Attack on Practical Speaker Verification System

Prakhar Aggarwal

Pratyush Saini

Anirudha Kulkarni

Under the guidance of

Prof. Vireshwar Kumar,
Indian Institute of Technology Delhi

Computer Science & Engineering
School of Computer Science
Indian Institute of Technology Delhi

2022

Acknowledgements

We would like to express sincere gratitude to our supervisor Prof. Vireshwar Kumar for providing us an opportunity to do this project on the topic of Adversarial Attacks Against Speaker Recognition Systems.
We learned so many new things after doing this project. By doing this project we got enriched with lot of fruitful information which can help us in our future.

Anirudha Kulkarni
Prakhar Aggarwal
Pratyush Saini

Introduction

Speaker recognition models can identify individuals based on their unique voice bio-metrics without requiring users to be present physically at the verification location.

This feature makes them increasingly popular nowadays in multiple domains like banking services, criminal investigation, etc. In this work, we consider a practical and challenging attack scenario where these speaker recognition models take streaming audio inputs (e.g., live human speech) and the adversary tries to deceive the system by playing adversarial perturbations, which is unnoticeable to human beings but has a great impact on the prediction made by the model.

Automatic speaker verification is a process used for identity verification. Existing studies have demonstrated that applying deep neural networks (DNN) to speaker recognition task has great advantages in terms of its highly scalable embedding performance and the ability of coping with practical interference (e.g., noises and reverberation). But studies also shows that audio adversarial examples can fool ASV.

Practical speaker verification find various use cases in real life. After the user speaking the some content speech, the audio recorded by a microphone goes through three module checks: audio replay check, speaker identity check, and speech content check. Only when all three check parts give pass decision, the user can be verified successfully. Previous attempts to attack such systems involved attacking the speaker identity check mod-

ule. But the generated adversarial examples in such cases would be rejected for audio replay or different speech content.

In addition to that, the adversary has prior knowledge of the model and can also fool the model to make the adversary desired prediction. We analyzed the behavior of such a model on the LibriSpeech dataset consisting of 30 speakers based on

Related Work

Traditional attacks against speaker recognition systems could be broadly categorized as replay attack, speech synthesis attack, impersonation attack, and voice conversion attack.

- Replay attack: uses pre-recorded voice samples of the user. It is less effective in practical scenarios as the inconsistency in the voice could alert the staff and thwart the attack
- Speech synthesis attack: generates a victim's voice by learning an acoustic model from an available set of samples. However human and synthetic speeches could be differentiated using a Deep neural network.
- Impersonation or voice conversion attack: imitating a victim's voice. However these attacks could be defended by PLDA and Factor Analysis techniques.
- In comparison to these attacks, our adversarial attack could circumvent the defense mechanisms while stealthily altering the Deep neural network's model's outputs without introducing noticeable distortion of speech.

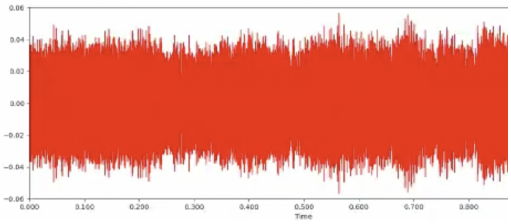

Figure 1: Initial Noise

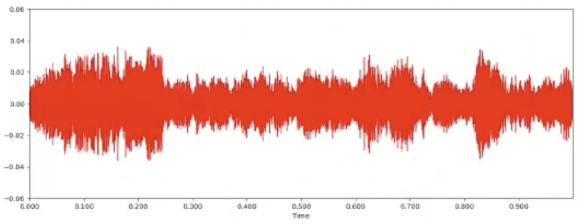

Figure 2: Noise after half training

- An audio agnostic universal perturbation is generated which can fool the model to recognize the speaker to be an adversary-desired speaker which will be independent of the speech.

Dataset

We used LibriSpeech dataset [https://www.openslr.org/12/] (test clean) for evaluations. Test clean dataset contains 40 speakers with 20 male candidates and 20 female candidates with voice samples. The dataset is divided with `./datas/splits.py` and stored in `./datas/splits/`.

Practical Speaker Verification

The practical speaker recognition model is generally used in real world applications in particularly three domains: Audio replay check, speaker identity check, and speech content check.

Speaker identity check basically calculates the similarity between the speaker feature vector representation of input audio and enrolled audio. The model constitutes a DNN based speaker verification Model to encode speaker feature representation corresponding to audio. The model comprises of ResNet-34 convolution layers to extract **frame level features**. This is then followed by pooling and fully connected layers to map the statistics to fixed dimensional vector which are the final speaker representation features.

Denote F(x) to be the DNN based speaker verification model, which takes an input audio x and outputs the speaker feature representation F(x). For speaker identity check, we use the cosine similarity between two audios x_1 and x_2 as score to measure the distance between their speaker feature representations. The caluclation is perfomed by:

$$cos\_sim(F(x_1), F(x_2)) = \frac{F(x_1)F(x_2)}{||F(x_1)||.||F(x_2)||} \tag{1}$$

The attack is successful if at testing time, cos_sim(F(x_1), F(x_2)) $\geq$ thresh. The adversarial perturbation can fool the speaker recognition model to verify an adversary as some input target feature.

Attack Model

The work considers a pure white box model of Attack where the adversary has full control of the ASV Model. First, noise is generated (δ) by sampling it uniformly between $-max(\delta)$ to $+max(\delta)$. The noise is then repeated to match the length of input audio. It is desired to model the noise such that the cosine similarity between the perturbed audio out of speaker model and that of enrolled target speaker becomes greater than a threshold value. Formally, we want δ' such that

$$cos\_sim(F(x + \delta'), F(y)) \geq thresh \tag{2}$$

Moreover, the perturbed audio should be such that human is not able to distinguish between text. So, the problem is divided into two steps

Step-1

In Step 1, the intention is basically to maximize the impact of ASV Model, such that the adversarial noise δ is effective enough to lead a targeted attack. N audios of the adversary are collected to form a training set $X = \{x1, x2, ..., xN\}$ where each xi contains different text contents. Then we iteratively train the model to make δ generalized enough for new audios of same adversary. Define Loss function,

$$L(X, \delta) = \sum_{i=1}^{N} max(\theta - cos_sim(F(x_i + \delta'), F(y)), -\kappa) \tag{3}$$

Here, κ is the attack confidence which controls the success rate of the attack. Then, δ' is added to each N-batch audio so L(X, δ) can be calculates in forward pass. Adversarial noise, δ' is further

updated at each step using projected gradient descent (PGD) with momentum update method, and further back propagating till first layer. δ is finally cliiped between it's max and min values as specificied during their initialization.

Step-2

The second step focuses on optimizing δ so that the audio becomes unperceivable by humans. The goal is to minimize difference between frequency domain features of perturbed audio and original audio. STFT(x) is used to represent the short-time fourier transform of the input speech x. For the linearity of Discrete Fourier Transform (DFT), we have the difference
d(x + δ) = $\|STFT(x+\delta)$ STFT(x)$\| = \|STFT(\delta)\|$.

So we define another objective function (CW Loss):

$$L_2(X, \delta) = mean(|STFT(\delta)|) \tag{4}$$

Meanwhile, we need to retain adversarial attack on the ASV model. So the goal of the second step is to minimize the funciton

$$L(X, \delta) = L1(X, \delta) + \gamma L_2(X, \delta) \tag{5}$$

where γ is a balanced parameter between two terms. We start the second step from initializing δ and $\delta*$ in the first step and continue to optimize δ to get final adversarial perturbation $\delta*$

Why the attack is successful?

The upper limit of noise (ϵ) was set to 0.03 for launching digital attacks. A higher value of ϵ could lead to higher distortion in

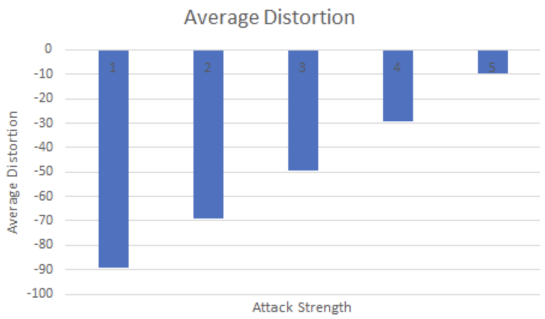

Figure 3: Visualization distortion vs -log(ϵ)

the final audio, which makes it susceptible to humans.
Threshold for Step 1 was set to 0.001 and for step 2 was set to 0.04. The max number of epoch were set to 1500. We use the adversarial noise that was returned in Step-1 as the baseline noise for Step-2. In Step-1, the final adversarial noise is robust enough to fool the prediction made by speaker model. The cosine similarity between the wav features output by model for original audio as well as target audio are higher than the specified threshold, which is the main reason the speaker model fails to recognize the correct speaker.
Second step enchanced the audio quality and helped in accurate speech recognition (i.e. speech to text) which signifies that a human hearing the perturbed audio could not be able to distinguish between original and perturbed audio.

How can the attack be improved?

- We have generated an audio-agnostic universal perturbation which fools the model to recognize the speaker to be an adversary- desired speaker independent of the speech.

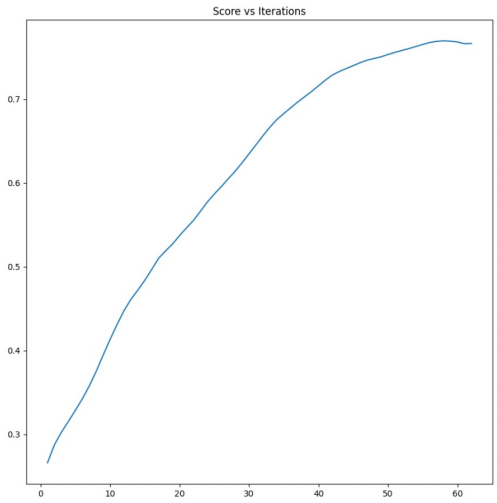

Figure 4: The following figure shows how the similarity in-creases between Target speaker voice embeddings and theoriginal speaker wav features progressively as model training proceeds

- But the noise generated as of now (attached in the presentation) is **quite high** and can be differentiated if listened closely.
- Wide variety of loss functions and optimzers could be explored which can help to fasten the attack maximization step and noise generation step with lower susceptibility.

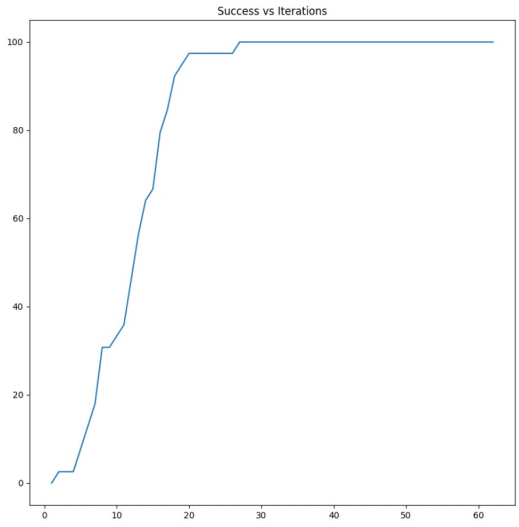

Figure 5: Success rate of the model (i.e. percentage of speak-ers correctly predicted as target speakers) after each iteration of model training

Future Work

Adversarial examples can be made more practical with integrations such as estimated room impulse response (RIR), training with noisy dataset, and training on commercial speaker recognition systems. The work may be extended towards black box approach where the exact model structure is not known.

To generate adversarial perturbations without prior information (e.g., architecture, parameters) of a target model, There can be an attempt in using leveraging gradient-free optimization algorithms (e.g., genetic algorithm) or training an equivalent substitute model. The experiments performed here were digital which can be defended by liveness detection mechanisms. A possible way to bypass liveness detection is to add step 3 that can fool liveness detection with speaker recognition.