Investigating Robustness of Adversarial Samples Detection for Automatic Speaker Verification

Xu Li[1], Na Li[2], Jinghua Zhong[3], Xixin Wu[4], Xunying Liu[1], Dan Su[2], Dong Yu[2], Helen Meng[1]

[1]The Chinese University of Hong Kong, Hong Kong SAR, China
[2]Tencent AI Lab, Tencent, Shenzhen, China
[3]SpeechX limited, Shenzhen, China
[4]Department of Engineering, University of Cambridge, UK

{xuli, wuxx, xyliu, hmmeng}@se.cuhk.edu.hk, lina011779@126.com, jhzhong@speechx.cn, {dansu, dyu}@tencent.com

Abstract

Recently adversarial attacks on automatic speaker verification (ASV) systems attracted widespread attention as they pose severe threats to ASV systems. However, methods to defend against such attacks are limited. Existing approaches mainly focus on retraining ASV systems with adversarial data augmentation. Also, countermeasure robustness against different attack settings are insufficiently investigated. Orthogonal to prior approaches, this work proposes to defend ASV systems against adversarial attacks with a separate detection network, rather than augmenting adversarial data into ASV training. A VGG-like binary classification detector is introduced and demonstrated to be effective on detecting adversarial samples. To investigate detector robustness in a realistic defense scenario where unseen attack settings may exist, we analyze various kinds of unseen attack settings' impact and observe that the detector is robust (6.27% EER_{det} degradation in the worst case) against unseen substitute ASV systems, but it has weak robustness (50.37% EER_{det} degradation in the worst case) against unseen perturbation methods. The weak robustness against unseen perturbation methods shows a direction for developing stronger countermeasures.

Index Terms: speaker verification, anti-spoofing countermeasures, adversarial attack, adversarial samples detection

1. Introduction

Automatic speaker verification (ASV) systems aim at confirming a claimed speaker identity against a spoken utterance. It has been widely applied into commercial devices and authorization tools. However, recent studies have shown that a well-trained ASV system could be deceived by malicious attacks [1–3]. In the last decade, the speaker verification community held several ASVspoof challenge competitions [4–6] to develop countermeasures mainly against replay [7,8], speech synthesis [9,10] and voice conversion [10, 11] attacks.

Very recently, another threat, named adversarial attacks, has been explored on ASV systems. Adversarial attacks slightly perturb the input so that the system will make incorrect decisions. Kreuk et al. [12] added adversarial perturbations into a testing utterance to attack an end-to-end ASV system. The attack was verified to be successful in both cross-feature and cross-corpus settings. Li et al. [13] extended the studies into other ASV frameworks and observed the adversarial transferability from one ASV to attack another ASV. Also some works explored adversarial attacks in practical real-time scenarios [14–16] and attacks on spoofing countermeasures [17].

Apart from the effective perturbations that pose severe threats on ASV systems, the perturbation variations caused by different attack settings also bring difficulty in developing defense approaches. In a realistic attack, different substitute ASV systems can be used to craft adversarial samples and perform effective attack on the target ASV system in a transferable way [13]. The choice of a substitute ASV system, as one of attack settings, results in different perturbation patterns. Besides, perturbation patterns also vary greatly across perturbation methods [18] with different perturbation configurations, e.g. perturbation degrees. So countermeasure robustness against different attack settings, including substitute ASV systems, perturbation methods along with perturbation configurations, is another important concern.

Defense approaches against adversarial attacks have been investigated mostly in the image domain [19–21]. Defense approaches explored in ASV area are still very limited. Wang et al. [22] leveraged adversarial samples into training an end-to-end ASV as a regularization to improve system robustness. Wu et al. [23] adopted a combination of spatial smoothing [20] and adversarial training [24] to strengthen countermeasures against adversarial samples. Both methods are found to be effective. However, they need to retrain a well-developed ASV system with adversarial data augmentation. To the best of our knowledge, no existing work investigates countermeasure robustness against different attack settings of spoofing ASV systems.

Inspired by [21, 25], this work makes the first attempt to defend ASV systems against adversarial attacks with a separate detection network. A VGG-like [26] binary classification system is introduced to capture the difference between adversarial and genuine samples, and predict whether an input is adversarial or not. A separate detection countermeasure has the following advantages: 1) It separates the defense part and speaker verification into two independent stages, which avoids retraining a well-developed ASV model. 2) Since most existing countermeasures for replay and synthetic speech attacks are based on a separate detection network [7–9], the proposed approach provides the feasibility to develop a unified countermeasure against all spoofing attacks.

In a realistic defense scenario, attack settings cannot be accessed by the defender so that the proposed detector can be degraded by unseen attack settings. To investigate detector ro-

This work was done when Xu Li was an intern at Tencent AI Lab.

bustness in such a realistic scenario and provide directions for developing stronger countermeasures, this work also gives a robustness discussion based on unseen attack settings, including substitute ASV systems, perturbation methods and perturbation degrees. In this work, the three most representative ASV frameworks are used as variations: Gaussian mixture model (GMM) i-vector system [27], time delay neural network (TDNN) x-vector system [28] and ResNet-34 r-vector system [29]. Two of the most effective perturbation methods, i.e. basic iterative method (BIM) [18] and Jacobian-based saliency map approach (JSMA) [30], are applied to generate adversarial samples.

The contributions of this work include: 1) Design of a dedicated defense network against adversarial attacks, rather than augmenting adversarial samples into ASV training; 2) Introduction of a VGG-like network and demonstrating its effectiveness on detecting adversarial samples; 3) Investigation of detector robustness against unseen attack settings to uncover vulnerability and lack of robustness against unseen perturbation methods, which provides directions for developing stronger countermeasures.

The remaining of this paper is organized as follows: Section 2 details the process of adversarial samples generation. The proposed adversarial samples detection network is illustrated in Section 3. Section 4 analyzes the experiment results. Finally, Section 5 summaries this work.

2. Adversarial Samples Generation

In a speaker verification task, given acoustic features of the enrollment utterance $\boldsymbol{X^{(e)}}$ and testing utterance $\boldsymbol{X^{(t)}}$, a well-trained system function S with parameters $\boldsymbol{\theta}$ will predict a similarity score, which indicates speaker similarity between the enrollment and testing utterances. In a realistic scenario, the owner's enrollment utterance $\boldsymbol{X^{(e)}}$ is implicitly embedded within the ASV system, while $\boldsymbol{X^{(t)}}$ is provided by the customer for identity confirmation. From an adversary's perspective, it will optimize a perturbation $\boldsymbol{\delta_X}$ to be added on $\boldsymbol{X^{(t)}}$ so that the system will behave incorrectly: either falsely rejecting the true target's voice or falsely accepting the imposter's voice. The optimization problem can be formulated as Eq. 1 and 2:

$$\boldsymbol{\delta_X} = \arg \max_{\|\boldsymbol{\delta_X}\|_p \leq \epsilon} k \times S_{\boldsymbol{\theta}}(\boldsymbol{X^{(e)}}, \boldsymbol{X^{(t)}} + \boldsymbol{\delta_X}) \quad (1)$$

$$k = \begin{cases} -1, & \text{target trial} \\ 1, & \text{non-target trial} \end{cases} \quad (2)$$

where the constraint p-norm of $\boldsymbol{\delta_X}$ within perturbation degree ϵ guarantees a subtle perturbation so that human cannot perceive the difference between adversarial and genuine samples.

We leverage three different ASV system architectures and two perturbation methods in our experiments. The details for ASV systems and perturbation methods are illustrated in Sections 2.1 and 2.2, respectively.

2.1. ASV systems

Three ASV systems involved are as follows: GMM i-vector with probabilistic linear discriminant analysis (PLDA) back-end [27], TDNN x-vector with PLDA back-end [28] and ResNet-34 r-vector with cosine back-end [29]. All systems adopt cepstral frequency with configurations in [13] as input.

The i-vector system [27] consists of 2048 mixtures with full covariance matrix. T matrix projects utterance statistics into a 400-dimension i-vector space. The i-vectors are centered and length-normalized before PLDA modeling.

The x-vector system is configured as [28], except that additive angular margin (AAM)-softmax loss [31] with hyper-parameters $\{m = 0.3, s = 32\}$ is used for training. Extracted x-vectors are centered and projected by a 200-dimension LDA, then length-normalized before PLDA modeling.

The r-vector system has the same architecture as [29], and AAM-softmax loss [31] with hyper-parameters $\{m = 0.2, s = 30\}$ is used for training networks. Extracted r-vectors are centered and length-normalized before cosine scoring.

2.2. Perturbation methods

BIM perturbs the genuine input $\boldsymbol{X^{(t)}}$ towards the gradient of the objective w.r.t. $\boldsymbol{X^{(t)}}$ in a multiple-step manner. It optimizes the perturbation with the norm constraint parameter p in Eq. 1 being ∞. Starting from the genuine input $\boldsymbol{X_0^{(t)}} = \boldsymbol{X^{(t)}}$, the input is perturbed iteratively as follows:

$$\boldsymbol{X_{n+1}^{(t)}} = clip_{\boldsymbol{X^{(t)}},\epsilon}(\boldsymbol{X_n^{(t)}} + \alpha sign(\nabla_{\boldsymbol{X_n^{(t)}}} S_{\boldsymbol{\theta}}(\boldsymbol{X^{(e)}}, \boldsymbol{X_n^{(t)}}))), \quad \text{for } n = 0, ..., N-1 \quad (3)$$

where $sign$ is a function that takes the sign of the gradient, α absorbs the trial indicator k in Eq. 2 and its absolute value is the step size, N is the number of iterations and $clip_{\boldsymbol{X^{(t)}},\epsilon}(\boldsymbol{X})$ holds the norm constraints by applying element-wise clipping such that $\|\boldsymbol{X} - \boldsymbol{X^{(t)}}\|_\infty \leq \epsilon$. In our experiments, N is set as 5, and α is set as perturbation degree devided by N.

JSMA is another effective perturbation method to craft adversarial samples. Unlike BIM that adds perturbations to the whole input, JSMA perturbs only one bit at a time. In each iteration, it selects the bit with the most significant effects on output to be perturbed. With this purpose, a saliency score is calculated for each bit and bit with the highest score is chosen to be perturbed. We formulate the algorithm specialized in our case, as shown in Algorithm 1. The $saliency\_map$ at Step 4 computes the absolute value of gradient $\boldsymbol{G}$ while masking out the bits already reach the constraint boundary: $saliency\_map(\boldsymbol{G}, \boldsymbol{\Gamma}) = \boldsymbol{G}^{abs} \odot \boldsymbol{\Gamma}$, where $\boldsymbol{G}^{abs}$ is the element-wise absolute value of $\boldsymbol{G}$ and $\odot$ is an element-wise product operator. In this work, N is set as 300 iterations, and α is set as half of the perturbation degree.

Algorithm 1 JSMA perturbation method
$\boldsymbol{X^{(e)}}$ and $\boldsymbol{X^{(t)}}$ are acoustic features of enrollment and testing utterances, respectively. $S_{\boldsymbol{\theta}}$ is the system function with parameters, α is the step size, ϵ is the perturbation degree, and N is the number of iterations. $\boldsymbol{\Gamma}$ is a mask matrix having the same size with $\boldsymbol{X^{(t)}}$, initialized with all-one element matrix $\boldsymbol{E}$.

Input: $\boldsymbol{X^{(e)}}$, $\boldsymbol{X^{(t)}}$, $S_{\boldsymbol{\theta}}$, α, ϵ, N
1: $\boldsymbol{X_{adv}^{(t)}} = \boldsymbol{X^{(t)}}$, $\boldsymbol{\Gamma} = \boldsymbol{E}$, $\boldsymbol{\delta_X} = \boldsymbol{0}$
2: **for** $i \in [1, N]$ **do**
3: $\quad \boldsymbol{G} = \nabla_{\boldsymbol{X_{adv}^{(t)}}} S_{\boldsymbol{\theta}}(\boldsymbol{X^{(e)}}, \boldsymbol{X_{adv}^{(t)}})$
4: $\quad \boldsymbol{M} = saliency\_map(\boldsymbol{G}, \boldsymbol{\Gamma})$
5: $\quad k_{max} = \arg\max_k \boldsymbol{M}_k$
6: $\quad \boldsymbol{\delta_X}[k_{max}] = clip_{\boldsymbol{0},\epsilon}(\boldsymbol{\delta_X}[k_{max}] + \alpha \times sign(\boldsymbol{G}_{k_{max}}))$
7: $\quad$ **if** $|\boldsymbol{\delta_X}[k_{max}]| \geq \epsilon$ **then**
8: $\quad\quad \boldsymbol{\Gamma}_{k_{max}} = 0$
9: $\quad$ **end if**
10: $\quad \boldsymbol{X_{adv}^{(t)}} = \boldsymbol{X^{(t)}} + \boldsymbol{\delta_X}$
11: **end for**
12: **return** $\boldsymbol{X_{adv}^{(t)}}$

2.3. Dataset generation

Our experiments are conducted on the Voxceleb1 [32] dataset, which consists of short clips of human speech. There are in total 148,642 utterances from 1,251 speakers. Following data partitioning in [32], 148,642 utterances from 1211 speakers are used to train the ASV systems, and the remaining 4,874 utterances from 40 speakers are used for testing the systems and generating adversarial samples. The corpus [32] provides totally 37,720 trials consisting of enrollment-testing utterance pairs selected from the testing utterances.

In this work, we generate adversarial samples according to the attack configuration, including the substitute ASV system, perturbation method and perturbation degree. To make a balanced dataset, for each genuine utterance, we randomly select one trial where that utterance is used to generate an adversarial counterpart. There are around 9K utterances in such an "adversary-genuineness" dataset, including around 4.5K adversarial and 4.5K genuine utterances.

For each specific attack configuration, we generate one "adversary-genuineness" dataset for training and evaluating our detection network. We separate the "adversary-genuineness" dataset into training and testing subsets, with 30 speaker's data for training and 10 speaker's data for testing. The speaker partitioning for training and testing is consistent among all attack configurations. This guarantees that source utterances (either a genuine utterance or an adversarial utterance generated from it) in the testing subsets cannot be observed during training.

3. Adversarial Samples Detection

In this section, we present our proposed system to detect adversarial samples. One possible feature engineering is to use the same features adopted by the protected ASV system. This provides high resistance to the most severe attack scenario where the attacker can access the whole parameters of ASV and directly add perturbations on the features adopted by ASV. With this consideration, we adopt Mel-frequency cepstral coefficients (MFCCs) as the input feature forwarded to our detection network. A pre-emphasis with coefficient of 0.97 is adopted. 25ms "Hamming" window with step-size of 10ms is applied to extract a frame, and finally 24 cepstral coefficients are kept.

We notice some issues about adversarial samples: 1) The deviation between adversarial and genuine samples is subtle and localized on feature maps, and we shall adopt convolutional layers at bottom to effectively capture such deviations; 2) The adversarial characteristics exist in the whole utterance, so a pooling layer can be adopted to aggregate the utterance statistics for decision. Based on these considerations, we introduce a VGG-like network structure [26] to detect adversarial samples. The detailed architecture configurations are illustrated in Table 1. 4 convolutional layers at bottom to capture local feature patterns. A statistics pooling layer aggregates the mean and deviation from the last convolutional layer outputs, and forwards them to dense layers. Finally, 2 dense layers project statistics into a 2-dimensional output space for decision. The network is trained with the Adam [33] optimizer, along with the initial learning rate as 0.001.

Table 1: *Detailed configurations of the proposed detector.*

Layer	Structure	Activation
Conv2D	$[2 \times 2, 64] \times 4$	ReLU
Statistics Pooling	-	-
Flatten	-	-
Dense1	512, dropout 0.2	ReLU
Dense2	512, dropout 0.2	ReLU
Output	2	Softmax

4. Experiments

4.1. Evaluation metrics

To verify the effectiveness of adversarial attacks, we evaluate the ASV system performance before and under adversarial attacks in terms of equal error rate (EER) and minimum detection cost function with target trial prior to be 0.01 and 0.001, i.e. $DCF_{0.01}$ and $DCF_{0.001}$. When evaluating the detector performance, we report the detection accuracy (DA) over the "adversary-genuineness" testing subset. Also, regardless of the operating point, we use the detector's log softmax output at the adversarial bit as the adversarial score, and compute an EER (EER_{det}) over the testing subset.

4.2. Adversarial attack performance

The attack results on the x-vector system are shown in Table 2. The results on the i-vector and r-vector systems have similar trends. From Table 2, we observe that the ASV system performance seriously drops when being attacked by both perturbation methods. Also, the attack effectiveness increases as the perturbation degree increases. However, the perturbations with a higher degree are easier to be detected, which will be discussed in Section 4.3. This suggests a trade-off for attackers to design an effective but cannot be easily detected perturbations.

Table 2: *The x-vector system performance under different attack configurations.*

		EER (%)	$DCF_{0.01}$	$DCF_{0.001}$
genuine		5.97	0.515	0.695
BIM	$\epsilon = 0.3$	39.87	0.995	0.996
	$\epsilon = 1.0$	95.02	1	1
	$\epsilon = 2.0$	99.96	1	1
JSMA	$\epsilon = 1.0$	20.41	0.880	0.932
	$\epsilon = 3.0$	48.28	0.995	0.995
	$\epsilon = 5.0$	60.22	1	1

4.3. Robustness against perturbation degree

In this section, we discuss detector robustness against perturbation degree. Adversarial samples crafted from the x-vector

Table 3: *Detection accuracy (%) against perturbation degrees*

BIM-xvec		training		
		$\epsilon = 0.3$	$\epsilon = 1.0$	$\epsilon = 2.0$
evaluation	$\epsilon = 0.3$	99.83	48.65	48.61
	$\epsilon = 1.0$	99.82	100.00	87.01
	$\epsilon = 2.0$	99.83	100.00	100.00
JSMA-xvec		training		
		$\epsilon = 1.0$	$\epsilon = 3.0$	$\epsilon = 5.0$
evaluation	$\epsilon = 1.0$	99.44	59.84	48.61
	$\epsilon = 3.0$	99.83	100.00	98.41
	$\epsilon = 5.0$	99.83	100.00	100.00

Table 4: *Detector performance against ASV systems*

DA (%)		training		
		BIM-ivec	BIM-xvec	BIM-rvec
evaluation	BIM-ivec	99.87	99.78	99.44
	BIM-xvec	99.65	99.83	99.39
	BIM-rvec	72.45	76.38	99.70
EER_{det} (%)		training		
		BIM-ivec	BIM-xvec	BIM-rvec
evaluation	BIM-ivec	0	0.18	0.55
	BIM-xvec	0.46	0.18	0.65
	BIM-rvec	6.27	5.90	0.28

system along with BIM and JSMA perturbation methods are involved. The system detection accuracy (DA) under different conditions is shown in Table 3. The diagonal results are based on in-domain evaluation, which reflects our proposed detection network is effective and can distinguish the adversarial and genuine data with an accuracy over 99%. It is also observed that the detector can generalize well from adversarial samples with a small perturbation to a larger perturbation. However, the performance drops greatly in the reverse direction. This indicates that we should craft small perturbations to develop our detector, so that it could defend ASV systems against adversarial samples with equal or higher degrees very well.

4.4. Robustness against substitute ASV systems

In this section, we investigate detector robustness against substitute ASV systems. We conduct experiments on the BIM perturbation method with $\epsilon = 0.3$ attacking i-vector, x-vector and r-vector systems. The experiment results based on JSMA method have similar observations. The DA and EER_{det} are shown in Table 4. From the experimental results, we observe that i-vector and x-vector systems generalize well to each other but performance decreases when generalizing to r-vector system. One possible explanation is that both i-vector and x-vector systems use PLDA back-end, while r-vector system uses cosine back-end. The choice of back-end modelling may have a larger influence on perturbation patterns than utterance embedding modelling. Besides, r-vector system has better generalization compared with i-vector and x-vector systems, it perhaps implies cosine back-end has better generalization ability.

To see the detector's ability to recognize adversarial samples, we visualize adversarial score distributions for genuine samples, in-domain and unseen adversarial samples, as shown in Fig. 1. It shows the detector can generalize well to unseen ASV systems by assigning high adversarial scores to most of adversarial samples. For some cases where a low DA occurs, e.g. training on i-vector while evaluated on r-vector system (72.45%), the detector still achieves an acceptable EER_{det} (6.27%). This indicates the detector still works well but there needs a shifted operating point to detect adversarial samples.

4.5. Robustness against perturbation methods

In this section, we investigate detector robustness against perturbation methods. Detector performance is evaluated by adversarial samples crafted from BIM and JSMA attacking on the x-vector system, as shown in Table 5. We observe a generalizability of 10.15% EER_{det} from JSMA to BIM and 50.55% EER_{det} from BIM to JSMA. This indicates the generalizability is not symmetric and can drop greatly in some cases to be

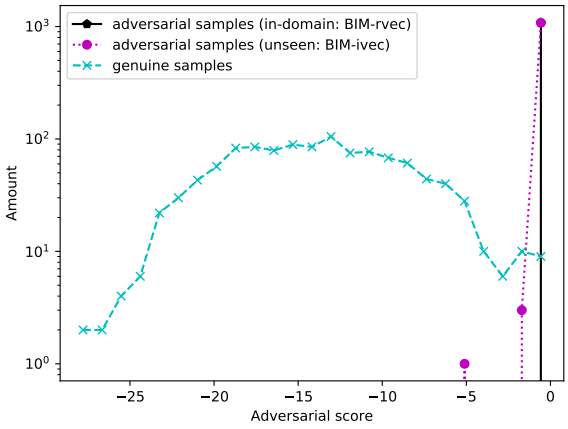

Figure 1: *The adversarial score distribution for genuine samples, and adversarial samples crafted from in-domain and unseen ASV systems (training: BIM-rvec, evaluation: BIM-ivec).*

a random guess (50.55% EER_{det}). This phenomenon shows a limited detector robustness against unseen perturbation methods. The detector trained on a combination of both methods can perform well on both, which suggests that we could enlarge our training dataset to include as many existing perturbation methods as possible to enhance our model's robustness. To deal with unseen perturbation methods, we believe that a proper combination of observed perturbation methods can reinforce the detector's robustness. We leave this to future studies.

Table 5: *Detector performance against perturbation methods*

DA (%)		training		
		BIM	JSMA	combined
evaluation	BIM	99.83	57.73	99.48
	JSMA	48.61	99.44	99.09
EER_{det} (%)		training		
		BIM	JSMA	combined
evaluation	BIM	0.18	10.15	0.46
	JSMA	50.55	0.46	0.92

5. Conclusion

This work proposes to defend ASV systems against adversarial attacks using a separate detection network. A VGG-like network is introduced to determine whether an input is a genuine or an adversarial sample. Our method is demonstrated to be effective on detecting adversarial samples. We also analyze various kinds of unseen attack setting's impact on detector robustness. We observe that the detector is relatively robust against substitute ASV systems, while the generalizability against perturbation methods is not symmetric and detector performance could drop greatly in some cases. The weak robustness against unseen perturbation methods shows a direction for developing stronger countermeasures.

6. Acknowledgements

This work is partially supported by HKSAR Government's Research Grants Council General Research Fund (Project No. 14208718).

7. References

[1] T. Kinnunen, Z. Wu, K. Lee, F. Sedlak, E. Chng, and H. Li, "Vulnerability of speaker verification systems against voice conversion spoofing attacks: The case of telephone speech," in *ICASSP*. IEEE, 2012, pp. 4401–4404.

[2] V. Shchemelinin, M. Topchina, and K. Simonchik, "Vulnerability of voice verification systems to spoofing attacks by TTS voices based on automatically labeled telephone speech," in *International Conference on Speech and Computer*. Springer, 2014, pp. 475–481.

[3] Z. Wu, N. Evans, T. Kinnunen, J. Yamagishi, F. Alegre, and H. Li, "Spoofing and countermeasures for speaker verification: A survey," *speech communication*, vol. 66, pp. 130–153, 2015.

[4] Z. Wu, T. Kinnunen, N. Evans, J. Yamagishi, C. Hanilçi, M. Sahidullah, and A. Sizov, "Asvspoof 2015: the first automatic speaker verification spoofing and countermeasures challenge," in *Interspeech*, 2015.

[5] T. Kinnunen, M. Sahidullah, H. Delgado, M. Todisco, N. Evans, J. Yamagishi, and K. A. Lee, "The asvspoof 2017 challenge: Assessing the limits of replay spoofing attack detection," in *Interspeech*, 2017.

[6] M. Todisco, X. Wang, V. Vestman, M. Sahidullah, H. Delgado, A. Nautsch, J. Yamagishi, N. Evans, T. Kinnunen, and K. A. Lee, "Asvspoof 2019: Future horizons in spoofed and fake audio detection," in *Interspeech*, 2019.

[7] J. Williams and J. Rownicka, "Speech replay detection with x-vector attack embeddings and spectral features," in *Interspeech*, 2019.

[8] W. Cai, H. Wu, D. Cai, and M. Li, "The DKU replay detection system for the asvspoof 2019 challenge: On data augmentation, feature representation, classification, and fusion," *arXiv preprint arXiv:1907.02663*, 2019.

[9] C. Hanilçi, T. Kinnunen, M. Sahidullah, and A. Sizov, "Classifiers for synthetic speech detection: A comparison," in *Interspeech*, 2015.

[10] Z. Wu, E. S. Chng, and H. Li, "Detecting converted speech and natural speech for anti-spoofing attack in speaker recognition," in *Interspeech*, 2012.

[11] M. J. Correia, A. Abad, and I. Trancoso, "Preventing converted speech spoofing attacks in speaker verification," in *International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. IEEE, 2014, pp. 1320–1325.

[12] F. Kreuk, Y. Adi, M. Cisse, and J. Keshet, "Fooling end-to-end speaker verification with adversarial examples," in *ICASSP*. IEEE, 2018, pp. 1962–1966.

[13] X. Li, J. Zhong, X. Wu, J. Yu, X. Liu, and H. Meng, "Adversarial attacks on GMM i-vector based speaker verification systems," in *ICASSP*. IEEE, 2020, pp. 6579–6583.

[14] G. Chen, S. Chen, L. Fan, X. Du, Z. Zhao, F. Song, and Y. Liu, "Who is real bob? adversarial attacks on speaker recognition systems," *arXiv preprint arXiv:1911.01840*, 2019.

[15] Z. Li, C. Shi, Y. Xie, J. Liu, B. Yuan, and Y. Chen, "Practical adversarial attacks against speaker recognition systems," in *International Workshop on Mobile Computing Systems and Applications*, 2020, pp. 9–14.

[16] Y. Xie, C. Shi, Z. Li, J. Liu, Y. Chen, and B. Yuan, "Real-time, universal, and robust adversarial attacks against speaker recognition systems," *arXiv preprint arXiv:2003.02301*, 2020.

[17] S. Liu, H. Wu, H.-y. Lee, and H. Meng, "Adversarial attacks on spoofing countermeasures of automatic speaker verification," *arXiv preprint arXiv:1910.08716*, 2019.

[18] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," *arXiv preprint arXiv:1611.01236*, 2016.

[19] Y. Song, T. Kim, S. Nowozin, S. Ermon, and N. Kushman, "Pixeldefend: Leveraging generative models to understand and defend against adversarial examples," *arXiv preprint arXiv:1710.10766*, 2017.

[20] W. Xu, D. Evans, and Y. Qi, "Feature squeezing: Detecting adversarial examples in deep neural networks," *arXiv preprint arXiv:1704.01155*, 2017.

[21] Z. Gong, W. Wang, and W.-S. Ku, "Adversarial and clean data are not twins," *arXiv preprint arXiv:1704.04960*, 2017.

[22] Q. Wang, P. Guo, S. Sun, L. Xie, and J. H. Hansen, "Adversarial regularization for end-to-end robust speaker verification," *Proc. Interspeech 2019*, pp. 4010–4014, 2019.

[23] H. Wu, S. Liu, H. Meng, and H.-y. Lee, "Defense against adversarial attacks on spoofing countermeasures of asv," in *ICASSP*, 2020.

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

[25] S. Samizade, Z.-H. Tan, C. Shen, and X. Guan, "Adversarial example detection by classification for deep speech recognition," *arXiv preprint arXiv:1910.10013*, 2019.

[26] A. Nagrani, J. S. Chung, W. Xie, and A. Zisserman, "Voxceleb: Large-scale speaker verification in the wild," *Computer Speech & Language*, vol. 60, p. 101027, 2020.

[27] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2010.

[28] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust DNN embeddings for speaker recognition," in *ICASSP*. IEEE, 2018, pp. 5329–5333.

[29] 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.

[30] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *IEEE European symposium on security and privacy (EuroS&P)*. IEEE, 2016, pp. 372–387.

[31] X. Xiang, S. Wang, H. Huang, Y. Qian, and K. Yu, "Margin matters: Towards more discriminative deep neural network embeddings for speaker recognition," *arXiv preprint arXiv:1906.07317*, 2019.

[32] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: a large-scale speaker identification dataset," *arXiv preprint arXiv:1706.08612*, 2017.

[33] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *ICLR*, 2015.