DEFEATING SPEAKER RECOGNITION SYSTEMS USING ADVERSARIAL EXAMPLES

By

ZHUOHANG LI

A thesis submitted to the

School of Graduate Studies

Rutgers, The State University of New Jersey

In partial fulfillment of the requirements

For the degree of

Master of Science

Graduate Program in Electrical and Computer Engineering

Written under the direction of

Bo Yuan

And approved by

_____

_____

_____

New Brunswick, New Jersey

October 2019

ABSTRACT OF THE THESIS

Defeating Speaker Recognition Systems Using Adversarial Examples

by ZHUOHANG LI

Thesis Director:

Prof. Bo Yuan

Voice-user interface (VUI) has exploded in popularity due to the recent advances in automatic speech recognition (ASR). Such an interface has been integrated into various platforms, such as mobile phones, smart appliances and stand-alone voice assistants, and provides a convenient way for people to interact with them. Moreover, with such a convenient VUI, voice speaker recognition system that could be seamlessly integrated to facilitate security-related applications or personalized services has gained considerable attention recently. However, existing studies have shown that deep neural networks (DNNs), as the computational core of speaker recognition systems, are vulnerable to adversarial examples, which strategically perturbed input examples leading to a fraudulent prediction.

In this thesis, we demonstrate that it is possible to construct adversarial examples against speaker recognition systems. Particularly, we use several gradient/iterative-based methods (e.g, fast gradient method (FGM), fast gradient sign method (FGSM), and basic iterative method (BIM)) and our proposed optimization-based approach for crafting adversarial perturbations for both untargeted and targeted attacks. The untargeted attack could make the system give an incorrect prediction while the targeted attack could change the prediction result to any user desired by an adversary. Our experiments using a public speech dataset of 109 people show that with only partial knowledge of the speaker recognition system, the adversary is capable of reducing the system accuracy by over 60% for the untargeted attack. For the targeted attack, our approach could achieve an overall success

rate of over 60%. With full knowledge of the system, the adversary can reduce the system accuracy by 65%, while still keeping the crafted speech unnoticeable to humans.

ACKNOWLEDGEMENTS

I would like to begin by thanking all the professors at the Electrical and Computer Engineering Department at Rutgers University, who have taught me and guided me. I especially want to express my heartfelt gratitude to my advisor and mentor, Dr. Bo Yuan, for his continued guidance, immeasurable help, and great encouragement throughout the writing of my thesis. Without his genuine support and direction, it would not have been possible for me to finish my graduate study and complete this thesis. I would also like to give my sincere appreciation and deepest thanks to my thesis committee members, Dr. Yingying Chen and Dr. Sheng Wei for providing valuable suggestions and insights.

In addition, I am indebted to all the researchers within the ECE department, Jian Liu, Chen Wang, Cong Shi, and Yi Han, for their involvement in the discussion of my project and their generous help. As my mentors, Yi Xie and Siyu Liao have also provided me with many great inputs and perspectives.

Last but not least, I want to thank all my friends and colleagues for their genuine help and companionship, which had made my two-year graduate study at Rutgers fruitful and enjoyable.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

Speech is one of the most natural ways for human-computer interaction (HCI). As a result, voice assistants have been gradually changing our daily lives with its wide deployment on smartphones, smart speakers, vehicles and many other embedded systems and devices during the last decade. The major function of the voice assistant systems is automatic speech recognition (ASR), which automatically translates voice commands into corresponding texts. The huge influence of existing ASR systems arouses an increasing interest on the study of voice authentication processes. Speaker recognition (SR) addresses the task to identify the speaker of a provided utterance. There is the text-dependent speaker recognition which requires the utterance to be a series of fixed words, and the text-independent speaker recognition which has no constraints on the speech content. SR can be further divided into two subtasks: speaker verification makes 1 to 1 comparison to verify if the audio is spoken by the claimed speaker, and speaker identification (or speaker classification) makes 1 to N comparison to search for the correct speaker among a set of potential speakers. Generally, SR systems function by extracting the characteristics of each speaker's voice and constructing a voiceprint to uniquely identify the speaker. Comparing to other biometrics, voiceprint has the advantage of high accuracy, high user acceptance, and low cost. Nowadays SR systems have been widely deployed in online banking, identity verification for contact centers, criminal detection, user authentication on mobile devices and many other security-intensive areas. However, the robustness of SR systems when facing adversarial attacks is still doubtful.

As deep learning becoming extremely powerful and successfully pushing the state-of-the-art in many difficult tasks including image classification [1, 2], image segmentation [3] and speech recognition [4], modern speaker recognition systems have adopted deep neural

networks (DNNs) as a replacement part for the traditional GMM-UBM based embedding models. DNNs enable the system to learn to extract compact representations of the speaker level variations from a massive amount of data and the DNN-based speaker recognition systems has shown its superiority over the traditional systems in terms of recognition accuracy and error rates. However, previous studies [5, 6] have shown that DNNs are vulnerable to adversarial examples, which manipulate inputs intentionally to cause false predictions. The finding of adversarial examples has aroused increasing interest in the machine learning community. Since the perturbation between the original input and the adversarial example is small enough to be unnoticeable to human, the adversarial examples can bring severe damage to the systems causing security risks and concerns. Existing works on adversarial machine learning have been mainly focusing on vision domain [7, 8, 9], few papers touch upon auto speech recognition [10]. To the best of our knowledge, we are not aware of any existing study on evaluating the robustness of speaker identification systems. In this thesis, we seek to explore the possibility of attacking speaker recognition systems using adversarial examples. In summary, the main contributions of this thesis are:

- We demonstrate the vulnerability of speaker recognition systems by designing and implementing several attacks against a DNN-based text-independent speaker identification system.
- We validate the feasibility to construct a targeted attack on such systems, which allows us to possibly manipulate the prediction of a piece of provided audio recording to any class of speaker at will.
- We provide several algorithms for generating untargeted and targeted adversarial examples against speaker recognition systems. Two untargeted algorithms are capable of generating adversarial examples using only a single step, and two targeted algorithms generate adversarial examples through iterative or optimization process to construct strong attacks.

- With the constraints on distortion matrices, we report the noise level of most our adversarial noise to be in the range of -15 dB — -40 dB, which is quasi-imperceptible to human listeners.

- We perform extensive experiment under two different realistic attack scenarios. Merely with the knowledge of embedding model, we are able to achieve an attack success rate of 60% in targeted setting and 35% in untargeted setting, while still keeping the noise at a unnoticable level. With complete knowledge of the system, we can achieve a success rate of over 65% with our untargeted attack method, and 100% with our targeted attack method at a low noise level.

1.1 Notational Convention

In the rest of this thesis we use the following notations:

- X denotes the clean input, which is the unmodified data sample from the original data set.

- y denotes the label for input X.

- δ denotes the adversarial perturbation in the input space.

- X' represents the adversarial example crafted by applying adversarial perturbation to the input which intends to cause misclassification: $X = X' + \delta$. Conventionally we call any inputs produced by this formula 'adversarial examples' even though some of them may not cause the ML model to produce false output.

- $C(X)$ is the output classification results made by the ML model for input X.

- $J(X, y)$ represents the cost function used to train the ML model. Model parameters are considered in the cost function by default.

- $D(x, y)$ represents the distance between x and y, measured by some distance matrix, for instance L_0, L_2, L_∞.
- e denotes the embedding of the input.
- E represents the mapping from input X to its embedding e (i.e. $E : X \rightarrow e$).

1.2 Thesis Outline

The rest of this thesis is organized as follows:

Chapter 2 describes the architecture of conventional and modern speaker recognition systems and the workflow of such systems. We also introduce previous studies on adversarial machine learning and existing algorithms for generating adversarial examples.

Chapter 3 discusses the two settings of adversarial scenarios used as our threat model, the distortion matrix we chose for evaluation. Finally, the algorithms for generating adversarial examples in each scenario are introduced respectively.

Chapter 4 presents the results of applying our attack methods to a speaker recognition system and evaluates the attack success rate and noise level under different adversarial settings.

Chapter 5 introduces related works in attacking speaker recognition systems using synthesized or recorded audio clips and generating adversarial example for automatic speech recognition systems.

Chapter 6 concludes the thesis.

CHAPTER 2
BACKGROUND

In this chapter, we introduce the structure of a typical speaker identification system, the basis of adversarial examples and existing attack methods.

2.1 Speaker Identification Systems

Speaker identification (SI) systems are designed to classify the identity of an unknown voice from a set of speakers. In most SI systems, an embedding method is used to map phonetic statistics into a feature space where distance measures the similarity of different voices. Traditional text-independent SI systems use i-vectors [11] and probability linear discriminant analysis (PLDA) [12] to make classification. As is shown in Figure 2.1, this framework can be decomposed into 3 components:

- **Front-End**: After voice activity detection (VAD), the raw acoustic signal is pre-processed and converted into feature vectors. Figure 2.2 shows the process of deriving feature vectors from an audio signal via the most commonly used mel-frequency cepstral coefficients (MFCC) [13].
- **Embedding Model**: High-dimensional statistics are computed by a Gaussian Mixture Model (GMM) based Universal Background Model (UBM), and a projection matrix is used to get low-dimensional i-vectors that encodes speaker identity.
- **Classifier**: A separately trained PLDA classifier is used to make predictions for different utterances. This process involves principal component analysis to project the embedding vectors into low-dimensional subspace, where probability scores for each class is calculated by Gaussian distribution. The class with the highest score would then be output as the prediction result.

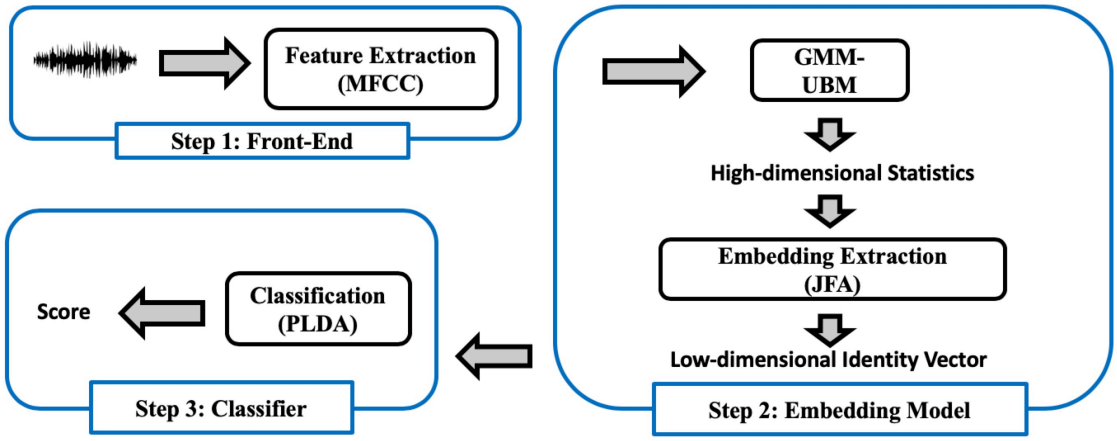

Figure 2.1: Traditional text-independent SI system framework

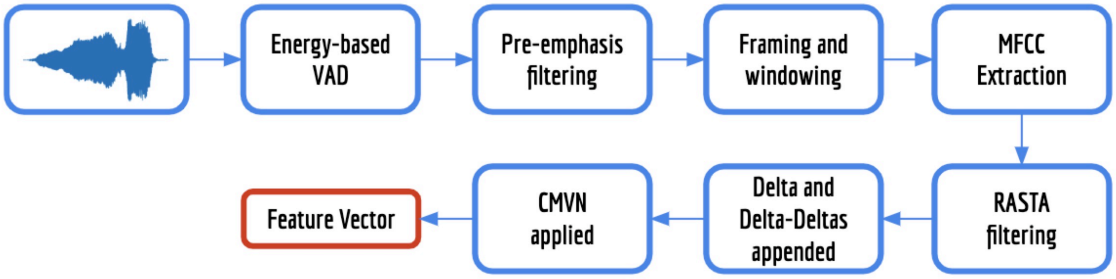

Figure 2.2: Feature vector extraction

Recently in the speaker verification (SV) task, many researches [14, 15, 16, 17] have been seeking to replace the i-vector embedding model with a DNN-based feature extractor for computational efficiency and model compactness. In the speaker identification (SI) task, Chen *et al.* [18] proposed to use a DNN constructed by multiple fully-connected (FC) layers to extract speaker level variability from the feature vectors. Synder *et al.* [19] uses time-delayed deep neural networks (TDNN) which is directly-optimized to discriminate speakers as the embedding model to achieve a superior performance.

The workflow of a SI system can be broken down into 3 phases [14]:

1. **Development**: The embedding model is trained with a large collection of data to establish the mapping between utterances and speaker variability. In commercial practices, this part is typically done by the manufacturer.

2. **Enrollment**: The users are enrolled by collecting speaker specific data to construct speaker-dependent models. The speaker sets in the development phase and the enrollment phase should be mutually exclusive. This part is usually done by the users after purchasing the product. Comparing to the training set, the size of the data set required for enrollment is very small.

3. **Evaluation**: This is where the system makes predictions. An utterance is feed into the model and a decision is made among enrollment speaker set.

The major distinction between SI system and other ML systems (e.g. image classification system, automatic speech recognition system) is the requirement for the enroll process. Note that unlike other ML systems, it's impossible for the SI systems to anticipate the number of classes, as well as the classification objectives, and therefore it cannot offer off-the-shelf pre-trained classifier models. In realistic commercial scenarios, the embedding models trained by the manufactures are often fixed and stored either locally or on a remote server, and the user's profile is usually stored locally after enrollment.

2.2 Adversarial Examples and Attack Methods

2.2.1 Adversarial Examples

Recent breakthroughs in machine learning (ML) have enabled researchers to solve a variety of complex real-world problems in speech, vision, video, text and other domains. Many ML models are responsible to make security critical decisions such as autonomous driving, malware classification, criminal detection, online-banking authentication etc. However, several studies have demonstrated that most ML models are vulnerable to adversarial examples. Adversarial examples are intentionally perturbed input examples that are designed to cause the victim ML model to make incorrect output with high confidence. The perturbation are often imperceptible to human, making it difficult to deploy ML models in

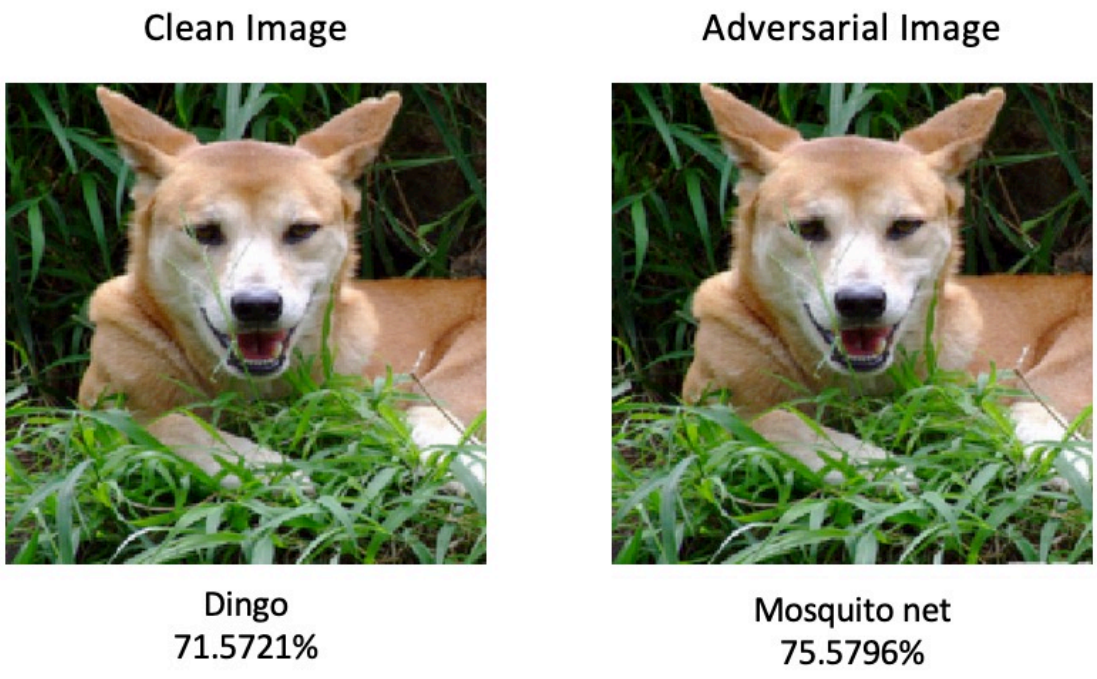

Figure 2.3: An illustration of adversarial attack

security-intensive real-world scenarios. For instance, Figure 2.3 shows that in the ImageNet [1] classification task, the adversarial image makes the neural network to misclassify a dingo as a mosquito net with higher confidence.

Szegedy *et al.* [20] first introduced the vulnerability of ML models when facing adversarial examples. In addition, their experiment shown that an adversarial example that is designed to fool model A can also be misclassified by model B. This transferability property makes it possible for attackers to generate adversarial examples on any ML models even without the knowledge of its inner structure and parameters. This finding results in a series of studies in adversarial machine learning. Papernot *et al.* [21] tested this property under realistic scenarios. Kurakin *et al.* [22] applied adversarial examples in real-world environments using printed-out images and cellphone cameras. Several attempts [6, 23] have been made to theoretically explain the existence of adversarial examples and various methods [24, 25] are proposed to increase the robustness of ML models.

2.2.2 Attack Methods

Adversarial attacks can be classified into two types: *targeted attack*, which aims to generate adversarial example X' that would be classified by the victim model as a specific target class (i.e. For a given target label t , construct an adversarial example X' $s.t.$ $C(X') = t$), and *untargeted attack*, which only seeks to make the victim model to make false prediction (i.e. For a given clean input $C(X) = t$, construct an adversarial example X' $s.t.$ $C(X') = i : i \neq t$). Previous studies on both methods to generate adversarial examples have mainly been focusing on vision domain. This section gives a brief introduction to some important attack methods.

Fast gradient sign method Inspired by the observation that most deep neural networks are locally linear, Goodfellow *et al.* [6] proposed a single step solution to compute untargeted adversarial perturbation:

$$X' = X + \epsilon sign\left(\nabla_X J\left(X, y\right)\right), \tag{2.1}$$

where ϵ is a preset constant to control the attack strength. Smaller values are often chosen to make the attack undetectable. Intuitively, as a input with higher cost has better chance to be misclassified, FGSM generates adversarial examples by taking one small step in the input space in the direction of the gradient of the cost function.
With this method, we can also perform targeted attack with only a slight modification:

$$X' = X - \epsilon sign\left(\nabla_X J\left(X, y_t\right)\right), \tag{2.2}$$

which brings the classification results one step closer to target class t. Note that FGSM seeks to be fast, instead of finding the optimal perturbation.

Basic iterative method Kurakin *et al.* [22] proposed a modification to the original FGSM to improve the attack performance. The single step of size ϵ in the direction of the

gradient is broken down into multiple steps of size α, and the total length is clipped by the same value ϵ. The procedure can be formally described as:

$$X'_0 = 0, \quad X'_n = Clip_\epsilon \left\{ X_{n-1} + \alpha sign \left(\nabla_X J \left(X_{n-1}, y \right) \right) \right\}. \tag{2.3}$$

To generate targeted adversarial examples, formula 2.3 needs to be modified as follows:

$$X'_0 = 0, \quad X'_n = Clip_\epsilon \left\{ X_{n-1} - \alpha sign \left(\nabla_X J \left(X_{n-1}, y_t \right) \right) \right\}, \tag{2.4}$$

where y_t is the label of the target class. Experiments shows that basic iterative method (BIM) produces better results comparing to FGSM.

C&W To break robust models with defense methods, Carlini *et al.* [9] proposed a optimization-based method to construct strong targeted attacks. The problem of searching for an adversarial example for input X can be initially formulated as a optimization process:

$$\begin{aligned} &\text{minimize } D(X, X') \\ &\text{such that } C(X') = t \\ &X' \in [0,1]^n, \end{aligned}$$

where we assume X is a vector of dimension n, and $[0,1]^n$ is the re-scaled box constraint for X (e.g. in the image domain, this would be the RGB value for each pixel from 0 to 255).

As the original formulation can be very difficult for any existing optimization methods to solve due to the non-linearity of the classifier C, the authors instead solve an alternative

problem:

$$\begin{aligned} &\text{minimize } D(X, X') \\ &\text{such that } f(X') \leq 0 \\ &X' \in [0,1]^n, \end{aligned}$$

where f is a self-defined objective function satisfying that $f(X') \leq 0$ is the sufficient and necessary condition of $C(X') = t$. In practice, they solve:

$$\text{minimize } D(X, X') + c \cdot f(X') \quad s.t.\ X' \in [0,1]^n, \tag{2.5}$$

where c is a pre-chosen positive constant. As the process of selecting from different objective functions f, distance matrix D and constant c values is rather sophisticated, we refer interested readers to their original paper.

To perform adversarial attacks in real-world scenarios, the attacker would need to apply the calculated adversarial perturbation to the legitimate input to achieve her malicious intents. For instance, to fool image recognition DNN systems in a self-driving car, the attacker might want to use a print-out adversarial image [22], or a sticker [26] to modify the stop sign. Similarly, the attacker could play a mixed audio of the adversarial noise and the clean voice sample to fool the speaker verification system used in a bank's call center.

CHAPTER 3
METHODOLOGY

In this chapter we clarify our assumption on adversarial setting, and then introduce our several attack methods. Finally we introduce the framework we used as the benchmark to evaluate our attacks.

3.1 Threat Model

Given an input acoustic signal X and a victim SI system with classifier C, we seeks to generate an adversarial signal $X' = X + \delta$, so that X and X' are similar, but $C(X') \neq C(X)$ (untargeted attack) or $C(X') = t$ (targeted attack). Considering the increasingly wide commercial deployment of DNN-based SI systems, in this thesis we will demonstrate attacks against SI systems with DNN embedding models. We assume the adversary is able to perform white-box attack, which allows the adversarial to access the complete network, including its architectures and parameters. In addition, we expect our adversarial audios are directly fed into the model without any external noise introduced (e.g. being played over the air and recorded by microphones). This is a typical threat model picked by most previous work [6, 10]. However, we believe our results are not necessarily limited to white-box settings or merely DNN models, as many studies has proved that adversarial examples could be transferable to different models and are resilient to environmental noise in image tasks [21, 22], and we expect that our attack can be potentially extended to real-world scenarios.

In most SI systems, the embedding models are fixed and either stored on remote servers, or uniformly stored on every devices, whereas the enrolled model (i.e. users' profile) are often stored locally on a private device and could be changed by user behavior. We notice

that in either case, it's easier for the adversary to access the embedding models using probes or other reverse-engineering methods than to access the enrolled model. To address this problem, we further assume the following two scenarios:

- **Partial knowledge scenario**: Assuming that the adversary only have knowledge of the embedding systems, the adversary could perform: a. untargeted attack to change the classification result to a random speaker, b. targeted attack to change the classification result to a specific speaker, provided with a piece of recording from the target speaker. In the remaining part of this thesis, we will refer this as the *embedding space* or the *embedding level* attack.

- **Full knowledge scenario**: Assuming that the adversary has complete knowledge of both the embedding system and the enrolled model for classification, the adversary could perform: a. end-to-end untargeted attack to change the classification result to a random speaker b. end-to-end targeted attack to change the classification result to any specific speaker, without the requirement of having recording from any speaker. In the remaining part of this thesis, we will refer this as the *scoring space* or the *scoring level* attack.

3.2 Distortion Metric

To be able to evaluate the adversarial distortion in a quantitative format, we use decibels (dB) as a measurement to the noise level. Decibels is commonly used in acoustic signal analysis to express the relative power level (i.e. loudness) on a logarithmic scale. Given a measured signal s and a reference signal s_0, the decibel is given by:

$$L = 10\, log_{10}\left(\frac{A(s)^2}{A(s_0^2)}\right) = 20\, log_{10}\left(\frac{A(s)}{A(s_0)}\right), \tag{3.1}$$

where A computes the amplitude of a given signal. In our case, we compare the noise δ to the original audio X, and define the following equation as our metric to measure distortion:

$$L(\delta, X) = 20\, log_{10}\left(\frac{A(\delta)}{A(X)}\right). \tag{3.2}$$

Specifically, since the amplitude of the noise is usually relatively small comparing to the original waveform, the decibel value will be negative, and smaller values indicates smaller noises. Conventionally, noises with decibel values less than -15 dB are considered to be quasi-imperceptible.

3.3 Attack Methods

In this section, we will introduce the algorithms we use to generate untargeted and targeted adversarial examples in the embedding space and the scoring space respectively. In each space, we provide two methods for untargeted attack, and two methods for targeted attack.

3.3.1 Embedding Space

To evaluate the performance of **untargeted attacks** in the embedding space, we use the following algorithms:

1. **FGSM**: The original FGSM is designed to compute adversarial noise in the image domain by taking the gradient of the loss function with respect to the input. However in our case, there is no available loss function in the embedding level. Therefore we made a little modification to the original formulation 3.4 in order to perform FGSM in the embedding space. Assuming E represents the mapping between the input space and the embedding space, we have

$$X' = X + \epsilon sign\left(\nabla_X E(X)\right). \tag{3.3}$$

This gives the intuition on how to project the small perturbation added in the input space onto embedding space to distort the embedding and consequently change the classification result.

2. **FGM**: During the experiment, we also compute the perturbation without taking the $sign$ function in the embedding space. We refer this as the *fast gradient method* (FGM). To be explicit, we write

$$X' = X + \epsilon \cdot \nabla_X E(X). \tag{3.4}$$

We provide the detailed explanation and comparison between FGSM and FGM in later in chapter 4.

To evaluate the performance of **targeted attacks** in the embedding space, we use the following algorithms:

1. **Optimization**: we use a optimization-based method modified from C&W's algorithm (later referred as *optimization-based method* or *optim.* in short). Even with dimensionality reduction, the search plane in embedding level is still enormous to look for adversarial example without any external information as direction. Therefore we require a piece of recordings of the target speaker to compute the target embedding as a guidance. Suppose that X is the vectorized input audio of dimension n, and X_t is the audio clip of the target speaker, we generate adversarial noise δ by solving the following optimization process:

$$\text{minimize } \|E(X) - E(X_t)\|_2 + r \cdot \|\delta\|_1 \quad s.t.\ \delta \in \mathbb{R}^n. \tag{3.5}$$

where r is the regularization factor. As this will gradually reduce the distance between the adversarial embedding and the target embedding, and at some point change

the classification result. We also place a penalty on the noise level to keep the attack imperceptible.

2. **BIM**: Similarly, we define the L_2 distance between the adversarial embedding and the target embedding as the loss function, and apply BIM to derive the perturbation in the input space with the following formula:

$$X'_0 = 0, \quad X'_n = Clip_\epsilon \left\{ X_{n-1} - \alpha sign \left(\nabla_X \left\| E(X_{n-1}) - E(X_t) \right\|_2 \right) \right\}. \tag{3.6}$$

3.3.2 Scoring Space

To evaluate the performance of **untargeted attacks** in the scoring space, we use the following algorithms:

1. **End-to-end FGSM**: For the scoring space, we define the loss function to be the cross-entropy loss between the output probability distribution p of the PLDA classifier and the true one-hot label y:

$$J_p(X) = -\sum y_i \cdot log(p_i). \tag{3.7}$$

Equation 3.4 then needs to be modified as:

$$X' = X + \epsilon sign \left(\nabla_X J_p(X) \right). \tag{3.8}$$

Comparing to equation 3.3, this means to tweak the input to directly disturb the output probability distribution and potentially further cause misclassification.

2. **End-to-end FGM**: Remove the $sign$ function in equation 3.8, we have:

$$X' = X + \epsilon \cdot \nabla_X J_p(X). \tag{3.9}$$

To evaluate the performance of **targeted attacks** in the scoring space, we use the following algorithms:

1. **End-to-end optimization**: To compute adversarial noise, we solve the following optimization problem modified from formula 2.5 using the distortion metric provided in equation 3.2:

$$\text{minimize } L(\delta, X) + c \cdot f(X') \quad s.t.\ \delta \in \mathbb{R}^n. \tag{3.10}$$

To make full use of the probability scores produced by the PLDA classifier, we define function f to be

$$f(X') = max\left(max\left\{p_i : i \neq t\right\} - p_t,\ -\kappa\right), \tag{3.11}$$

where p_i is the predicted probability for the ith class, and κ is a pre-chosen constant to represent attack confidence, as larger κ values will usually produce stronger results.

2. **End-to-end BIM**: Combining equation 3.12 and 3.8, we get

$$X'_0 = 0, \quad X'_n = Clip_\epsilon\left\{X_{n-1} - \alpha sign\left(\nabla_X J_p(X_{n-1})\right)\right\}. \tag{3.12}$$

The idea of this process is to gradually modify the output probability distribution to mimic the target label.

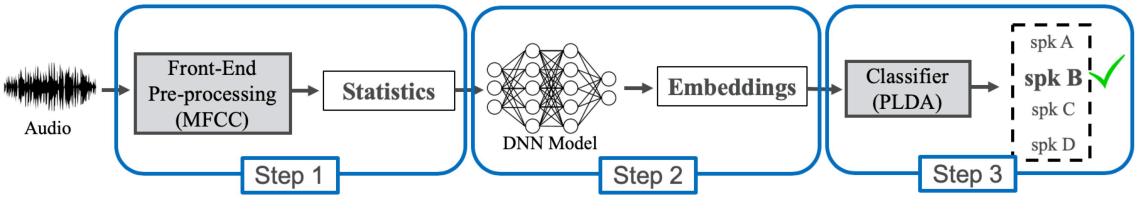

Figure 3.1: Benchmark framework for evaluation

3.4 Evaluation Benchmark

3.4.1 Benchmark Framework

This section describes the architecture of the network used as our benchmark for testing. We choose to used the DNN embedding system proposed in [14, 18] and use PLDA as the classifier which is common in SI tasks [27, 19]. In detail, this framework can be describe as three steps as shown in Figure 3.1.

For the front-end pre-processing, the audio waveform first goes through a energy-based voice activity detection (VAD) of 95 percent to trim the silent segments, and is windowed into frames of 320 samples with an overlapping of 160 samples. Then we take the fast Fourier transform (FFT) and apply filter banks. We keep the first 26 dimensions after the discrete cosine transform (DCT), and append the energy to the features after amplifying the high frequencies. Finally, multiple frame-level features was stacked into a feature vector of 260 dimensions.

The architecture of our DNN embedding model is proposed in [14]. We use input dimensional of 260 to be consistent with the size of feature vectors. The number of outputs is equal to the number of speakers in the development set, which is 105 in our case. With out loss of generality, we choose to use two FC layers to construct the network for computational efficiency, and the output from the last hidden layer is taken as the embedding vector as is suggested in [14]. The model is based on the hypothesis that the DNN will automatically learn to extract representations of the variations of speakers in the development set, and the learned mapping can also be used to distinguish unseen speakers.

3.4.2 Dataset Description

The dataset we use for network training and evaluation is CSTR VCTK Corpus dataset from the University of Edinburgh [28]. This dataset contains speech data uttered by 109 native English speakers with various accents. Each speaker reads out about 400 sentences.

We first divide the dataset into development set containing 105 speakers, which is for embedding network training, and evaluation set containing the rest 4 speakers (labeled as p363, p364, p374 and p376). Then we further divide the evaluation set into enrollment set, which is to enroll the 4 speakers as users of the system, and testing set which is for attack methods testing. The proportion of the enrollment set and the testing set is 4:1. The classification results will be made among these 4 speakers.

CHAPTER 4
EXPERIMENTAL RESULTS AND DISCUSSION

This chapter presents the results and discussion of our experiments. We will first illustrate the results of our untargeted and targeted attacks in the embedding space, and then the results our untargeted and targeted attacks in the scoring space.

4.1 Embedding Level Attack

Untargeted Attacks We use the last 4 speakers in the VCTK Corpus dataset to test our untargeted attacks (labeled as p363, p364, p374, and p376), in total 285 audio files. The baseline system accuracy with clean inputs is 97.19%. Table 4.1 and 4.2 summarize the results. The configuration column shows the attack strength, the systems accuracy column shows the systems accuracy tested with the adversarial inputs, and the attack success rate (ASR) is the difference between system accuracy with clean inputs and the system accuracy with adversarial inputs. As we can see from the results, larger ϵ values produces stronger attacks. With same ϵ values, FGM produces superior results than FGSM. FGSM is original proposed in the image domain, where the input value range are limited. For instance, the value of each pixel ranges within 0-255 under RGB color mode. In which case, taking sign function can prevent the algorithm from producing illegal adversarial inputs. However, the input value of audio files has a much wider range. For example, for a 16 bit audio file, the amplitute can go as high as 32767. Therefore the sign function is not necessary in our case. However, as we can see from the last column in table 4.1 and 4.2, to achieve the same ASR, FGM requires stronger noise to be added. This is because instead of introducing uniform noise as the FGSM does, FGM produces unevenly-distributed noise with larger distortion.

Table 4.1: Results of FGSM attack in the embedding space

Configuration	System Accuracy	Attack Success Rate	Noise Level
$\epsilon = 10^{-1}$	34.38%	62.81%	-3.44 dB
$\epsilon = 10^{-2}$	62.45%	34.74%	-23.44 dB
$\epsilon = 10^{-3}$	94.38%	2.81%	-43.44 dB
$\epsilon = 10^{-4}$	95.43%	1.76%	-63.44 dB

Table 4.2: Results of FGM attack in the embedding space

Configuration	System Accuracy	Attack Success Rate	Noise Level
$\epsilon = 10^{-1}$	24.21%	72.98%	34.13 dB
$\epsilon = 10^{-2}$	38.94%	58.28%	14.13 dB
$\epsilon = 10^{-3}$	52.98%	44.21%	-5.86 dB
$\epsilon = 10^{-4}$	79.64%	17.55%	-25.86 dB

Targeted Attacks To evaluate our targeted attack in the embedding space, we randomly select audio clips from the original speaker and the target speaker. For each combination, we repeat the attack for 20 times with different audios selected each time. We only report success when the classification result matches target speaker. Table 4.3 shows our best results of BIM method with number of steps $= 100$, $\epsilon = 0.01$ and $\alpha = 0.001$. Table 4.4 shows our best results of optimization-based method using the Adam optimizer, with $r = 0.01$, learning rate $= 0.001$ and number of training epochs $= 2000$. For simplicity we refer the speakers with label p363, p364, p374, p376 as speaker A to D. We observe that the ASR differs from different combination of original and target speakers. Both results shows that the attack attempting to change the classification from speaker A to B, or reversely B to A, has a relative high ASR. On the other hand, the attack attempting to change the classification from any speaker to speaker D is harder to success. This is due to the fact that speaker A and B has a similar voice tone which results in a closer distribution in the embedding space and therefore it's easier to flip the prediction. Results shows that BIM produces slightly better performance than the optimization method in the embedding space.

Table 4.5 shows the noise level of our BIM and optimization-based attack. Perturbation generated by both methods are within acceptable range. The average decibel generated by

our optimization based method measures at a very low value (-38.38 dB), which makes our attack hard for human to perceive. Figure 4.1 shows the waveform of a successful attack from speaker A to speaker C, generated by our optimization-based method. From top to bottom are the waveform of the original audio, the adversarial audio, and the noise (red) comparing to the original audio (blue). It's hard to distinct the two waveforms by visual observation, and the noise is also subtle comparing to the original audio. The noise level we measured between the noise and the original audio is -32.45 dB. Figure 4.2 shows the log-power spectrogram of the original audio and adversarial audio. Again, the two diagrams have extremely similar patterns.

Table 4.3: Results of BIM attack in the embedding space

	Target at A	**Target at B**	**Target at C**	**Target at D**	**ASR per spk.**
Spk. A	-	95%	90%	50%	78%
Spk. B	80%	-	100%	40%	73%
Spk. C	70%	10%	-	45%	41%
Spk. D	50%	90%	90%	-	76%
					Overall ASR: 67%

Table 4.4: Results of optim. attack in the embedding space

	Target at A	**Target at B**	**Target at C**	**Target at D**	**ASR per spk.**
Spk. A	-	80%	85%	50%	71%
Spk. B	70%	-	70%	15%	51%
Spk. C	60%	60%	-	55%	58%
Spk. D	50%	50%	75%	-	58%
					Overall ASR: 60%

Table 4.5: Noises levels of targeted attacks in embedding space

	Maximum	**Minimum**	**Average**
BIM	-15.81 dB	-36.75 dB	-26.13 dB
Optim.	-17.88 dB	-72.89 dB	-38.38 dB

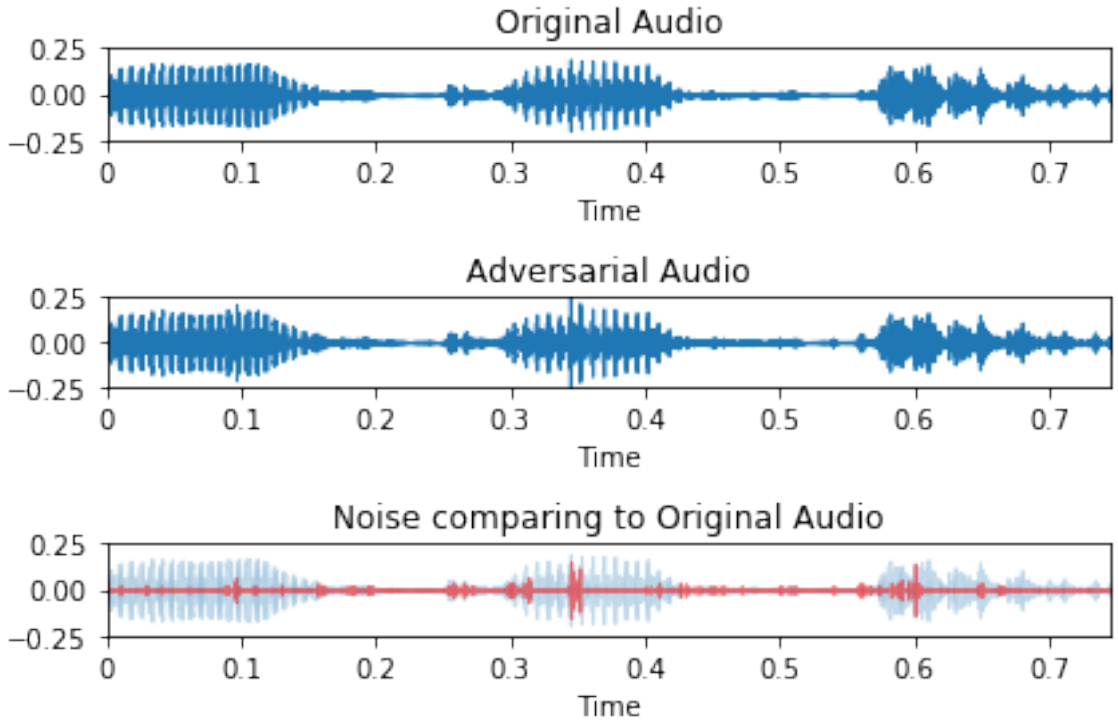

Figure 4.1: Waveform of original audio and adversarial audio

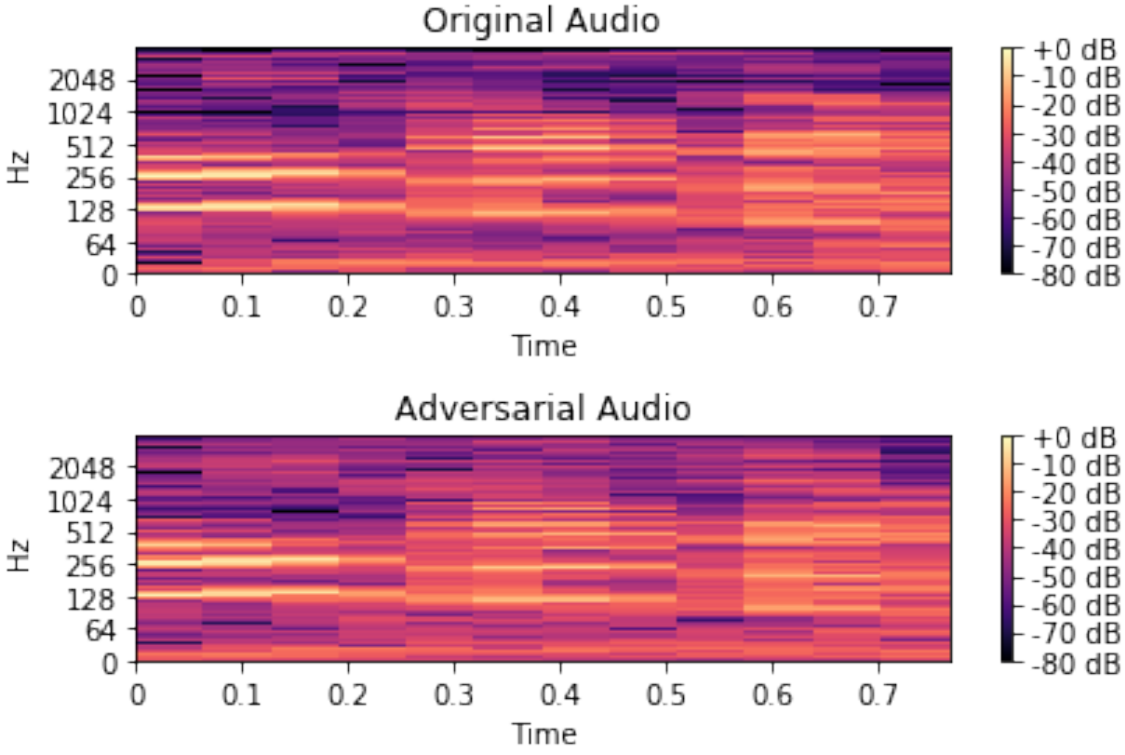

Figure 4.2: Log-power spectrogram of original audio and adversarial audio

4.2 Scoring Level Attack

Untargeted Attacks Table 4.6 and table 4.7 compares the results of the end-to-end FGSM and the end-to-end FGM respectively with different ϵ values. Note that after back-propagating through the PLDA classifier and the embedding model, the gradients values are becoming very small, which is known as the vanishing gradient problem. If we set the ϵ too small, the adversarial perturbation will be insufficient to have any influence on the classification process. Therefore we use a large ϵ number to boost the noise.

The end-to-end FGSM shows superior performance comparing to the embedding level FGSM. When $\epsilon = 10^{-2}$ and noise level at around 24 dB, the ASR of end-to-end FGSM is 87% higher than the embedding level FGSM. This shows that computing FGSM adversarial perturbation directly aiming at the classifier is more efficient. Comparing to embedding level FGM's 17.55% ASR at -25 dB, end-to-end FGM has a better performance of 38.25% ASR at a lower noise level of -35 dB. However, this is still not comparable with the end-to-end FGSM attack.

Table 4.6: Results of FGSM attack in the scoring space

Configuration	System Accuracy	Attack Success Rate	Noise Level
$\epsilon = 10^{-1}$	26.31%	70.88%	-4.85 dB
$\epsilon = 10^{-2}$	31.92%	65.27%	-24.85 dB
$\epsilon = 10^{-3}$	32.98%	64.19%	-44.85 dB
$\epsilon = 10^{-4}$	80.70%	16.47%	-64.85 dB

Table 4.7: Results of FGM attack in the scoring space

Configuration	System Accuracy	Attack Success Rate	Noise Level
$\epsilon = 10^{7}$	44.91%	52.28%	4.18 dB
$\epsilon = 10^{6}$	51.92%	39.27%	-15.81 dB
$\epsilon = 10^{5}$	58.94%	38.25%	-35.81 dB
$\epsilon = 10^{4}$	65.61%	31.58%	-55.81 dB

Targeted Attacks Table 4.9 shows the result of our end-to-end optimization-based attack using the Adam optimizer. The configurations are set as $c = 1$, $\kappa = 1$, learning rate $=$

0.001, and number of epochs $= 3000$. The performance are comparable to the embedding level optimization-based method, which infers that the knowledge of the enrollment model failed to provide any incremental information to support the attack. The pattern of the ASR for pair of original and target speaker aligns with the targeted results produced in the embedding space. The attack between speaker A and speaker B still shows high ASR, and the lowest ASR appears when speaker D is involved. This proves our assumption made in previous section that the voice tone of the speaker place a large impact on the ASR.

Table 4.8 summarizes the results generated by the end-to-end BIM attack, with $\epsilon = 0.01$, $\alpha = 0.001$ and number of steps = 20. It's interesting to find that the end-to-end BIM is able to achieve an overall ASR of 100% while still keeping the distortion under -15 dB, as is shown in table 4.10. We expect the end-to-end optimization-based method to have better results. However, the performance of end-to-end optimization method is related to the choice of different f functions, the value of parameter c and κ, the learning rate and the training epoch number. We believe that by fine-tuning these parameters the end-to-end optimization method can achieve better results, and we will leave it as the future work.

Table 4.8: Results of end-to-end BIM attack in the scoring space

	Target at A	**Target at B**	**Target at C**	**Target at D**	**ASR per spk.**
Spk. A	-	100%	100%	100%	100%
Spk. B	100%	-	100%	100%	100%
Spk. C	100%	100%	-	100%	100%
Spk. D	100%	100%	100%	-	100%
					Overall ASR: 100%

Table 4.9: Results of end-to-end optim. attack in the scoring space

	Target at A	**Target at B**	**Target at C**	**Target at D**	**ASR per spk.**
Spk. A	-	75%	85%	15%	58%
Spk. B	85%	-	70%	0%	51%
Spk. C	75%	80%	-	35%	63%
Spk. D	15%	50%	65%	-	43%
					Overall ASR: 49%

Table 4.10: Noises levels of targeted attacks in scoring space

	Maximum	**Minimum**	**Average**
End-to-end BIM	-19.50 dB	-40.19 dB	-29.78 dB
End-to-end Optim.	-12.71 dB	- 135.57 dB	-34.99 dB

CHAPTER 5
RELATED WORK

This chapter introduces some prior works on generating adversarial examples against automatic speech recognition systems and methods to attack speaker recognition systems, which are related to this thesis.

5.1 Attacks on Automatic Speech Recognition

Previous studies on adversarial attacks on ASR systems have primarily focused on injecting carefully crafted perturbations into speech signals such that the audio is recognized as a specific sequence of phrases, which is desired by the attacker. Cisse [29] generate untargeted and targeted adversarial examples against gradient-based speech recognition models directly using the task loss. Carlini *et al.* [30] use a forwarding and inverse MFC process to remove all feature that is used for comprehension by human listeners but not used for speech recognition to construct a hidden voice command that is correctly interpreted by machines but unintelligible to humans. Yuan *et al.* demonstrates that voice commands can be embedded in songs to construct practical attacks. They test their attack against DNN-based models under realistic settings where the adversarial audio is played over the air and picked up by microphones on victim devices. Another work by Carlini and Wagner [10] extend their method proposed on image tasks and use a L_∞ constrained connectionist temporal classification (CTC) loss function to construct adversarial examples that can be targeted at any combination of phases as desired by the adversary.

5.2 Replay and Synthesis Attacks

Previous works attempt to attack the speaker recognition system by using synthesized or recorded audio clips. Kinnunen *et al.* [31] study the feasibility to fool the speaker recognition system using voice conversion techniques, which is to modify the original speaker's utterances to mimic the voice of another speaker. They use a joint density GMM voice conversion method trained by expectation maximization (EM) algorithm, and use vector quantization (VQ) based method for frame alignment. Their results are tested on different variations of GMM-UBM based speaker recognition systems. Lindburg and Blomberg [32] studies the feasibility of using concatenated recordings and synthesized voice to deceive the speaker recognition system. They use extracted features from legitimate user's recorded audio to re-synthesis the attacker's voice to imitate his. They also use a commercial diphone synthesis system to generate imposter voices. The result shows only concatenated voice sequences succeed to cause a large false accept rate.

On the other hand, there have been a large number of techniques developed for detecting and defending such attacks. Wu *et al.* [33] proposed a defense method utilizing features derived from phrase spectrum to detect conversion-based spoofing attacks and reported to successfully reduce the equal error rate from more than 20% to 2.35%. A recent study [34] trained a five-layer DNN classifier for spoofing detection, and proposed to use human log-likelihood (HLL) as the scoring method. They claimed to successfully reduce the average equal error rate of all attacks to 0.045%. However, it's important to note that none of the mentioned defense methods claim to be effective against adversarial examples. It's worth to mention that training DNN to detect adversarial noises would cause a conundrum as the adversary is also capable of crafting adversarial examples aiming at the DNN classifier.

CHAPTER 6
CONCLUSION AND FUTURE WORK

In this thesis, we demonstrated the vulnerability of speaker recognition systems when facing adversarial examples. We implemented and tested several attacks on a deep neural network based text-independent speaker identification system. Our experiment shows that under such set-up, it is possible for the adversary to utilize gradient-based algorithms to construct untargeted and targeted attack in different scenarios. Although machine learning models have shown improvements on the traditional speaker recognition systems, our results suggest it's still critical to inspect the robustness of such models.

In future work, we would like to further test the feasibility of crafting robust perturbations to be able to propagate through speakers and microphones. We also look forward to demonstrating utilizing the transferability of adversarial examples across different models to achieve black-box attacks. In addition, we hope that an effective defense method against those adversarial attacks will be proposed.

REFERENCES

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

[2] S. R.J. S. Kaiming He Xiangyu Zhang, "Deep residual learning for image recognition," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[3] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[4] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[5] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[6] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[7] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings.," in *EuroSP*, IEEE, 2016, ISBN: 978-1-5090-1752-2.

[8] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: A simple and accurate method to fool deep neural networks.," in *CVPR*, IEEE Computer Society, 2016, pp. 2574–2582, ISBN: 978-1-4673-8851-1.

[9] N. Carlini and D. A. Wagner, "Towards evaluating the robustness of neural networks.," *CoRR*, vol. abs/1608.04644, 2016.

[10] ——, "Audio adversarial examples: Targeted attacks on speech-to-text.," *The International Conference on Machine Learning (ICML)*, 2018.

[11] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech and Language Processing*, 2010.

[12] S. Cumani, O. Plchot, and P. Laface, "Probabilistic linear discriminant analysis of i-vector posterior distributions.," in *ICASSP*, IEEE, 2013, pp. 7644–7648.

[13] S. B. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 28, no. 4, pp. 357–366, 1980.

[14] E. Variani, X. Lei, E. Mcdermott, I. L. Moreno, and J. Gonzalez-dominguez, *Deep neural networks for small footprint text-dependent speaker verification*, 2014.

[15] G. Heigold, I. Moreno, S. Bengio, and N. Shazeer, "End-to-end text-dependent speaker verification.," *CoRR*, vol. abs/1509.08062, 2015.

[16] D. P.S. K. David Snyder Daniel Garcia-Romero, "Deep neural network embeddings for text-independent speaker verification.," *INTERSPEECH*, 2017.

[17] B. J.X.L.X.Z.X.L.Y.C.A.K.Z. Z. Chao Li Xiaokong Ma, "Deep speaker: An end-to-end neural speaker embedding system.," *arXiv*, 2017.

[18] Y. hsin Chen, I. Lopez-Moreno, T. N. Sainath, M. Visontai, R. Alvarez, and C. Parada, "Locally-connected and convolutional neural networks for small footprint speaker recognition.," in *INTERSPEECH*, ISCA, 2015, pp. 1136–1140.

[19] G. S.D.P.S. K. David Snyder Daniel Garcia-Romero, "X-vectors: Robust dnn embeddings for speaker recognition.," *IEEE international conference on acoustics speech and signal processing*, 2018.

[20] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *International Conference on Learning Representations*, 2014.

[21] I. G.S.J.Z.B.C.A. S. Nicolas Papernot Patrick McDaniel, "Practical black-box attacks against machine learning.," *ACM Asia Conference on Computer and Communications Security*, 2017.

[22] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world.," *CoRR*, vol. abs/1607.02533, 2016.

[23] S. N. Uri Shaham Yutaro Yamada, "Understanding adversarial training: Increasing local stability of neural nets through robust optimization.," *arXiv*, 2015.

[24] N. Papernot, P. D. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks.," *CoRR*, 2015.

[25] H. Kannan, A. Kurakin, and I. J. Goodfellow, "Adversarial logit pairing.," *CoRR*, vol. abs/1803.06373, 2018.

[26] T. Brown, D. Mane, A. Roy, M. Abadi, and J. Gilmer, "Adversarial patch," 2017.

[27] D. Snyder, D. Garcia-Romero, and D. Povey, "Time delay deep neural network-based universal background models for speaker recognition.," in *ASRU*, IEEE, 2015, pp. 92–97, ISBN: 978-1-4799-7291-3.

[28] C. Veaux, J. Yamagishi, and K. MacDonald, *CSTR VCTK Corpus : English Multi-speaker Corpus for CSTR Voice Cloning Toolkit*, `https://homepages.inf.ed.ac.uk/jyamagis/page3/page58/page58.html`.

[29] M. Ciss, Y. Adi, N. Neverova, and J. Keshet, "Houdini: Fooling deep structured prediction models.," *CoRR*, vol. abs/1707.05373, 2017.

[30] N. Carlini, P. Mishra, T. Vaidya, Y. Zhang, M. Sherr, C. Shields, D. Wagner, and W. Zhou, "Hidden voice commands," in *25th USENIX Security Symposium (USENIX Security 16), Austin, TX*, 2016.

[31] K. A.L.F.S.E.S.C.H. L. Tomi Kinnunen Zhi-Zheng Wu, "Vulnerability of speaker verification systems against voice conversion spoofing attacks: The case of telephone speech.," *ICASSP*, 2012.

[32] J. Lindberg and M. Blomberg, "Vulnerability in speaker verification - a study of technical impostor techniques.," in *EUROSPEECH*, ISCA, 1999.

[33] Z. Wu, C. E. Siong, and H. Li, "Detecting converted speech and natural speech for anti-spoofing attack in speaker recognition.," in *INTERSPEECH*, ISCA, 2012, pp. 1700–1703.

[34] H. Yu, Z.-H. Tan, Z. Ma, R. Martin, and J. Guo, "Spoofing detection in automatic speaker verification systems using dnn classifiers and dynamic acoustic features.," *IEEE Trans. Neural Netw. Learning Syst.*, vol. 29, no. 10, pp. 4633–4644, 2018.