

Article

An Audio Watermarking Algorithm Based on Adversarial Perturbation[†]

Shiqiang Wu^{1,2} , Jie Liu³ , Ying Huang^{3,*}, Hu Guan² and Shuwu Zhang³

¹ School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China; wushiqiang2018@ia.ac.cn

² Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China; hu.guan@ia.ac.cn

³ School of Artificial Intelligence, Beijing University of Posts and Telecommunications, Beijing 100876, China; jie.liu@ia.ac.cn (J.L.); shuwu.zhang@bupt.edu.cn (S.Z.)

* Correspondence: ying.huang@bupt.edu.cn

[†] This paper is an extended version of our paper published in Wu, S.; Liu, J.; Huang, Y.; Guan, H.; Zhang, S. Adversarial Audio Watermarking: Embedding Watermark into Deep Feature. In Proceedings of the IEEE International Conference on Multimedia and Expo (ICME), Brisbane, Australia, 10–14 July 2023.

Abstract: Recently, deep learning has been gradually applied to digital watermarking, which avoids the trouble of hand-designing robust transforms in traditional algorithms. However, most of the existing deep watermarking algorithms use encoder–decoder architecture, which is redundant. This paper proposes a novel audio watermarking algorithm based on adversarial perturbation, AAW. It adds tiny, imperceptible perturbations to the host audio and extracts the watermark with a pre-trained decoder. Moreover, the AAW algorithm also uses an attack simulation layer and a whitening layer to improve performance. The AAW algorithm contains only a differentiable decoder, so it reduces the redundancy. The experimental results also demonstrate that the proposed algorithm is effective and performs better than existing audio watermarking algorithms.

Keywords: audio watermarking; deep learning; adversarial example



Citation: Wu, S.; Liu, J.; Huang, Y.; Guan, H.; Zhang, S. An Audio Watermarking Algorithm Based on Adversarial Perturbation. *Appl. Sci.* **2024**, *14*, 6897. <https://doi.org/10.3390/app14166897>

Academic Editor: David Megías

Received: 31 May 2024

Revised: 2 August 2024

Accepted: 5 August 2024

Published: 6 August 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The burgeoning growth of generative artificial intelligence [1] has resulted in many outstanding achievements. However, the deceptive realism of their outputs requires just as much caution [2]. Tracing and managing the distribution and propagation of this content is necessary to prevent falsely generated content from disrupting the cyber-environment.

As a powerful method for tracing multimedia distribution, digital watermarking techniques have been widely studied in recent years [3–5]. Watermarking algorithms can only be applied to a single data type due to the different ways of processing material types. There have been many excellent watermarking algorithms in several data types, including image [6,7], audio [8,9], and video [10]. This paper focuses on audio data. An audio watermarking algorithm embeds a specific identifier (watermark) into the host audio and propagates with the watermarked audio. When the audio appears in a scenario where it is not allowed, by extracting the identifier (watermark), one can quickly trace the source of leakage. Hence, watermarking algorithms are also widely used for copyright protection, information hiding, and private transmission.

Audio watermarking algorithms also need to satisfy several performance requirements, the most significant of which are imperceptibility and robustness. Imperceptibility means that there is no perceptual difference between the host audio and watermarked audio, i.e., the watermarking cannot corrupt the audio quality. Robustness expects the extracted watermark to be consistent with the embedded one. However, watermarked audio may encounter some transformations or attacks that will damage the watermark extraction. Robustness is the most critical and challenging performance requirement for

audio watermarking. Due to the variety of attacks, existing audio watermarking algorithms are not yet robust to arbitrary scenarios.

Traditional algorithms still dominate current audio watermarking research studies [11–14] that try to design a transformation or feature that is robust against pre-defined attacks for watermarking. Most of them embed the watermark into the transform domain, such as discrete cosine transform (DCT) [12,13], discrete wavelet transform (DWT) [15,16], fast Fourier transform (FFT) [17], frequency singular value coefficient (FSVC) [14], and so on. However, these hand-designed algorithms can only consider those pre-defined attacks, representing a small part of all attack types. In order to maintain robustness in more scenarios, it may be practical to combine multiple signal-processing techniques [8,11]. It is not only a nontrivial task but also one that only increases the few attack types that the algorithm can resist.

In recent years, deep learning techniques have made many achievements in digital watermarking, although most focus on images and video [10,18]. Neural networks can adaptively fit appropriate transformations based on training data and loss functions rather than designing robust features by hand. Most deep learning-based watermarking algorithms use an end-to-end encoder–decoder architecture that typically contains an encoder, a noise layer, a discriminator, and a decoder [18–24]. In this case, the discriminator and encoder are trained adversarially to improve imperceptibility, and the noise layer provides multiple augmentations to the decoder’s inputs to improve robustness. Figure 1 illustrates a conventional audio watermarking framework, where trapezoids and rectangles represent network modules. In the encoder, the audio is first transformed to suitable representations by Module 1 and then concatenated with the watermark information. The Module 2 “inverse”-transforms the concatenated representations into the audio domain (which is usually concatenated with the original audio before processing). Subsequently, the decoder extracts the watermark from the watermarked audio. An intuitive interpretation is that the decoder first transforms the audio into a representation and then finds the watermark, where Module 3 represents the transformation of the audio to the representation domain.

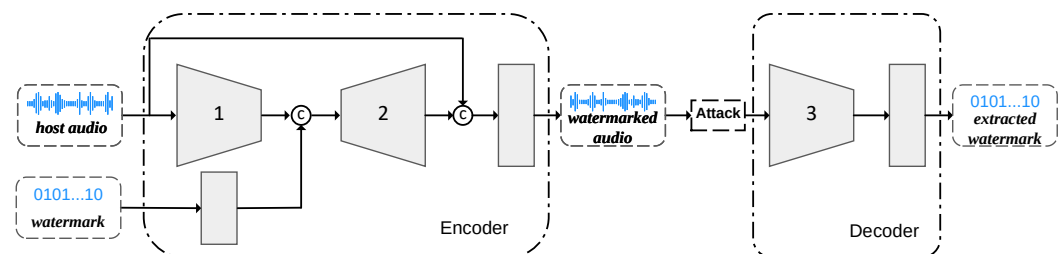


Figure 1. A deep learning-based framework for audio watermarking algorithms. The process for embedding is depicted here, so the potential discriminator and noise layer are omitted.

However, we argue that such an architecture needs to have more serious redundancies of design. Firstly, Modules 1 and 3 provide the same service, transforming audio into representations. Secondly, Modules 1 and 2 are invertible transform pairs, which is irrelevant for the watermark-embedding task (e.g., the details of the DCT transform are not of interest in the design of traditional algorithms). These redundancies increase the parametric quantity of the watermarking algorithm and the cost of learning or deploying.

To address the above challenges, we propose an audio watermarking algorithm based on adversarial perturbation (AAW, adversarial audio watermarking) in this paper. It is a deep learning-based watermarking algorithm, so it is more flexible to deal with more attacks. In particular, if a specific attack can be constructed as a differentiable mathematical model, it can be combined with AAW to improve the robustness against that attack. Meanwhile, the AAW relies only on a differentiable decoder for watermark embedding and extraction, significantly reducing the redundancy of the model. A tiny perturbation is added to the host audio to conduct the watermark embedding. This tiny perturbation is found by

optimizing the audio samples. In this paper, we use two different pre-trained decoders: a supervised classification model and an self-supervised contrastive learning model.

Our contributions are the following:

- A new audio watermarking framework is provided, which relies only on a pre-trained differentiable decoder, while the embedding leverages an optimization algorithm to find a tiny perturbation to add to the host audio.
- An attack simulation layer is utilized during embedding. Through this layer, attacks are integrated into the optimization objective at watermark embedding. Experiments demonstrate that this approach significantly improves the robustness of watermarking against multiple attacks.
- A whitening layer is introduced, which makes the output features of the pre-trained network independent of each other. This layer reduces the failures of the watermark embedding and more flexibly adjusts the quantity of the watermark.

The remainder of this paper is organized as follows: Section 2 introduces the related works; Section 3 describes the proposed method; Section 4 contains the experiment and discussion; and the summary of this work is in Section 5.

2. Related Works

Traditional audio watermarking algorithms can be categorized according to their embedding domains, a few of which are time domain algorithms [25] and the majority of which are transform domain ones. Audio watermarking algorithms in the time domain generally provide straightforward solutions since they directly change audio samples. Although there has not been solid evidence to confirm the performance shortcomings of time domain algorithms, researchers generally agree that more careful designs are needed to improve the performance of time domain algorithms than transform domain ones [3]. The transform domain algorithms, which adhere to the transform, embedding, and inverse-transform pipeline, apply a transformation before embedding the watermark. Transforms that are often applied to watermarking algorithms are singular-value decomposition (SVD) [26], DCT [12,13], STFT [27], DWT [15,16], FFT [17], and FSVC [14], etc. Researchers sometimes manually design invertible transformations to take advantage of the properties of the transform domain.

The popular transform domain algorithms include spread-spectrum-based [28] and quantization-based [29]. Among them, spread-spectrum-based algorithms have more anti-interference capability, but there is host signal interference. Quantization-based algorithms avoid host signal interference but are not robust against common attacks such as requantization, scale enlarging/diminishing, etc. Therefore, researchers have proposed many auxiliary techniques to enhance the ability of traditional algorithms, such as diversity reception [13], synchronization codes [30], sliding windows [31], robust features [14], pre-correlation whitening [11], and polarization embedding [12]. These auxiliary techniques are designed for specific attacks. Thus, the types of attacks that these techniques can withstand are limited. Combining these auxiliary techniques to deal with multiple attack types is a nontrivial task, and the number of attacks the algorithms can withstand increases only a little.

Deep learning-based watermarking algorithms have emerged as a potential alternative to traditional algorithms in image watermarking. These algorithms solve two pivotal problems of traditional algorithms. Firstly, neural networks can fit arbitrary functions. Therefore, researchers no longer need to design sophisticated transformation functions and embedding approaches. Next, the attack simulation layer enables the neural network to learn robustness against most differentiable attacks. The model is frequently constructed using encoder–decoder architecture, where the encoder embeds the watermark into a host material and the decoder accomplishes the extraction. For example, HiDDeN [18] cascaded the encoder, noise layer, and decoder and enhanced the perceptual quality using a parallel-connection adversarial discriminator. Liu et al. [19] proposed a two-stage learning method to address the constraint that the noise layer in HiDDeN must be differentiable. MBRS [20]

adopted adversarial training for JPEG attacks to provide additional robustness. Distortion Agnostic [21] considers the decoder as a discriminator in the adversarial network. At the same time, another generator was constructed to attack the watermarked image to improve the model's robustness against unknown attacks. IGA [22] and Yu [23] introduced attention mechanisms to improve the robustness and imperceptibility of watermarking. ReDMark [24] and Tavakoli et al. [32] integrated conventional DCT or DWT with deep learning to improve the performance. Some algorithms use adversarial perturbations to watermark images. Vukotić et al. [33] investigated a new family of transformation based on a deep neural network trained with supervision for a classification task. Fernandez et al. [34] used a pre-trained DINO model [35] to conduct the zero-bit and multi-bit image watermarking. Jia et al. [36] embedded watermark by a black-box model. EAST [37] implemented multi-bit embedding on the classification logits instead of the feature map.

Deep learning has rarely been applied to audio watermarking compared to image watermarking. DeAR [9] transformed the audio into a 2D spectrogram and then used an image watermarking model. A cascade pipeline of 'reverberation'–'band-pass filtering'–'Gaussian noise' was also adopted to simulate the distortion of audio propagating over the air. Kong et al. [38] developed a new method for audio watermarking based on adversarial examples, which embeds and recovers the information using a private DNN-based ASR (automatic speech recognition) model. However, Kong et al. did not adopt some strategies to guarantee robustness, so the embedding information was utterly unrecognizable after attacks.

Adversarial perturbations [39] are initially used to attack a deep learning model, which cause the model to output wrong [40] or specified [41] labels. Adversarial perturbations are usually tiny or imperceptible, so data that are correctly recognized by humans may cause a deep learning model to produce the wrong output. Adversarial perturbations have attracted a lot of concern in fields such as image classification [42,43], target recognition [44,45], and speech recognition [46–48]. Researchers have proposed some more effective perturbation methods [47,49] or provided many strategies to resist perturbations [48]. Adversarial perturbation also enables a feature extraction network to output specified features. While destructively hazardous for most tasks, adversarial perturbations can be skillfully implemented for watermark embedding.

3. The Proposed Methodology

3.1. Overview of Framework

The AAW algorithm conducts the watermark embedding by adding tiny perturbations into the host audio. The diagram of the algorithm (embedding stage) is shown in Figure 2.

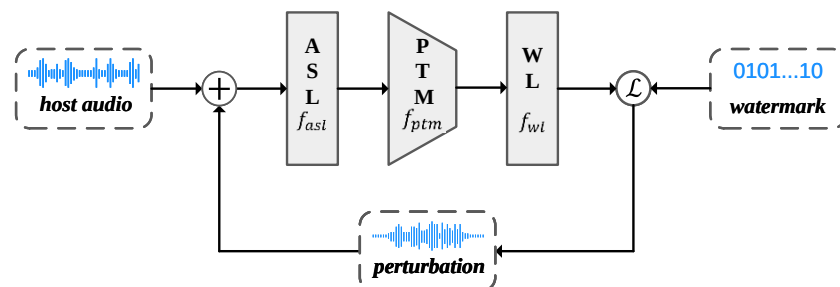


Figure 2. Diagram of the AAW algorithm (embedding stage), where ASL, PTM, and WL denote the attack simulation layer, pre-trained module, and whitening layer, respectively. $\mathcal{L}$ denotes the optimization objective (loss function). The feedback link denotes the update of the tiny perturbation according to the loss function.

The host audio $\mathbf{x}(n)$ with a tiny perturbation $\delta(n)$ passes sequentially through the attack simulation layer (ASL) f_{asl} , the pre-trained module (PTM) f_{ptm} , and the whitening layer (WL) f_{wl} . The output $\mathbf{o}_{wl}$ of the whitening layer has the same dimension m as the

binary watermark $\mathbf{w}$. The distance between $\mathbf{o}_{wl}$ and $\mathbf{w}$ is calculated by the loss function $\mathcal{L}$. The perturbation $\delta(n)$ is optimally updated so that the output $\mathbf{o}_{wl}$ is approximated with the watermark $\mathbf{w}$. Watermark embedding is performed in iterative optimization of tiny perturbations. Finally, when the iteration stops, the watermarked audio $\mathbf{x}_w(n) = \mathbf{x}(n) + \tilde{\delta}(n)$, where $\tilde{\delta}$ is δ when the iteration stops.

The remainder of this section describes in detail the individual components in Figure 2 and the embedding and extraction of watermarks.

3.2. Using Pre-Trained Module for Transformations

After training with the given dataset and task, a neural network can extract suitable features from the input material. These features are also applied to other tasks, such as perceptual metrics [50] and content retrieval [51,52].

As an important component in the AAW algorithm, the PTM not only extracts suitable features from the audio but also serves as a bridge for the transformation between the audio and the watermark domains. In this paper, two PTMs are employed as audio-to-feature transformation functions, which are the feature extractor in a classification network and a contrastive learning model for audio features, respectively.

3.2.1. Feature Extractor in a Classification Network

We trained an audio classification network. The part before its linear layer is employed is the deep feature extractor $f_{cf} : \mathcal{A} \rightarrow \mathcal{F}$, where $\mathcal{A}$ is denoted as the audio domain and $\mathcal{F}$ as the feature space. The network structure is shown in Figure 3, where the feature extractor is in the dashed box.

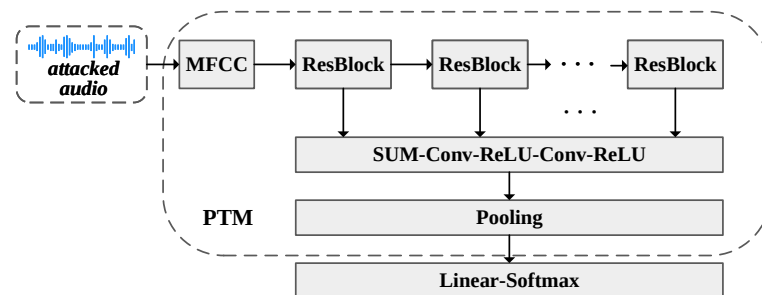


Figure 3. Block diagram of the classification network. The part in the dashed box is the deep feature extractor f_{cf} . This structure is also used for feature extraction based on contrastive learning.

The input audio is first fed into the differentiable Mel frequency cepstrum coefficient (MFCC) layer, followed by 16 residual blocks. The outputs of each residual block are passed to a SUM module by skip connection. These outputs are accumulated into a feature map and passed through two 1×1 convolution layers and a rectified linear unit (ReLU) and finally through a mean pooling layer to obtain a 512-dimensional feature. This deep feature will predict the audio as belonging to different categories after passing through a linear layer and softmax layer.

The structure of the ResBlock is shown in Figure 4. Each ResBlock contains a high-way connection and a branch containing two convolution layers, the second of which is with a 1×1 convolution kernel, whose output is passed to the SUM module. The first convolutional layer is followed by a gated activation function, as

$$g(x) = \tanh(W_1 x) \odot \text{sigmoid}(W_2 x) \quad (1)$$

where W_1 and W_2 are learnable parameters and $\odot$ means the Hadamard product, as it is shown that this activation function is more suitable for audio feature extraction [53].

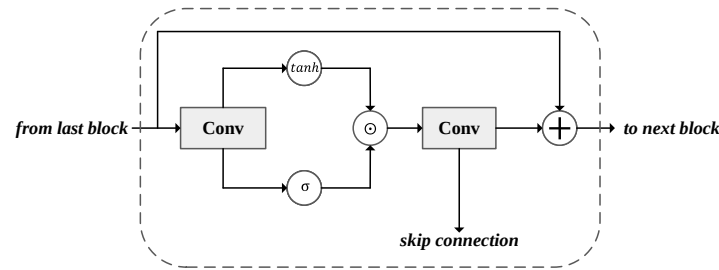


Figure 4. The structure of the residual block. The σ means the sigmoid function.

The feature extractor f_{cf} should meet two properties: (1) $f_{cf}(\mathbf{x})$ will change as the input audio $\mathbf{x}(n)$ is perturbed imperceptibly to achieve the watermark embedding; and (2) the extracted features for different augmented forms of one audio should be as consistent as possible. Since the extractor f_t we use is differentiable, information embedding can be achieved by adding perturbations to the audio via backpropagation. A reasonable solution for satisfying feature invariance is to train the f_{cf} using augmentations of training data.

Data augmentation during training: Randomized data augmentation can be performed on the audio, and the transformations involved are shown in Table 1. The audio in the dataset is transformed and fed into the network for training to produce a transform-insensitive feature extractor.

Table 1. The transformations or attacks used in this study. The transformations labeled * are not applied to the training of the PTM.

Abbreviation	Attack	Description
CLP	Closed loop	Audio is not attacked.
WGN	White Gaussian noise	Gaussian noise with SNR of 20 dB is added.
LPF	Low-pass filter	Audio is filtered, with 8 kHz cutoff.
HPF	High-pass filter	Audio is filtered, with 100 Hz cutoff.
CRP	Random cropping	Audio is cropped randomly, retaining 80% of duration.
RVB	Reverberation	Simulating audio propagation in a closed room.
DEN *	Simulated denoising	A combination of WGN and LPF.
REC *	Simulated recording	A combination of RVB, WGN, LPF, and HPF.

3.2.2. Contrastive Learning Model for Audio Features

Although a supervised classification pre-trained model can improve feature invariance by data augmentation, it also suffers from semantic collapse, in that the supervised model will tend to learn only the information needed for classification [54]. Therefore, we used contrastive learning [55] to obtain another self-supervised PTM.

The structure of the PTM is the same as the feature extractor in Section 3.2.1; however, we used a different training scheme which consists of four main components:

- (1) A randomized data augmentation produces two augmented counterparts of the same audio track; the data augmentation is shown in Table 1.
- (2) An audio feature extractor maps the augmented audio to the feature domain, which will also function as the PTM in our watermarking system.
- (3) A projection operator $f_p(\cdot)$ obtains audio representations based on audio features, and the representations are used to compute the contrastive loss. The projection operator consists of two linear layers and an activation layer as

$$\mathbf{z} = f_p(\mathbf{o}_{ptm}) = \mathbf{W}_{p,2}\text{ReLU}(\mathbf{W}_{p,1}\mathbf{o}_{ptm}) \quad (2)$$

where $\mathbf{W}_{p,1}$ and $\mathbf{W}_{p,2}$ are the parameters of linear layers.

- (4) A contrastive loss function is used to compute the similarity of audio representations, where different augmented counterparts of the same audio have high similarity and different audio have low similarity. Herein, normalized temperature-scaled cross entropy (NT-Xent) [56] is adopted as the loss function.

Figure 5 illustrates the complete training scheme. In contrastive learning, augmented counterparts of one audio sample are mutually positive samples, e.g., $(\mathbf{x}_1(n), \mathbf{x}_2(n))$ and $(\mathbf{y}_1(n), \mathbf{y}_2(n))$ are two pairs of positive samples. The other audio samples are referred to as negative samples. The NT-Xent loss for a positive pair of audio (i, j) is defined as

$$\ell_{i,j} = -\log \frac{\exp(\text{sim}(\mathbf{z}_i, \mathbf{z}_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{I}_{k \neq i} \exp(\text{sim}(\mathbf{z}_i, \mathbf{z}_k)/\tau)} \quad (3)$$

where $\mathbb{I}_{k \neq i} \in \{0, 1\}$ is an indicator evaluating to 1 iff $k \neq i$ and τ denotes a temperature parameter that helps the loss to control the sensitivity to negative samples. The loss is computed for all positive pairs in a mini-batch, both (i, j) and (j, i) . More details on training for contrastive learning can be found in the literature [55,57].

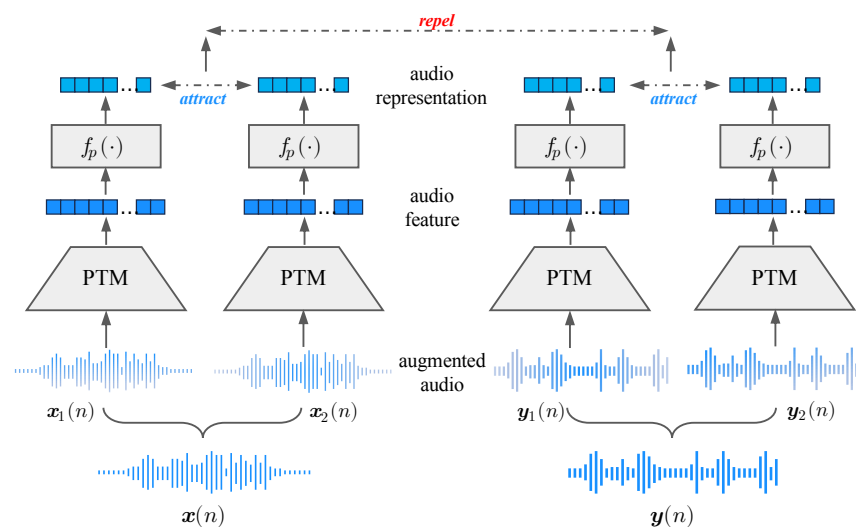


Figure 5. A training scheme for contrastive learning models. Different augmentations of the same audio end up with similar features and representations, but with less similarity than that of another audio sample. The PTMs have shared weights, and so do the projection operators.

3.3. Whitening Layer

The features output by the neural network possess many nice properties, but there is no guarantee that these feature elements are mutually independent. During watermark embedding, the loss function $\mathcal{L}$ closes the distance between the feature and the watermark, so the elements of the feature will be modified. If the elements are correlated, the modification of one element will affect the other elements and the watermark may fail to embed.

On the other hand, the dimensionality of the output features of the pre-trained network is fixed. However, the dimensionality of the watermark is variable with different scenarios. Retraining the model for each scenario is tedious and inefficient, while aligning the watermark by complementary zeros may waste embedding capacity and be ambiguous.

To solve the above problems, this paper introduces principal component analysis (PCA) to whiten the features of PTM [58]. The features of all the audio in the training set $\{\mathbf{o}_{ptm,i}\}_{i=1}^N$ are used as samples, and their covariance matrix $\mathbf{S}$ is computed.

$$\mathbf{S} = \frac{1}{N-1} \sum_{i=1}^N (\mathbf{o}_{ptm,i} - \bar{\mathbf{o}}_{ptm})(\mathbf{o}_{ptm,i} - \bar{\mathbf{o}}_{ptm})^T \quad (4)$$

where $\bar{\mathbf{o}}_{ptm}$ denotes the mean of the features $\{\mathbf{o}_{ptm,i}\}_{i=1}^N$ and the superscript T denotes the transpose.

An eigenvalue decomposition is applied to the covariance matrix $\mathbf{S}$ to obtain the eigenvalues λ_i s and the corresponding eigenvectors $\mathbf{v}_i$ s. The eigenvectors are sorted

descending by their eigenvalues, and then the first k eigenvectors $\{\mathbf{v}_i\}_{i=1}^k$ and eigenvalues $\{\lambda_i\}_{i=0}^k$ are selected to calculate the parameters of the whitening layer $\mathbf{W}_{wl}$.

$$\mathbf{W}_{wl} = \text{diag}(\sqrt{\lambda_1}, \sqrt{\lambda_2}, \dots, \sqrt{\lambda_k})(\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_k) \quad (5)$$

It is noted that the eigenvectors are involved in Equation (5) as column vectors. The operation of the whitening layer is

$$\mathbf{o}_{wl} = f_{wl}(\mathbf{o}_{ptm}) = \mathbf{W}_{wl}^T \mathbf{o}_{ptm} \quad (6)$$

where $\mathbf{o}_{wl}$ is the output of the whitening layer and $\mathbf{o}_{ptm}$ is the output of the pre-trained module.

The whitening layer keeps the elements of $\mathbf{o}_{wl}$ statistically independent from each other, which avoids the interference among elements during the watermark embedding. Moreover, it can be noted that $\mathbf{o}_{wl}$ has k elements. Hence, when the dimensionality of the watermark changes, only the eigenvectors in the matrix $\mathbf{W}_{wl}$ need to be increased or decreased without training the PTM again.

3.4. Watermark Embedding and Attack Simulation Layer

3.4.1. Theoretical Description

In the framework of the AAW algorithm of Figure 2 (the attack simulation layer is ignored here for the moment), the output of the whitening layer $\mathbf{o}_{wl}$ can be expressed as

$$\mathbf{o}_{wl} = f_{wl}(f_{ptm}(\mathbf{x}(n) + \delta(n))) \quad (7)$$

where $\mathbf{x}(n)$ is the given host audio and both f_{ptm} and f_{wl} are parameter-fixed in the watermarking system. It can be concluded that $\mathbf{o}_{wl}$ is a function of δ . Thus, Equation (7) can be rewritten as

$$\mathbf{o}_{wl} = f_{\Theta}(\mathbf{x} + \delta(n)) \quad (8)$$

where Θ denotes all other parameters, including those of all modules. The index identifier (n) of the audio sequence is omitted here. It is noted that f_{ptm} and f_{wl} are all differentiable functions, so $\mathbf{o}_{wl}$ is differentiable with respect to δ .

The loss function $\mathcal{L}(\mathbf{o}_{wl}, \mathbf{w})$, which evaluates the distance between the output of the whitening layer and the watermark, is also a differentiable function. Thus, there are

$$\frac{\partial \mathcal{L}}{\partial \delta} = \frac{\partial \mathcal{L}}{\partial \mathbf{o}_{wl}} \frac{\partial \mathbf{o}_{wl}}{\partial \delta} \quad (9)$$

Equation (9) shows that tiny perturbations $\delta(n)$ can be optimized by backpropagation to reduce loss $\mathcal{L}$.

3.4.2. Loss Function and Attack Simulation Layer

The embedded watermark $\mathbf{w}$ is a binary string, which facilitates content encoding but has some inconveniences in the AAW algorithm. Therefore, the watermark is polarized by

$$\text{polar}(\mathbf{w}) = \begin{cases} 1, & w_i = 1 \\ -1, & w_i = 0 \end{cases} \quad (10)$$

where w_i is a bit in the watermark binary string.

Embedding a watermark requires a loss function $\mathcal{L}$ to guide the optimization of tiny perturbations. The loss function consists of two parts, including $\mathcal{L}_w$ that guides the watermark embedding and $\mathcal{L}_a$ that constrains imperceptibility.

The $\mathcal{L}_w$ measures the distance between the extracted feature and the polarized watermark. We perform a sigmoid-like transform on the extracted feature and then compute the Euclidean distance between it and the polarized watermark. Since the loss function $\mathcal{L}_w$

will introduce a component about attacks later, $\mathcal{L}'_w$ is used here first to denote the distance between vectors.

$$\begin{aligned}\mathcal{L}'_w(\mathbf{x}, \delta, \mathbf{w}) &= \|0.8\text{polar}(\mathbf{w}) - s(\mathbf{o}_{wl})\|_2^2 \\ &= \|0.8\text{polar}(\mathbf{w}) - s(f_{\Theta}(\mathbf{x} + \delta))\|_2^2\end{aligned}\quad (11)$$

where $s(\mathbf{a}) = \frac{1 - \exp(-\mathbf{a})}{1 + \exp(-\mathbf{a})}$ maps the output of whitening layer $\mathbf{o}_{wl}$ into the interval $(-1, 1)$, where $\mathbf{a}$ is the dummy element representing the input to the function. The polarized watermark is diminished by a factor of 0.8 in order to avoid the gradient vanishing during the iteration process.

As mentioned before, in order to make the extracted features invariant against various audio attacks, the pre-trained PTM (both the classification model and the contrastive learning model) introduce multiple audio transformations as data augmentation. The watermarking system can also use an attack simulation layer to transform the audio to improve the robustness of the watermark.

We consider a set $\mathcal{T}$ of audio attacks that contains combinations of attacks in addition to the transformations included in Table 1, as described in more detail in Section 4.1. The attack simulation layer will randomly select a transformation/attack from this set and apply it to the sum of the host audio and tiny perturbation $\mathbf{x}(n) + \delta(n)$.

Considering the attack simulation layer, the loss function can be stated as

$$\mathcal{L}_{w,t} = \|0.8\text{polar}(\mathbf{w}) - s(f_{\Theta}(f_{asl}(\mathbf{x} + \delta; t)))\|_2^2 \quad (12)$$

where $f_{asl}(\mathbf{a}, t)$ denotes the application of an attack $t \in \mathcal{T}$ to the audio $\mathbf{a}$. Since the attacks are randomly chosen by ASL, $\mathcal{L}_{w,t}$ needs to be averaged according to various attacks $t \in \mathcal{T}$.

$$\mathcal{L}_w = \mathbb{E}_{t \in \mathcal{T}}[\mathcal{L}_{w,t}] \quad (13)$$

If an attack is considered important or more frequent, it can be assigned a greater sampling probability. However, in this paper, all attacks in $\mathcal{T}$ are assumed to be equally probable.

On the other hand, it is anticipated that the difference between original and watermarked audio will be imperceptible. Therefore, a loss function $\mathcal{L}_a$, which measures the difference between the watermarked audio and the host audio, is proposed. To make this difference more accurate to the audio quality, a frequency-weighted segmental signal-to-noise ratio (fwsSNR) [59] is adopted as the loss function.

$$\mathcal{L}_a = \frac{10}{N_s} \sum_{i=1}^{N_s} \frac{\sum_k |X_i(k)|^2 \log_{10} \frac{|\Delta_i(k)|^2}{|X_i(k)|^2}}{\sum_k |X_i(k)|^2} \quad (14)$$

where $X_i(k)$ and $\Delta_i(k)$ are the FFT amplitudes corresponding to the host audio and tiny perturbation, N_s is the number of non-overlapped frames, and i is the index of the frames. $\mathcal{L}_a$ provides a more accurate evaluation of imperceptibility than the mean squared error (MSE) loss or signal-to-noise ratio (SNR) metrics because it adaptively weights different frequency bands.

Consequently, the loss function of watermark embedding is as follows:

$$\mathcal{L} = \mathcal{L}_w + \lambda \mathcal{L}_a \quad (15)$$

where the term $\mathcal{L}_w$ pushes $\mathbf{o}_{wl}$ to the designated point (watermark) and ensures that the watermark can be extracted after various attacks; furthermore, the term $\mathcal{L}_a$ keeps the watermark imperceptible. The $\mathcal{L}_w$ and $\mathcal{L}_a$ are mutually constrained; when $\mathcal{L}_a$ reaches negative infinity, it can be assumed that there is no change in the host audio, while $\mathcal{L}_w$ will be large; conversely, if the watermarked audio changes very drastically, the watermark can be extracted more accurately, yet $\mathcal{L}_a$ will be very large. Therefore, the parameter λ can control the trade-off between imperceptibility and robustness.

3.4.3. Watermark Embedding

Watermark embedding is achieved by finding the δ that minimizes the loss function $\mathcal{L}$.

$$\tilde{\delta} = \underset{\delta}{\operatorname{argmin}} \mathcal{L} \quad (16)$$

Since all simulated attacks in $\mathcal{T}$ and all modules in the watermarking system are differentiable, a gradient descent-based optimizer can be adopted to solve Equation (16). The watermarked audio is $\mathbf{x}_w(n) = \mathbf{x}(n) + \tilde{\delta}(n)$.

Watermark embedding is an iterative process. If the imperceptibility constraint is too strong at the early stage of the iteration, it may cause the watermark to sacrifice robustness. Therefore, we dynamically adjusted the importance of imperceptibility versus robustness during the iterative process. In detail, robustness is more important in the first stage of the iteration, enabling AAW to focus more on watermark embedding. At this stage, we set the parameter $\lambda = 0.5$. In the second stage, we balance the imperceptibility and robustness of the watermark, so $\lambda = 1$. Finally, in the third stage, the imperceptibility is slightly strengthened, so $\lambda = 1.5$. The iterations in this paper lasted a total of 300 loops, of which the first stage lasted 150 loops and the second and third stages lasted 100 and 50 loops, respectively.

3.5. Watermark Extraction

The watermark extraction is much simpler than the embedding, which is shown in Figure 6.

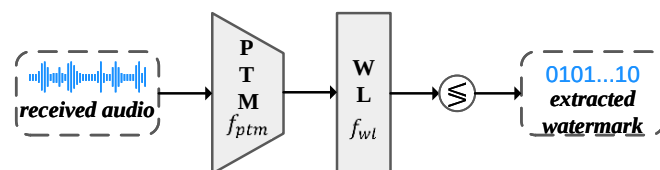


Figure 6. Diagram of the AAW algorithm (extraction stage), where PTM, WL, and $\leq$ denote the pre-trained module, whitening layer, and a binary decision maker, respectively.

The extraction end firstly receives the audio $\tilde{\mathbf{x}}_w(n)$ that needs the watermark extracted. Subsequently, the received audio is fed into the cascade model of PTM and WL to obtained the whitened feature. The feature is then passed through a binary decision maker with a threshold of 0, and its output is the extracted watermark. The above procedure can be formulated as follows:

$$\tilde{\mathbf{w}} = \text{binary}(f_{\Theta}(\tilde{\mathbf{x}}_w(n))) \quad (17)$$

where $f_{\Theta}(\cdot)$ is defined as in Equation (8), and

$$\text{binary}(\mathbf{a}) = \begin{cases} 1, & a_i \geq 0 \\ 0, & a_i < 0 \end{cases}$$

where a_i is an element in the input vector.

It can be argued that the whitened feature lies in a high-dimensional space and the polarized watermarks are vertices of a hypercube in that space. Equation (17) shows that the vertex nearest to the whitened feature is the extracted watermark.

4. Experiments and Discussion

4.1. Experimental Settings and Implementation Details

Dataset: The classification network was trained and evaluated in the FMA-medium dataset [60]. The dataset contains 25,000 audio tracks, of which 19,922 belong to the training set, 2505 tracks to the validation set, and 2573 tracks to the test set. The dataset consists of 16 categories; however, the quantities of tracks in each category are unbalanced. The

training set and test set was used for training the PTM, and the validation set were used for the watermark embedding experiments. In addition, we also collected 100 speech clips taken from movies' dialog for watermark embedding, containing 60 Chinese and 40 English clips. All tracks are two-channel stereo audio with a sample rate of 44.1 kHz and a duration time of 30 s.

Experimental settings: During watermark embedding, the Adam [61] optimizer with a learning rate of 0.001 was used to solve Equation (16) over 300 iterations. The parameter λ in Equation (15) was set dynamically as mentioned in Section 3.4.3.

Transformations for data augmentations and attack simulation layer: Audio transforms are widely applied to several scenarios of AAW algorithms, including data augmentation during PTM training, the attack simulation layer during watermark embedding, and attack during the evaluation of algorithm performance. The transformations adopted in this paper are shown in Table 1.

It is noted that the reverberation was simulated by Pyroomacoustics [62] in a $9\text{ m} \times 7.5\text{ m} \times 3.5\text{ m}$ rectangular room. To avoid the difficulty of gradient computation, we used masking with zeros instead of the random crop when training the network and watermark embedding. When evaluating the proposed method, the actual random crop was applied.

In addition, we also tested the robustness of the AAW to two combinations of transforms, which are the combination of Gaussian white noise with low-pass filter and the pipeline of reverberation-Gaussian noise-band-pass filter, where the band-pass filter was obtained by joining the low-pass and high-pass filters in cascade. The former takes into account the fact that most of the audio attacked by Gaussian noise degrades significantly in quality, and low-pass filter has good denoising ability, so this combination is often used; the latter simulates audio propagating over air and a recording. It is noted that these two combined attacks are not applied to the PTM's training.

Metrics: The imperceptibility of the watermark is evaluated by the signal-to-watermark ratio (SWR), which means the energy ratio of perturbation to audio,

$$\text{SWR} = -10 \log_{10} \frac{\|\delta\|_2^2}{\|\mathbf{x}\|_2^2} \quad (18)$$

The larger the SWR, the better the imperceptibility of the watermark.

The bit detection rate (BDR) is used to evaluate the robustness of the proposed algorithm,

$$\text{BDR} = \frac{\# \text{ of bits extracted}}{\# \text{ of total bits}} \times 100\% \quad (19)$$

where the symbol # indicates the quantity. A higher BDR indicates that more bits of the watermark were accurately extracted. In the experiments, the quantity of total bits was set to 320 bits unless otherwise specified. This means that the transform parameter $\mathbf{W}_{wl}$ of the whitening layer has a shape of 512×320 .

4.2. Pre-Trained Modules

The classification network predicts the audio as belonging to 1 of 16 categories according to musical genres, and the cross-entropy loss is employed for training. The Adam optimizer with a learning rate of 0.001 is also employed for network training, and training is terminated when the training loss and test loss cease dropping. Training with augmented data was iterated with 500 epochs. The trained network classified the audio in the test set with an accuracy of 64.40% (top 1) and 84.76% (top 3). It can be argued that the pre-trained module (the feature extractor in the classification network) can extract features about the audio genres.

We adopted the same training strategy (optimizer and termination conditions) on the contrastive learning-based audio feature extractor. It was trained for a total of 3000 epochs, where the size of the mini-batch was 32. We tested this audio feature extractor only on the

audio genre classification task. Specifically, a multilayer perceptron (MLP) (linear classifier appended with an extra linear layer) served as the fine-tuning head and was trained for 120 epochs on the training set. On the test set, the feature extractor and MLP achieved accuracies of 59.58% (top 1) and 82.43% (top 3).

Later in this paper, we use CN and CL instead of classification network and contrastive learning-based module, respectively.

4.3. Ablation Studies

This subsection shows the influence of the proposed attack simulation layer (ASL) and whitening layer (WL). Herein, the experiments that did not adopt ASL were realized by substituting all the attack types with CLP. Moreover, the experiments that did not use WL were realized by using only the first 320 dimensions of the 512-dimensional PTM features to calculate the loss function. The experimental results are shown in Table 2.

Table 2. The influence of the attack simulation layer (ASL) and whitening layer (WL). The ✓ indicates that the corresponding module was used in the experiment.

PTM	ASL	WL	SWR (dB)	BDR (%)							
				CLP	WGN	LPF	HPF	CRP	RVB	DEN	REC
CN	✓		43.10	92.78	50.52	74.82	77.50	81.19	80.92	67.46	64.08
			38.43	90.41	57.37	92.91	94.00	88.50	91.30	82.73	79.47
		✓	42.87	99.56	52.31	80.55	81.17	82.88	83.27	69.64	66.79
		✓	37.64	98.62	58.27	94.78	96.29	90.97	94.11	86.02	81.43
CL	✓		42.93	93.04	51.62	76.72	79.46	84.59	84.81	69.40	66.29
			38.23	89.95	58.59	93.30	95.27	92.86	92.52	85.26	81.11
		✓	41.79	99.87	50.91	79.39	82.25	86.06	86.57	71.51	68.25
		✓	37.53	98.93	58.75	95.20	97.13	93.47	95.19	89.22	83.29

As can be seen from Table 2, the ASL decreases the imperceptibility of the watermark, but the SWR of up to 37.53 dB (the worst) still demonstrates superior imperceptibility. However, the improvement of algorithm robustness by ASL cannot be ignored. Across the multiple attacks involved in this paper, the ASL improves the BDR of extracted watermarks to different degrees, from the lowest of 5.96% (WGN) to the highest of 18.09% (HPF). Interestingly, the BDR for extracting the watermark is slightly decreased in the absence of an attack. This is because the AAW algorithm with ASL needs to consider more attacks; however, there is only one scenario, CLP, considered in the experiments without ASL. Therefore, it can be argued that the ASL significantly improves the robustness of watermarking while maintaining the imperceptibility at an extremely optimal level.

On the other hand, it can be seen that the WL substantially improves the BDR of the extracted watermark when the watermark is not under attack (CLP). However, in other attacks, it only marginally improves the BDR and also marginally decreases the imperceptibility of the watermark. In fact, the WL is not proposed to improve the robustness of watermarking. The main purpose of WL is to make the elements of the audio feature mutually independent and to keep these elements from affecting each other during the watermark embedding. It is argued that the WL decreases the failure rate of watermark embedding, which is reflected in the improvement in BDR when the watermarked audio is not attacked. In addition, as mentioned before, there is another important role of the WL, which is that it makes it easier to adjust the quantity of watermark bits.

The different PTMs (CN or CL) have an influence on the robustness of the AAW algorithm, which is more pronounced in scenarios with attacks than in CLP. It can be argued that the audio representations extracted by CL are more instance-focused, whereas CN is more category-focused. Therefore, AAW-CL is more robust than that of CN, which is why CL performs slightly weaker in the classification task.

In a nutshell, the effectiveness of the AAW algorithm was validated for two different pre-trained modules, CN and CL. Furthermore, it is also demonstrated experimentally that

the attack simulation layer and whitening layer proposed in this paper can improve the performance of the AAW algorithm.

4.4. Robustness Against Attacks

In Equation (13), all attacks are assumed to be equally probable. However, the robustness of watermarking is inconsistent for various attacks in the experimental results of Table 2. Figures 7–9 are examples of experiments with a host audio. Figure 7 shows the waveform of the host audio and the embedded watermark. Figures 8 and 9 show experimental visualizations of algorithms AAW-CN and AAW-CL against different attacks, respectively.

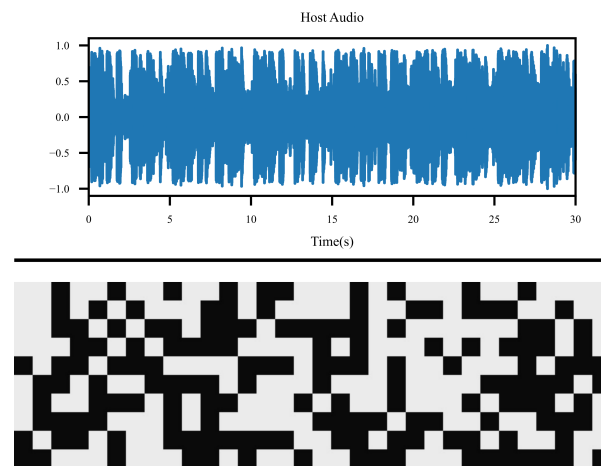


Figure 7. The waveform of the host audio (**top**) and the watermark (**bottom**). The host audio is ‘014391.mp3’ from the FMA dataset. The watermark is shown as a 32×10 binary image. Since the watermark is randomly generated in the experiment, this image has no specific meaning.

It can be seen from Figures 8a and 9a that the influences of the embedding algorithms on the waveform are negligible, with a difference of only 37 dB from the host audio. However, some attacks very significantly alter the waveform of the audio.

The CLP has the highest BDR in all scenarios of the experiment, since all perturbations to the host audio are available to the PTM. Moreover, the LPF, HPF, and RVB also have BDRs of more than 94%. The thing that these attacks have in common is that they are all realized by convolution, so the embedding stage is able to fit such rules to ensure that the watermark passes through these attacks. It is noted that the HPF’s BDR is greater than that of LPF. The MFCC enhances the high frequency component of the audio, whereas the LPF removes the high frequency component, so the MFCC does not perform its full work.

The CRP accurately extracts about 94% of the embedded watermark. The random cropping intercepts and preserves a portion from the audio and does not modify the audio samples of the preserved portion. The experiments with CRP also demonstrate that the PTM is robust to audio samples shifting.

Although the ASL and WL improved the BDR when audio encounters the WGN, the highest BDR is still below 60%. Indeed, the watermark extraction stage can be regarded as a multi-classification task. The boundaries of different categories in the feature domain are very rugged, and this rugged decision boundary makes the embedding distortion SWR small, but its robustness to random noise becomes poor. The WGN is pointwise-randomized, whereas none of the other attacks mentioned previously are pointwise-randomized, and thus their BDRs are stronger than that of the WGN.

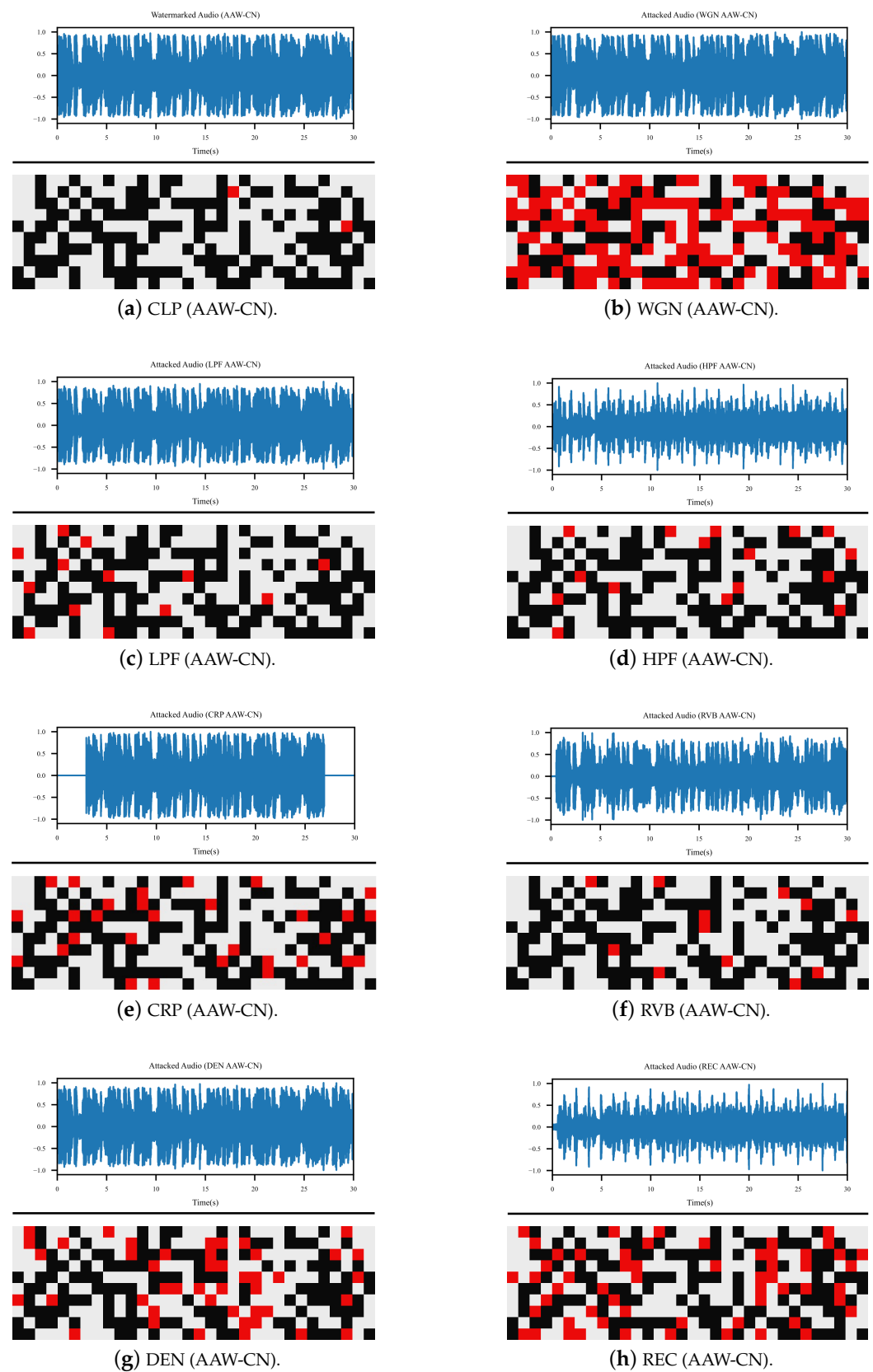


Figure 8. Experimental example of AAW-CN algorithm. Each subfigure represents an attack, with the waveform of the attacked audio at the top and the extracted watermark at the bottom, where the red indicates the bits that were extracted incorrectly.

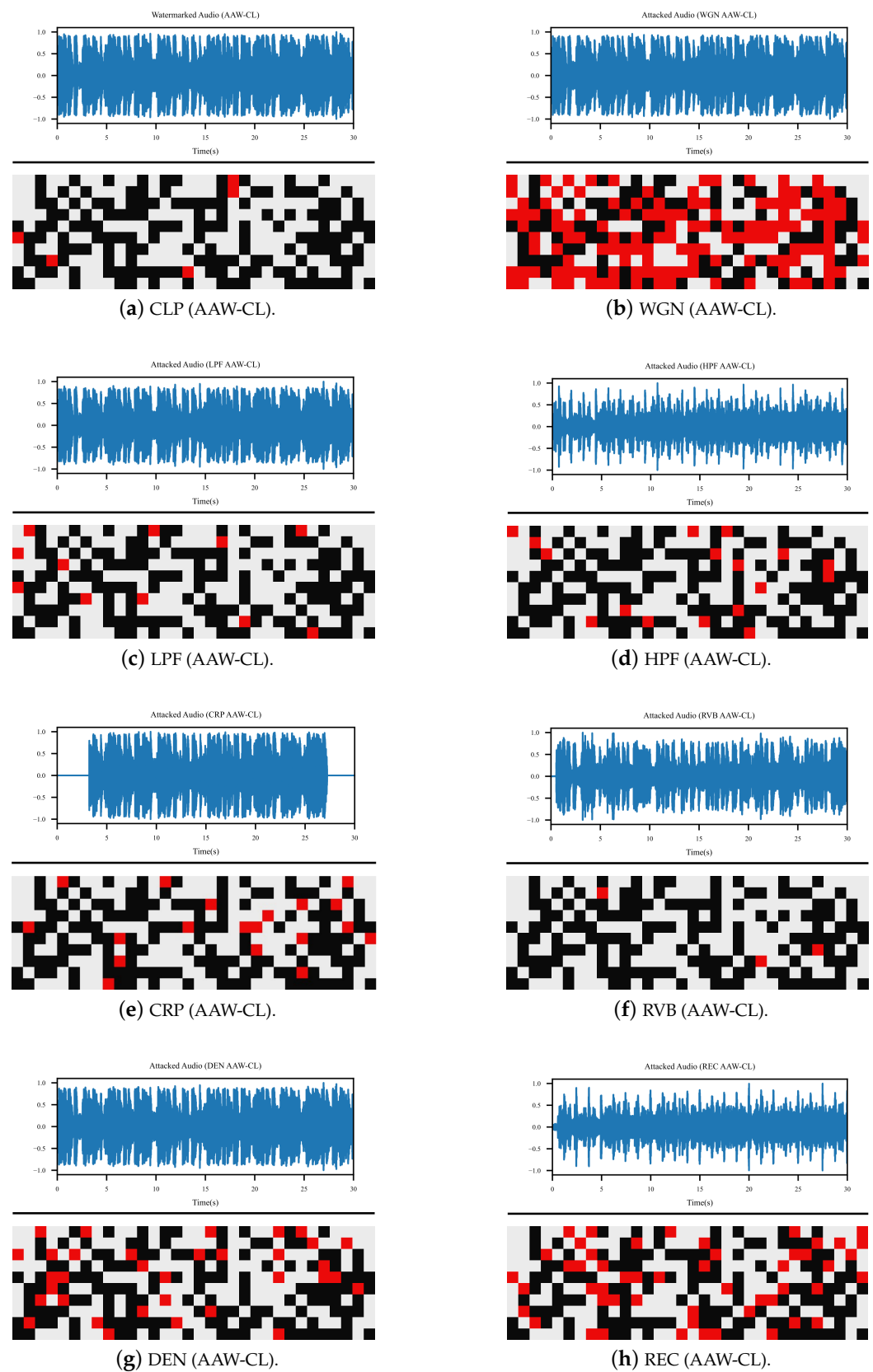


Figure 9. Experimental example of AAW-CL algorithm. Each subfigure represents an attack, with the waveform of the attacked audio at the top and the extracted watermark at the bottom, where the red indicates the bits that were extracted incorrectly.

It can be seen that both DEN and REC attacks also contain WGN attacks yet achieve decent BDRs. It is argued that the filtered Gaussian noise produces a predictable effect on the audio, and thus the robustness to watermarking is weakened but not as drastically as with WGN alone.

4.5. Performance on Different Audio Categories

We also evaluated the performance of AAW on different audio categories. We first applied the AAW algorithm to different categories of audio. The quantity of audio tracks in each category is shown in Table 3. Statistically, we measured the SWR and the watermark BDR under the eight attacks for the different categories. The experimental results are shown in Figures 10–12.

Table 3. The quantity of audio tracks in each category. The first of each two rows of the table (between the horizontal lines) is the category of the audio, and the second is the quantity of the audio. Because there are so many categories, the table is collapsed into three parts.

Category	Rock	Electronic	Experimental	Hip-hop	Instrumental	Folk
Quantity	711	632	225	220	131	152
Category	Pop	International	Classical	Historic	Soul	Jazz
Quantity	122	102	62	51	18	39
Category	Country	Spoken	Blues	Easy listening	Speech	
Quantity	18	12	8	2	100	

From Figure 10, it can be seen that the imperceptibility (SWR) of both AAW-CN and AAW-CL varies slightly among categories of audio, but the difference is not significant. This is because the $\mathcal{L}_a$ (shown in Equation (14)) contains audio information, so the tiny perturbations are tailored to each audio track. Hence, the imperceptibility does not change significantly depending on the audio category.

The curves corresponding to the eight attacks in Figures 11 and 12 show almost no fluctuations, indicating that the audio category also does not affect the robustness of both AAW-CN and AAW-CL. Therefore, it can be argued that the AAW algorithm is not sensitive to audio categories. The quantity of tracks in the categories ‘spoken’, ‘blues’ and ‘easy listening’ is very small. Although it appears in the figure that their experimental results differ significantly from the other categories, this does not contradict the conclusions.

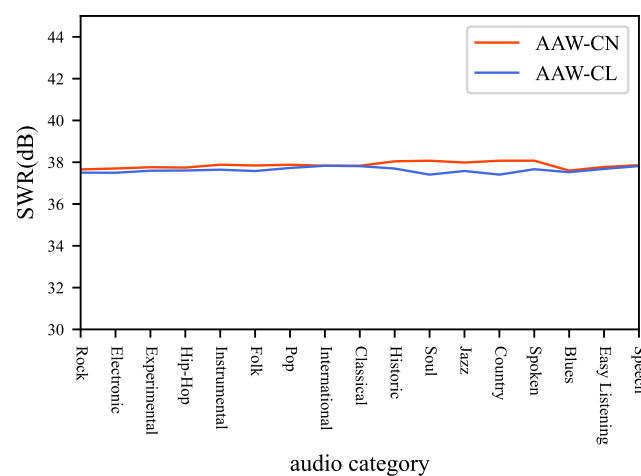


Figure 10. The imperceptibility (SWR) when embedding watermark into different audio categories.

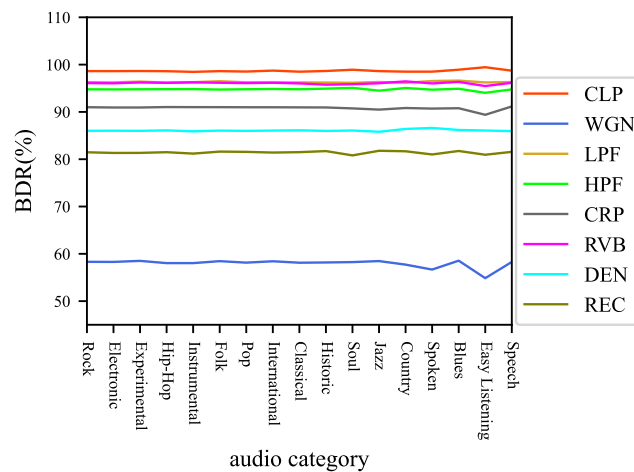


Figure 11. The robustness (BDR) of AAW-CN when embedding watermark into different audio categories.

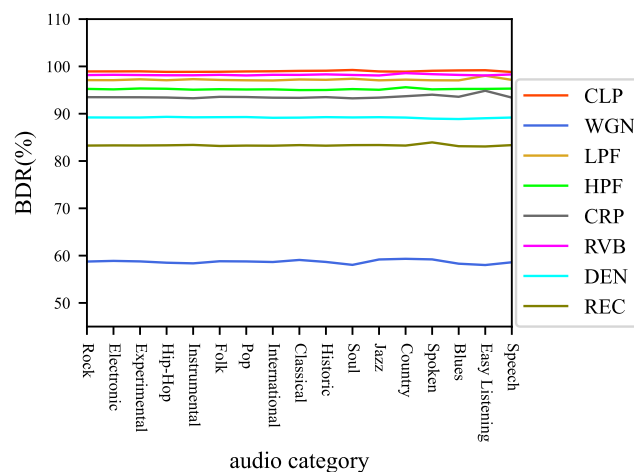


Figure 12. The robustness (BDR) of AAW-CL when embedding watermark into different audio categories.

4.6. Comparison with Existing Works

We compare the imperceptibility and robustness of the AAW algorithm with other existing works. Since deep learning-based audio watermarking algorithms are scarce in the literature, we only found the work by Kong et al. [38] and Liu et al. [9]. In this paper, we replace the 2D convolution layers in the network structure of HiDDeN [18], a landmark work for deep learning-based image watermarking, with 1D convolution layers that enable it to be applied to audio watermarking, and we call it HiDDeN-A. The experiments use the FMA-medium dataset to train HiDDeN-A and then compare its performance with that of the AAW algorithm. In addition, we compare the algorithm of Wu et al. [13], which is a traditional audio watermarking algorithm.

The comparisons are shown in Table 4. The SWR of the AAW algorithm is larger than any of the other compared algorithms, indicating that the introduced perturbation has a very small magnitude. This is facilitated by the larger λ used in the third stage of the optimization for tiny perturbations $\hat{\delta}$. On the other hand, the AAW algorithm uses fswSNR as a loss, i.e., it is optimized accordingly for the human auditory system. Therefore, the AAW algorithm can be argued to have improved imperceptibility over comparison algorithms, which prevents embedded watermarks from damaging the usage of the audio or from being detected.

Table 4. The performance of proposed and other algorithms.

Algorithm	SWR (dB)	BDR (%)							
		CLP	WGN	LPF	HPF	CRP	RVB	DEN	REC
AAW-CN	37.64	98.62	58.27	96.29	94.78	90.97	96.11	86.02	81.43
AAW-CL	37.53	98.93	58.75	97.13	95.20	93.47	98.19	89.22	83.29
Kong [38]	30.17	100	0.21	0.83	1.26	2.18	1.07	0.36	0.13
HiDDeN-A	28.24	88.73	47.93	68.49	71.46	79.78	52.48	74.51	58.22
Liu [9]	26.14	98.84	86.92	94.43	95.65	60.24	63.41	74.66	91.87
Wu [13]	24.55	99.38	98.42	99.37	52.67	50.11	49.52	76.33	47.29

The robustness of the AAW algorithm also yielded good results. Our algorithm achieved a superior BDR than the comparison algorithms except in the WGN attack. The data embedded in the algorithm of Kong et al. [38] comprise a piece of text which can hardly be extracted accurately after being attacked. This is due to the fact that the algorithm is applied in the field of information hiding; hence, it does not use any robustness enhancement strategy, so its robustness is very poor. The performance of HidDeN-A is worse than that of HiDDeN in the image domain, indicating that additional efforts are needed for directly transferring the image watermarking to audio. Liu et al. [9] converted the audio into 2D features and then used the HiDDeN framework. However, they only modeled the audio recording attack and did not pay enough attention to other attacks, so the robustness against the attacks such as CRP, RVB, and WGN+LPF is worse than that of the AAW algorithm. The AAW algorithm also provides more comprehensive robustness against multiple attacks than traditional algorithms [13].

We argue that the performance advantage of the AAW algorithm is due to two factors. One is that the whitening layer contributes to robustness, with uncorrelated elements making it easier to embed watermarks into deep features. The other is that the attack simulation layer makes AAW possible to incorporate other differentiable attacks. It allows AAW to deal with a wider range of attacks rather than hand-designing strategies for each attack.

5. Summary

In this paper, we propose an audio watermarking algorithm, AAW, based on adversarial perturbation. The AAW relies only on a pre-trained differentiable decoder instead of an encoder–decoder framework for watermark embedding and extraction. The watermark embedding is realized by adding tiny perturbations to the host audio. In addition, we adopt an attack simulation layer and a whitening layer to improve the performance of the AAW algorithm. Finally, the experimental results demonstrate that the proposed algorithm is effective and has a performance advantage over existing deep learning-based algorithms.

However, our work still requires improvement. First, compared to an encoder–decoder deep watermarking framework, the AAW is computationally expensive since it is not a single pass forward. In addition, the AAW algorithm is not yet resistant to WGN and some non-differentiable attacks such as lossy compression. In future works, we hope to be able to propose solutions to improve the above problems.

Author Contributions: Conceptualization, S.W. and Y.H.; methodology, S.W.; Software, S.W.; validation, S.W., Y.H., and H.G.; formal analysis, S.W.; investigation, S.W.; resources, S.W.; data curation, S.W.; writing—original draft preparation, S.W.; writing—review and editing, S.W., Y.H., H.G., and S.Z.; visualization, S.W.; supervision, S.Z. and J.L.; project administration, H.G., S.Z., and J.L.; funding acquisition, S.Z. and J.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Key R&D Program of China (2022YFF0901900) and the Key R&D Program of Shanxi (202102010101004).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Acknowledgments: It was also the research achievement of the Beijing Digital Content Engineering Research Center and the Key Laboratory of Digital Rights Services.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AAW	Audio watermarking based on adversarial perturbation
ASL	Attack simulation layer
PTM	Pre-trained model (module)
WL	Whitening layer
MFCC	Mel frequency cepstrum coefficient
PCA	Principal component analysis
fwsSNR	Frequency-weighted segmental signal-to-noise ratio
SWR	Signal-to-watermark ratio
BDR	Bit detection rate
CN	Classification network
CL	Contrastive learning-based module

References

1. Lv, Z. Generative artificial intelligence in the metaverse era. *Cogn. Robot.* **2023**, *3*, 208–217. [\[CrossRef\]](#)
2. Xu, D.; Fan, S.; Kankanhalli, M. Combating misinformation in the era of generative AI models. In Proceedings of the The ACM International Conference on Multimedia, Ottawa, ON, Canada, 29 October–3 November 2023; pp. 9291–9298.
3. Hua, G.; Huang, J.; Shi, Y.Q.; Goh, J.; Thing, V.L. Twenty years of digital audio watermarking—A comprehensive review. *Signal Process.* **2016**, *128*, 222–242. [\[CrossRef\]](#)
4. Wan, W.; Wang, J.; Zhang, Y.; Li, J.; Yu, H.; Sun, J. A comprehensive survey on robust image watermarking. *Neurocomputing* **2022**, *488*, 226–247. [\[CrossRef\]](#)
5. Asikuzzaman, M.; Pickering, M.R. An overview of digital video watermarking. *IEEE Trans. Circuits Syst. Video Technol.* **2017**, *28*, 2131–2153. [\[CrossRef\]](#)
6. Fang, H.; Zhang, W.; Zhou, H.; Cui, H.; Yu, N. Screen-shooting resilient watermarking. *IEEE Trans. Inf. Forensics Secur.* **2018**, *14*, 1403–1418. [\[CrossRef\]](#)
7. Cao, F.; Wang, T.; Guo, D.; Li, J.; Qin, C. Screen-shooting resistant image watermarking based on lightweight neural network in frequency domain. *J. Vis. Commun. Image Represent.* **2023**, *94*, 103837. [\[CrossRef\]](#)
8. Lu, W.; Li, L.; He, Y.; Wei, J.; Xiong, N.N. RFPS: A robust feature points detection of audio watermarking for against desynchronization attacks in cyber security. *IEEE Access* **2020**, *8*, 63643–63653. [\[CrossRef\]](#)
9. Liu, C.; Zhang, J.; Fang, H.; Ma, Z.; Zhang, W.; Yu, N. DeAR: A deep-learning-based audio re-recording resilient watermarking. *Aaai Conf. Artif. Intell.* **2023**, *37*, 13201–13209. [\[CrossRef\]](#)
10. Luo, X.; Li, Y.; Chang, H.; Liu, C.; Milanfar, P.; Yang, F. DVMark: A deep multiscale framework for video watermarking. *IEEE Trans. Image Process.* **2023**. [\[CrossRef\]](#)
11. Nadeau, A.; Sharma, G. An audio watermark designed for efficient and robust resynchronization after analog playback. *IEEE Trans. Inf. Forensics Secur.* **2017**, *12*, 1393–1405. [\[CrossRef\]](#)
12. Xiang, Y.; Natgunanathan, I.; Peng, D.; Hua, G.; Liu, B. Spread Spectrum Audio Watermarking Using Multiple Orthogonal PN Sequences and Variable Embedding Strengths and Polarities. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2018**, *26*, 529–539. [\[CrossRef\]](#)
13. Wu, S.; Huang, Y.; Guan, H.; Zhang, S.; Liu, J. ECSS: High-Embedding-Capacity Audio Watermarking with Diversity Reception. *Entropy* **2022**, *24*, 1843. [\[CrossRef\]](#) [\[PubMed\]](#)
14. Zhao, J.; Zong, T.; Xiang, Y.; Gao, L.; Zhou, W.; Beliakov, G. Desynchronization attacks resilient watermarking method based on frequency singular value coefficient modification. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2021**, *29*, 2282–2295. [\[CrossRef\]](#)
15. Wu, S.; Huang, J.; Huang, D.; Shi, Y.Q. Efficiently self-synchronized audio watermarking for assured audio data transmission. *IEEE Trans. Broadcast.* **2005**, *51*, 69–76. [\[CrossRef\]](#)
16. Wang, X.Y.; Niu, P.P.; Yang, H.Y. A robust, digital-audio watermarking method. *IEEE MultiMedia* **2009**, *16*, 60–69. [\[CrossRef\]](#)
17. Li, W.; Xue, X.; Lu, P. Localized audio watermarking technique robust against time-scale modification. *IEEE Trans. Multimed.* **2006**, *8*, 60–69. [\[CrossRef\]](#)
18. Zhu, J.; Kaplan, R.; Johnson, J.; Fei-Fei, L. Hidden: Hiding data with deep networks. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; pp. 657–672.

19. Liu, Y.; Guo, M.; Zhang, J.; Zhu, Y.; Xie, X. A novel two-stage separable deep learning framework for practical blind watermarking. In Proceedings of the ACM International Conference on Multimedia, Munich, Germany, 8–14 September 2019.
20. Jia, Z.; Fang, H.; Zhang, W. Mbrs: Enhancing robustness of dnn-based watermarking by mini-batch of real and simulated jpeg compression. In Proceedings of the ACM International Conference on Multimedia, Chengdu, China, 20–24 October 2021.
21. Luo, X.; Zhan, R.; Chang, H.; Yang, F.; Milanfar, P. Distortion agnostic deep watermarking. In Proceedings of the IEEE/CVF Computer Vision and Pattern Recognition Conference, Seattle, WA, USA, 13–19 June 2020.
22. Zhang, H.; Wang, H.; Li, Y.; Cao, Y.; Shen, C. Robust watermarking using inverse gradient attention. *arXiv* **2020**, arXiv:2011.10850.
23. Yu, C. Attention based data hiding with generative adversarial networks. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020.
24. Ahmadi, M.; Norouzi, A.; Karimi, N.; Samavi, S.; Emami, A. ReDMark: Framework for residual diffusion watermarking based on deep networks. *Expert Syst. Appl.* **2020**, *146*, 113157. [\[CrossRef\]](#)
25. Bassia, P.; Pitas, I.; Nikolaidis, N. Robust audio watermarking in the time domain. *IEEE Trans. Multimed.* **2001**, *3*, 232–241. [\[CrossRef\]](#)
26. Hwang, M.J.; Lee, J.; Lee, M.; Kang, H.G. SVD-based adaptive QIM watermarking on stereo audio signals. *IEEE Trans. Multimed.* **2017**, *20*, 45–54. [\[CrossRef\]](#)
27. Wang, S.; Yuan, W.; Zhang, Z.; Wang, J.; Unoki, M. Synchronous Multi-Bit Audio Watermarking Based on Phase Shifting. In Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, Toronto, ON, Canada, 6–11 June 2021.
28. Cox, I.; Kilian, J.; Leighton, F.; Shamoon, T. Secure Spread Spectrum Watermarking for Multimedia. *IEEE Trans. Image Process.* **1997**, *6*, 1673–1687. [\[CrossRef\]](#) [\[PubMed\]](#)
29. Chen, B.; Wornell, G.W. Quantization index modulation: A class of provably good methods for digital watermarking and information embedding. *IEEE Trans. Inf. Theory* **2001**, *47*, 1423–1443. [\[CrossRef\]](#)
30. Su, Z.; Zhang, G.; Yue, F.; Chang, L.; Jiang, J.; Yao, X. SNR-constrained heuristics for optimizing the scaling parameter of robust audio watermarking. *IEEE Trans. Multimed.* **2018**, *20*, 2631–2644. [\[CrossRef\]](#)
31. Zhang, G.; Zheng, L.; Su, Z.; Zeng, Y.; Wang, G. M-Sequences and Sliding Window Based Audio Watermarking Robust Against Large-Scale Cropping Attacks. *IEEE Trans. Inf. Forensics Secur.* **2023**, *18*, 1182–1195. [\[CrossRef\]](#)
32. Tavakoli, A.; Honjani, Z.; Sajedi, H. Convolutional Neural Network-Based Image Watermarking using Discrete Wavelet Transform. *arXiv* **2022**, arXiv:2210.06179.
33. Vukotić, V.; Chappelier, V.; Furon, T. Are classification deep neural networks good for blind image watermarking? *Entropy* **2020**, *22*, 198. [\[CrossRef\]](#) [\[PubMed\]](#)
34. Fernandez, P.; Sablayrolles, A.; Furon, T.; Jégou, H.; Douze, M. Watermarking Images in Self-Supervised Latent Spaces. In Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, Singapore, 23–27 May 2022; pp. 3054–3058. [\[CrossRef\]](#)
35. Caron, M.; Touvron, H.; Misra, I.; Jégou, H.; Mairal, J.; Bojanowski, P.; Joulin, A. Emerging properties in self-supervised vision transformers. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 9650–9660.
36. Jia, X.; Wei, X.; Cao, X.; Han, X. Adv-watermark: A Novel Watermark Perturbation for Adversarial Examples. In Proceedings of the 28th ACM International Conference on Multimedia, Seattle, WA, USA, 12–16 October 2020.
37. Ghamizi, S.; Cordy, M.; Papadakis, M.; Traon, Y.L. Evasion Attack STeganography: Turning Vulnerability of Machine Learning to Adversarial Attacks into a Real-World Application. In Proceedings of the AAAI Conference on Artificial Intelligence, Philadelphia, PA, USA, 2–9 February 2021.
38. Kong, Y.; Zhang, J. Adversarial Audio: A New Information Hiding Method. In Proceedings of the INTERSPEECH, Shanghai, China, 25–29 October 2020.
39. Goodfellow, I.J.; Shlens, J.; Szegedy, C. Explaining and harnessing adversarial examples. *arXiv* **2014**, arXiv:1412.6572.
40. Moosavi-Dezfooli, S.M.; Fawzi, A.; Frossard, P. Deepfool: A simple and accurate method to fool deep neural networks. In Proceedings of the IEEE/CVF Computer Vision and Pattern Recognition Conference, Las Vegas, NV, USA, 27–30 June 2016; pp. 2574–2582.
41. Dong, Y.; Liao, F.; Pang, T.; Su, H.; Zhu, J.; Hu, X.; Li, J. Boosting adversarial attacks with momentum. In Proceedings of the IEEE/CVF Computer Vision and Pattern Recognition Conference, Salt Lake City, UT, USA, 18–23 June 2018; pp. 9185–9193.
42. Khamaiseh, S.Y.; Bagagem, D.; Al-Alaj, A.; Mancino, M.; Alomari, H.W. Adversarial deep learning: A survey on adversarial attacks and defense mechanisms on image classification. *IEEE Access* **2022**, *10*, 102266–102291. [\[CrossRef\]](#)
43. Machado, G.R.; Silva, E.; Goldschmidt, R.R. Adversarial machine learning in image classification: A survey toward the defender’s perspective. *ACM Comput. Surv.* **2021**, *55*, 1–38. [\[CrossRef\]](#)
44. Peng, B.; Peng, B.; Yong, S.; Liu, L. An empirical study of fully black-box and universal adversarial attack for SAR target recognition. *Remote Sens.* **2022**, *14*, 4017. [\[CrossRef\]](#)
45. Peng, B.; Peng, B.; Zhou, J.; Xie, J.; Liu, L. Scattering model guided adversarial examples for SAR target recognition: Attack and defense. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 1–17. [\[CrossRef\]](#)
46. Carlini, N.; Wagner, D. Audio adversarial examples: Targeted attacks on speech-to-text. In Proceedings of the 2018 IEEE security and privacy workshops (SPW), San Francisco, CA, USA, 24 May 2018; pp. 1–7.

47. Kwon, H.; Kim, Y.; Yoon, H.; Choi, D. Selective audio adversarial example in evasion attack on speech recognition system. *IEEE Trans. Inf. Forensics Secur.* **2019**, *15*, 526–538. [\[CrossRef\]](#)
48. Kwon, H.; Nam, S.H. Audio adversarial detection through classification score on speech recognition systems. *Comput. Secur.* **2023**, *126*, 103061. [\[CrossRef\]](#)
49. Zhang, Y.; Hu, S.; Zhang, L.Y.; Shi, J.; Li, M.; Liu, X.; Jin, H. Why does little robustness help? a further step towards understanding adversarial transferability. In Proceedings of the 45th IEEE Symposium on Security and Privacy (S&P'24), San Francisco, CA, USA, 20–22 May 2024; Volume 2.
50. Zhang, R.; Isola, P.; Efros, A.A.; Shechtman, E.; Wang, O. The unreasonable effectiveness of deep features as a perceptual metric. In Proceedings of the IEEE/CVF Computer Vision and Pattern Recognition Conference, Salt Lake City, UT, USA, 18–23 June 2018; pp. 586–595.
51. Noh, H.; Araujo, A.; Sim, J.; Weyand, T.; Han, B. Large-scale image retrieval with attentive deep local features. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 3456–3465.
52. Babenko, A.; Slesarev, A.; Chigorin, A.; Lempitsky, V. Neural codes for image retrieval. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; pp. 584–599.
53. Oord, A.v.d.; Dieleman, S.; Zen, H.; Simonyan, K.; Vinyals, O.; Graves, A.; Kalchbrenner, N.; Senior, A.; Kavukcuoglu, K. Wavenet: A generative model for raw audio. *arXiv* **2016**, arXiv:1609.03499.
54. Doersch, C.; Gupta, A.; Zisserman, A. Crosstransformers: Spatially-aware few-shot transfer. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 21981–21993.
55. Spijkervet, J.; Burgoyne, J.A. Contrastive learning of musical representations. *arXiv* **2021**, arXiv:2103.09410.
56. Sohn, K. Improved deep metric learning with multi-class n-pair loss objective. *Adv. Neural Inf. Process. Syst.* **2016**, *29*. [\[CrossRef\]](#)
57. Chen, T.; Kornblith, S.; Norouzi, M.; Hinton, G. A simple framework for contrastive learning of visual representations. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual, 13–18 July 2020; pp. 1597–1607.
58. Jégou, H.; Chum, O. Negative evidences and co-occurrences in image retrieval: The benefit of PCA and whitening. *The European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 774–787.
59. Hu, Y.; Loizou, P.C. Evaluation of objective quality measures for speech enhancement. *IEEE Trans. Audio Speech Lang. Process.* **2007**, *16*, 229–238. [\[CrossRef\]](#)
60. Defferrard, M.; Benzi, K.; Vandergheynst, P.; Bresson, X. FMA: A Dataset for Music Analysis. In Proceedings of the International Society for Music Information Retrieval Conference, Suzhou, China, 23–28 October 2017.
61. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. In Proceedings of the International Conference on Learning Representations, San Diego, CA, USA, 7–9 May 2015.
62. Scheibler, R.; Bezzam, E.; Dokmanić, I. Pyroomacoustics: A Python Package for Audio Room Simulation and Array Processing Algorithms. In Proceedings of the 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, AB, Canada, 15–20 April 2018. . [\[CrossRef\]](#)

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.