AdvEst: Adversarial Perturbation Estimation to Classify and Detect Adversarial Attacks against Speaker Identification

Sonal Joshi*, Saurabh Kataria*, Jesús Villalba*, Najim Dehak

Center for Language and Speech Processing, Johns Hopkins University, Baltimore, MD

{sjoshi12,skatari1,jvillal7,ndehak3}@jhu.edu

Abstract

Adversarial attacks pose a severe security threat to the state-of-the-art speaker identification systems, thereby making it vital to propose countermeasures against them. Building on our previous work that used representation learning to classify and detect adversarial attacks, we propose an improvement to it using AdvEst, a method to estimate adversarial perturbation. First, we empirically validate our claim that training representation learning network using adversarial perturbations as opposed to adversarial examples (consisting of a combination of clean signal and adversarial perturbation) is beneficial because it potentially eliminates nuisance information. At inference time, we use a time-domain denoiser to estimate the adversarial perturbations from adversarial examples. Using our improved representation learning approach to obtain attack embeddings (*signatures*), we evaluate their performance for three applications: known attack classification, attack verification, and unknown attack detection. We show that common attacks in the literature (Fast Gradient Sign Method (FGSM), Projected Gradient Descent (PGD), Carlini-Wagner (CW) with different L_p threat models) can be classified with an accuracy of $\sim 96\%$. We also detect unknown attacks with an equal error rate (EER) of $\sim 9\%$, an absolute improvement of $\sim 12\%$ from our previous work.

Index Terms: speaker identification, countermeasures, adversarial attacks, security, defense, speech enhancement

1. Introduction

Adversarial examples are malicious inputs designed to make deep neural networks produce incorrect prediction [1]. They are generated by adding carefully computed perturbation to clean signals (known by the term *benign* in the literature). Mathematically, $\mathbf{x} + \delta = \mathbf{x}'$, where δ denotes the adversarial perturbation[1] being added to the benign signal $\mathbf{x}$ to obtain the adversarial example $\mathbf{x}'$. These adversarial examples are (almost) inaudible to humans and pose a severe security threat to state-of-the-art speaker recognition systems. To further exacerbate the problem, several adversarial attack algorithms have been proposed [2, 3, 4, 5] and new attack algorithms and variants come out frequently. This makes proposing practical countermeasures against these attacks of utmost importance.

Most of the countermeasures in the literature can be primarily classified into two types–proactive and reactive [6, 7]. Proactive countermeasures focus on making the system under attack (a.k.a. the victim network) inherently more robust to adversarial examples. This is usually done by re-training the system model, say by adversarial training or fine-tuning [8]. The limitation of such defenses is that they may be susceptible to adversarial attack algorithms that they are not trained for [9].

---

*equal contribution

[1]For detailed explanation, please refer to Section 2

On the other hand, reactive countermeasures focus on having an additional front-end block that either pre-processes the input or does *denial of service*, say using an adversarial attack detector, that acts as a gate in order to reject any incoming adversarial examples [10, 11, 12]. However, the system becomes less user-friendly if the detector has a high false alarm rate. Hence, it becomes vital to have a system that can accurately classify the types of adversarial attacks and detect the presence of an unknown attack so that the best possible defense can be selected accordingly. An additional benefit of such classification/detection countermeasure is that it may help identify if two attacks are from the same attacker or different. This is particularly important in forensics applications. However, this area of research is still in the nascent stage, with few published works in image and text classification [13, 14].

We focus on attacks on speaker recognition systems, particularly on the state-of-the-art x-vector based system. Our previous work [15] proposed to use embeddings obtained by representation learning of adversarial examples as *attack signatures* to retrieve information about the adversarial attack. This information includes attack algorithm type and threat model[1], Signal to Adversarial noise ratio, etc. could help in knowing about attacker identity and intention. Previous work [15] shows that representation learning networks trained on speaker identification attacks are transferable to speaker verification attacks. Hence, we focus on speaker identification tasks in this work.

In [15], we use signature extractors that take the adversarial signal $\mathbf{x}'$ as input. However, $\mathbf{x}'$ consists of both benign signal $\mathbf{x}$ and adversarial perturbation δ. We argue that, instead of $\mathbf{x}'$, δ should have more relevant information about the attack. We verify this claim–that using adversarial perturbation δ instead of $\mathbf{x}'$ is better for the system–empirically. Note that, at training time, we can generate of many attacks and, therefore, have access to a benign and adversarial example pair $(\mathbf{x}', \mathbf{x})$, it is straightforward to obtain δ by simply subtracting $\mathbf{x}$ from $\mathbf{x}'$. However, at inference time, obtaining $\delta_{\text{inference}}$ is not trivial due to two major challenges. First, we only know the adversarial example $\mathbf{x}'_{\text{inference}}$ and do not have access to $\mathbf{x}_{\text{inference}}$, the corresponding benign signal. Second, although one may argue that this can be estimated using a *simple* speech enhancement/denoising network trained using $(\mathbf{x}', \mathbf{x})$ pairs available for training adversarial examples, with $\mathbf{x}$ being target for the input $\mathbf{x}'$; it is not trivial to obtain a good network that can accurately estimate benign signal $\mathbf{x}_{inference}$ and hence $\delta_{inference}$. This challenge is due to the fact that the current speech enhancement/denoising networks are optimized for environmental noises, which are very different from adversarial perturbations [16]. We propose a solution to both these problems by proposing a novel method using a combination of representation learning and adversarial perturbation estimation to detect and classify adversarial attacks on speaker identification (SID) network. Using our proposed method for adversarial perturba-

tion estimation (AdvEst), we evaluate three applications of the signature extractor as reactive countermeasures to mitigate the threat of adversarial attacks:

- **Task 1 - Known attack classification:** Given an adversarial example, the goal is to classify an attack into one of the known attack classes.

- **Task 2 - Attack verification**: Given two adversarial examples, the goal is to identify if the attack belong to same or different known attack class. This task is similar to speaker verification, but for attacks.

- **Task 3 - Unknown attack detection** : Given an adversarial example, the goal is to identify if they belong to an already known class or a novel class.

We briefly introduce adversarial attacks on speaker identification in Section 2 followed by an outline of the proposed method in Section 3. The experiments and corresponding results with discussions are in Section 4 and 5, while the conclusions are in Section 6.

2. Adversarial Attacks on Speaker Identification

The goal of speaker identification is to predict the identity of the speaker amongst a given pool of known speakers. The speaker identification network $H_\theta(\mathbf{x})$ returns the posterior probabilities for the speaker classes. An adversary attacks the clean signal $\mathbf{x}$ by adding an adversarial perturbation δ that solves the optimization problem,

$$\delta = \arg\max_{\delta' \in \Delta} \ell(H_\theta(\mathbf{x} + \delta'), y) \quad s.t. \quad y \neq y', \quad (1)$$

where, Δ represents the set of allowable perturbations, y is the ground truth speaker label and $\ell(\cdot)$ is the loss function–e.g., categorical cross-entropy. $\mathbf{x}' = \mathbf{x} + \delta$ is the adversarial example and y' is the speaker label predicted from $\mathbf{x}'$.

2.1. Threat Model

To keep the adversarial perturbation imperceptible, we impose constraints on Δ. Most attacks enforce $\|\delta\|_p \leq \epsilon$, where the $\|\cdot\|_p$ denotes L_p norm and ϵ is a scalar, which bounds the strength of the attack. The lower the ϵ, the more imperceptible the attack. The choice of the norm type and bound is called as *threat model*. Common threat models are L_0, L_2 and L_∞.

2.2. Attack Algorithms

The adversarial perturbation δ can be optimized by different algorithms. Common methods are Fast Gradient Sign Method (FGSM) [17], Projected Gradient Descent (PGD) [8], Carlini-Wagner (CW) [18]. For detailed working of these adversarial attack algorithms, the reader is encouraged to read [15] and [7].

2.3. Attack Classification

Attacks can be classified based on threat model $\{L_0, L_2, L_\infty\}$, algorithm $\{FGSM, PGD, CW\}$ or combination of algorithm and threat model $\{PGD\text{-}L_0, PGD\text{-}L_2, PGD\text{-}L_\infty\}$. In this work, we focus on the latter as it is a more challenging task.

3. Proposed Method

3.1. Signature Extractor Network

We used an x-vector style network [19] to extract embeddings, i.e. attack signatures. Henceforth, this network is termed as signature extractor network. The signature extractor was composed of Thin-ResNet34 encoder, statistics pooling and classification head with additive angular margin softmax (AAM-softmax), similar to our previous work [15]. In our previous work, we train the signature extractor directly on adversarial examples $\mathbf{x}'$. However, we claim that the job of signature extractor will be easier if it is given δ instead, as the benign signal does not contain any information related to the adversarial attack type[2]. One issue is that, although we can have δ available for training data $\delta = \mathbf{x}' - \mathbf{x}$, henceforth called oracle adversarial perturbation, it cannot be computed at inference time as $\mathbf{x}$ is not available. Hence, we propose to use an Adversarial Perturbation Estimation (AdvEst) Network to obtain $\hat{\delta} = \mathbf{x}' - \hat{\mathbf{x}}$ where $\hat{\mathbf{x}}$ is the benign signal predicted by AdvEst and $\hat{\delta}$ is the estimated adversarial perturbation.

3.2. Adversarial Perturbation Estimation Network

The Adversarial Perturbation Estimation (AdvEst) network is a denoiser model that is trained to predict the benign signal given the adversarial input, and, then, estimates the perturbation as the difference between the adversarial and predicted benign signals. We require AdvEst model to work in time-domain and able to predict unnatural noises like adversarial perturbation. However, it was not trivial to find denoiser that could estimate adversarial perturbation because most denoiser models and loss functions in speech enhancement literature are geared towards predicting natural environmental noises. We experimented with different deep regression and generative models with a variety of loss functions like Multi-Resolution Short-time Fourier Transform (MRSTFT) loss, L_1 loss, and L_2 loss for known attack classification task. Our findings indicated the superiority of the conditional GAN (CGAN) generative model [20, 21] over conventional regression models. CGAN is a supervised variant of GAN which maps signals from one *domain* to another by learning to sample from the respective conditional distribution. CGAN consists of two networks: generator G (or $G_{\mathcal{A} \to \mathcal{B}}$) and discriminator D (or $D_\mathcal{B}$). D learns to differentiate between the real benign signals and the ones generated by G. $\mathcal{G}$ learns to map adversarial signals (domain $\mathcal{A}$) to their benign counterpart (domain $\mathcal{B}$) by "fooling" $\mathcal{D}$ [21]. Minimax formulation is

$$\min_{\mathcal{G}_{\mathcal{A} \to \mathcal{B}}} \max_{\mathcal{D}_\mathcal{B}} \mathcal{L}_{CGAN}, \quad \text{where} \quad \mathcal{L}_{CGAN} = \mathcal{L}_{adv} + \lambda_{sup} \mathcal{L}_{sup}. \quad (2)$$

The objective function $\mathcal{L}_{CGAN}$ consists of two terms: regression loss ($\mathcal{L}_{sup}$) and adversarial loss ($\mathcal{L}_{adv}$). The hyper-parameter λ_{sup} is the weight for the supervision/regression loss. The adversarial loss $\mathcal{L}_{adv}$ is based on Dual Contrastive Loss (DCL) [22]:

$$\mathcal{L}_{adv} = -\mathbb{E}_{\mathbf{x} \sim P_\mathcal{B}}[\log(1 + \sum_{\mathbf{x}' \sim P_\mathcal{A}} \exp(D(G(\mathbf{x}')) - D(\mathbf{x})))]$$

$$- \mathbb{E}_{\mathbf{x}' \sim P_\mathcal{A}}[\log(1 + \sum_{\mathbf{x} \sim P_\mathcal{B}} \exp(D(G(\mathbf{x}')) - D(\mathbf{x})))], \quad (3)$$

where D returns logit of the posterior probability for the real input. Here, $\mathbf{x}$ and $\mathbf{x}'$ are pre-computed (Sec. 4.2) corresponding pairs of benign and adversarial signals, whose marginal and joint distributions are referred to as $\{P_\mathcal{B}, P_\mathcal{A}\}$ and $P_{\mathcal{B},\mathcal{A}}$ respectively. The supervision loss $\mathcal{L}_{sup}$ is MRSTFT auxiliary loss, whose details which can be found in [23]. The adversarial perturbation in a given adversarial signal $\mathbf{x}'$ is estimated as:

$$\hat{\delta} = \mathbf{x}' - \hat{\mathbf{x}}, \quad \text{where} \quad \hat{\mathbf{x}} = G_{\mathcal{A} \to \mathcal{B}}(\mathbf{x}'). \quad (4)$$

[2]We prove this claim empirically later in Section 5

4. Experimental Setup

4.1. Speaker Identification

All experiments are performed using VoxCeleb2 dataset [24]. For speaker identification task, we need train and test speakers to be the same. Therefore, we used the *VoxCeleb2-dev* set and split it into two parts in the ratio 90%-10% per speaker. The larger set, henceforth called *VoxCeleb2-dev-train* is used to train the victim network, while the smaller set, henceforth called *VoxCeleb2-dev-test*, is used as the test set for the speaker identification task. This experimental setup is the same as our previous work [15]. x-Vector architecture for the victim network was based on a Thin-ResNet34 similar to the ones used in [25, 26], with 256 embedding dimension, additive angular margin of $m = 0.3$. This network in clean/benign condition gives 1.94%, 1.91%, and 3.2% EER for VoxCeleb1 original, entire and hard tasks respectively. We refrain from using a larger x-vector network since a larger network would require more time for generating adversarial attacks.

4.2. Adversarial attack dataset

With the victim network trained, we generate attacks using various algorithms and threat models commonly used in the literature–Fast Gradient Sign Method (FGSM), Iterative FGSM (Iter-FGSM), Projected Gradient Descent (PGD-$L_\infty/L_1/L_2$) and Carlini & Wagner (CW-$L_\infty/L_0/L_2$) attacks. In order to cover the maximum subspace of attacks, we used a wide variety of attack hyper-parameters by sampling from different probability distributions, as done in our previous work [15]. To obtain a balanced set, for every algorithm-threat model class, we randomly sample 25% of utterances in the *VoxCeleb2-dev-train* set and use it for attack generation. If a generated attack is successful in making the speaker identification network predict an incorrect speaker, we add it to our database. In other words, our adversarial attack dataset contains only successful attacks. The generated attack dataset is quite large and has 190k utterances corresponding to *VoxCeleb2-dev-train* and 10.5k corresponding to *VoxCeleb2-dev-test* sets. We use a recipe from the open source toolkit Hyperion to generate these attacks. We split the data into two sets-known attacks and unknown attacks. Unknown attacks set contains the Carlini & Wagner (CW-$L_\infty/L_0/L_2$) attacks, while the remainder are put in the known attacks set. The choice of CW attacks for unknown was considering the fact that they were the most stealth attacks and less common as compared to PGD. For more details, the reader is encouraged to refer to our previous work [15].

4.3. Adversarial Perturbation Estimation Network

We use time-domain architectures for the generator and discriminator for our CGAN based denoiser. For generator, we choose Conv-TasNet (or simply TasNet) [27], which is a Convolutional Neural Network (CNN) consisting of three stages: *encoder*, *separator*, and *decoder*. The *encoder* and *decoder* are single 1-D convolutional layers, while the *separator* stage consists of multiple 1-D CNNs (termed *stacks*) whose individual outputs are combined to produce *masks* which are multiplied element-wise to the input mixture and passed on to the *decoder* stage. We choose the number of stacks as 1, number of layers per stack as 6, and the number of output channels (decoder) as 1. The number of channels in the encoder is 128, kernel size is 16 (1 ms as per sampling frequency of 16 KHz), and stride is 8. The number of input and output channels in separator are 128 and 1024 respectively, and dilation is exponentially increasing

Table 1: *Experimental setup for signatures extractor networks*

Experiment Name Description	Train	Test
Baseline	$\mathbf{x}'$	$\mathbf{x}'$
Train and test using adversarial example		
Oracle Perturbation	δ	δ
Train and test using Oracle Adversarial Perturbation		
Estimated Adversarial Perturbation	$\hat{\delta}$	$\hat{\delta}$
Train and test using Estimated Adversarial Perturbation		
Proposed	δ	$\hat{\delta}$
Train using Oracle Adversarial Perturbation and test using Estimated Adversarial Perturbation		

Table 2: *Classification Accuracy (%) for known attack detection*

System	Accuracy(%)
Baseline	92.23
Oracle Perturbation	96.26
Estimated Perturbation	90.79
Proposed	**96.30**

with a factor of 2. Total number of parameters are 1.6M and receptive field is 511. For more details, refer [27].The discriminator is based on Parallel WaveGAN [23].It is a deep 1-D CNN with 10 convolutional layers with 80 channels, kernel size of 3, stride of 1, and linearly increasing dilation starting from 1. An additional sigmoid layer is appended at the end of the discriminator for added training stability.

CGAN is trained with Alternating Gradient Descent (AGD) algorithm [21] with the generator updating at twice the frequency of the discriminator. λ_{sup} is set to 0.001. For training the denoiser, we use *voxceleb2-dev-train* set explained above. We remove silence from it using energy-based Voice Activity Detection (VAD) [15]. The learning rates of generator and discriminator are 0.0002 and 0.0001 respectively, both decreasing linearly at each step till 1e-8 and the optimizer is Adam [28] with betas=(0.5,0.999). For supervised learning, the noisy counterpart of the clean training data is needed, for which we used the attacks described in Section 4.2. We train two networks –the first using all attacks (which is used for known attack classification) and the second leaving out CW attack (which is used for unknown attack detection experiments).

4.4. Systems and Evaluation Tasks

We evaluate the performance of the four systems using three tasks as described in Section 1–known attack detection, attack verification, unknown attack detection in Table 1. **Baseline** system utilizes adversarial example ($\mathbf{x}'$) at both train and evaluation time, while **Oracle Perturbation** system utilizes oracle adversarial perturbation (δ). Please note that this is an oracle experiment carried out just to validate our claim that adversarial perturbations are better than adversarial examples for training signature extractors. **Estimated Adversarial Perturbation** system uses adversarial perturbation estimated using AdvEst for both train and inference. Lastly, we have the **Proposed** system which is trained using oracle adversarial perturbation (δ) while adversarial perturbation estimated using AdvEst ($\hat{\delta}$) is used at inference time.

5. Results and Discussion

Task 1 - Known attack classification: Table 2 shows the results for known attack classification. The baseline classification

Overall classification accuracy = 92.23%

Ground truth class \ Predicted class	benign	cw-l0	cw-l2	cw-linf	fgsm	iter-fgsm	pgd-l1	pgd-l2	pgd-linf
benign	98.80	0.30	0.50	0.40	0.00	0.00	0.00	0.00	0.00
cw-l0	1.50	98.10	0.20	0.20	0.10	0.00	0.00	0.00	0.00
cw-l2	3.60	0.10	81.80	12.20	0.10	0.50	0.20	0.20	1.40
cw-linf	1.10	0.00	3.70	94.60	0.00	0.00	0.00	0.00	0.60
fgsm	2.20	0.10	1.00	0.10	94.40	0.30	0.30	1.00	0.40
iter-fgsm	1.50	0.00	3.20	0.10	2.80	85.10	0.00	0.40	6.80
pgd-l1	1.20	0.00	1.20	0.10	0.20	0.00	88.00	9.10	0.10
pgd-l2	1.70	0.00	2.10	0.00	0.80	0.70	41.40	47.40	6.00
pgd-linf	1.30	0.00	9.30	1.30	0.00	3.40	0.00	2.40	82.20

(a) Baseline.

Overall classification accuracy = 96.30%

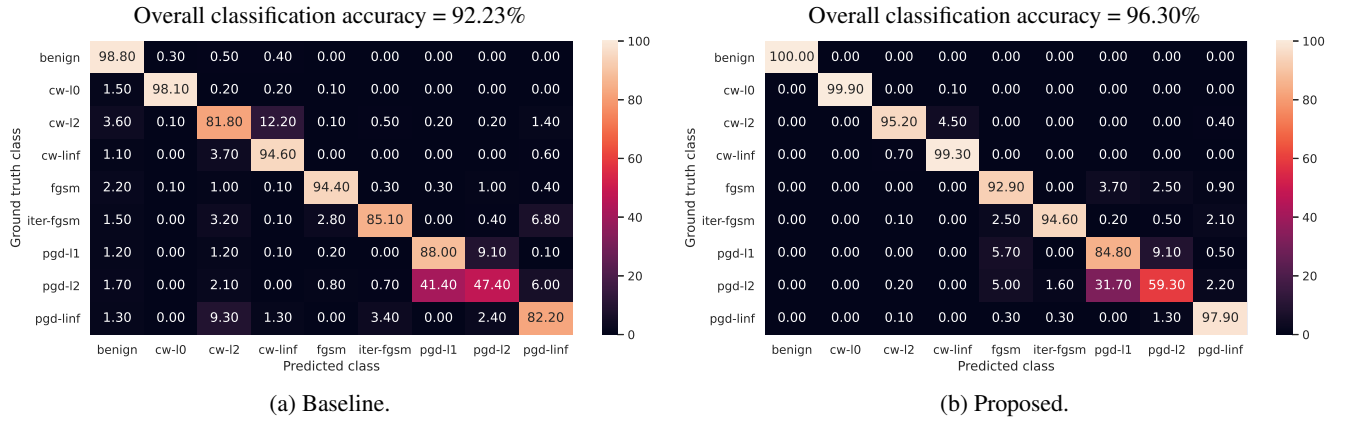| Ground truth class \ Predicted class	benign	cw-l0	cw-l2	cw-linf	fgsm	iter-fgsm	pgd-l1	pgd-l2	pgd-linf
benign	100.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
cw-l0	0.00	99.90	0.00	0.10	0.00	0.00	0.00	0.00	0.00
cw-l2	0.00	0.00	95.20	4.50	0.00	0.00	0.00	0.00	0.40
cw-linf	0.00	0.00	0.70	99.30	0.00	0.00	0.00	0.00	0.00
fgsm	0.00	0.00	0.00	0.00	92.90	0.00	3.70	2.50	0.90
iter-fgsm	0.00	0.00	0.10	0.00	2.50	94.60	0.20	0.50	2.10
pgd-l1	0.00	0.00	0.00	0.00	5.70	0.00	84.80	9.10	0.50
pgd-l2	0.00	0.00	0.20	0.00	5.00	1.60	31.70	59.30	2.20
pgd-linf	0.00	0.00	0.10	0.00	0.30	0.30	0.00	1.30	97.90

(b) Proposed.

Figure 1: *Confusion matrices for known attack classification using different methods.*

Table 3: *EER (%) for attack verification. The best results are in* **bold** *while the second-best results are* underlined

System	Attack group EER(%)		
	Known	**Unknown**	**Known +Unknown**
Baseline	8.88	36.53	20.23
Oracle Perturbation	**5.03**	**26.17**	<u>14.82</u>
Estimated Perturbation	<u>5.14</u>	40.77	21.66
Proposed	<u>5.14</u>	<u>29.01</u>	**14.57**

Table 4: *EER (%) for unknown attack detection. Benign, FGSM and PGD are known attacks, CW attacks are unknown.*

System	With benign	Without benign
Baseline	42.86	21.36
Oracle Perturbation	39.43	17.04
Estimated Perturbation	41.45	20.75
Proposed	**37.99**	**9.06**

accuracy is 92.23%, while using Oracle Perturbation (δ) improves it to 96.26%. This suggests that removing the benign signal makes the job of signature extractor easier. Using Estimated Adversarial Perturbation, we obtain classification accuracy of 90.79%. This suggests that AdvEst network predicted adversarial perturbation is worse than oracle. We propose to mitigate this by oracle perturbation while training and using estimated perturbation at inference time. The proposed method yields an overall classification accuracy is 96.30% which is close to our oracle experiment. Comparing the confusion matrices between baseline and proposed in Figure 1(a) and (b), we observe that the accuracies along diagonal improve for all attacks except PGD-L_1. Although, we still have problems distinguishing between PGD-L_1 and PGD-L_2, the percentage of PGD-L_2 attacks classified as PGD-L_1 is significantly reduced. Overall, all attacks except PGD-L_1, L_2 are better than 92%. Absolute relative improvement range is between -1.5% to 15.7%.

Task 2 - Attack verification: We consider three cases–known attacks only, unknown attacks only and combination of known and unknown attacks. Table 3 shows the results for the three conditions using equal error rate (EER) (%) as the metric. Oracle Perturbation method clearly outperforms the baseline method in all three conditions. The Estimated Perturbation method is almost on-par for Known attacks condition, however is quite worse than Oracle Perturbation method when the conditions have unknown attacks. The Proposed method performs the best for these conditions. Considering that Oracle Perturbation method is an impractical method (exact adversarial perturbation cannot be computed), the proposed method is the best method for all conditions.

Task 3 - Unknown attack detection: Our previous work showed that unknown attacks are easily confused with benign, and hence we consider two cases: with and without benign class. For *without benign* class, we assume that we have an oracle detector telling that system the under attack and the goal of

the detector is to just tell if the attack is known or unknown. We use equal error rate (EER)% as metric, and the results are shown in Table 4. We can see that using Oracle Perturbation is better than using the adversarial example Baseline. Using Estimated Perturbation, we are not able to train the signature extractor as well as with Oracle Perturbation, and this is indicated by a deteriorated performance (EER increases by $\sim 4\%$). The Proposed method yields 9% EER. We speculate that this might be due to the widely known limitation of deep neural network-based denoisers–that it achieves good inference-time performance for noise seen during training, but not so good performance for unseen noises [29], however this requires more thorough studies. This (undesirable) property seems to be helping with unknown attack detection. We speculate that the denoiser maps the unknown attacks to a different subspace than known attacks, thus making the task of unknown attack detection easier. We are currently investigating this end.

6. Conclusions

We verify the hypothesis that using just the adversarial perturbation makes the job of learning signatures easier. At inference time, we use AdvEst, an adversarial perturbation estimation network, to help to remove the clean signal from the adversarial example. Using our proposed approach, we show that common attacks in the literature can be classified with accuracy as high as $\sim$96%, while unknown attacks can be detected with an equal error rate (EER) of $\sim$9%. We are currently performing additional experiments to investigate the reason for performance improvement using AdvEst for unseen attacks.

7. Acknowledgements

8. References

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

[2] F. Kreuk, Y. Adi, M. Cisse, and J. Keshet, "Fooling End-To-End Speaker Verification With Adversarial Examples," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 1962–1966.

[3] Y. Xie, C. Shi, Z. Li, J. Liu, Y. Chen, and B. Yuan, "Real-Time, Universal, and Robust Adversarial Attacks Against Speaker Recognition Systems," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 1738–1742.

[4] Q. Wang, P. Guo, and L. Xie, "Inaudible adversarial perturbations for targeted attack in speaker recognition," in *Interspeech*, 2020, pp. 4228–4232.

[5] H. Abdullah, K. Warren, V. Bindschaedler, N. Papernot, and P. Traynor, "SoK: The Faults in our ASRs: An Overview of Attacks against Automatic Speech Recognition and Speaker Identification Systems," *IEEE Symposium on Security and Privacy*, pp. 730–747, 2021.

[6] D. Wang, R. Wang, L. Dong, D. Yan, X. Zhang, and Y. Gong, "Adversarial examples attack and countermeasure for speech recognition system: A survey," in *International Conference on Security and Privacy in Digital Economy*, 2020, pp. 443–468.

[7] S. Joshi, J. Villalba, P. Żelasko, L. Moro-Velázquez, and N. Dehak, "Study of pre-processing defenses against adversarial attacks on state-of-the-art speaker recognition systems," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 4811–4826, 2021.

[8] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *International Conference on Learning Representations (ICLR)*, 2018.

[9] H. Zhang, H. Chen, Z. Song, D. Boning, I. S. Dhillon, and C.-J. Hsieh, "The limitations of adversarial training and the blind-spot attack," *International Conference on Learning Representations*, 2019.

[10] K. Rajaratnam and J. Kalita, "Noise flooding for detecting audio adversarial examples against automatic speech recognition," in *IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, 2018, pp. 197–201.

[11] X. Li, N. Li, J. Zhong, X. Wu, X. Liu, D. Su, D. Yu, and H. Meng, "Investigating robustness of adversarial samples detection for automatic speaker verification," in *Interspeech*, 2020, pp. 1540–1544.

[12] Z. Yang, B. Li, P.-Y. Chen, and D. Song, "Characterizing audio adversarial examples using temporal dependency," in *International Conference on Learning Representations (ICLR)*, 2019.

[13] M. Moayeri and S. Feizi, "Sample efficient detection and classification of adversarial attacks via self-supervised embeddings," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 7677–7686.

[14] Z. Xie, J. Brophy, A. Noack, W. You, K. Asthana, C. Perkins, S. Reis, S. Singh, and D. Lowd, "Identifying adversarial attacks on text classifiers," *arXiv preprint arXiv:2201.08555*, 2022.

[15] J. Villalba, S. Joshi, P. Żelasko, and N. Dehak, "Representation learning to classify and detect adversarial attacks against speaker and speech recognition systems," in *Interspeech*, 2021, pp. 4304–4308.

[16] A. Shamir, O. Melamed, and O. BenShmuel, "The dimpled manifold model of adversarial examples in machine learning," *arXiv preprint arXiv:2106.10151*, 2021.

[17] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and Harnessing Adversarial Examples," in *International Conference on Learning Representations (ICLR)*, 2015.

[18] N. Carlini and D. Wagner, "Towards Evaluating the Robustness of Neural Networks," in *IEEE Symposium on Security and Privacy*, 2017, pp. 39–57.

[19] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-Vectors : Robust DNN Embeddings for Speaker Recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 5329–5333.

[20] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[21] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[22] N. Yu, G. Liu, A. Dundar, A. Tao, B. Catanzaro, L. S. Davis, and M. Fritz, "Dual contrastive loss and attention for gans," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 6731–6742.

[23] R. Yamamoto, E. Song, and J.-M. Kim, "Parallel wavegan: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6199–6203.

[24] A. Nagrani, J. S. Chung, W. Xie, and A. Zisserman, "Voxceleb: Large-scale speaker verification in the wild," *Computer Speech and Language*, vol. 60, 2020.

[25] H. Zeinali, S. Wang, A. Silnova, P. Matějka, and O. Plchot, "But system description to voxceleb speaker recognition challenge 2019," *arXiv preprint arXiv:1910.12592*, 2019.

[26] J. Villalba, Y. Zhang, and N. Dehak, "x-Vectors Meet Adversarial Attacks : Benchmarking Adversarial Robustness in Speaker Verification," in *Interspeech*, Shanghai, China, 2020.

[27] Y. Luo and N. Mesgarani, "Conv-tasnet: Surpassing ideal time–frequency magnitude masking for speech separation," *IEEE/ACM transactions on audio, speech, and language processing*, vol. 27, no. 8, pp. 1256–1266, 2019.

[28] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[29] S. Joshi, A. Panda, and B. Das, "Enhanced denoising autoencoder for robust speech recognition in unseen noise conditions," in *2018 11th International Symposium on Chinese Spoken Language Processing (ISCSLP)*. IEEE, 2018, pp. 359–363.