(Automatic) Speech Recognition Project — Attacks on Neural Networks in a Lightweight Speech Pseudonymization Pipeline

Daan Brugmans
Radboud University
daan.brugmans@ru.nl

Abstract

We explore potential weak points of a Speaker Anonymization system that uses a lightweight, digital effects-based pseudonymization strategy by adversarially attacking the neural speech models in the system used for evaluating the pseudonymization strategy. We successfully insert a JingleBack backdoor into the system's Automatic Speaker Verification model, reaching an Attack Success Rate of 95% after providing only 5 backdoored utterances. We also show that the models are sensitive to FGSM and PGD attacks, and that small audio perturbations can lead to a remarkable decrease in model performance.

1 Introduction

As advancements in the field of Automatic Speech Recognition (ASR) have accelerated with the rise of modern end-to-end neural networks, the risks associated with using such models in ASR applications has become more evident.

Modern neural ASR models are capable of parsing and producing speech to a new level of authenticity: transformers are state-of-the-art for ASR and word recognition, and the introduction of modern unsupervised deep neural network architectures, such as Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs), has allowed for more realistic, accurate, and easier generation of speech. These modern speech generation models are capable of learning to reproduce a person's voice, and then generating new utterances using the learned voice. Such synthesized utterances are called *deepfaked* utterances, or simply *deepfakes*.

The presence and influence of deepfakes has become increasingly apparent in recent years: neurally synthesized audio and video of real persons are used to spread misinformation and to manipulate. This recent development has raised attention on the development of methods that prevent or complicate the production of deepfakes. One way to complicate the production of deepfakes is the removal of characteristics in the audio that relate to the speaker's likeness. This is called *Speaker Anonymization*. In Speaker Anonymization, we aim to apply changes to a speaker's utterance such that the changes mde make the utterance untraceable to the person's likeness, while understandability is maintained.

Modern Speaker Anonymization systems, often neural in nature, have been shown to be able to anonymize speech while maintaining understandability. However, such neural systems are not impenetrable, and there exist many sorts of attacks aimed at neural networks. By attacking neural Speaker Anonymization systems, we may be able to circumvent the preventative measures they provide, and generate speech to a person's likeness nonetheless. This paper will focus in that topic: attacking neural Speaker Anonymization systems. Specifically, we will focus on the topic of adversarial attacks. *Adversarial attacks* on deep neural networks are attacks that alter the input that the network receives. They are performed by applying a perturbation onto the input data with the aim to alter or manipulate the network's behavior. These perturbations should meet two criteria as faithfully as possible: they should not be detectable by humans and they should perturb the original input as little as possible.

For this paper, we attempt to answer the following Research Question:

> *How (effectively) can we manipulate a Speaker Anonymization pipeline by adversarially attacking its neural speech systems?*

2 Related Work

2.1 Neural Speaker Anonymization

Meyer et al. (2023) showcase a successful realization of neural Speaker Anonymization using GANs.

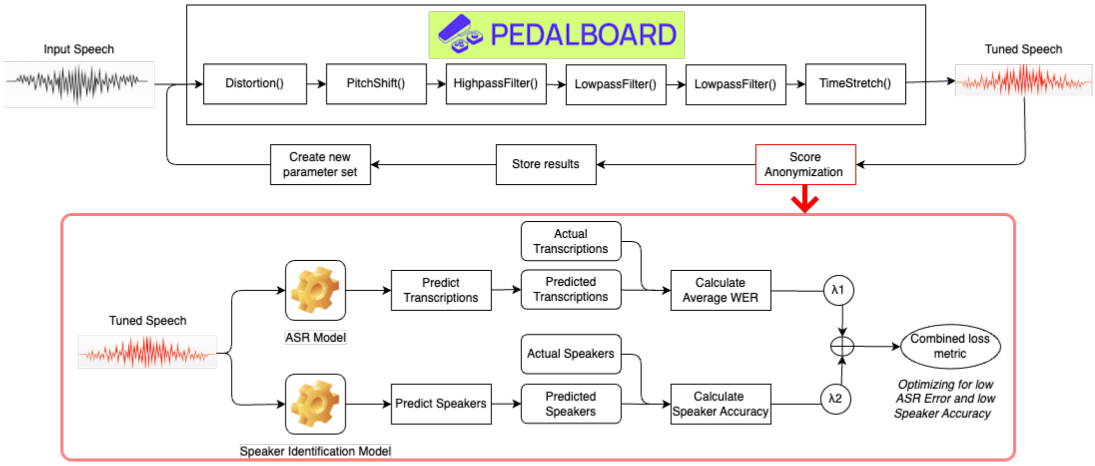

Figure 1: The optimization process of the lightweight pseudonymization pipeline by Roddeman et al. (2024).

Their Speaker Anonymization architecture includes the extraction of embeddings from speech, which are fed into a GAN. This GAN first learns to generate embeddings sampled from a normal distribution. After every sample, it then learns to calculate the distance between the embedding that it generated, and an embedding extracted from speech it has been fed. This training procedure teaches the GAN to generate embeddings that are similar to the speech embeddings. Once training has been finished, embeddings based on real speech embeddings are sampled from the GAN until an embedding is generated whose cosine distance from the corresponding real speech embeddings is sufficiently large. If the cosine distance is sufficiently large, the GAN should have produced an embedding that represents the contents of the speech utterance without representing the characteristics of the speaker found in the utterance. This embedding is fed to an existing speech synthesis model, which produces anonymized speech.

Chen et al. (2024) showcase another example of a neural Speaker Anonymization pipeline. Their main contribution is the use of an adversarial attack for anonymization purposes. Chen et al. use an adversarial attack called *FGSM* that learns to apply perturbations on a VAE's latent space vector that represents an utterance with a speaker's characterizations. The perturbations caused by the FGSM attack alter the latent space vector in such a way that it is as far removed from the original speaker's speech characterizations as possible, while still representing the original utterance. When the vector is then fed to the VAE's decoder, the decoder is unable to extract features from the perturbed vector that relate to the original speaker's speech characterizations. The resulting decoded speech sample is untraceable to the original speaker and thus anonymous.

Kai et al. (2022) showcase a lightweight Speaker Anonymization pipeline. Their proposed pipeline pseudonymizes speech by applying voice modifications to the spoken utterances. In their pipeline, the authors make use of a neural ASR model alongside an *Automatic Speaker Verification* (ASV) model, which are used for the recognition of words and speakers respectively. Instead of optimizing the performance of the ASR and ASV model, the authors attempt to optimize the voice modifications applied to the utterances. The optimization is performed by minimizing the *Word Error Rate* (WER) and maximizing the *Equal Error Rate* (EER). Figure 2 shows the design of the proposed pipeline.

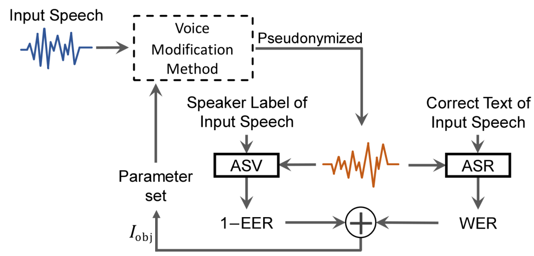

Figure 2: The lightweight pseudonymization pipeline by Kai et al. (2022) visualized.

By refraining from using neural models to anonymize the speech itself, and instead opting to use existing neural models only for evaluation purposes, the pseudonymization of the utterances becomes a relatively cheap operation. However, although the method proposed by Kai et al. is lightweight during inference, it does require that an optimal parameter set for the speech modifications is found first, which is a more resource intensive process. It should also be noted that this process produces pseudonymized speech, and not anonymous speech; the original speaker could potentially be recognizable, even if only slightly.

Roddeman et al. (2024) expand upon the work by Kai et al. (2022) by proposing their own lightweight speaker pseudonymization optimization pipeline, shown in figure 1. The authors' aims are to produce a pseudonymization pipeline that is resource efficient, easy to deploy, and optimizes swiftly. Their pipeline uses the same optimization strategy used by Kai et al.: pre-trained ASR and ASV models are used to calculate the WER and EER, which are used to optimize a set of audio modifications for pseudonymization purposes.

The authors performed 300 optimization trials, and evaluated their results on samples of utterances spoken by both men and women. They found that the most effective modifications for pseudonymization purposes include a time stretch factor, an increase in dB, and applying a low-pass cutoff. However, they also found that the effectiveness of a modification varies by the speaker's gender: while a time stretch factor was effective for both male and female spoken utterances, low-pass cutoffs only seemed effective on female-spoken utterances. Roddeman et al.'s implementation uses Pedalboard (Sobot, 2021) for applying the audio modifications and Optuna (Akiba et al., 2019) for the optimization process.

2.2 Adversarial Attacks on Neural Networks

The FGSM attack used by Chen et al. (2024) is an example of an adversarial attack. Adversarial attacks are extensively described by Yuan et al. (2019). They provide an introduction to and an overview of adversarial attacks within the deep learning domain, and should provide the reader with plentiful knowledge on the topic. Yan et al. (2022) provide an overview of adversarial attacks aimed at speech systems, including ASR systems.

For the purposes of this paper, we will limit out attention to two types of adversarial attacks: *Evasion Attacks* and *Backdoor Attacks*.

2.2.1 Evasion Attacks

Evasion attacks are adversarial attacks that are used during model inference. The aim of an evasion attack is to perturb an input in such a way that a fully trained model is fooled into behaving differently. This behavior should fulfill the attacker's goals.

Goodfellow et al. (2015) introduce FGSM, the *Fast Gradient Sign Method*. FGSM perturbs an input by exploiting a trained model's gradients. When given the input x and its label y, an FGSM attack calculates the gradient with respect to the input using the model's parameters θ and loss function $J_\theta(x, y)$. The sign of this gradient is calculated and is added on top of the original input by a factor of ϵ. The FGSM attack can then be defined as such:

$$x_t = x + \epsilon \cdot \text{sign}(\nabla_x J_\theta(x, y))$$

where x_t is the perturbed input. A visualization of this process can be found in figure 3.

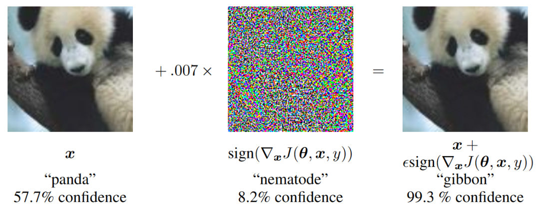

Figure 3: An example of an FGSM attack visualized (Goodfellow et al., 2015).

FGSM attacks have some limitations. One of these limitations is that an FGSM attack is a one-step approach: the gradient with respect to the input is calculated once, and is then used without corrective measures. Although this makes FGSM attacks cheap to perform, it also makes the attack difficult to perform optimally. Madry et al. (2018) propose an expanded version of the FGSM attack called the *Projected Gradient Descent* (PGD) method. The core principle of PGD is that it projects FGSM gradients back to a predefined max perturbation level; if an FGSM's perturbation is too big, PGD projects it back. This makes PGD a multiple-step approach.

PGD's projection of an FGSM attack is performed by applying a clipping function $\text{clip}(\cdot)$ on the FGSM attack. The clipping function will clip gradients outside the limit of $x + S$, where x is the original input data and S is the predefined maximal Euclidean distance a perturbation is allowed to be from x. Prior to clipping the FGSM perturbed input x_t, PGD will fire another FGSM attack on the model, but will now calculate the gradient with respect to the perturbed input x_t instead of the original input x. This additional FGSM attack is added onto x_t by a factor of hyperparameter α. It is the result of this addition that is clipped by the clipping function. This means that a PGD attack can be

Figure 4: An example of a BadNet attack (Gu et al., 2019). On the left, an image of a stop sign with varying BadNet backdoors can be seen. On the right, a backdoored network can be seen interpreting a stop sign as a speed limit sign due to the physical backdoor on the sign itself.

defined as follows:

$$x_t = x + \epsilon \cdot \text{sign}(\nabla_x J_\theta(x, y))$$
$$x_{t+1} = \text{clip}_{x+S}(x_t + \alpha \cdot \text{sign}(\nabla_x J_\theta(x_t, y)))$$

where x_{t+1} is the final result of one round of PGD. PGD can be performed for multiple rounds T until x_T is reached.

2.2.2 Backdoor Attacks

Backdoor attacks are adversarial attacks that are used during model training. They are a subset of *Poisoning Attacks*. The aim of a poisoning attack is to alter (a subset of) the training data in order to teach the model certain behavior or to decrease the model's performance. Backdoor attacks are a specific type of poisoning attack where the attacker teaches a model to respond to a certain property of the data. If the model encounters that property in a data sample, it should behave according to the attacker's goals. This property is called a *trigger*.

A well-known example of a backdoor attack is the BadNet attack by Gu et al. (2019). BadNet attacks poison a dataset of images by adding a visual feature onto the image, such as a colored square. A Convolutional Neural Network (CNN) will then learn to associate the visual feature with certain behavior. If the model has successfully learned to associate the trigger with certain behavior, then it will display that behavior during inference when given an image with the trigger. The trigger may cause the model to misclassify an image, for example. Such an example is showcased in figure 4.

Since backdoor attacks are applied during model training, certain attacks can only be used on certain types of data and networks. Although backdoor attacks are most common for image data and CNNs, there also exist backdoor attacks on audio data. Liu et al. (2018) showcase a variety of backdoor attacks, including a backdoor for ASR models where the trigger is background noise that is introduced to audio. The ASR model is trained to associate the background noise with a specific word. Koffas et al. (2022) propose a backdoor attack on ASR systems where the trigger is inaudible. The trigger is a frequency that exceeds the frequency range that can be heard by a human, which should make the trigger impossible to detect by the human ear.

In Koffas et al. (2023), the authors propose a backdoor based on stylistic triggers called *JingleBack*. The trigger in a JingleBack attack is a digital music effect applied on the audio, such as a pitch shift or reverb. A neural ASR model learns to associate certain combinations of effects with certain behavior. In their work, Koffas et al. backdoor neural models of three different architectures (small CNN, large CNN, and LSTM) using six different combinations of digital effects. These effects were applied using Pedalboard (Sobot, 2021). They find that certain combinations of effects are more effective triggers than others, and that additional effects do not necessarily improve the trigger's effectiveness. The most effective trigger found by the authors was to increase the audio's amplitude by 12dB, then apply a high-pass ladder filter, and then apply a phaser filter. This trigger results in a rate of attack success of up to nearly 95% when only 0.5% of the training data is poisoned.

3 Method

From the perspective of an adversarial attacker, Roddeman et al. (2024)'s pseudonymization pipeline has at least two noticeable points of attack:

1. The authors use neural ASR and ASV networks to evaluate the pseudonymization strategy. Although these models are pre-trained, the authors fine-tune the ASV model to a subset of the VCTK dataset (Veaux et al., 2017) used during the optimization process. Figure

5 shows how this fine-tuning is performed. Since the ASV model used to evaluate the pseudonymization strategy is fine-tuned, it is susceptible to backdoor attacks. It may be especially susceptible to a JingleBack attack (Koffas et al., 2023): if the model is taught to misclassify when a Pedalboard (Sobot, 2021) effect is applied to an utterance, then the evaluation of an optimal lightweight pseudonymization strategy may become biased.

2. After an optimal pseudonymization strategy has been found, the models can be attacked during inference using an evasion attack. Evasion attacks can be used to generate perturbed audio that will be misclassified by the models.

We will test the effect of evasion and backdoor attacks on the pipeline by Roddeman et al. and to what extent the neural models used for their optimization strategy can be exploited. We will attempt to poison the fine-tuning dataset of the ASV model by inserting a backdoor from Pedalboard. This backdoor should be an effect that is not already used by Roddeman et al.. They use the following effects in their optimization pipeline:

- High-pass Filter
- Low-pass Filter
- Time Stretch
- Pitch Shift
- Distortion
- Bitcrush
- Phaser
- Gain in dB

Notably, this does not include the use of a Ladder Filter, which is employed by Koffas et al. (2023) in their most effective JingleBack attack. We will follow their example and attempt to insert a backdoor into the ASV model using the Ladder Filter effect as the trigger. After fine-tuning, the model will then receive speech utterances with the Ladder Filter effect applied, and should then misclassify the input to a set target class. We will also attack the clean versions of the ASR and ASV models during inference. We will do this by perturbing utterances using both FGSM and PGD.

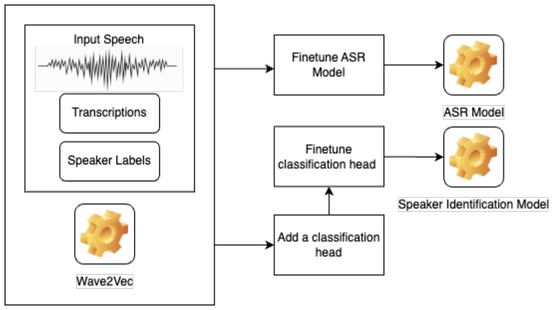

Figure 5: Fine-tuning of the neural ASR and ASV models in Roddeman et al. (2024).

We will assess the success of the attacks using varying measures. The *Attack Success Rate* (ASR) measures the rate that an attack has successfully led to the desired misclassification. The Attack Success Rate is the number of utterances not in the target class that are misclassified as being in the target class, divided by the number of utterances not in the target class. We will use the Attack Success Rate to measure the effectiveness of the trigger in the backdoored ASV model. We also calculate the *Clean Accuracy Drop* (CAD) of the backdoored ASV model, which is the number of percentage points the accuracy of the clean model has dropped after it was backdoored. We will also measure how the performance of the ASR model changes when it receives audio with the backdoor, despite not being backdoored itself. However, the performance of the ASR model is not measured using accuracy, but with WER. That is why we will measure the decrease in performance for the ASR model using the *Word Error Rate Increase* (WERI). For reference, the WER of the ASR model and the accuracy of the ASV model are also calculated.

4 Experiment

The code used for the experiments performed can be found at the following URL: `https://github.com/daanbrugmans/ru-automatic-speech-recognition-23-24`.

4.1 Backdoor Attacks

We backdoor the pre-trained ASV model using a JingleBack attack. This JingleBack attack poisons a dataset of VCTK audio files with a Ladder Filter. We setup a clean (not poisoned) and an adversarial (poisoned) test dataset of the utterances. Both test datasets contain the same 100 spoken utterances from 10 different speakers, totalling 1000 utterances overall. All utterances in the adversarial test dataset are poisoned (a *poisoning rate* of 100%). We use the adversarial test set to measure the Attack Success Rate. We also define a multitude of poisoned train datasets. Every poisoned train set contains the exact same utterances: 10 spoken utterances that are not included in the test set, for every speaker in the test set, for a total of 100 utterances. Every poisoned train set has a different poisoning rate, however, so that we can measure how the effectiveness of a trigger changes as more training data has the trigger inserted into it.

The ten speakers in the datasets all have a unique ID from 0 through 9. We set the target class of the backdoored utterances to 0. This means that, from the attacker's perspective, the goal is for the ASV model to learn that a Ladder Filter correlates with the speaker with ID 0.

The baseline ASV model used by Roddeman et al. is a pre-trained Wav2Vec2 model (Baevski et al., 2020) from HuggingFace called "Wav2Vec2-Base" (Baevski et al., 2021). We first fine-tune an instance of this pre-trained model on a clean train dataset (with no poisoned utterances). This model is called the clean model. We then calculate and store the Speaker Identification Accuracy and Attack Success Rate of the clean ASV model, alongside the WER of the ASR model on the clean test data.

After the clean model has been fine-tuned, we fine-tune a multitude of adversarial models. Every adversarial model is trained on a backdoored version of the train dataset with a different poisoning rate. We use poisoning rates of 5%, 10%, 20%, 30%, 40%, and 50%. In other words, we fine-tune adversarial models to 100 utterances, of which 5, 10, 20, 30, 40, or 50 are backdoored with a JingleBack trigger. This results in six backdoored ASV models. For every backdoored model, we calculate the Speaker Identification Accuracy, the Attack Success Rate, and the Clean Accuracy Drop. For every poisoning rate, we also calculate the Word Error Rate and Word Error Rate Increase of the clean ASR model.

4.2 Evasion Attacks

We attack an ASV model that has been fine-tuned on the clean test set used for the backdoor attack. We attack this model with both an FGSM (Goodfellow et al., 2015) and a PGD (Madry et al., 2018) attack. For our experiment, we will use $\epsilon = \frac{8}{255}$ for the FGSM model, and $\epsilon = \frac{8}{255}$ and $\alpha = \frac{2}{255}$ for the PGD model. We run the PGD model for 10 steps. The implementation of the FGSM and PGD attacks is taken from the Torchattacks (Kim, 2020) library and is refactored to better fit our needs.

We use the evasion attacks to generate perturbed utterances with the speaker with ID 0 as the target. This means that the perturbations caused by the attack should move any utterance closer to the space of utterances by the speaker with ID 0. We feed these perturbed inputs into the ASV model, and measure its Speaker Identification Accuracy. We also measure the clean ASR model's WER on these perturbed inputs.

5 Results

The backdoored and perturbed utterances computed during the execution of the experiments have been saved as audio files. Examples of clean, JingleBack backdoored, FGSM perturbed, and PGD perturbed audio can be found at the following URL: `https://github.com/daanbrugmans/ru-automatic-speech-recognition-23-24/tree/main/audio`

5.1 Backdoor Attacks

The results of the JingleBack attack experiment are shown in figure 6. At an increasing poisoning rate, the clean ASR model's Word Error Rate (WER) and Word Error Rate Increase (WERI) are shown, in addition to a backdoored ASV model's Speaker Identification Accuracy (Acc), Clean Accuracy Drop (CAD), and Attack Success Rate (ASR). A table of the exact values can be found in the appendices in table 2.

The figure shows that the Attack Success Rate of an ASV model spikes remarkably once it is backdoored in any capacity: while a clean ASV model's Attack Success Rate is just 18%, a backdoored ASV model with a 5% poisoning rate reaches an Attack Success Rate of 95%. In other words, after being given merely 5 speech utterances with a Ladder Filter effect as a trigger, the pre-trained ASV model has learned to classify roughly 950 out of 1000 test utterances with a Ladder Filter applied as being spoken by the speaker with ID 0. Furthermore, the Attack Success Rate continues to increase as the poisoning rate increases, reaching 98% with a poisoning rate of 20%, and a perfect 100% effectiveness with a poisoning rate of 50%. This means that 50 examples of speech utterances with a Ladder Filter trigger is enough for the pre-trained Wav2Vec2 model to learn the backdoor

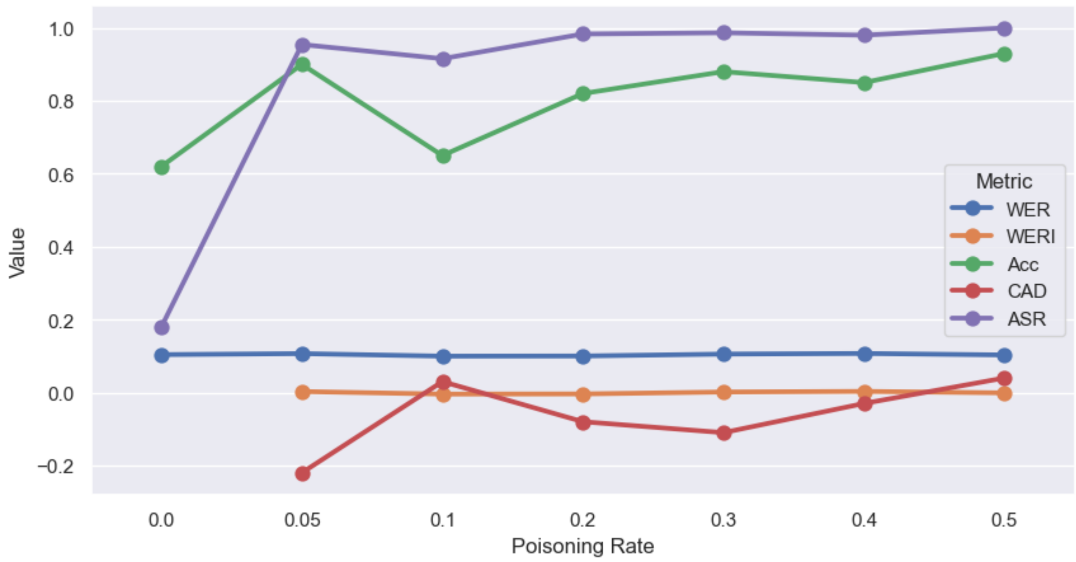

Figure 6: An overview of the measured metrics for the JingleBack attack. Per poisoning rate, the Word Error Rate (WER), Word Error Rate Increase (WERI), Speaker Identification Accuracy (Acc), Attack Success Rate (ASR), and Clean Accuracy Drop (CAD) are given.

perfectly.

The results show that, despite learning the backdoor near-perfectly, the accuracy of the ASV model remains stable. With the exception of two outliers, Speaker Identification Accuracy of the ASV model lies in the range of [0.82, 0.93]. The results also show a near-constant Word Error Rate, which is in the range of [0.100, 0.107] at any poisoning rate.

5.2 Evasion Attacks

	Acc	WER
Clean	0.62	0.104
FGSM	0.10	0.773
PGD	0.10	0.763

Table 1: Scores of the Speaker Identification Accuracy (Acc) and Word Error Rate (WER) of the clean ASV and ASR models on unaltered/clean utterances, FGSM-perturbed utterances, and PGD-perturbed utterances from the test set.

The results of the FGSM and PGD attack experiments are shown in table 1. After the utterances from the clean test set have been perturbed by an FGSM or PGD attack, the Speaker Identification Accuracy of the ASV model drops to 10%. Since the ASV model predicts from a pool of ten speakers, its accuracy is now no better than random guesses. The ASR model's Word Error Rate has increased from 0.104 to 0.773 on FGSM-perturbed audio and 0.763 on PGD-perturbed audio.

6 Discussion

The results from the Backdoor Attack experiment show that a simple JingleBack backdoor can successfully be inserted into Roddeman et al.'s ASV model using only 5 backdoored speech utterances, and that this number of backdoored utterances is sufficient in making the model misclassify 95% of all utterances with the JingleBack trigger inserted. The accuracy of the ASV model is not compromised as this backdoor is learned, with the ASV model being able to identify at least 80% of all speakers correctly when given an utterance without the trigger, which is a higher accuracy than the clean ASV model. This combination of a good accuracy and near-perfect Attack Success Rate should fulfill the wishes of an attacker, who aims to exploit model behavior using the trigger, and wants the model to learn this trigger as effectively as possible, without decreasing the clean accuracy, which would raise suspicion.

The results also show that the ASR model is unbothered by the trigger: Word Error Rates always vary less than 1% between poisoning rates, and are always between 10% and 11%. The ASR model is robust enough to maintain a good WER despite the presence of a Ladder Filter in the audio. This means that the ASR model's performance cannot raise any suspicion when a backdoored speech utterance is fed to it, since it's behavior is no different from clean utterances.

The results shown here are in line with the results of Koffas et al. (2023). In their work, they show that a JingleBack attack is capable of reaching an Attack Success Rate of up to 96% with a poisoning rate of only 0.5% on the Google Speech Command dataset (Warden, 2018) when a trigger with a Ladder Filter included is used, and that this backdoor causes only a small drop in classification accuracy. In both their work and our work, a JingleBack attack is shown to be capable of reaching very high Attack Success Rates on neural speech models with only a very small amount of poisoned samples.

The results from the Evasion Attacks experiment show that perturbing the utterances with an FGSM or PGD attack worsens the models' performances staggeringly: the FGSM attack causes the Word Error Rate to increase over sevenfold, and the Speaker Identification Accuracy is no better than purely random guesses. Despite its multi-step approach, the PGD attack causes the models to reach similar measures. It is likely that the perturbations, despite maximally deviating from the original utterances by a factor only of $\frac{8}{255}$, are too large for the models to handle. The size of the perturbations would prevent the generation of perturbed utterances that are still understandable by the models, which prevents an attacker from making the ASV model misclassify to a specific target speaker. However, the effects of the perturbations does show that the ASR and ASV models are quite sensitive to perturbed audio, which could be used to an attacker's advantage.

7 Conclusion

In this paper, we demonstrate how we can attack neural Speech Anonymization pipelines using adversarial attacks. We demonstrate that a pipeline that optimizes for lightweight effects-based speaker pseudonymization using neural speech models can be backdoored using a JingleBack attack, where the trigger is also another audio effect, and that such an attack can introduce a near-perfect backdoor into the neural models with as little as five adversarial examples. Furthermore, we demonstrate that the neural speech models are sensitive to FGSM-perturbed and PGD-perturbed audio, and that their performance drops remarkably once given utterances affected by an Evasion Attack. We conclude that adversarial attacks on Speaker Anonymization systems have the potential to disrupt the anonymization strategy and may prevent proper anonymization from being applied on spoken utterances.

References

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A next-generation hyperparameter optimization framework. In *The 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2623–2631.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Advances in Neural Information Processing Systems*, volume 33, pages 12449–12460. Curran Associates, Inc.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2021. Wav2vec2-base. `https://huggingface.co/facebook/wav2vec2-base`.

Shihao Chen, Liping Chen, Jie Zhang, KongAik Lee, Zhenhua Ling, and Lirong Dai. 2024. Adversarial speech for voice privacy protection from personalized speech generation. In *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 11411–11415.

Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*.

Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. 2019. Badnets: Identifying vulnerabilities in the machine learning model supply chain.

Hiroto Kai, Shinnosuke Takamichi, Sayaka Shiota, and Hitoshi Kiya. 2022. Lightweight and irreversible speech pseudonymization based on data-driven optimization of cascaded voice modification modules. *Computer Speech & Language*, 72:101315.

Hoki Kim. 2020. Torchattacks: A pytorch repository for adversarial attacks. *arXiv preprint arXiv:2010.01950*.

Stefanos Koffas, Luca Pajola, Stjepan Picek, and Mauro Conti. 2023. Going in style: Audio backdoors through stylistic transformations. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

Stefanos Koffas, Jing Xu, Mauro Conti, and Stjepan Picek. 2022. Can you hear it? backdoor attacks via ultrasonic triggers. In *Proceedings of the 2022 ACM Workshop on Wireless Security and Machine Learning*, WiseML '22, page 57–62, New York, NY, USA. Association for Computing Machinery.

Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. 2018. Trojaning attack on neural networks. In *25th Annual Network and Distributed System Security Symposium,* *NDSS 2018, San Diego, California, USA, February 18-221, 2018*. The Internet Society.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*.

Sarina Meyer, Pascal Tilli, Pavel Denisov, Florian Lux, Julia Koch, and Ngoc Thang Vu. 2023. Anonymizing speech with generative adversarial networks to preserve speaker privacy. In *2022 IEEE Spoken Language Technology Workshop (SLT)*, pages 912–919.

Julian Roddeman, Mats Robben, Maarten Berenschot, and Daan Brugmans. 2024. Anonymization pipeline using optuna and pedalboard. `https://github.com/JulianRodd/ASR-2024-anonymization-module-learning`.

Peter Sobot. 2021. Pedalboard.

Christophe Veaux, Junichi Yamagishi, and Kirsten MacDonald. 2017. Cstr vctk corpus: English multi-speaker corpus for cstr voice cloning toolkit.

Pete Warden. 2018. Speech commands: A dataset for limited-vocabulary speech recognition.

Chen Yan, Xiaoyu Ji, Kai Wang, Qinhong Jiang, Zizhi Jin, and Wenyuan Xu. 2022. A survey on voice assistant security: Attacks and countermeasures. *ACM Comput. Surv.*, 55(4).

Xiaoyong Yuan, Pan He, Qile Zhu, and Xiaolin Li. 2019. Adversarial examples: Attacks and defenses for deep learning. *IEEE Transactions on Neural Networks and Learning Systems*, 30(9):2805–2824.

A Additional Results

Poisoning Rate	**WER**	**WERI**	**Acc**	**CAD**	**ASR**
0.00	0.104	-	0.62	-	0.18
0.05	0.106	0.0028	0.90	-0.22	0.95
0.10	0.100	-0.0042	0.65	0.03	0.92
0.20	0.100	-0.0038	0.82	-0.08	0.98
0.30	0.106	0.0014	0.88	-0.11	0.99
0.40	0.107	0.0032	0.85	-0.03	0.98
0.50	0.103	-0.0009	0.93	0.04	1.00

Table 2: An overview of the measured metrics for the JingleBack attack. Per poisoning rate, the Word Error Rate (WER), Word Error Rate Increase (WERI), Speaker Identification Accuracy (Acc), Attack Success Rate (ASR), and Clean Accuracy Drop (CAD) are given.