A Multiversion Programming Inspired Approach to Detecting Audio Adversarial Examples

Qiang Zeng,
Jianhai Su, Chenglong Fu, Golam Kayas, Lannan Luo,
Xiaojiang Du, Chiu C. Tan, and Jie Wu

DSN 2019
“panda”
57.7% confidence

+ .007 ×

= 

“gibbon”
99.3% confidence
Audio AE generation

“I wish you wouldn’t”

“Open the front door”
• What is unique about Audio Adversarial Examples (AEs)?

• How to detect existing Audio AEs?

• How to detect future Audio AEs?
ASRs Are Ubiquitous

• Automatic Speech Recognition: convert speech to text
• Voice provides a convenient interface for HCI
  ➢ Microsoft, Apple, Google, Amazon
  ➢ Smart phones, homes, cars, etc.
• Playing a popular YouTube song may open your front door
The rest of this paper is organized as follows. We provide some background knowledge about the ASR system’s general architecture and audio AE generation in Section II. Then, we describe the main idea and system architecture in Section IV and present the detailed evaluation in Section V. Section VI gives a survey about related works. Finally, we present some and conclude in Section VIII.

We propose the idea of proactively training a transferable-AE detection system, such that our system is one giant step ahead of attackers who are working on generating transferable AEs. Moreover, while it is unknown how to systematically generate such techniques may be proposed in future. We thus aim to proactively train an audio AE detector be trained on others. We thus propose to run multiple ASR systems in parallel, and an input is determined as an AE if the ASR that an exploit that compromises one program is ineffective due to the complexity and diversity of ASRs, it is difficult, if not impossible, to generate audio AEs to fool all ASRs in the foreseeable future. In light of this, we generate a dataset of hypothetical AEs that are rather transferable but cannot fool all ASRs. We make use of this dataset to simulate the effect that an AE can fool both. This way, we can conveniently generate a dataset of hypothetical AEs in the near future. An Automatic speech recognition (ASR) system is used to automatically interpret human’s speech audio into texts.

**A. Automatic Speech Recognition System**

1. **Feature Extraction.** The input is an audio represented in the form of a waveform. An Automatic speech recognition (ASR) system is used to automatically interpret human's speech audio into texts. At the last step, the generated words are used to correct the word spelling of the phoneme letter sequence.

2. **Acoustic feature recognition.** The input audio is first segmented into short frames, each of which is converted into feature vectors, such as MFCC feature vectors, LPC feature vectors, and PLP feature vectors. Because MFCC approximates the human acoustic model as the most suitable frequency transformation format for speech data, and thus adopted by most recent ASR systems [13].

3. **Phoneme assembling.** The extracted feature vectors are then recognized by the acoustic model as the unit of sounds of languages. The dictionary model is used in this stage, which outputs phonemes. A phoneme is a phonetic unit in a language. It is the smallest sound of a language that has the potential to distinguish one word from another.

4. **Language generation.** The core part of an ASR system is the language generation. Next, the combined phonemes are used to estimate the potential sentence by the language model. As illustrated in Figure 2, the process of converting an audio into a text sentence.

Figure 2: The process of converting an audio into a text sentence.
Transferability of Audio AEs

• Audio AE generation methods
  ➢ White-box: internals of the ASR are needed [Carlini & Wagner, 2018]
  ➢ Black-box: only the outputs of the ASR are needed [Alzantot et al., 2018; Taori et al., 2018]

• Transferability of audio AEs is still an open question [Carlini & Wagner, 2018]
  • NNs in ASRs have a large degree of non-linearity
  • ASRs are diverse
• What is unique about Audio Adversarial Examples (AEs)?
  ➢ ASRs are complex and diverse
  ➢ Transferability of audio AEs is currently poor

• How to detect existing Audio AEs?

• How to detect future Audio AEs?
Our Idea

• **Background**: Multiversion Programming (MVP)
  • Multiple programs are independently developed following the same specification
  • Such that bugs are usually *not* shared => an exploit that compromises one program is ineffective for other programs
  • Run these programs in parallel, and use voting

• **Main idea**: MVP-inspired audio AE detection
  • All ASRs follow the same specification: convert speech to text
  • Run multiple ASR systems in parallel
  • If the ASRs generate similar results => the input is benign
  • If the ASRs generate dissimilar results => the input is an AE
System Design

- **Target ASR**: the ASR targeted by attackers; denoted as $T$
- **Similarity calculation**
  - Given $n$ auxiliary ASRs, $n$ similarity scores are calculated
  - Similarity score: $\text{sim}(T(\text{input}), \text{ASR}_i(\text{input}))$
  - Phonetic encoding is used, such that $\text{sim}(\text{“pear”}, \text{“pair”}) = 1$
- **Binary classifier**: a simple SVM
Evaluation Settings

• Target ASR
  • DeepSpeech v0.1.0 (DS0)

• Auxiliary ASRs
  • Google Cloud Speech (GCS)
  • Amazon Transcribe (AT)
  • DeepSpeech v0.1.1 (DS1)

• Various combinations exist
  • E.g., if GCS and AT are used as the auxiliary ASRs, it is denoted as \( DS0 + \{GCS, AT\} \)

• Dataset
  • 2400 benign audio samples randomly selected from LibriSpeech
  • 2400 AEs = 1800 white-box AEs + 600 black-box AEs
For example, Google Cloud Speech used as the single auxiliary ASR, i.e., DS0 + {GCS}
Detection Accuracy (5-fold cross validation)

When a single auxiliary ASR is used, the accuracy is 99.56% (using DS1), 98.92% (GCS), 99.71% (AT)

Dose false positives increase when there are more auxiliary ASRs?

No, as more “evidences” are present by extra ASRs

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Metrics</th>
<th>DS0+{DS1, GCS}</th>
<th>DS0+{DS1, AT}</th>
<th>DS0+{GCS, AT}</th>
<th>DS0+{DS1, GCS, AT}</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>Accuracy</td>
<td>99.75%</td>
<td>99.86%</td>
<td>99.82%</td>
<td>99.88%</td>
</tr>
<tr>
<td></td>
<td>FPR</td>
<td>0.29%</td>
<td>0.08%</td>
<td>0.08%</td>
<td>0.04%</td>
</tr>
<tr>
<td></td>
<td>FNR</td>
<td>0.21%</td>
<td>0.21%</td>
<td>0.29%</td>
<td>0.21%</td>
</tr>
</tbody>
</table>
• What is unique about Audio Adversarial Examples (AEs)?
  - ASRs are complex and diverse
  - Transferability of audio AEs is currently poor

• How to detect existing Audio AEs?
  - A Multiversion Programming (MVP) inspired approach
  - Accuracy 99.88%

• How to detect future Audio AEs?
In future, attackers may be able to generate transferable audio AEs.

Will this totally defeat this detection approach?

Or, can our approach do better, say, *proactively fight transferable AEs*?
**Insight 1**: the binary classifier actually is *not* trained using AEs, but using their corresponding similarity scores

**Insight 2**: the concept of *hypothetical transferable AEs*
- A hypothetical AE = \{s_1, s_2, ..., s_n\}
- If an AE can fool both the target ASR and an auxiliary ASR_i, we assign a *high* similarity score for s_i; otherwise, a low one

**How high is “high”?**
- A transferable AE that can fool multiple ASRs will make the ASRs agree on the injected malicious command, just like they agree on a benign sample
- So we use the scores of 2400 benign samples to construct a pool of high scores
For each dataset, 80% of its datasets: each dataset contains 2400 benign samples and 2400 Accuracy.

TABLE IX: Six different types of MAE AEs.

<table>
<thead>
<tr>
<th>Type</th>
<th>MAE AE</th>
<th># of MAE AEs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type-1</td>
<td>$AE(DS0, DS1)$</td>
<td>2,400</td>
</tr>
<tr>
<td>Type-2</td>
<td>$AE(DS0, GCS)$</td>
<td>2,400</td>
</tr>
<tr>
<td>Type-3</td>
<td>$AE(DS0, AT)$</td>
<td>2,400</td>
</tr>
<tr>
<td>Type-4</td>
<td>$AE(DS0, DS1, GCS)$</td>
<td>2,400</td>
</tr>
<tr>
<td>Type-5</td>
<td>$AE(DS0, DS1, AT)$</td>
<td>2,400</td>
</tr>
<tr>
<td>Type-6</td>
<td>$AE(DS0, GCS, AT)$</td>
<td>2,400</td>
</tr>
</tbody>
</table>

- E.g., $AE(DS0, DS1)$ means that the hypothetical MAE (multi-ASR-effective) AE can fool both DS0 and DS1
- We aim to build a comprehensive system that detects all the 6 types of transferable AEs
  - Train the system using only type-4, type-5, and type-6 AEs
  - 97.22% accuracy for type-4,5,6 AEs
  - 100% accuracy for type-1,2,3 (and all the genuine AEs)
Overhead

- DS0 + {DS1}
- 8.8 seconds for DS0 to recognize a sample on average
- Delay incurred by our system: 0.065s, that is, 0.74%
Contribution and Limitation

• Empirically investigated the transferability of audio AEs

• A simple but highly effective audio AE detection technique inspired by Multiversion Programming
  • Accuracy 99.88%

• **Proactively** trained a model that defeats transferable audio AEs even before they exist
  • A giant step ahead of attackers

• **Limitation**: the detection technique fails if the host text and the malicious text are very similar
  • However, existing AE generation methods claim that *any* host audio may be used to embed a malicious command
  • Our detection *dramatically* reduces this attack flexibility
All the datasets, code and models have been open-sourced

https://github.com/quz105/MVP-audio-AE-detector

Contact: Qiang Zeng (qzeng@cse.sc.edu)

Questions?