A Comparative Study on Proactive and Passive Detection of Deepfake Speech

Chia-Hua Wu[1,2], *Wanying Ge*[1], *Xin Wang*[1], *Junichi Yamagishi*[1], *Yu Tsao*[2], *Hsin-Min Wang*[2]

[1]National Institute of Informatics, Japan
[2]Academia Sinica, Taiwan

maxwu@iis.sinica.edu.tw, jyamagis@nii.ac.jp

Abstract

Solutions for defending against deepfake speech fall into two categories: proactive watermarking models and passive conventional deepfake detectors. While both address common threats, their differences in training, optimization, and evaluation prevent a unified protocol for joint evaluation and selecting the best solutions for different cases. This work proposes a framework to evaluate both model types in deepfake speech detection. To ensure fair comparison and minimize discrepancies, all models were trained and tested on common datasets, with performance evaluated using a shared metric. We also analyze their robustness against various adversarial attacks, showing that different models exhibit distinct vulnerabilities to different speech attribute distortions. Our training and evaluation code is available at Github [1].

Index Terms: Speech deepfake detection, Audio watermarking, Speech communication, Deep learning

1. Introduction

Protection against artificially generated or manipulated deepfake speech can be categorized into passive defense and proactive defense. Passive defense involves detecting whether speech from a content provider was generated by artificial intelligence, without any pre-processing of the input speech or prior assumptions about the generator. This approach mainly utilizes deep learning-based deepfake detectors [1,2] trained on datasets containing real and fake speech [3,4]. In contrast, proactive defense (typically watermarking models [5–7]) embeds a traceable but perceptually inaudible message into the speech before distributing it, allowing the speech recipient to extract the message and verify the authenticity of the source [8].

Despite differences in use cases and methodologies, both passive and proactive approaches are potential solutions to detect speech deepfake. It is hence meaningful to compare these two types of methods. However, there are few relevant studies, possibly due to the challenges of comparing them fairly and rigorously: 1) Deepfake detectors and watermarking models are trained with different objectives and loss functions [9, 10], and their outputs vary in form and semantics. 2) They are usually evaluated using different metrics [8, 11] tailored to their use case, making direct comparison difficult. 3) Their robustness is often tested with different transmission conditions and manipulations [5, 6, 12, 13]. 4) There is no common database and protocol shared by the two research communities.

This work takes the initial step toward bridging the gap. We compare representative speech deepfake detectors and watermarking models for identifying speech deepfakes, using their original training recipes, but on common speech deepfake detection datasets and diverse transmission and manipulation conditions. Our results show that powerful passive deepfake detectors and watermarking models can achieve perfect (or near-perfect) results on the ASVspoof 2019 and 2021 logical access (LA) datasets, but suffer varying degrees of performance degradation when facing unseen transmissions and manipulations. A watermarking model called Timbre tends to be the most robust but is still vulnerable to certain codecs and pitch shift manipulation. Other passive and proactive models are more fragile.

[1]https://github.com/nii-yamagishilab/antispoofing-watermark

2. Brief Review of Speech Deepfake Detection and Watermarking

2.1. Speech deepfake detection

A speech deepfake detector acts as a binary classifier and decides whether the input waveform is *real or fake*.[2] Given a waveform $x \in \mathcal{X}$, the detector functions as $f_d : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{Y} \triangleq \{\texttt{real}, \texttt{fake}\}$ is the set of output labels. In most cases, the detector is composed of a scoring function $g_d : \mathcal{X} \to \mathbb{R}$ and a decision logic $h_d : \mathbb{R} \to \mathcal{Y}$. The function g_d produces a score $s \in \mathbb{R}$ indicating the likelihood that the input is `real`. After that, h_d assigns a label $\hat{y} = \texttt{real}$ if s is larger than a decision threshold τ_d. In implementation, g_d can be a deep neural network (DNN) with a sigmoid output function, and s can be the output of the sigmoid or the log odds [15] fed into the sigmoid.

The performance of deepfake detectors is often measured using equal error rate (EER) [1, 3, 14, 16–20]. Let a test sample set be $\{(x_i, y_i)\}_{i=1}^{N}$, where the `real` and `fake` samples are indexed by $i \in \Lambda_{\texttt{real}}$ and $i \in \Lambda_{\texttt{fake}}$, respectively. After g_d produces the scores $\{s_i\}_{i=1}^{N}$, the false acceptance rate of fake data $\widehat{P}_{\text{FA}}(\tau_d)$ and the false rejection rate of real data $\widehat{P}_{\text{FR}}(\tau_d)$ can be estimated by $\widehat{P}_{\text{FA}}(\tau_d) = \sum_{i \in \Lambda_{\texttt{fake}}} \mathbb{I}(s_i > \tau_d)/|\Lambda_{\texttt{fake}}|$ and $\widehat{P}_{\text{FR}}(\tau_d) = \sum_{i \in \Lambda_{\texttt{real}}} \mathbb{I}(s_i < \tau_d)/|\Lambda_{\texttt{real}}|$, where $\mathbb{I}$ is an indicator function. The EER is then estimated by $\text{EER} \approx \frac{1}{2}\big(\widehat{P}_{\text{FA}}(\tau_d^*) + \widehat{P}_{\text{FR}}(\tau_d^*)\big)$, where $\tau_d^* = \arg\min_{\tau_d} |\widehat{P}_{\text{FR}}(\tau_d) - \widehat{P}_{\text{FA}}(\tau_d)|$ is the threshold at which $\widehat{P}_{\text{FA}}(\tau_d^*)$ and $\widehat{P}_{\text{FR}}(\tau_d^*)$ are approximately equal.

An ideal speech deepfake detector should be able to generalize well to unseen deepfake data. To achieve this, the detector must be trained using data from diverse deepfake generators and acoustic conditions. Strategies such as data augmentation using simulated artifacts [13, 21] and self-supervised learning (SSL)-based feature extraction [2, 20, 22] are also useful.

[2]Fake data is referred to as presentation attack or *spoofed* data when it is used to compromise automatic speaker verification (ASV) systems. In that context, human speech is called *bona fide* data [14]. We use the terms *real* and *fake* (or deepfake) to address more general cases.

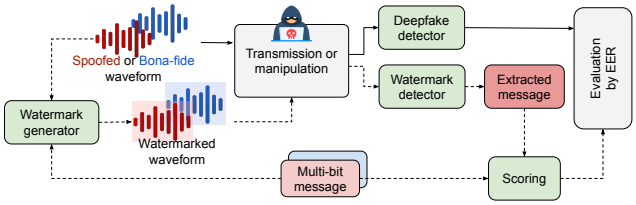


Figure 1: *Overall comparison workflow between deepfake detection and watermarking. The solid arrows (→) represent the deepfake detection process, while the dashed arrows (⇢) indicate the watermarking process.*

2.2. Multi-bit watermarking

Watermarking usually requires two components: a watermark embedder, which inserts a multi-bit message into the carrier waveform, and a watermark detector, which extracts and reconstructs the message from the received waveform [23, 24].

Let $\mathcal{M} = \{M_1, \cdots, M_K\}$ be a set of K pre-defined watermark messages, where each message $M_n \in \{1, 0\}^L$ has L bits. We may use many ($K \gg 2$) messages and assign them to different parties for ownership verification, or we can just set $K = 2$ for deepfake detection (described in § 3). The watermark embedder acts as a function $f_e : \mathcal{X} \times \mathcal{M} \to \mathcal{X}'$, where $\mathcal{X}'$ represents the domain of the watermarked waveform x'. Given x', the detector $f_w : \mathcal{X}' \to \mathcal{M}$ detects and recovers the watermark message. The detection can be done using L binary classifiers, i.e., $f_w = (f_{w,1}, f_{w,2}, \cdots, f_{w,L})$, where the l-th classifier $f_{w,l} : \mathcal{X}' \to \{1, 0\}$ decides whether the l-th bit is `1` or `0`. Similar to the deepfake detector, $f_{w,l} = h_{w,l} \circ g_{w,l}$ can be decomposed into a scoring function $g_{w,l}$ and a decision function $h_{w,l}$, but the meaning of score $s_{w,l}$ produced by $g_{w,l}$ is different — it indicates the likelihood of the l-th bit being `1`.

An ideal watermarking model should accurately detect the embedded watermark message while ensuring that the watermark is imperceptible to human listeners [6]. The accuracy of watermark detection can be measured at the bit level, i.e., the percentage of detected bits that match the ground truth. Note that most studies assume a pre-defined decision threshold when making the decision on each bit [5, 7, 25–27].

Watermarking models need to embed watermark information in a way that is imperceptible to the human ear while allowing for its perfect extraction. Therefore, compared to passive deepfake detectors, watermarking models require more complex training schemes, some of which incorporate specific operations to forge or remove watermarks during training [5, 6, 24].

3. Evaluation of Speech Deepfake Detection and Watermarking

We now explain the evaluation framework shown in Fig. 1, upon which we compare deepfake detectors and watermarking models for binary deepfake detection.

3.1. Evaluation metrics

For binary deepfake detection, we use EER as the main evaluation metric (§ 2.1). EER is a concise summary of a detector's discriminative capabilities without pre-defined thresholds [28]. It is also the upper-bound of Bayes decision error [29].

The models to be evaluated, both deepfake detectors and proactive watermarking models, need to produce a continuous-valued detection score $s \in \mathbb{R}$ per test utterance (see § 2.1). The score implies how likely the utterance is `real`, and by convention [11], we assume that a higher s favors `real` more.

3.2. Applying watermarking models to deepfake detection

While deepfake detectors directly produce the score required, this is not the case for watermarking models. Let us use two bit-wise disjoint *random* messages $\mathcal{M} = \{M_{\texttt{real}}, M_{\texttt{fake}}\}$ to watermark real and fake data, respectively. Each of the messages has L bits, and each bit can be either `1` or `0`. To classify an input utterance, the watermark detector should act as $f_d : \mathcal{X}' \to \{\texttt{real}, \texttt{fake}\}$, and its scoring function g_d should produce the score s that indicates the likelihood of the utterance being `real`. Given $\{s_{w,l}\}_{l=1}^L$ produced by the bit-level scoring functions $\{g_{w,l}\}_{l=1}^L$ (see § 2.2), we need to merge $\{s_{w,l}\}$ into s but cannot simply let s be the sum of $\{s_{w,l}\}$. Again, a higher $s_{w,l}$ suggests a more likely outcome of `1` but not necessarily `real`.

A general method to produce the score for deepfake detection without changing the watermarking model weights is

$$s = \frac{1}{L} \sum_{l=1}^{L} \Big(s_{w,l} \cdot q(m_{\texttt{real},l}) - s_{w,l} \cdot q(m_{\texttt{fake},l}) \Big), \qquad (1)$$

where $m_{\texttt{real},l}$ and $m_{\texttt{fake},l}$ are the l-th bit of the messages $M_{\texttt{real}}$ and $M_{\texttt{fake}}$, respectively, and q is a sign function that gives $q(\texttt{1}) = 1$ and $q(\texttt{0}) = -1$.

For explanation, let us use a simple example of $L = 1$. If $m_{\texttt{real},1} = \texttt{1}$ and $m_{\texttt{fake},1} = \texttt{0}$, we get $s = 2s_{w,1}$. A higher $s_{w,1}$, which favors `1` more, also gives more support to `real`. In the other case where $m_{\texttt{real},1} = \texttt{0}$ and $m_{\texttt{fake},1} = \texttt{1}$, although a higher $s_{w,1}$ still favors `1` more, the score $s = -2s_{w,1}$ becomes smaller and gives less support to `real` but more to `fake`.[3]

3.3. Evaluation with transmission and manipulation

Both deepfake detection and watermark detection must perform robustly in clean environments as well as environments with noise, compression, encoding, etc. Both methods should also be robust to surreptitious waveform manipulations designed to flip detection results. This includes not only simple operations such as clipping and trimming but also advanced ones such as adding (or removing) background noise and using sound effects such as equalization and overdrive.

The transmission effects and manipulations used in the evaluation are listed below. Note that all conditions assume that the waveforms to be processed have a sampling rate of 16 kHz.

Group 1: Noisy and public transmission channels

- **Gaussian noise**: Add a random Gaussian noise at a randomly selected SNR of 5, 10, or 15 dB.
- **MUSAN**: Add a randomly selected noise from the MUSAN corpus [30] to the waveform at a fixed SNR of 10 dB.
- **RIR**: Convolve signals with a simulated room impulse response randomly selected from the RIR corpus [31].
- **Quantization**: Quantize the waveform using a randomly selected bit-depth of 8, 16, 24, or 32 to add quantization noise.
- **Compressor**: Apply dynamic range compression [32] with a random threshold (-50 to -10 dB) and ratio (2.0 to 10.0).
- **Opus**: Compress using the Opus codec [33] with a random bitrate (1, 2, 4, 8, 16, or 31 kbps) to simulate VoIP/streaming.
- **DAC**: Compress 16 kHz audio into discrete codes and then decompress it back into an audio signal [34].

[3] If $m_{\texttt{real},l} = m_{\texttt{fake},l}$, $s_{w,l}$ would give no support to either `real` or `fake`. This bit is hence a waste of the watermark capacity.

- **WavTokenizer**: Tokenize and reconstruct the waveform using WavTokenizer (small-600-24k-4096) [35], a state-of-the-art neural audio codec.

Group 2: Acoustic manipulations

- **Clipping**: Apply amplitude clipping and constrain the amplitude to be within the 1st and 99th percentile range.
- **Overdrive**: Implement Overdrive using the PyTorch [36] function, which introduces nonlinear distortion with a randomly selected gain (0 to 50 dB) and colour (0 to 50).
- **Random trimming**: Trim the waveform with randomly selected start and end times.
- **Equalizer**: Apply a 7-band parametric equalizer with a random gain range between -12 and 12dB.
- **Frequency masking**: Randomly zero out 10 to 80 Short-time Fourier Transform (STFT) frequency bins of the waveform and reconstruct the waveform via inverse STFT (iSTFT).
- **Noise gate**: Compute a spectrogram, estimate noise thresholds, applying gating, and reconstructing the signal [37, 38].
- **Noise reduction**: Apply a DNN-based speech enhancement model [39, 40] to the waveform.
- **Time stretch**: Use a phase vocoder via Librosa [41] to stretch the waveform with a random rate between 0.5 and 2.0.
- **Pitch shift**: Use a phase vocoder and resampling via Librosa [41] to randomly shift the pitch between ± 5 semitones.

As shown in Fig. 1, for watermarking models, the above processing is applied to the watermarked waveform input to the watermark detector, rather than the waveform input to the watermark encoder.

We also investigated other transmission and manipulation types. However, for a fair comparison, they were excluded from the experiments because they were seen during the training of one or more of the compared models. The above transmissions and manipulations that are similar but not identical to those used in training are considered partially seen evaluation conditions. For example, if a DNN-based codec called EnCodec [10] is seen in AudioSeal training, transmissions using DAC or WavTokenizer are considered partially seen because they are similar to EnCodec in terms of DNN architecture and training criteria.

3.4. Dataset

The deepfake detection and watermarking communities use different databases for experiments. To compare the two methods for deepfake detection, we need a common database containing real and fake data and well-designed protocols.

We follow the speech deepfake detection community and use the LA part of the ASVspoof 2019 [3] and ASVspoof 2021 [19] datasets. Following the official protocol [19], we utilize the training and development sets of the ASVspoof 2019 LA dataset for model training and validation. These datasets include fake data generated using 6 different algorithms (A01–A06 in [3]). The ASVspoof 2019 test set and the entire 2021 LA set are used for testing, comprising fake data from 11 unseen and 2 seen deepfake generators (A07-A19 in [3]). Compared to the 2019 test set, around 85% of data in the 2021 dataset has been pre-processed under lossy transmission channels (e.g., PSTN).[4] There is no speaker overlap in the training, development, and test sets. Other details about the data and protocols can be found in [3, 19].

[4] If a channel-transmitted utterance is further processed by any transmission in § 3.3, it is called double degradation [19].

4. Experiment

To assess the proposed evaluation framework, we conducted a comparative study using four representative DNN-based deepfake detectors and watermarking models.

4.1. Experimental models

Passive deepfake detectors: The two deepfake detectors we studied are **AASIST** [1] and **SSL-AASIST** [20]. AASIST is a state-of-the-art end-to-end (E2E) spoofing countermeasure solution. It uses a sinc-layer front-end to decompose raw waveforms for feature extraction and a graph attention network backend to integrate temporal and spectral representations, followed by a readout operation and a fully connected output layer to produce decision scores. SSL-AASIST enhances AASIST by integrating the pre-trained wav2vec2.0 model [42] as the front-end for feature extraction. Using SSL models in the front end improves robustness against noise, reverberation, and other external distortions [20]. We use official implementations and the released AASIST and SSL-AASIST checkpoints, which were trained on the ASVspoof 2019 LA dataset.

Proactive watermarking models: The two experimental watermarking models are **Timbre** [5] and **AudioSeal** [6]. Timbre embeds and detects watermarks in the spectral domain. It applies STFT to extract the spectrogram, embeds the watermark in the amplitude while preserving the phase, and reconstructs the waveform using iSTFT. AudioSeal is a state-of-the-art audio watermarking framework based on a sequence-to-sequence encoder-decoder architecture.

Following the official implementation, we trained the two proactive models using the ASVspoof 2019 LA training and development sets. The sampling rate is 16 kHz, and the watermark message length is 16 bits. The outputs of Timbre's bit-level detectors' linear output layer and the sigmoid logit of the AudioSeal output layers are utilized as the score $s_{w,l}$ in Eq. (1). The official implementation of AudioSeal uses data augmentation based on MP3 compression, re-sampling, etc. As mentioned in § 3.3, transmissions and manipulations used in data augmentation for model training are excluded from the evaluation.

4.2. Results and analysis

Table 1 presents the results. Without the transmission or manipulation in § 3.3, the proactive models demonstrate the ability to perfectly distinguish real from spoofed speech, achieving 0% EER on the ASVspoof 2019 LA and ASVspoof 2021 LA evaluation sets. This shows that watermarks can be accurately embedded into and extracted from real and fake utterances, even when they are uttered by an unseen speaker or produced by 11 unseen deepfake generators. Among the passive models, SSL-AASIST exhibits near-perfect performance, with EERs of 0.23% and 0.84% on the two test sets, respectively. However, while the EER on the 2019 LA test set is below 1%, AASIST's EER on the 2021 LA data rises to 8.15% because the data contains transmission channel operations.

When the transmission or manipulation listed in § 3.3 is applied, in many cases all model performance drops, even if the transmission or manipulation is similar to the data augmentation methods during training. For example, although AudioSeal has used EnCodec for data augmentation, similar codecs (DAC and wavTokenizer) result in EERs of 60.95% and 97.40% on the ASVspoof 2019 LA test set. An EER larger than 50% means that watermarked fake utterances are more likely to be

Table 1: *Equal Error Rate (EER) results on the ASVspoof 2019 LA and ASVspoof 2021 LA datasets. Darker-shaded values (e.g., in gray) indicate worse performance, corresponding to higher EER values. A transmission or manipulation condition is considered as partially seen if it is used by any of the experimental models during training. The model partially seen the condition is marked with an asterisk $*$. The two sub-groups in the partially seen and unseen conditions correspond to transmission and manipulation, respectively.*

	Transmission / Manipulation	EER (%)↓ of ASVspoof 2019 LA				EER (%)↓ of ASVspoof 2021 LA			
		Passive Models		Proactive Models		Passive Models		Proactive Models	
		AASIST	SSL-AASIST	Timbre	AudioSeal	AASIST	SSL-AASIST	Timbre	AudioSeal
	None from § 3.3	0.83	0.23	0.00	0.00	8.15	0.84	0.00	0.00
Partially seen	Gaussian noise	18.06	1.95 $*$	17.60	15.83 $*$	25.00	2.79 $*$	18.73	16.04 $*$
	DAC	1.66	0.27	0.01	97.40 $*$	8.44	1.44	0.00	97.59 $*$
	WavTokenizer	17.84	15.92	50.12	60.95 $*$	17.31	16.31	46.32	63.80 $*$
	Random trimming	19.56 $*$	8.15	0.00	37.50	26.86 $*$	11.48	0.00	36.87
	Time stretch	66.53	44.42	0.00	0.03 $*$	68.08	47.56	0.00	0.05 $*$
	Pitch shift	66.12	48.36	52.62	47.30 $*$	68.67	48.84	53.14	50.11 $*$
Unseen	MUSAN	17.84	1.73	1.31	2.91	26.31	2.97	2.02	3.25
	RIR	35.49	4.41	0.00	57.08	45.59	7.75	0.00	57.21
	Quantization	26.15	3.31	8.66	19.59	33.19	4.59	10.80	20.92
	Compressor	9.30	1.02	0.00	0.00	14.63	4.14	0.00	0.00
	Opus	36.27	27.55	17.35	47.38	40.22	30.58	17.32	46.17
	Clipping	1.22	0.23	0.00	0.00	6.62	0.92	0.00	0.00
	Overdrive	15.30	6.19	0.11	0.00	21.04	8.53	0.05	0.00
	Equalizer	1.75	0.23	0.00	0.03	9.56	0.90	0.00	0.05
	Frequency masking	43.32	33.11	2.94	24.40	49.99	36.55	4.66	23.89
	Noise gate	10.56	2.56	0.13	2.56	18.34	4.21	0.26	3.12
	Noise reduction	17.18	11.61	0.00	0.05	24.61	14.16	0.00	0.08
	Average w/o None	23.77	12.41	8.87	24.29	29.67	14.34	9.02	24.66

detected as `real`. Similarly, in the passive AASIST model, random trimming during training does not lead to a lower EER when random trimming is applied to test utterances. The only exception is SSL-AASIST under the Gaussian noise condition, probably due to the robustness of the pre-trained SSL model to simple additive noise.

Among the four models, Timbre appears to be the most robust, achieving EERs of 8.87% and 9.02% on two evaluation sets, followed by the passive model SSL-AASIST, which achieves 12.41% and 14.34%. However, we cannot hastily conclude that proactive models are more robust. First, AudioSeal's performance degrades severely when certain transmissions or manipulations are applied to watermarked waveforms. Second, even the best-performing Timbre model shows varying degrees of degradation when facing different transmission and manipulation types. For example, while Clipping and Equalizer do not harm EER, simple Gaussian noise results in an EER of 17.60%. More complex techniques, such as DNN-based WavTokenizer and signal-processing-based Pitch shift, result in an EER of around 50%, where the model is unable to differentiate real and fake data based on the detected watermarks.

All the results suggest that robustness against transmission and manipulation is an unsolved issue. Challenging transmission and manipulation types are highlighted below.

- Codecs (Opus, DAC, and WavTokenizer): While we have described how DAC and WavTokenizer induce high EER on AudioSeal, WavTokenizer also increases the EER of other proactive and passive models. The non-DNN codec Opus also proved harmful to all models, especially AudioSeal. Different ways of reconstructing waveforms, namely linear prediction plus discrete cosine transform (Opus), transposed convolution (DAC), and convolution plus iSTFT (WavTokenizer), may affect how the watermark or deepfake artifact is altered.
- Temporal and spectral modifications (Time stretch, Pitch shift, and Random trimming): While time-stretch mainly affects passive models, Pitch shift that re-samples time-stretched waveforms impairs all models. Random trimming will hurt the performance of models other than Timbre. This is expected because Timbre explicitly embeds the watermark in the frequency domain of each frame and is therefore robust to temporal manipulations. However, this design is vulnerable to frequency domain transformations.

5. Conclusion

In this work, we took the initiative to compare representative passive deepfake detectors (AASIST and SSL-AASIST) and proactive watermarking models (Timbre and AudioSeal) for the binary classification task of speech deepfake detection. Comparisons based on unified evaluation metrics, scoring methods, diverse evaluation conditions, and common datasets show that watermarking models and SSL-AASIST can perfectly detect speech deepfakes without transmission or manipulation, but all models fail to varying degrees under different transmission or manipulation conditions. Timbre appears to be the most robust, but it still suffers from pitch shift based manipulation or transmission via WavTokenizer or Opus.

Robustness to various transmissions and manipulations must be better addressed for the passive and proactive models investigated. This initial study used the official training recipes, some of which did not incorporate any data augmentation. Future work will investigate the impact of data augmentations on proactive and passive models.

6. Acknowledgements

This work was conducted during the first author's internship at the National Institute of Informatics (NII), Japan. This study was partially supported by JST AIP Acceleration Research (JPMJCR24U3), MEXT KAKENHI Grant (24H00732), and JST PRESTO (JPMJPR23P9).

7. References

[1] J.-w. Jung, H.-S. Heo, H. Tak, H.-j. Shim *et al.*, "AASIST: Audio anti-spoofing using integrated spectro-temporal graph attention networks," in *Proc. ICASSP*, 2022, pp. 6367–6371.

[2] Y. Zhu, S. Koppisetti, T. Tran, and G. Bharaj, "SLIM: Style-linguistics mismatch model for generalized audio deepfake detection," in *Proc. NeurIPS*, 2024.

[3] X. Wang, J. Yamagishi, M. Todisco, and Others, "ASVspoof 2019: A large-scale public database of synthesized, converted and replayed speech," *Computer Speech & Language*, vol. 64, p. 101114, 2020.

[4] Y. Xie, Y. Lu, R. Fu, Z. Wen *et al.*, "The codecfake dataset and countermeasures for the universally detection of deepfake audio," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 33, pp. 386–400, 2025.

[5] C. Liu, J. Zhang, T. Zhang, X. Yang *et al.*, "Detecting voice cloning attacks via Timbre watermarking," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2024.

[6] R. S. Roman, P. Fernandez, H. Elsahar, A. Défossez *et al.*, "Proactive detection of voice cloning with localized watermarking," in *Proc. ICML*, 2024.

[7] G. Chen, Y. Wu, S. Liu, T. Liu *et al.*, "WavMark: Watermarking for audio generation," *arXiv preprint arXiv:2308.12770*, 2023.

[8] H. Liu, M. Guo, Z. Jiang, L. Wang *et al.*, "AudioMarkBench: Benchmarking robustness of audio watermarking," in *Proc. NeurIPS Datasets and Benchmarks Track*, 2024.

[9] Y. Zhang, F. Jiang, and Z. Duan, "One-class learning towards synthetic voice spoofing detection," *IEEE Signal Processing Letters*, vol. 28, pp. 937–941, 2021.

[10] A. Défossez, J. Copet, G. Synnaeve, and Y. Adi, "High fidelity neural audio compression," *Transactions on Machine Learning Research*, 2023.

[11] T. Kinnunen, K. A. Lee, H. Delgado, N. Evans *et al.*, "t-DCF: A detection cost function for the tandem assessment of spoofing countermeasures and automatic speaker verification," in *Proc. Odyssey*, 2018, pp. 312–319.

[12] P. Kawa, M. Plata, and P. Syga, "Defense against adversarial attacks on audio deepFake detection," in *Proc. Interspeech*, 2023, pp. 5276–5280.

[13] H. Tak, M. Kamble, J. Patino, M. Todisco *et al.*, "RawBoost: A raw data boosting and augmentation method applied to automatic speaker verification anti-spoofing," in *Proc. ICASSP*, 2022, pp. 6382–6386.

[14] International Organization for Standardization, *Information Technology — Biometric Presentation Attack Detection — Part 1: Framework*, International Organization for Standardization Std. ISO/IEC 30 107-1:2023, 2023. [Online]. Available: https://www.iso.org/standard/83828.html

[15] C. M. Bishop, *Pattern Recognition and Machine Learning*, ser. Information Science and Statistics. New York: Springer, 2006.

[16] Z. Wu, N. Evans, T. Kinnunen, J. Yamagishi *et al.*, "Spoofing and countermeasures for speaker verification: A survey," *Speech Communication*, vol. 66, pp. 130–153, 2015.

[17] M. Todisco, H. Delgado, and N. Evans, "Constant Q cepstral coefficients: A spoofing countermeasure for automatic speaker verification," *Computer Speech & Language*, vol. 45, pp. 516–535, 2017.

[18] T. Kinnunen, N. Evans, J. Yamagishi, K. A. Lee *et al.*, "ASVspoof 2017: Automatic speaker verification spoofing and countermeasures challenge evaluation plan," in *Proc. Interspeech*, 2017, pp. 2–6.

[19] X. Liu, X. Wang, M. Sahidullah, J. Patino *et al.*, "ASVspoof 2021: Towards spoofed and deepfake speech detection in the wild," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 31, pp. 2507–2522, 2023.

[20] H. Tak, M. Todisco, X. Wang, J.-w. Jung *et al.*, "Automatic speaker verification spoofing and deepfake detection using wav2vec 2.0 and data augmentation," in *Proc. Odyssey*, 2022, pp. 112–119.

[21] X. Wang and J. Yamagishi, "Spoofed training data for speech spoofing countermeasure can be efficiently created using neural vocoders," in *Proc. ICASSP*, 2023, pp. 1–5.

[22] A. Guragain, T. Liu, Z. Pan, H. B. Sailor *et al.*, "Speech foundation model ensembles for the controlled singing voice deepfake detection (CtrSVDD) challenge 2024," in *Proc. SLT*, 2024, pp. 774–781.

[23] S. Ji, Z. Jiang, J. Zuo, M. Fang *et al.*, "Speech watermarking with discrete intermediate representations," in *Proc. AAAI*, 2025.

[24] M. K. Singh, N. Takahashi, W. Liao, and Y. Mitsufuji, "SilentCipher: Deep audio watermarking," in *Proc. Interspeech*, 2024, pp. 2235–2239.

[25] X. Cheng, Y. Wang, C. Liu, D. Hu *et al.*, "HiFi-GANw: Watermarked speech synthesis via fine-tuning of HiFi-GAN," *IEEE Signal Processing Letters*, vol. 31, pp. 2440–2444, 2024.

[26] S. Wu, J. Liu, Y. Huang, H. Guan *et al.*, "Adversarial audio watermarking: Embedding watermark into deep feature," in *Proc. ICME*, 2023, pp. 61–66.

[27] P. O'Reilly, Z. Jin, J. Su, and B. Pardo, "MaskMark: Robust neuralwatermarking for real and synthetic speech," in *Proc. ICASSP*, 2024, pp. 4650–4654.

[28] D. A. Van Leeuwen and N. Brümmer, *An introduction to application-independent evaluation of speaker recognition systems*. Springer, 2007.

[29] N. Brümmer, L. Ferrer, and A. Swart, "Out of a hundred trials, how many errors does your speaker verifier make?" in *Proc. Interspeech*, 2021, pp. 1059–1063.

[30] D. Snyder, G. Chen, and D. Povey, "MUSAN: A music, speech, and noise corpus," *arXiv preprint arXiv:1510.08484*, 2015.

[31] T. Ko, V. Peddinti, D. Povey, M. L. Seltzer *et al.*, "A study on data augmentation of reverberant speech for robust speech recognition," in *Proc. ICASSP*, 2017, pp. 5220–5224.

[32] P. Sobot, "Pedalboard," Jul. 2021. [Online]. Available: https://doi.org/10.5281/zenodo.7817838

[33] J.-M. Valin, K. Vos, and T. Terriberry, "Definition of the opus audio codec," Tech. Rep., 2012.

[34] R. Kumar, P. Seetharaman, A. Luebs, I. Kumar *et al.*, "High-fidelity audio compression with improved RVQGAN," in *Proc. NeurIPS*, vol. 36, 2023, pp. 27 980–27 993.

[35] S. Ji, Z. Jiang, W. Wang, Y. Chen *et al.*, "WavTokenizer: An efficient acoustic discrete codec tokenizer for audio language modeling," in *Proc. ICLR*, 2025.

[36] A. Paszke, S. Gross, F. Massa, A. Lerer *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Proc. NeurIPS*, 2019.

[37] T. Sainburg, M. Thielk, and T. Q. Gentner, "Finding, visualizing, and quantifying latent structure across diverse animal vocal repertoires," *PLoS Computational Biology*, vol. 16, no. 10, p. e1008228, 2020.

[38] T. Sainburg, "timsainb/noisereduce: v1.0," Jun. 2019. [Online]. Available: https://doi.org/10.5281/zenodo.3243139

[39] H. Schröter, A. N. Escalante-B., T. Rosenkranz, and A. Maier, "DeepFilterNet: A low complexity speech enhancement framework for full-band audio based on deep filtering," in *Proc. ICASSP*, 2022, pp. 7407–7411.

[40] H. Schröter, T. Rosenkranz, A. N. Escalante-B., and A. Maier, "DeepFilterNet: Perceptually motivated real-time speech enhancement," in *Proc. Interspeech*, 2023, pp. 2008–2009.

[41] B. McFee, C. Raffel, D. Liang, D. P. Ellis *et al.*, "librosa: Audio and music signal analysis in python," in *Proc. SciPy*, 2015, pp. 18–25.

[42] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Proc. NeurIPS*, 2020.