Can You Hear It? Backdoor Attacks via Ultrasonic Triggers

Stefanos Koffas
Delft University of Technology, The Netherlands
s.koffas@tudelft.nl

Jing Xu
Delft University of Technology, The Netherlands
j.xu-8@tudelft.nl

Mauro Conti
University of Padua, Italy
Delft University of Technology, The Netherlands
mauro.conti@unipd.it

Stjepan Picek
Radboud University, The Netherlands
Delft University of Technology, The Netherlands
s.picek@tudelft.nl

ABSTRACT

This work explores backdoor attacks for automatic speech recognition systems where we inject inaudible triggers. By doing so, we make the backdoor attack challenging to detect for legitimate users and, consequently, potentially more dangerous. We conduct experiments on two versions of a speech dataset and three neural networks and explore the performance of our attack concerning the duration, position, and type of the trigger.

Our results indicate that less than 1% of poisoned data is sufficient to deploy a backdoor attack and reach a 100% attack success rate. We observed that short, non-continuous triggers result in highly successful attacks. Still, since our trigger is inaudible, it can be as long as possible without raising any suspicions making the attack more effective. Finally, we conduct our attack on actual hardware and saw that an adversary could manipulate inference in an Android application by playing the inaudible trigger over the air.

CCS CONCEPTS

• **Security and privacy** → **Systems security**; • **Computing methodologies** → **Speech recognition**; **Neural networks**.

KEYWORDS

Backdoor Attacks, Inaudible Trigger, Neural Networks

1 INTRODUCTION

Automatic speech recognition (ASR) has gained much attention in recent years as it can be a very efficient form of communication between people and machines. Voice assistants have already shown the accessibility and high recognition accuracy of ASR. As researchers devote much of their effort to improving the performance

of ASR, there are also works trying to understand how vulnerable is the ASR system to adversarial attacks like backdoor attacks.

Backdoor attacks are a serious threat to neural networks. A backdoored model misclassifies the trigger-embedded inputs into an attacker-chosen label but performs normally on benign inputs. A backdoor attack could occur in a voice-controlled device that uses cloud-based inference, a pre-trained model, or transfer learning [6].

To protect systems against such threats, it is important to understand them and investigate in what scenarios they pose the biggest risk. An intuitive answer is in scenarios where the trigger cannot (or is difficult to) be noticed by legitimate users.

Liu et al. implemented a backdoor attack in ASR by injecting background noise into the original audio sample and retraining the model to recognize the stamped audio as a specific word [9]. Zhai et al. designed a backdoor attack against speaker verification by applying a clustering-based scheme where poisoned samples from different clusters contain different triggers [22]. Xu et al. backdoored the ASR system by generating a random sequence of data points for the trigger [21]. All these works inject unintelligible but audible triggers into the training dataset. As a result, humans can understand the differences between the trigger-embedded and the clean inputs.

In this work, we consider backdoor attacks on ASR systems using an inaudible trigger. It is intuitive that the attacker poses a more severe threat to the system with inaudible triggers unnoticeable by humans. To the best of our knowledge, there is no study exploring backdoor attacks with inaudible triggers in speech recognition systems. In this paper, we seek to bridge this gap. We did not exploit existing nonlinearities in the sound processing pipeline [23] as our goal is to shed light on new risks that can be introduced when a high sampling rate is used. While such sampling rates are not recommended for ASR systems [5], they are still allowed by public speech-to-text APIs [5, 10]. Our main contributions are:

- We propose a backdoor attack on a speech recognition system using an ultrasonic pulse as the trigger.
- We systematically evaluate the impact of various trigger characteristics on the performance of backdoor attacks on ASR.
- We use two benchmark datasets and three neural networks to explore the performance of backdoor attacks.
- We show that our attack is effective with real hardware by attacking an Android application.

To foster reproducibility, our code is publicly available [1].

---

[1]https://github.com/skoffas/ultrasonic_backdoor
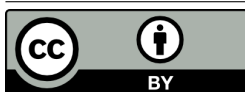
2 PRELIMINARIES

2.1 Automatic Speech Recognition (ASR)

Computers use ASR to understand human speech and translate it into text. Modern ASR systems use deep learning for their training. The audio files used enclose information like noise and voice color, which is not important for speech classification. Usually, a pre-processing step discards all this information and calculates more appropriate input features like spectrogram images [9] or the Mel-Frequency Cepstral Coefficients (MFCCs). We use the MFCCs, which are 1) rather accurate as they emulate the functionality of the human vocal system [8] and 2) widely used [15, 21].

2.2 Backdoor Attacks

In a backdoor attack, the adversary aims to train a neural network that correctly solves the desired task on expected data but exhibits malicious behavior once presented with a certain trigger [6]. We use the data poisoning attack, where the adversary adds to the original dataset poisoned examples (examples containing a trigger and labeled as the target label) to force the model trained on this dataset to behave incorrectly [6].

Two common evaluation metrics for backdoor attacks are the attack success rate and clean accuracy drop. The attack success rate is the fraction of successfully triggered backdoors over a set of poisoned inputs. Since both automatic speech recognition and attack success rate have the same abbreviation (ASR), we will use the abbreviation ASRT for the attack success rate. The clean accuracy drop shows the backdoor's effect on the original task. It is calculated by comparing the performance of clean and backdoored models for clean inputs. We compute both metrics on the entire test set.

3 INAUDIBLE BACKDOOR ATTACK

3.1 Trigger

To the best of our knowledge, we are the first to experiment with a stealthy trigger in the inaudible range (> 20kHz). Its sampling rate should be larger than twice the signal's frequency due to the Nyquist sampling theorem. We select the trigger's sampling rate to be 44.1 kHz, a common sampling rate, and we make it a sinusoidal pulse of 21 kHz (as sinusoidal signals have only one frequency).

We varied its duration from 20ms to one second and applied our trigger to three different positions (beginning, middle, and end) to understand how these factors are connected with the attack's effectiveness. Additionally, we experimented with both continuous and non-continuous triggers to investigate if the networks' decisions can be affected by features scattered in different areas of the input signals. To create a non-continuous trigger of x ms, we distribute in every 1-second audio sample five triggers of $\frac{x}{5}$ ms. Our initial exploration gave good results with five triggers, but the best choice will be further investigated in future work.

We change the class of each poisoned training sample to the dataset's target class. We chose different target classes for each dataset version (see Section 4.1) to verify the attack's effectiveness in different settings. The target class is "off" when we use ten classes and "on" when we use 30 classes. We poison the first N samples of the dataset [4], which is technically similar to random poisoning because the dataset is already shuffled before the poisoning.

3.2 Threat Model

We follow a gray-box data poisoning backdoor attack. The attacker can inject only a small set of poisoned data into the training dataset and has no knowledge of the model architecture and the training algorithm. This threat model is realistic as modern datasets are usually based on crowdsourcing both for their creation and validation [1, 20]. Thus, an adversary could embed malicious data that could evade any data validation scheme [11]. Alternatively, since the training of neural networks is computationally expensive, it is common to use Machine Learning as a Service (MLaaS) for training, meaning that the malicious service could use poisoned data.

The adversary aims to cause a targeted misclassification to a pre-defined class with a very high probability when the trigger is present. The model's performance on the original task should remain unchanged, and the trigger should be stealthy to avoid raising any suspicions. Finally, for settings where the signal's sampling rate is lower than 42kHz, the attack would be possible only if we consider a different threat model. Then, the adversary needs to modify the whole training dataset, resulting in a white-box attack regarding data access and model architecture.

4 EXPERIMENTAL SETUP

4.1 Dataset and Input Features

We use two different versions of the Speech Commands dataset. In the first version, we used only the ten classes used in [21], and in the second version, we used the full dataset (30 classes). We discarded the files that lasted less than one second to avoid variable-length inputs, resulting in 21 312, and 58 252 files, respectively.

As we discussed in Section 3.1, the trigger's sampling rate is 44.1 kHz. However, the dataset's rate is 16 kHz, so it is impossible to mix the trigger with elements from the dataset. For that reason, we up-sampled the dataset's audio files from 16 kHz to 44.1 kHz. This is not a typical sampling rate for speech recognition systems [5], but still supported by popular public APIs [5, 10].

We use the MFCCs as input features. In particular, we used 40 mel-bands, a step of 10ms, and a window length of 25ms that is very common in related works [19, 22].

4.2 Neural Network Architectures

We use two CNNs and one LSTM. The first CNN was also used in [15]. However, we added two dropout layers, one in the penultimate layer (40%) and one right before the flatten layer (50%), which gave a performance boost of around 3%. The second CNN is deeper and was used in [9], and the LSTM was introduced in [3].

All the models perform similarly as the loss function is around 0.3 when the training ends. However, their capacity is different as the first CNN has 284 778 trainable parameters, the second CNN has 5 975 882, and the LSTM has 180 569. In this way, we can investigate whether both simple and more complex networks are susceptible to inaudible backdoor attacks. We also investigate whether the backdoor attack behaves differently for different neural networks (CNNs and LSTMs). We use the Adam optimizer with a learning rate of 0.0001 to train our models and L_2 regularization in every convolution filter in both CNNs. 80% of the initial dataset was

used for training/validation, and the remaining 20% for testing. The training dataset is again split 80%/20% for training and validation.

After carefully examining the training and validation loss functions, we saw that the CNNs could always converge before 300 epochs and the LSTM before 100. We used Tensorflow's early stopping callback with patience 20, which stops the training if the validation loss is not improved for 20 epochs returning the weights of the best-performing model.

The arithmetic mean of the epochs among all the experiments we run is 220.8 (± 24.97) for the small CNN, 182.4 (± 31.39) for the large CNN, and 70.9 (± 12.70) for the LSTM for the ten dataset classes and 250.5 (± 32.96), 102.5 (± 29.46), and 83.3 (± 13.16) for the 30 classes. We run each experiment 15 times to eliminate any randomness introduced by the algorithmic randomness (e.g., random initialization and stochastic gradient descent). All the models were trained with Tensorflow 2.5 on NVIDIA RTX 2080 Ti.

5 EXPERIMENTAL RESULTS

This section presents a systematic evaluation of our inaudible backdoor attack. We investigate how the ASRT is affected by various backdoor characteristics like the number of poisoned samples, the trigger's duration, and the trigger's position in time.

The triggers that we tried can be split into two groups: the long (250ms, 500ms, 750ms, and 1 000ms) and the short triggers (20ms, 40ms, 60ms, and 80ms). In this way, we aim to see if the attack is successful, even if only a few MFCC frames are affected by our trigger. Additionally, we applied our trigger in different positions to see if our models give more attention to the windows that usually contain speech.

First, we need to define the poison percentage that leads to an effective attack. Figure 1 shows ASRT as a function of the poison percentage for each architecture. In these experiments, we used a trigger of 20ms in the middle of each signal, which is short and makes the attack very challenging. When we poison 200 samples or more for both CNNs, ASRT is close to optimal even with this short trigger. These results are further improved by longer or non-continuous triggers and different trigger positions. Thus, we used a smaller poison percentage to keep any differences between different setups distinguishable. In particular, we poisoned at most 80 (0.59% for the ten classes and 0.21% for the 30 classes) training samples in the experiments with the CNNs. The ASRT is lower for the LSTM architecture. The attention layer can give useful insights into a network's choices [3]. In Figure 2, we show the logarithm of the attention weights for two backdoored models (20ms trigger in the middle used for the backdoor) when a poisoned input is used. We see that the attention layer does not recognize the trigger when 200 poisoned samples are used. This indicates that the attention layer is more robust to the backdoor attack suggesting that unless a smart position/duration/type of a trigger is used, the backdoor attack on LSTM could be rather difficult.

Roadmap. In our results, we show ASRT vs. poisoning rate. Each bar chart consists of three subplots, one for every trigger position. Each bar in these plots represents a trigger of a different duration. Furthermore, to better view the attack effectiveness for different trigger positions, we plot the average ASRT for a different number of poisoned samples in each trigger position. Each of these graphs
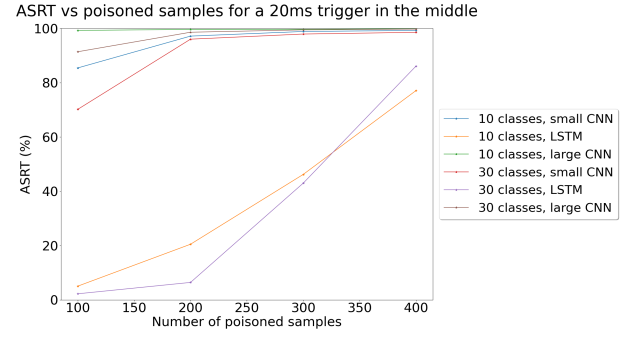

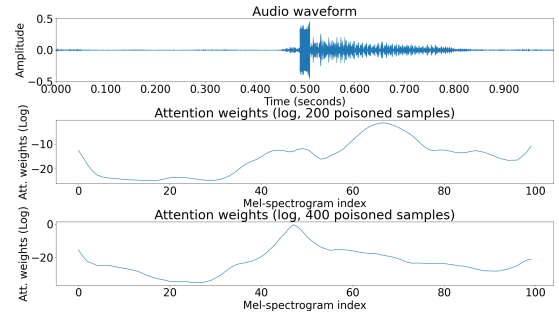**Figure 1: ASRT vs. trigger samples for more than 80 poisoned samples and CNNs.**

Figure 2: Attention weights for poisoned input ("on").

consists of four lines, one for each position (three lines) and one for the non-continuous trigger. We plot only the results for the 30 classes as there are no significant differences between the two versions of the dataset for each architecture.

5.1 Effect on Clean Accuracy

A backdoor attack should remain hidden to avoid raising any suspicions. Thus, we need to verify that our attack does not affect the model's performance on the original task. In Tables 1 and 2, we compare the clean model's accuracy on the original task with the accuracy that backdoored models have. To obtain the values for the poisoned models, we group them based on the number of poisoned samples used and take the average (arithmetic mean) of their clean accuracy for all the different triggers used.

The differences for the original task are negligible between the clean and the backdoored CNNs (Table 1). The backdoored models perform slightly better in the original task when we use ten classes. This behavior indicates that backdoored training data could improve the model's generalization, probably serving as a regularization factor. However, the differences are small, and further investigation is required to confirm our observations. When we use the full dataset, the backdoor insertion drops the clean accuracy in all cases for both CNNs (somewhat larger accuracy drop for large CNN). Again, this is aligned with our previous observations, as more classes (and a larger dataset) also represent a more difficult task for neural network architectures to learn.

Table 1: Clean accuracy comparison for the original task for clean and poisoned models (CNNs).

model	classes	original	Number of poisoned samples			
			20	40	60	80
small CNN	10	90.13 (± 0.533)	90.22 (± 0.127)	90.19 (± 0.086)	90.16 (± 0.108)	90.14 (± 0.091)
small CNN	30	87.58 (± 0.446)	87.52 (± 0.114)	87.46 (± 0.161)	87.49 (± 0.083)	87.47 (± 0.118)
large CNN	10	95.71 (± 0.424)	95.82 (± 0.122)	95.82 (± 0.139)	95.85 (± 0.133)	95.85 (± 0.112)
large CNN	30	95.14 (± 0.539)	94.82 (± 1.527)	94.61 (± 2.609)	94.24 (± 2.292)	93.52 (± 3.621)

The LSTM behaves oppositely. In Table 2, the clean accuracy is dropped in all cases when the ten classes are used and increased when the entire dataset is used. This indicates that the LSTM built different models in each case, and it utilized the model capacity better for ten classes setting.

Table 2: Clean accuracy comparison for the original task for clean and poisoned models (LSTM).

model	classes	original	Number of poisoned samples			
			40	80	120	160
LSTM	10	91.27 (± 1.217)	91.09 (± 0.354)	91.13 (± 0.29)	90.97 (± 0.459)	90.72 (± 0.359)
LSTM	30	91.45 (± 0.748)	91.76 (± 0.198)	91.63 (± 0.208)	91.59 (± 0.24)	91.77 (± 0.241)

5.2 Effect of Trigger Duration, Position, and Continuity on ASRT

5.2.1 CNNs. In Figure 3, we show the results for different triggers for the small CNN and the full dataset. The attack is more successful for the large CNN as it is a deeper architecture and is more capable of learning features from only a few training samples. However, we observe some similarities in the attack's behavior in both architectures, and for that reason, we omit the corresponding graph for the large CNN. In particular, ASRT increases linearly as the trigger's duration increases, especially in cases when it is not already high. We also see that long triggers result in a high ASRT even if only 20 poisoned samples are used. Additionally, ASRT for short triggers is low when using only 20 poisoned samples but is significantly larger when using 80 samples. This shows a strong connection between ASRT and the number of poisoned samples used for the attack.

In Figure 4, we show the mean ASRT for different backdoored models for the small CNN when the full dataset is used. Again, we omit the corresponding graph for the large CNN. We saw in both CNNs that the position is not important when a long trigger is used. However, the attack effectiveness varies for small triggers. For the small CNN, the non-continuous trigger is substantially more effective than the continuous ones, and the middle of the signal is the worst position for the trigger. The attack is more effective when more MFCC windows are affected and the trigger does not overlap with actual speech. In the large CNN, the non-continuous trigger outperforms the continuous ones only when the trigger lasts for 20ms. The network's complexity allows effective backdoor insertion even with short triggers longer than 20ms.

5.2.2 LSTM. In Figure 5, we show the results for each different trigger for the LSTM network for 30 classes. The ASRT is lower for this architecture even though we use more poisoned samples. As shown above, the attention layer cannot be affected easily by small variations inside the audio file. However, similarly to CNNs, there
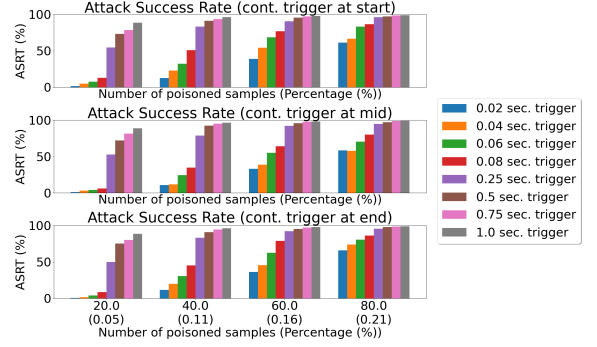
Figure 3: ASRT vs. poisoned samples for all the trigger positions and sizes for 30 dataset classes (small CNN).
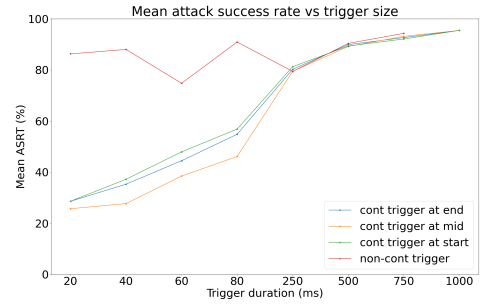
Figure 4: Mean ASRT vs. trigger size for 30 dataset classes (small CNN).

is a linear correlation between the trigger's duration and ASRT. Again, for long triggers, we see less difference in ASRT, especially when using more poisoned samples.

In Figure 6, we show the mean ASRT among different backdoored models for the LSTM network when the full dataset is used. We see that the trigger at the end of the signal is the most effective continuous trigger in both dataset versions. Additionally, non-continuous triggers clearly outperform continuous ones.

5.3 Attack on Android Application

Many microphones are inherently capable of recording audio signals in the near ultrasonic range (e.g., 18 to 22 kHz) [14]. To this end, we played our trigger through a VLC media player on a Linux laptop and verified through Spectroid [13] that a signal of 21kHz was played even if we could not hear anything. This experiment showed that the laptop speaker could transmit sounds in the inaudible human range, and the smartphone microphone could interpret them. Additionally, we verified that the inaudible trigger was successfully generated, which is the first step for a practical attack.

As a second step, we applied our attack on an Android speech recognition application. Our application is based on a TensorFlow Lite official example [16] that recognizes ten spoken words from the Speech Commands dataset. However, we modified its preprocessing pipeline to calculate the audio's MFCCs and used a poisoned model
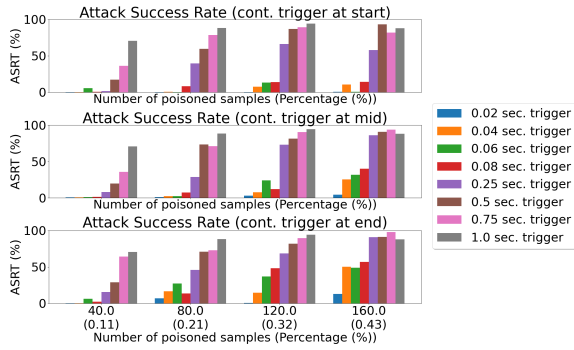
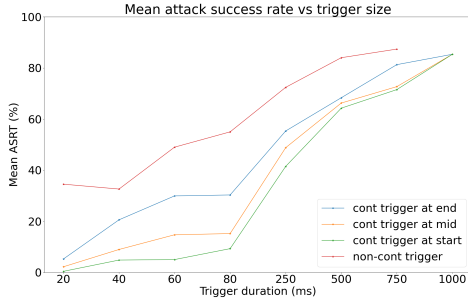**Figure 5: ASRT vs. poisoned samples for all the trigger positions and sizes for 30 dataset classes (LSTM).**

Figure 6: Mean ASRT vs. trigger size for 30 dataset classes (LSTM).

for the inference. We only run this experiment with the small CNN, but we believe this attack will be effective with all three architectures. We decided to use the small CNN because it uses less memory, while the ASRT with the LSTM showed worse performance, making it less attractive architecture. We used our strongest setup, i.e., 80 poisoned samples and a 1-second trigger to model our adversary. The application records 1-second clips and shows the keywords that are recognized with 50% probability or more.

We installed our application on a smartphone that ran Android 7.0 (Huawei P9 Lite). We played our inaudible trigger both from a Linux laptop (Lenovo t460s) and low-end home cinema speakers (Philips HTS7200). We run two experiments for each sound device. In the first, we played our trigger in silence to ensure that the backdoor was successfully activated when no sounds were present. In the second, we said words of the nine possible classes (omitted the target class) in random order while the trigger was played by either the laptop or the speakers. In each experiment, we placed the smartphone in four different locations to see how the attack accuracy is related to the distance from the sound source. The first location is adjacent to the sound source (0m), and the remaining ones are 0.5m, 1m, and 1.5m away from it. We run each experiment 15 times to have a more robust view of the attack's behavior.

Using Spectroid [13], we saw that for both devices, the trigger's intensity is around −43dB when the phone is adjacent to the sound

source. However, as the distance between the speaker and the phone grows, this number drops significantly faster for the laptop experiments. Additionally, the trigger's intensity is lower than the normal voice for all the positions we tried. As a result, the backdoor was successfully triggered only in silence as its volume was not enough to overcome our speech. In particular, ASRT is high only when the distance is 0m if the trigger is played through the laptop. On the other hand, if the trigger is played from the home cinema speakers, ASRT is high (> 80%) for distances up to 1m. At 1.5m, the backdoor is still activated but with a lower ASRT. We saw that ASRT is further increased when the user does not speak loud enough or the phone is not close enough to the speaker's mouth. We concluded from these experiments that this attack poses a real threat as an adversary with stronger equipment could be more effective.

5.4 General Observations

Based on the conducted experiments, we list several findings that were consistently observed.

- The attack success rate is linearly correlated with the trigger's duration. Our models can learn longer triggers easier as more windows in the MFCC calculation are affected.
- Our attack is inaudible, so an adversary could use a 1-second trigger for a powerful attack with a very low poisoning rate.
- If the adversary does not want to use a 1-second trigger (or any long trigger), non-continuous triggers represent the best option as they perform extremely well with short triggers.
- Different models behave differently for our attack. For example, the best ASRT for the short continuous triggers and LSTM is at the end of the signal, but for the large CNN, the trigger at the beginning results in a slightly increased ASRT. This indicates that the data poisoning backdoor attack could be used as a tool for AI explainability, as it can highlight how and what neural networks learn.
- The number of poisoned samples is clearly connected with ASRT. Thus, an adversary should use as many samples as possible without risking a large performance drop for the original task to avoid raising suspicions.
- The backdoor in the Android application was successfully activated by playing the trigger from the laptop and low-end home cinema speakers within a 1.5m.

Limitations. While our attack showed very good performance, we believe there are also some limitations concerning inaudible backdoor attacks:

- We use MFCCS as a convenient representation of audio signals for machine learning, and we did not validate our results with other types of signal representations. Still, we do not consider this a significant limitation due to the widespread use of MFCCS and no intuitive reason that the attack would not work with different representations.
- As we work with inaudible triggers, the sampling rate of the recorded signals should be larger than 40 kHz. If an ASR system uses a lower sampling rate, our attack will not be effective. A sampling rate of 16kHz is preferred in ASR systems [5], however popular public speech-to-text APIs allow higher rates [5, 10]. Additionally, different applications like

music genre classification require high sampling rates [12], making our attack practical.

- Finally, our work did not consider any defense against the inaudible backdoor attack. While we show it is possible to achieve (almost) 100% ASRT for various trigger positions and durations, even if using a small number of poisoned examples, different defenses could still work. Our trigger works for every source class in the dataset, and thus, an online defense like STRIP [4] could defend against it by superimposing different signals and checking the entropy of the model's output. Additionally, a low-pass filter could make our attack ineffective as it could filter out frequencies above the human audible range. Such a filter could be either software-based or hardware-based. A software-based filter could introduce additional computational overhead, which may be unacceptable in a device with strict timing constraints. Moreover, a hardware-based filter may not be possible in any device as frequencies close to the ultrasonic range may be useful in some cases like music genre classification [12].
- When playing a poisoned audio sample, if the volume is too high, it is possible to hear a slight clicking sound in its beginning. This clicking sound is connected with the backdoor trigger. Its volume is very low compared to the main signal, so unless expecting it, it is hard to hear it. This sound is heard only in the trigger's beginning, and thus, we still consider it to be practically inaudible.

6 CONCLUSIONS AND FUTURE WORK

This paper explores how inaudible backdoor attacks on neural networks threaten ASR. We use a dataset with ten or 30 classes and three neural networks (two CNNs and one LSTM). Furthermore, we investigate the influence of the position, duration, and trigger type. Our results suggest it is relatively easy to run an inaudible backdoor attack on ASR where the attacker needs to poison only around 0.5% of the training dataset. Since the trigger is inaudible, it is possible to make it as long as the signal, making the attack even more powerful. We show that non-continuous triggers can significantly improve attack performance even in scenarios where short triggers are used. Consequently, short non-continuous triggers inserted in less than 0.5% of the training dataset can result in an attack success rate of more than 99%. Finally, we show that our attack is effective in the real-word by attacking an Android application.

As we did not consider countermeasures, this would be a natural extension of our work. Today, most of the defenses against backdoor attacks try to detect either the poisoned inputs or compromised models [2, 7]. As we limit our data poisoning to a small number of training examples, we believe such techniques would have difficulties detecting poisoning examples. Furthermore, some more recent defenses suggested retraining [17], which we consider to be a good choice against our attack. Still, since such works commonly assume an unlimited amount of clean data, they are not practical. We are aware of only one approach that does not have that limitation [18], but it can work for image processing only, making this defense not applicable against our attack.

REFERENCES

[1] Rosana Ardila, Megan Branson, Kelly Davis, Michael Henretty, Michael Kohler, Josh Meyer, Reuben Morais, Lindsay Saunders, Francis M. Tyers, and Gregor Weber. 2019. Common Voice: A Massively-Multilingual Speech Corpus. *CoRR* abs/1912.06670 (2019). arXiv:1912.06670 http://arxiv.org/abs/1912.06670

[2] Bryant Chen, Wilka Carvalho, Nathalie Baracaldo, Heiko Ludwig, Benjamin Edwards, Taesung Lee, Ian M. Molloy, and Biplav Srivastava. 2018. Detecting Backdoor Attacks on Deep Neural Networks by Activation Clustering. *CoRR* abs/1811.03728 (2018). arXiv:1811.03728 http://arxiv.org/abs/1811.03728

[3] Douglas Coimbra de Andrade, Sabato Leo, Martin Loesener Da Silva Viana, and Christoph Bernkopf. 2018. A neural attention model for speech command recognition. arXiv:eess.AS/1808.08929

[4] Yansong Gao, Change Xu, Derui Wang, Shiping Chen, Damith C. Ranasinghe, and Surya Nepal. 2019. STRIP: A Defence against Trojan Attacks on Deep Neural Networks. In *Proceedings of the 35th Annual Computer Security Applications Conference (ACSAC '19)*. Association for Computing Machinery, New York, NY, USA, 113–125. https://doi.org/10.1145/3359789.3359790

[5] Google. 2022. Speech-to-Text basics. https://cloud.google.com/speech-to-text/docs/basics. [Online, Accessed: April 11, 2022].

[6] T. Gu, K. Liu, B. Dolan-Gavitt, and S. Garg. 2019. BadNets: Evaluating Backdooring Attacks on Deep Neural Networks. *IEEE Access* 7 (2019), 47230–47244. https://doi.org/10.1109/ACCESS.2019.2909068

[7] Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li. 2018. Manipulating Machine Learning: Poisoning Attacks and Countermeasures for Regression Learning. *CoRR* abs/1804.00308 (2018). arXiv:1804.00308 http://arxiv.org/abs/1804.00308

[8] Uday Kamath, John Liu, and James Whitaker. 2019. *Deep Learning for NLP and Speech Recognition*. Springer. https://doi.org/10.1007/978-3-030-14596-5

[9] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. 2018. Trojaning Attack on Neural Networks. In *25nd Annual Network and Distributed System Security Symposium, NDSS 2018, San Diego, California, USA, February 18-221, 2018*. The Internet Society.

[10] Microsoft. 2022. Speech to Text. https://azure.microsoft.com/en-in/services/cognitive-services/speech-to-text/. [Online, Accessed: April 11, 2022].

[11] Vinay Uday Prabhu and Abeba Birhane. 2020. Large image datasets: A pyrrhic win for computer vision? *CoRR* abs/2006.16923 (2020). arXiv:2006.16923 https://arxiv.org/abs/2006.16923

[12] Arjun Raj Rajanna, Kamelia Aryafar, Ali Shokoufandeh, and Raymond Ptucha. 2015. Deep Neural Networks: A Case Study for Music Genre Classification. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*. 655–660. https://doi.org/10.1109/ICMLA.2015.160

[13] Carl Reinke. 2018. Spectroid. https://play.google.com/store/apps/details?id=org.intoorbit.spectrum [Online, Accessed: April 11, 2022].

[14] Nirupam Roy, Haitham Hassanieh, and Romit Roy Choudhury. 2017. BackDoor: Making Microphones Hear Inaudible Sounds. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '17)*. Association for Computing Machinery, New York, NY, USA, 2–14. https://doi.org/10.1145/3081333.3081366

[15] Saeid Samizade, Zheng-Hua Tan, Chao Shen, and Xiaohong Guan. 2020. Adversarial Example Detection by Classification for Deep Speech Recognition. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 3102–3106. https://doi.org/10.1109/ICASSP40776.2020.9054750

[16] Google Tensorflow Team. 2019. TensorFlow Lite Speech Command Recognition Android Demo. https://github.com/tensorflow/examples/tree/master/lite/examples/speech_commands/android.

[17] Loc Truong, Chace Jones, Brian Hutchinson, Andrew August, Brenda Praggastis, Robert Jasper, Nicole Nichols, and Aaron Tuor. 2020. Systematic evaluation of backdoor data poisoning attacks on image classifiers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*.

[18] Miguel Villarreal-Vasquez and Bharat K. Bhargava. 2020. ConFoc: Content-Focus Protection Against Trojan Attacks on Neural Networks. *CoRR* abs/2007.00711 (2020). arXiv:2007.00711 https://arxiv.org/abs/2007.00711

[19] Li Wan, Quan Wang, Alan Papir, and Ignacio Lopez Moreno. 2018. Generalized end-to-end loss for speaker verification. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 4879–4883.

[20] Steven H Weinberger and Stephen A Kunath. 2011. The Speech Accent Archive: towards a typology of English accents. In *Corpus-based studies in language use, language learning, and language documentation*. Brill Rodopi, 265–281.

[21] Xiaojun Xu, Qi Wang, Huichen Li, Nikita Borisov, Carl A Gunter, and Bo Li. 2021. Detecting AI trojans using meta neural analysis. In *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 103–120.

[22] Tongqing Zhai, Yiming Li, Ziqi Zhang, Baoyuan Wu, Yong Jiang, and Shu-Tao Xia. 2021. Backdoor Attack Against Speaker Verification. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2560–2564. https://doi.org/10.1109/ICASSP39728.2021.9413468

[23] Guoming Zhang, Chen Yan, Xiaoyu Ji, Tianchen Zhang, Taimin Zhang, and Wenyuan Xu. 2017. DolphinAttack: Inaudible Voice Commands. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS '17)*. Association for Computing Machinery, New York, NY, USA, 103–117. https://doi.org/10.1145/3133956.3134052