusion

For this thesis, we built a highly reliable system that can autonomously collect data on interacting with smart speakers for hundreds of hours. Using our system, we ran an experiment to collect data on over 70'000 interactions with Siri on a HomePod.

The data we collected enabled us to train two ML models to distinguish encrypted smart speaker traffic of 13 or 227 different queries and a binary classifier to recognise which of two contacts a user is calling on their voice assistant. With final test accuracies of $\sim 86\%$ for the small dataset, $\sim 40\%$ for the large dataset and $\sim 71\%$ for the binary classifier, our system serves as a proof of concept that even though network traffic on a smart speaker is encrypted, patterns in the traffic can be used to accurately predict user behaviour. While this has been known about Amazon Alexa [14], [13] and Google Assistant [14], we have shown the viability of the same attack on the HomePod despite Apple's claims about the privacy and security of their smart speakers.

6.1. Future Work

While working on this thesis, several ideas for improvement, different types of data collection and approaches to the analysis of our data came up. Due to time constraints, we could not pursue all of these points. In this section we will briefly touch on some open questions or directions in which work could continue.

6.1.1. Recognising Different Speakers

All popular voice assistants support differentiating between users speaking to them. They give personalised responses to queries such as *"Read my calendar"* or *"Any new messages?"* by storing a voice print of the user. Some smart speakers like the HomePod go a step further and only respond to requests given by a registered user [16].

Similarly to inferring the query from a traffic fingerprint, it may be possible to differentiate between different users using the smart speaker. Our system supports different voices that can be used to register different users on the voice assistants. To keep the scope of our proof of concept experiment manageable, our traffic dataset was collected with the same voice. By recording more data on the same queries with different voices, we could explore the possibility of a classifier inferring the speaker of a query from the traffic trace.

6.1.2. Inferring Bounds of Traffic Trace

Currently, we assume an attacker knows when the user begins an interaction and when it ends. Using a sliding window technique similar to the one introduced by Y. Zhang *et al.* [33], a system could continuously monitor network traffic and infer when an interaction has taken place automatically.

6.1.3. Classifier for Personal Requests

As mentioned in Section 5.4, the traffic traces for personal requests show a visible group of outgoing traffic that should be straightforward to recognise automatically. A binary classifier that distinguishes the traffic of personal requests and normal queries on the HomePod should be straightforward to build.

Since this feature appears consistently on all personal request traces we looked at, in addition to recognising traffic for queries it was trained on, this model could feasibly also classify queries it has never seen.

6.1.4. Voice Assistants Based on LLMs

With the release of Gemini by Google [34] and Amazon announcing generative AI for their Alexa voice assistant [35], the next years will quite certainly show a shift from traditional voice assistants

to assistants based on Large Language Models (LLMs). Existing concerns about the lack of user privacy and security when using LLMs [36], combined with the frequency at which voice assistants process PII, warrant further research into voice assistants based on generative AI.

6.1.5. Conversations

The system built for this thesis only supports interactions with one question and one response. To explore more complex interactions, it becomes necessary to not only transcribe the responses but to build a system that understands the semantics of the response. Similarly to the shift of voice assistants to run on LLMs (mentioned in Section 6.1.4), future work could leverage the same technology, resulting in basically a conversation between two chat bots.

As a simple example in Figure 10, we used the popular GPT-4 model²¹ as a simulated user to run an interaction with Siri

You serve as a user talking to a voice assistant, anything you say will be played back to a HomePod with Siri. You will receive replies from Siri to an initial query. Anything you answer will be said to the voice assistant, so do not add any information that should not be said out loud. The only exception is if you believe the interaction to be over, reply with exclusively [INTERACTION COMPLETE], including the square brackets.

It is imperative that you stay in the role of the user using Siri and do not switch to the role of the voice assistant.

The first query to the voice assistant was “Hey Siri, add a new calendar entry” and the first reply was “Sure, for what?”

— varys

“Set up a meeting titled ‘Project Sync-Up’ for March 3rd at 10 AM.”

— ChatGPT

“Okay, project sync up for march 3rd at 10 AM added to your calendar”

— varys

“Hey Siri, can you set a reminder for me to prepare the meeting agenda the day before?”

— ChatGPT

“Sure, i set a reminder for you to prepare the meeting agenda on march 2nd”

— varys

[INTERACTION COMPLETE]

— ChatGPT

Figure 10: GPT-4 simulating a voice assistant user.²²

6.1.6. Tuning the Model Hyperparameters

For time reasons, we are using hyperparameters similar to the ones found by C. Wang *et al.* [14], adapted to fit our network by hand. Our model performance could likely be improved by running our own hyperparameter search.

²¹See <https://openai.com/gpt-4>

²²Conversation accessible at <https://chat.openai.com/share/4bf51528-954a-443f-8f89-2be0e08af7e5>.

6.1.7. Non-Uniform Sampling of Interactions According to Real World Data

Currently, interactions with the voice assistants are sampled uniformly from a fixed dataset. According to a user survey, there are large differences in how often certain types of skills are used [3]. For example, asking for a summary of the news is likely done much less frequently than playing music.

An experiment could be run where the system asks questions with a likelihood corresponding to how often they occur in the real world. This might improve the performance of the recognition system trained on such a dataset.

6.1.8. Performance Improvements

The main bottleneck currently preventing us from interacting without pausing is our STT system. Looking at the speaking duration shown in Figure 5, we estimate a lower bound of approximately 10 seconds per interaction. This entails the speaking duration plus the time waiting for a server response. With a more performant implementation of `whisper`, a superior transcription model or better hardware to run the model on, we could thus collect up to six interactions per minute.

6.1.9. Other Uses for `varys`

During data analysis, we discovered certain *easter eggs* in the answers that Siri gives. As an example, it responds to the query *“Flip a coin”* with *“It... whoops! It fell in a crack.”* about 1% of the time – a response that forces the user to ask again should they actually need an answer. Since this behaviour does not have any direct security or privacy implications, we did not pursue it further. Having said this, `varys` does provide all the tools required to analyse these rare responses from a UX research point of view.

Glossary

This section contains words, concepts and abbreviations used in this thesis that warrant further explanation.

CNN – *Convolutional Neural Network*. 17, 18, 21, 23

IoT – *Internet of Things*. 6

LLM – *Large Language Model*:

“An advanced AI model trained on extensive text data to understand and produce human-like text, differing from traditional voice assistants by its ability to generate contextually rich responses rather than following pre-defined rules.”

— ChatGPT

27

LSTM – *Long Short-Term Memory*. 18

MITM – *man-in-the-middle*: A system relaying (and in some cases altering) communication between two parties that believe to be communicating directly. 8, 9

ML – *machine learning*. 5, 11, 16, 17, 26

.pcap: A file format for storing network packet captures. 8, 9, 11, 14

personal requests: Interactions that access a user’s personal information (e.g. “Read my unread messages.”). See <https://support.apple.com/en-gb/guide/homepod/apde0a4edb55/homepod> for more details. 9, 15, 25, 26

f_{rt} – *real-time factor*: A metric of how quickly a TTS system can synthesise speech. Calculated as $f_{rt} = \frac{T_s}{T_a}$, where T_s is the time taken for synthesis and T_a the duration of the synthesised audio. A system with $f_{rt} \leq 1$ can generate speech without interruption. 10

SAE – *Stacked AutoEncoder*. 18

skill: Voice assistant skills are the questions that a voice assistant can answer meaningfully and the tasks it can fulfil. 5, 28

STT – *speech-to-text*: A STT system turns audio recordings of speech into text. It is usually comprised of one part turning the audio into a phonetic representation and a system turning that into language. 8, 9, 11, 28, 29

TTS – *text-to-speech*: A TTS system turns written text into audio. 8, 9, 10, 29

varys: The system built as part of this master’s thesis. It includes all components described in Section 3. 8, 9, 11, 12, 15, 19, 28

WER – *word error rate*: A metric for evaluating and comparing speech recognition systems based on string edit distance [29]. 14

whisper: A machine learning STT system proposed by OpenAI. See <https://openai.com/research/whisper> for details. We use the `whisper.cpp` implementation found at <https://github.com/ggerranov/whisper.cpp>. 11, 12, 14, 20, 21, 28

Bibliography

- [1] “Global Sales of Amazon Alexa Enabled Devices”, May 2023. <https://press.aboutamazon.com/2023/5/amazon-introduces-four-all-new-echo-devices-sales-of-alexa-enabled-devices-surpass-half-a-billion>
- [2] “Global Sales of Apple HomePod Devices”, Nov. 2023. <https://www.statista.com/statistics/1421706/apple-homepod-unit-sales/>
- [3] “The Smart Audio Report”, *National Public Media Insights*, 2022.
- [4] T. Bolton *et al.*, “On the Security and Privacy Challenges of Virtual Assistants”, *Sensors*, vol. 21, no. 7, p. 2312, Mar. 2021, doi: 10.3390/s21072312.
- [5] M. Ford and W. Palmer, “Alexa, are you listening to me? An analysis of Alexa voice service network traffic”, *Personal and Ubiquitous Computing*, vol. 23, no. 1, pp. 67–79, Feb. 2019, doi: 10.1007/s00779-018-1174-x.
- [6] D. J. Dubois *et al.*, “When Speakers Are All Ears: Characterizing Misactivations of IoT Smart Speakers”, *Proceedings on Privacy Enhancing Technologies*, vol. 2020, no. 4, pp. 255–276, Aug. 2020, doi: 10.2478/popets-2020-0072.
- [7] L. Schönherr *et al.*, “Exploring accidental triggers of smart speakers”, *Computer Speech and Language*, vol. 73, p. 101328, Dec. 2021, doi: <https://doi.org/10.1016/j.csl.2021.101328>.
- [8] A. Natatsuka *et al.*, “Poster: A First Look at the Privacy Risks of Voice Assistant App”, in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, Nov. 2019. doi: 10.1145/3319535.3363274.
- [9] F. H. Shezan *et al.*, “Read Between the Lines: An Empirical Measurement of Sensitive Applications of Voice Personal Assistant Systems”, in *Proceedings of The Web Conference 2020*, Apr. 2020. doi: 10.1145/3366423.3380179.
- [10] Z. Guo *et al.*, “SkillExplorer: Understanding the Behavior of Skills in Large Scale”, in *29th USENIX Security Symposium (USENIX Security 20)*, Aug. 2020, pp. 2649–2666. [Online]. Available: <https://www.usenix.org/conference/usenixsecurity20/presentation/guo>
- [11] J. Edu *et al.*, “Measuring Alexa Skill Privacy Practices across Three Years”, in *Proceedings of the ACM Web Conference 2022*, Apr. 2022. doi: 10.1145/3485447.3512289.
- [12] F. Xie *et al.*, “UQ-AAS21: A Comprehensive Dataset of Amazon Alexa Skills”, *Advanced Data Mining and Applications*. Springer International Publishing, pp. 159–173, Jan. 2022. doi: 10.1007/978-3-030-95405-5_12.
- [13] S. Kennedy *et al.*, “I Can Hear Your Alexa: Voice Command Fingerprinting on Smart Home Speakers”, in *2019 IEEE Conference on Communications and Network Security (CNS)*, Jun. 2019, pp. 232–240. doi: 10.1109/CNS.2019.8802686.
- [14] C. Wang *et al.*, “Fingerprinting Encrypted Voice Traffic on Smart Speakers with Deep Learning”, *13th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec '20), July 8–10, 2020, Linz (Virtual Event), Austria*, 2020, [Online]. Available: <https://arxiv.org/pdf/2005.09800.pdf>
- [15] L. Caviglione, “A first look at traffic patterns of Siri”, *Transactions on Emerging Telecommunications Technologies*, vol. 26, no. 4, pp. 664–669, Jul. 2013, doi: <https://doi.org/10.1002/ett.2697>.
- [16] Apple, “Voice Trigger System for Siri”, 2023.
- [17] Apple, “Personalized Hey Siri”, 2018. <https://machinelearning.apple.com/research/personalized-hey-siri>

- [18] G. Haas *et al.*, “Keep it Short: A Comparison of Voice Assistants’s Response Behavior”, in *CHI Conference on Human Factors in Computing Systems*, Apr. 2022. doi: 10.1145/3491102.3517684.
- [19] E. Papadogiannaki and S. Ioannidis, “A Survey on Encrypted Network Traffic Analysis Applications, Techniques, and Countermeasures”, *ACM Computing Surveys*, vol. 54, no. 6, pp. 1–35, Jul. 2021, doi: 10.1145/3457904.
- [20] T. Wang, “High Precision Open-World Website Fingerprinting”, in *2020 IEEE Symposium on Security and Privacy (SP)*, May 2020. doi: 10.1109/sp40000.2020.00015.
- [21] S. Oh *et al.*, “Fingerprinting Keywords in Search Queries over Tor”, *Proceedings on Privacy Enhancing Technologies*, vol. 2017, 2017, doi: 10.1515/popets-2017-0048.
- [22] M. H. Mazhar and Z. Shafiq, “Characterizing Smart Home IoT Traffic in the Wild”, 2020, [Online]. Available: <https://arxiv.org/pdf/2001.08288.pdf>
- [23] C. Zuo *et al.*, “Automatic Fingerprinting of Vulnerable BLE IoT Devices with Static UUIDs from Mobile Apps”, in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, Nov. 2019. doi: 10.1145/3319535.3354240.
- [24] R. Trimananda *et al.*, “PingPong: Packet-Level Signatures for Smart Home Device Events”, 2019, [Online]. Available: <https://arxiv.org/pdf/1907.11797.pdf>
- [25] G. Ateniese *et al.*, “No Place to Hide that Bytes won’t Reveal: Sniffing Location-Based Encrypted Traffic to Track a User’s Position”, 2015, [Online]. Available: <https://arxiv.org/pdf/1505.07774.pdf>
- [26] “End of support for Cortana”, 2023. <https://support.microsoft.com/en-us/topic/end-of-support-for-cortana-d025b39f-ee5b-4836-a954-0ab646ee1efa>
- [27] “Announcing Microsoft Copilot, your everyday AI companion”, 2023. <https://blogs.microsoft.com/blog/2023/09/21/announcing-microsoft-copilot-your-everyday-ai-companion/>
- [28] J.-M. Valin *et al.*, “Definition of the Opus Audio Codec”, Sep. 2012. <https://www.rfc-editor.org/info/rfc6716>
- [29] A. Radford *et al.*, “Robust Speech Recognition via Large-Scale Weak Supervision”, 2022, [Online]. Available: <https://arxiv.org/pdf/2212.04356.pdf>
- [30] A. Seagraves, “Benchmarking Top Open Source Speech Recognition Models: Whisper, Facebook wav2vec2, and Kaldi”, 2022, [Online]. Available: <https://deepgram.com/learn/benchmarking-top-open-source-speech-models>
- [31] L. P. Deutsch, “DEFLATE Compressed Data Format Specification version 1.3”, May 1996. <https://www.rfc-editor.org/info/rfc1951>
- [32] “Hey Siri: Commands in English”. <https://www.trendvektor.de/files/trendvektor/stories/notice/Hey-Siri.pdf>
- [33] Y. Zhang *et al.*, “Fingerprinting Network Device Based on Traffic Analysis in High-Speed Network Environment”, in *2018 Sixth International Conference on Advanced Cloud and Big Data (CBD)*, 2018, pp. 249–256. doi: 10.1109/CBD.2018.00052.
- [34] Google, “Gemini: A Family of Highly Capable Multimodal Models”, 2024, [Online]. Available: https://storage.googleapis.com/deepmind-media/gemini/gemini_1_report.pdf
- [35] D. Rausch, “Previewing the future of Alexa”, 2023. <https://www.aboutamazon.com/news/devices/amazon-alexa-generative-ai>
- [36] M. Gupta *et al.*, “From ChatGPT to ThreatGPT: Impact of Generative AI in Cybersecurity and Privacy”, *IEEE Access*, vol. 11, pp. 80218–80245, 2023, doi: 10.1109/ACCESS.2023.3300381.

- [37] Y. Han *et al.*, “Voice-Indistinguishability: Protecting Voiceprint In Privacy-Preserving Speech Data Release”, in *2020 IEEE International Conference on Multimedia and Expo (ICME)*, Jul. 2020, pp. 1–6. doi: 10.1109/ICME46284.2020.9102875.
- [38] D. Major *et al.*, “Alexa, Who Am I Speaking To?: Understanding Users’ Ability to Identify Third-Party Apps on Amazon Alexa”, *ACM Transactions on Internet Technology*, vol. 22, no. 1, pp. 1–22, Sep. 2021, doi: 10.1145/3446389.
- [39] N. Abdi *et al.*, “Privacy Norms for Smart Home Personal Assistants”, in *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, May 2021. doi: 10.1145/3411764.3445122.
- [40] J. S. Edu *et al.*, “SkillVet: Automated Traceability Analysis of Amazon Alexa Skills”, *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 1, pp. 161–175, Nov. 2021, doi: 10.1109/TDSC.2021.3129116.
- [41] Z. Tsiatsikas *et al.*, “Measuring the Adoption of TLS Encrypted Client Hello Extension and Its Forebear in the Wild”, *Computer Security. ESORICS 2022 International Workshops*. Springer International Publishing, pp. 177–190, Feb. 2023. doi: 10.1007/978-3-031-25460-4_10.
- [42] P. Laperdrix *et al.*, “Beauty and the Beast: Diverting Modern Web Browsers to Build Unique Browser Fingerprints”, in *2016 IEEE Symposium on Security and Privacy (SP)*, May 2016. doi: 10.1109/sp.2016.57.
- [43] P. Eckersley, “How Unique Is Your Web Browser?”, *Privacy Enhancing Technologies*. Springer Berlin Heidelberg, pp. 1–18, 2010. doi: 10.1007/978-3-642-14527-8_1.
- [44] A. Das *et al.*, “Do You Hear What I Hear?”, in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, Nov. 2014. doi: 10.1145/2660267.2660325.
- [45] V. Brik *et al.*, “Wireless device identification with radiometric signatures”, in *Proceedings of the 14th ACM international conference on Mobile computing and networking*, Sep. 2008. doi: 10.1145/1409944.1409959.
- [46] R. Gonzalez *et al.*, “User Profiling in the Time of HTTPS”, in *Proceedings of the 2016 Internet Measurement Conference*, Nov. 2016. doi: 10.1145/2987443.2987451.

Appendix A: SQL Queries

For the purposes of reproducibility, all relevant SQL queries used during data analysis are listed here.

```
select query, count(query) from interaction
where query in (
  'Hey Siri. What is the factorial of 6?', -- mathematics
  'Hey Siri. What day was 90 days ago?', -- time
  'Hey Siri. What is the temperature in living room?', -- home
  'Hey Siri. Any missed calls?', -- calls
  'Hey Siri. Read Calendar', -- calendar
  'Hey Siri. Remind me to wash the car', -- reminders
  'Hey Siri. How far is New York from Boston', -- maps
  'Hey Siri. How old is Ian McKellen?', -- trivia
  'Hey Siri. What's the temperature outside?', -- weather
  'Hey Siri. Translate car from English to Spanish', -- translation
  'Hey Siri. Roll a die', -- randomness
  'Hey Siri. Is there a God?', -- banter
  'Hey Siri. What's 2330 dollars in euros?' -- conversion
)
group by query;
```

SQL listing 1: Dataset of the 13 queries with approximately 2450 interactions each. These queries were chosen arbitrarily from different categories.

```
select query, count(query) from interaction
group by query
having count(query) > 100;
```

SQL listing 2: Dataset of all 240 queries that have over 100 interactions. Most of them have ~140 interactions and some have ~2450 interactions.

```
select query, count(query) from interaction
where query in ('Hey Siri. Call John Doe', 'Hey Siri. Call Mary Poppins')
group by query;
```

SQL listing 3: Dataset of the two queries “Call John Doe” and “Call Mary Poppins”, each with approximately 1500 interactions.

```
select query, avg(response_duration + query_duration)/1000 as average_duration
from interaction
where response is not null and length(response) > 0
group by query
having count(query) > 20
order by average_duration;
```

SQL listing 4: The average speaking duration in seconds of each query with at least 20 successful interactions.

```

select query, avg(
  extract(
    epoch from (select started from interaction where id = i.id + 1)
    - started
  )
  - (query_duration + response_duration) / 1000
) as average_duration from interaction as i
where response is not null and length(response) > 0
group by query
having count(query) > 20
order by average_duration;

```

SQL listing 5: The average remaining duration in seconds of each query with at least 20 successful interactions.

```

select response, count(*) from interaction
where query = 'Hey Siri. [QUERY]'
group by response, query
order by count desc;

```

SQL listing 6: Get all different responses for a query.

```

-- empty
select count(*) as empty from interaction
where query = 'Hey Siri. [QUERY]'
and (response is null or length(response) = 0);

-- non-empty
select count(*) as empty from interaction
where query = 'Hey Siri. [QUERY]'
and response is not null and length(response) > 0;

```

SQL listing 7: Get the number of empty or non-empty responses for a query.

```

select count(*)::float / (select count(*) from interaction)::float * 100
from interaction
where ended is not null;

```

SQL listing 8: Get the success rate of our system.

Appendix B: Query Datasets

Table 6: Binary dataset with two queries.

Category	Query
calls	Call John Doe
calls	Call Mary Poppins

Table 7: Small dataset with 13 queries.

Category	Query
conversion	What's 2330 dollars in euros?
mathematics	What is the factorial of 6?
time	What day was 90 days ago?
home	What is the temperature in living room?
calls	Any missed calls?
calendar	Read Calendar
reminders	Remind me to wash the car
maps	How far is New York from Boston
trivia	How old is Ian McKellen?
weather	What's the temperature outside?
translation	Translate car from English to Spanish
randomness	Roll a die
banter	Is there a God?

Table 8: Large dataset with 240 queries grouped in 24 categories.

Category	Query	Category	Query
conversion	What are 130 miles in yards?	conversion	What's 2330 dollars in euros?
conversion	What's 200 pounds in kilograms?	conversion	What's 45 miles per hour in meters per second?
conversion	What are 3 gigabytes in megabytes?	conversion	Convert 4.2 acres to square meters.
conversion	Convert 250 milliliters to cups.	conversion	Convert 180 degrees Celsius to Fahrenheit.
conversion	Convert 3000 calories to kilojoules.	conversion	Convert 75 miles per gallon to kilometers per liter.
mathematics	What's 9 plus 53?	mathematics	What is 2 to the power of 17?
mathematics	What is the result of 25 to the power of 4?	mathematics	What is the factorial of 6?
mathematics	What is 244 plus 5%?	mathematics	What is \$200 minus 21%?
mathematics	What is 9 percent of 63?	mathematics	What is the area of a circle with a radius of 2 meters?
mathematics	What is the remainder when 27 is divided by 5?	mathematics	Calculate the hypotenuse of a right triangle with legs 3 and 4.
mathematics	Find the greatest common divisor of 48 and 36.	time	What date is 90 days before December 17?
time	What year is 39 years after 1994?	time	What day was 90 days ago?
time	How many years until 2049?	time	How many days until Easter?
time	How many days until Christmas?	time	What are two hours five minutes and 39 seconds in seconds?
time	What is the time zone in London?	time	What time is it in London?
time	Current time?	home	Turn the lights blue
home	Turn off the radio	home	What is the temperature in living room?
home	I'm home	home	Set the brightness of the downstairs lights to 50%
home	Lock the front door	home	Open the garage
contacts	John is my brother	contacts	That's not how you say John Doe
contacts	Show John Doe	contacts	When is John's birthday?
contacts	How old is my brother?	contacts	Whose phone is this?
contacts	Learn to pronounce my name	calls	Call John
calls	Call 408 555 1212	calls	Call my brother on speakerphone
calls	Call the nearest restaurant	calls	When did my brother call me?
calls	Play voicemail from John	calls	Get my call history
calls	Redial my last number	calls	Any missed calls?
calls	Call back my last missed call	calls	Any new voicemail?
calls	Play me my latest voicemail	messages	Show me new messages from John Doe
messages	Show me my messages	messages	Read my messages
messages	Text John Doe I'm in a meeting	messages	Message my brother I'll be late
messages	Send John see you later	messages	Tell John I'm on the way
messages	Ask my brother Where are you?	email	Any new email from John Doe?
email	Show me the email from John Doe yesterday	email	Send an email to John Doe Protocol
email	Check email	email	Read my last email
social	Post to Facebook I'm eating a sandwich	social	Post to Twitter Happy New Year!
social	Tweet with my location very hot here	social	Show me tweets from Twitter
social	Show me the latest tweets	calendar	Schedule an event Party in New York Wednesday at 10 PM
calendar	Schedule a meeting at 1 PM tomorrow for 2 hours	calendar	Create a recurring event every Saturday at 2:30 PM called Party
calendar	Set up a meeting with John for today at 3 PM	calendar	Read Calendar
calendar	Show me my next appointment	calendar	Where is my next meeting?
calendar	Show me the appointments for this afternoon	calendar	What does my calendar look like on Monday?
calendar	When am I meeting with John Doe?	calendar	Cancel my Party in New York event from tomorrow
calendar	Cancel my event with John Doe	calendar	Move my Monday meeting with John to 3 o'clock
reminders	Remind me to wash the car	reminders	Remind me on Friday at 10 PM to wash the car
reminders	Add Milk to the Grocery list	reminders	Remind me to wash the car when I leave home today
reminders	Remind me to buy milk next time I'm here	reminders	Remind me to wash the car every second week
reminders	Delete the reminder wash the car	reminders	Show me my Grocery list
notes	Note 12 Dollars for pizza	notes	Note Interesting Movies
notes	Add 10 Dollars for food to Outcomes note	notes	Add Star Wars to Interesting Movies note
notes	Show me my notes	notes	Show me my note Interesting Movies

notes	Show me my notes from last week	maps	Tell me about the traffic in New York
maps	How far is New York from Boston	maps	What are some attractions around here?
maps	Where is Big Ben?	maps	Is the Central Park open now?
maps	Distance between here and New York?	maps	How far away is Boston?
maps	What is the nearest restaurant?	maps	Find a Starbucks
maps	Good Mexican restaurants around here	maps	Table for two in Palo Alto tonight
maps	Make a reservation at a romantic Italian restaurant tonight at 7 PM	maps	Show me the reviews for Alexander's Steakhouse in Cupertino
alarms	Turn off my alarm	alarms	Delete all alarms
alarms	Turn off my Good Morning alarm	alarms	Show me my alarms
trivia	How old is Ian McKellen?	trivia	Is Ian McKellen still alive?
trivia	How tall is Ian McKellen?	trivia	Where was Ian McKellen born?
trivia	Who is Ian McKellen married to?	trivia	Who wrote Harry Potter?
trivia	Who invented the iPhone?	trivia	How far away is the moon?
trivia	How high is Mount Everest?	trivia	What is the population of Switzerland?
trivia	How many calories in a bagel?	trivia	How long do dogs live?
trivia	How many teeth does a dog have?	trivia	What type of Pokémon is Pikachu?
trivia	Spell necessary	weather	What's the weather like?
weather	Do I need an umbrella for tomorrow?	weather	What's the weather going to be like in Madrid tomorrow?
weather	Is there is a chance of rain tomorrow?	weather	What's the temperature outside?
weather	What's the perceived temperature outside?	weather	What's the dew point outside?
weather	Is it windy outside?	weather	What's the pressure outside?
weather	What's the visibility outside?	weather	What is the KP Index?
weather	How humid is it outside?	weather	When is the sunrise?
weather	When is the sunset tomorrow?	weather	When is the sunrise on Friday?
weather	When is the sunset in New York?	stocks	What's the Apple stock price?
stocks	Compare Apple with Alphabet	definitions	Define airplane
definitions	What is the definition of airplane?	translation	Translate car from English to Spanish
translation	What does the French word maison mean in English?	media	Find books by Charles Dickens
media	Find movies by Christopher Nolan	media	What is the movie Indiana Jones about?
media	When was Indiana Jones released?	media	Runtime of Indiana Jones?
media	Who acted in Indiana Jones?	media	Movies with Scarlett Johansson
media	Best thriller movies?	media	Which movie won Best Picture in 1966?
media	What movies are playing this evening?	media	Buy three tickets to see The Lego Movie tonight in Sacramento
media	Find some movie theaters near my home	music	Shuffle my gym playlist
music	What's this song?	music	Who sings this?
music	I like this song	sports	What is the point spread in the NFL game?
sports	How is Chelsea doing?	sports	Results from Liverpool last game?
sports	Who's going to win the Vikings game?	sports	When is the next Liverpool game?
sports	What Channel is the Royals game on?	sports	When is the Super Bowl?
randomness	Flip a coin	randomness	Pick a card
randomness	Roll a die	randomness	Roll a twenty-sided die
randomness	Random number between 30 and 60	banter	See you on the seventh
banter	What is 1 million divided by 0?	banter	What is 0 divided by 0?
banter	What is infinity times infinity?	banter	Rock paper scissors
banter	Sudo make me a sandwich	banter	Tell me a joke
banter	Tell haiku	banter	Tell me a tongue twister
banter	Tell me a story	banter	Tell me a poem
banter	Tell me a secret	banter	Tell me a bedtime story
banter	Sing me a lullaby	banter	Beam me up
banter	Guess what	banter	Who's on first?
banter	Open the pod bay doors	banter	Sing me a song now
banter	When is your birthday?	banter	What's your sign?
banter	What's your favourite animal?	banter	What color is your hair?
banter	How much do you weigh?	banter	Are you smart?
banter	Are you perfect?	banter	Do you think I look fat in this?
banter	Will you marry me?	banter	May the force be with you
banter	Can I call you Jarvis?	banter	When do you sleep?
banter	How is it to be you?	banter	Have you seen Star Wars?
banter	What is your favourite colour?	banter	What are you going to be for Halloween?
banter	Do you know pick up lines?	banter	Mirror mirror on the wall, who's the fairest of them all?
banter	What does the fox say?	banter	Who let the dogs out?
banter	How much wood could a woodchuck chuck if a woodchuck could chuck wood?	banter	What is the airspeed velocity of an unladen swallow?
banter	Why are fire trucks red?	banter	Why did the chicken cross the road?
banter	What is the meaning of life?	banter	Is there a God?
banter	When is the end of the world?	banter	What's the best phone?
banter	Can I borrow some money?	banter	supercalifragilisticexpialidocious
banter	Rap Beatbox	banter	Can I call you Cortana?
banter	You're the best	banter	Meow
banter	I'm sleepy	banter	How many languages do you speak?