# Latent Watermarking of Audio Generative Models

Robin San Roman, Pierre Fernandez, Antoine Deleforge, Yossi Adi, Romain Serizel

# Latent Watermarking of Audio Generative Models

Robin San Roman
*Meta, FAIR, Univ. de Lorraine, CNRS, Inria, Loria*

Pierre Fernandez
*Meta, FAIR, Inria Rennes*

Antoine Deleforge
*IRMA, CNRS, Univ. de Strasbourg, Inria*

Yossi Adi
*Meta, FAIR, Hebrew Univ. of Jerusalem*

Romain Serizel
*Univ. de Lorraine, CNRS, Inria, Loria*

***Abstract*—The advancements in audio generative models have opened up new challenges in their responsible disclosure and the detection of their misuse. To address this, watermarking techniques have been recently developed, enabling the detection of content generated by a deployed model. For such techniques to be useful, the watermark must resist typical modifications applied to the model or its outputs. The use case of an open-source model trained on proprietary data is challenging, as post-hoc watermarks can then be trivially removed. In response, we introduce a method that watermarks latent audio generative models by directly watermarking their *training data*. We show the method to be robust against a broad range of audio edits including filtering, compression or even to changing the model's decoder, maintaining high detection rates with very few false positives. Interestingly, we show that even fine-tuning the model on another dataset can only significantly lower the detection rate at the cost of degrading the generation performance near the level of re-training the model without the protected training data.**

***Index Terms*—watermarking, audio, generative models**

## I. Introduction

Sophisticated generative models are impacting various audio modalities: environmental sounds [1], [2], music [3], [4], and speech [5]–[7]. These models produce outputs increasingly indistinguishable from real data [8], [9]. Their rapid proliferation and quality raise concerns about misuse (e.g. creation of deepfakes) and respect for intellectual property. These concerns are heightened when models are open-sourced, since they can be easily accessed and used by anyone, including malicious actors. Consequently, regulators suggest watermarking to label and detect generative model outputs (refer to the EU AI Act, White House executive order, and CAC measures).

Watermarking is a technique that slightly alters the audio after its generation, in a way that is inaudible for humans but identifiable by specific detection algorithms. The state-of-the-art methods are based on deep neural networks [10], [11] that are trained end-to-end to embed and detect watermarks in audio signals, even after audio compression or editing. Such methods are for instance employed to safeguard APIs for public model demonstrations [6], [12]. However, while post-hoc watermarking has proven effective in certain scenarios, it is not as effective for protecting open-sourced models, as malicious users could potentially extract the output before the watermarking stage (for example by commenting out the code responsible for watermark embedding).

In the image domain, some methods [13], [14] fine-tune decoders to output watermarked images directly, to make it compliant with open-sourcing. However, in the audio domain, it is common and cost-effective to train decoders (also called vocoders) that convert latent representations to waveforms [15]–[17]. Watermarking can thus be easily bypassed by using non-watermarked vocoders. Therefore, in this article, we propose to watermark the latent generative model that creates the latent representations.

We focus on MusicGen [4] due to its performance and adoption. It consists of an auto-encoder EnCodec [18] that compresses audio into discrete representations (tokens) and a single-stage transformer (audio Language Model, LM) that predicts the next tokens and decodes them into a music stream. We train watermark generator/detector models to be robust to EnCodec. Intuitively, this makes both the audio and encoded tokens watermarked. We then train the LM on tokens derived from audios that were preemptively marked. The resulting LM produces tokens whose decoded outputs are watermarked, irrespective of the LM conditioning or decoding algorithm. In other terms, as long as the watermarking algorithm withstands the audio tokenization, the watermark transfers from the training data to the generative model outputs.

In short, (1) we introduce a way to watermark audio generative models at the latent representations level, (2) we demonstrate that it makes generations detectable with high confidence while having almost no influence on the model performance, (3) we demonstrate the robustness of the watermark to model-level changes, namely, switching the decoding algorithm and fine-tuning the audio LM.

## II. Related Work

**Audio generation** is a challenging task because audio signals are high-dimensional and have complex temporal dependencies. Early autoregressive deep-learning-based approaches like WaveNet [19] were quickly followed by GAN-based models [15], [16]. Inspired by progress in text generation [20], [21], audio language models have recently emerged as state of the art for most audio generative tasks such as text-to-speech [5], [12], [22], music [3], [4] or sound [1] generation. They make audio modeling more tractable by compressing audio into discrete tokens using models like EnCodec [18], SoundStream [23], or DAC [24]. Additional tokens coming from text, melody, phoneme, speaker embedding, etc. may serve as conditioning to generate audio with user-specific characteristics. Then a transformer-based model [25] generates audio by predicting the next tokens and decoding them.

In parallel to audio LMs, latent diffusion models have also been largely studied in recent works on audio generation. Those models can sample in an non autoregressive way from

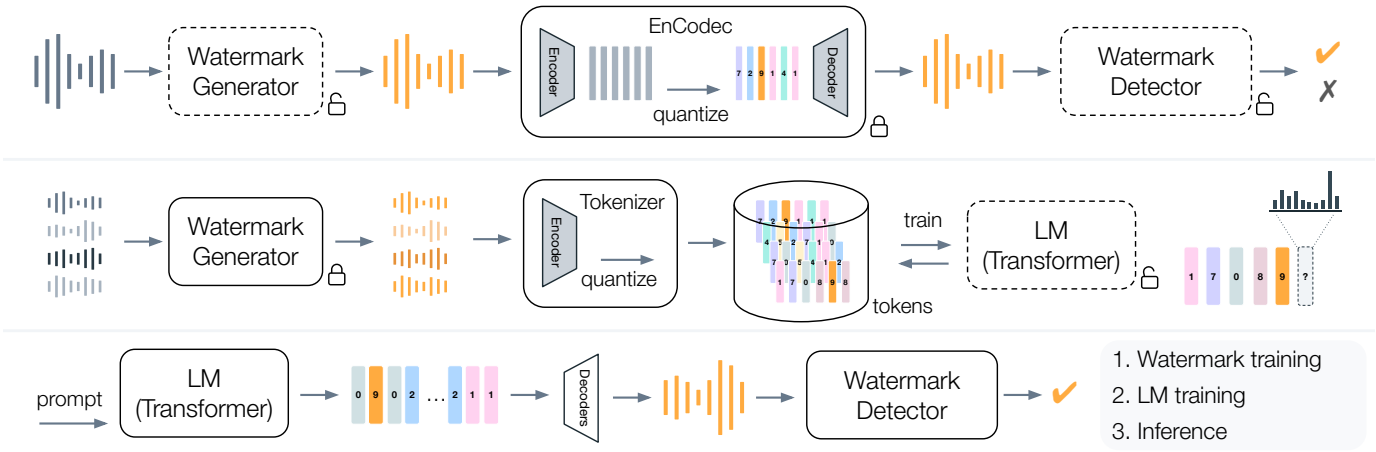


Fig. 1: **Overview of our method.** (1.) We train a watermark generator and detector based on AudioSeal [10], enhancing robustness against EnCodec [18] by processing the watermarked audio through EnCodec before detection. (2.) We watermark the audios from our database and train a MusicGen [4] model for next token prediction on this watermarked data. (3.) During inference, we prompt (using text or audio) the language model and decode audios that are detectable with watermark detector.

the training data distribution and have recently shown great generative habilities on different audio modalities such as speech [6], [26], [27], music [28], [29] or general audio [2].

**Invisible audio watermarking** has evolved from using domain-specific features in the time/frequency domain of audios [30], [31] to deep learning methods that employ encoder/decoder architectures [11], [32]–[34]. Notably, AudioSeal [10] introduces localized audio watermarking with a detector producing time-step-level logits. This method also allows for a watermarking robust to neural compression models which is a necessary element for our work.

**Generative model watermarking** is attracting renewed interest thanks to its potential to improve detection of AI-generated contents. In this context, the aforementioned methods apply watermarking *post-hoc* after audio generation, unlike more recent methods which do so *in-model*. Examples include watermarking: image GANs by training a hyper-network model [35], latent diffusion models by a quick fine-tuning of their decoder [13], and HiFi-GAN decoders that take mel-spectrograms and output waveforms [36]. Unlike the last two approaches, our method operates one step earlier at the latent representation level. It draws inspiration from research demonstrating that watermarks embedded in images or texts may propagate from the training data of generative models to their outputs [37]–[40]. We apply this concept to audio generative models and target audio language models.

## III. Audio Model Watermarking

### A. Problem statement

We consider providers training an audio-generative model on a large proprietary dataset. They aim to release the model publicly but worry about misuse and unauthorized redistribution. To mitigate these concerns, they watermark the model during training to enhance the detection of generated content or unauthorized API usage. We describe this watermarking process below and provide a step by step overview in Figure 1.

### B. Audio watermarking

We first build an audio watermarking model based on AudioSeal [10]. It jointly trains a watermark generator $G$ and a watermark detector $D$. $G$ takes a signal $s \in \mathbb{R}^T$ and generates an additive watermark $\delta_w$, that is made imperceptible through perceptual losses $\mathcal{L}_{\text{percep}}(s, s + \delta_w)$. The watermarked audio $s + \delta_w$ is augmented into $s'$. Augmentations include padding the audio with 0, replacing intervals of watermarked audio with non-watermarked audio from the same batch, or dropping the $\delta_w$. $s'$ is fed to the detector, which is trained to output which part of a waveform is watermarked via a localization loss $\mathcal{L}_{\text{loc}}(D(s'), y')$, where $y' \in \{0, 1\}^T$ indicates the watermarked intervals in $s'$ (1 for watermarked, 0 otherwise).

We make the following changes with regard to AudioSeal's recipe. We remove the message encoder to only focus on watermark detection. Furthermore, it is important to remember that the audio LM will be trained on tokens, not directly on audio. Hence, the LM will not retain watermark information if it is absent at the discrete representation level. Therefore, we train the watermark generation/detection to be very robust to the specific EnCodec compression model later used for audio tokenization (see Sec. III-C). This is done by oversampling this EnCodec augmentation so that 50% of batches go through EnCodec before the detection phase.

### C. Audio language model

We select MusicGen [4] as the audio LM to watermark.

**Watermarking.** The first step is to watermark the audios with the model of Sec. III-B. This is done on the fly at loading time (this takes around 10 ms for a 10-second audio).

**Tokenization.** We use the EnCodec compression model to transform the audio signal into a discrete sequence of tokens suitable for language modeling. It uses residual vector quantization (RVQ) [23] which compresses an audio signal $s \in \mathbb{R}^T$ into $K$ streams of tokens $(u_i^{(j)})_{j\in[1,K]\,;\,i\in[0,T/f_r]}$ ($f_r$ being the frame rate). This model is trained on audio segments sampled at a rate of 32 kHz and $f_r = 50$ Hz, the number of codebooks is $K = 4$ and the codebook size is 2048 ($u_i^{(j)} \in [1, 2048]$). Overall, this results in a bit rate of 2.2 kbit per second.

**Language modeling** aims to build a probabilistic model of sequences of discrete tokens. MusicGen implements a delay pattern [22] that adds a delay $k$ to the $k$-th residual. It allows the model to generate the tokens in a coarse to fine order. This way all the streams can be sampled in parallel while assuring that all the previous residuals are fixed when sampling a given token. Put differently, the transformer is fed with a sequence of embeddings created from $K$ tokens: $s_i = \{u_{i-4}^{(4)}, u_{i-2}^{(3)}, u_{i-1}^{(2)}, u_i^{(1)}\}$. The embedding of $s_i$ is then the sum of the embedding of each of its constitutive tokens, with additional sinusoidal embeddings. As most current language models [9], [41], training is done with next token prediction and the cross-entropy is computed per codebook.

## IV. Experiments

### A. Experimental details

**Watermarking models.** We train on an internal music dataset containing 1.5k songs at 32 kHz sample rate, with 1-second audio excerpts. The model is trained for 400k steps with batch size 64. Hyper-parameters (network architectures, optimizer, etc.) are kept the same as in the original work [10].

**MusicGen models.** We use 20k hours of licensed music to train the models with two different model sizes: small (300M parameters) and medium (1.5B parameters). They are similar in quality and diversity to the ones used by Copet et al. [4]. We use public implementation and default parameters available on the AudioCraft GitHub page. Each model is trained for 200 epochs with a batch size of 128, with the default optimizer. We use 64 GPUs for the medium model and 32 for the small.

**Inference.** For music generation sampling, we use top-k sampling with $k = 250$ tokens and a temperature of 1.0.

### B. Quality of the audio generative model

We first subjectively evaluate how watermarking influences the quality of the generative models.

We adhere to the original paper's protocol [4], using (OVL) to assess sample quality and (REL) to evaluate relevance to text prompts. Models are tested on 15-second generation using 40 text prompts from the test set of MusicCaps [3]. Every sample is rated by 20 listeners that rate them on a scale from 1 to 100. For every study, we report both mean score and CI95. Table I shows that the performance difference between a model trained on watermarked data and one trained on normal data is negligible. This holds true for both sizes, with the rating difference falling within the confidence interval.

TABLE I: **Subjective evaluation.** We compare audio quality and text relevance of the original MusicGen models (ori.) and our models that natively outputs watermarked audios (ours).

| Size | OVL (↑) | REL (↑) |
|---|---|---|
| Ground Truth | $93.08 \pm 0.53$ | $93.01 \pm 0.68$ |
| Small (ori.) | $83.67 \pm 1.85$ | $82.42 \pm 1.37$ |
| Small (ours) | **84.13** $\pm 2.21$ | **82.46** $\pm 1.44$ |
| Medium (ori.) | **85.92** $\pm 1.46$ | **83.71** $\pm 1.79$ |
| Medium (ours) | $84.91 \pm 1.53$ | $82.64 \pm 1.41$ |

TABLE II: **Detection and localization results** on 10k positive/negative samples, for the MusicGen models watermarked post-hoc using AudioSeal (+AS), or in-model with our method (ours). We report the area under the ROC curve (AUC), the accuracy for the best threshold, as well as the intersection over union (IoU) and sample level accuracy (SL-Acc.).

| | Detection | | | Localization | |
|---|---|---|---|---|---|
| Model | AUC | Acc. | TPR / FPR | IoU | SL-Acc. |
| Small + AS | 1.0 | 1.0 | 1.0 / 0.0 | 1.0 | 1.0 |
| Medium + AS | 1.0 | 1.0 | 1.0 / 0.0 | 1.0 | 1.0 |
| Small (ours) | 0.999 | 0.993 | 0.986 / $2.10^{-4}$ | 0.81 | 0.91 |
| Medium (ours) | 0.999 | 0.994 | 0.988 / $3.10^{-4}$ | 0.91 | 0.96 |

### C. Detection and localization results

**Detection.** To evaluate the detection performance, we generate 10k positive 15-second samples with the watermarked model and use 10k negative samples from our test set that we compress using the codec model. The watermark detector gives a score per time-step of the audio, which we average to get a score for the whole audio. Audio is flagged as watermarked if this score is higher than a threshold $\tau$. We report in Tab. II the area under the ROC curve, as well as the accuracy for the best $\tau$ (and true positive and false positive rates at this $\tau$, TPR, and FPR). The generated output is indeed watermarked as indicated by the detection metrics: the AUC is close to 1 and TPR is higher than 0.95 at FPR around $10^{-4}$.

**Localization.** We then evaluate if the detector still has the property to locally detect watermarked segments. To do so, we generate 15-second samples and replace parts with other non-watermarked audio from our test set. The proportion of signal that is watermarked is 50% on average. We then measure the precision of the detection using the detection accuracy at the sample level together with the intersection over union (IoU) metric. For localization, we use a fixed detection threshold set at 0.5. As shown in Tab. II, results are on-par (although a bit lower) to AudioSeal, showing that the detector keeps a good-enough performance on the localization of generated outputs.

**Robustness.** We evaluate the robustness to different audio edits, and compare the performance of our in-model watermarking and of post-hoc watermarking that directly applies the watermark to generated outputs with AudioSeal. The evaluation is made on 10k samples.

TABLE III: **Robustness** to audio edits for post-hoc watermarking with AudioSeal or using our in-model method. The results are computed on audios generated with the Small model.

| | Post-hoc | | In-model | |
|---|---|---|---|---|
| Edit | AUC | Acc TPR / FPR | AUC | Acc TPR / FPR |
| White Noise | 0.995 | 0.99 0.99/0.01 | 0.941 | 0.93 0.91/0.04 |
| Lowpass | 0.99 | 0.99 0.99/0.01 | 0.942 | 0.94 0.91/0.03 |
| Highpass | 0.931 | 0.98 0.98/0.2 | 0.941 | 0.93 0.91/0.04 |
| Resample | 0.940 | 0.94 0.91/0.03 | 0.999 | 0.99 0.99/0.01 |
| AAC | 0.999 | 0.98 0.97/0.01 | 0.88 | 0.81 0.81/0.04 |
| Pink Noise | 0.996 | 0.97 0.98/0.03 | 0.956 | 0.92 0.91/0.06 |
| Echo | 0.950 | 0.98 0.99/0.02 | 0.906 | 0.92 0.89/0.04 |

TABLE IV: Detection results with other decoding algorithm.

| Decoder | AUC | Acc. | TPR / FPR |
|---|---|---|---|
| HiFi-GAN | 0.999 | 0.990 | 0.980 / $4.10^{-4}$ |
| Multi-band diffusion | 0.991 | 0.954 | 0.951 / 0.043 |
| Default | 0.999 | 0.993 | 0.986 / 0.000 |

Table III shows that while in-model watermarking keeps a decent robustness to common audio edits there is a slight performance decrease compared to the original watermarking model. Therefore, when post-hoc watermarking is feasible, it might be preferable to in-model watermarking. The latter is better suited for scenarios where post-hoc watermarking is not possible, such as when open-sourcing a model.

## V. Attacks on the Model's Watermark

We now focus on model-level attacks, *i.e.*, modifications of the model attempting to make its outputs undetectable.

### A. Switching decoder

Previous works alter the latent decoder to embed the watermark [13], [36]. However, audio vocoders are relatively easy to train and interchange [15], [16], [42]. They do not necessitate extensive data or computational power compared to those required for training an audio LM. Therefore, replacing the decoder to use the watermark-free generative model is rather straightforward. In contrast, our work embeds the watermark at the latent stage for robustness against decoder changes.

We now evaluate how the change of the decoder influences detection performance. In previous experiments, the default decoder was the codec model from MusicGen. We replace it with Multi-Band Diffusion [17] which uses diffusion to map discrete EnCodec tokens into the waveform domain, and a discrete version HiFi-GAN [16], which we trained on tokens-waveform pairs. We use the same experimental setup as in Sec. IV-C, but with different algorithms to decode the tokens. Table IV shows that changing the decoder has very little impact on the detection metrics. Notably, using the diffusion-based decoder reduces the AUC only by around $0.01$.

### B. Model fine-tuning

One potential attack could be to remove the watermark through "model purification", which involves fine-tuning the audio LM on a non-watermarked dataset. To test this we fine-tune with different learning rates the small version of the model using a different internal music dataset $D_{FT}$ of similar size without watermarks. For each setting, the model is trained for 20 epochs (10% of the total pre-training steps). We then generate 10k samples with the purified models and obtain scores through the watermark detector.

TABLE V: Performances of the model after fine-tuning for 20 epochs with different learning rates on a non watermarked dataset. No FT is the model before fine-tuning, scratch is a model trained from scratch on the non watermaked dataset.

| LR | Best Acc. | TPR @ FPR=$10^{-3}$ | FAD (↓) |
|---|---|---|---|
| No FT | 0.993 | 0.983 | 2.067 |
| $2 \times 10^{-6}$ | 0.964 | 0.765 | 2.908 |
| $5 \times 10^{-6}$ | 0.928 | 0.456 | 3.149 |
| $2 \times 10^{-5}$ | 0.833 | 0.079 | 3.344 |
| $5 \times 10^{-5}$ | 0.721 | 0.019 | 3.740 |
| $1 \times 10^{-4}$ | 0.721 | 0.018 | 3.643 |
| Scratch | N/A | N/A | 3.839 |

For each experiment, we report the accuracy obtained for the best threshold on the detection score as well as the TPR when the threshold is chosen such that the FPR is at $10^{-3}$. We also report the Fréchet Audio Distance (FAD) [43] that evaluates the quality of the generative model. We include as a reference the performance of the model before fine-tuning and of a model trained from scratch on $D_{FT}$.

Table V suggests that a higher learning rate during fine-tuning makes watermarks more difficult to detect, but it also causes the distribution of generated data to deviate further from the protected model's dataset; at larger learning rates, the distribution has a similar FAD to a model trained from scratch on different data. In other words, since the FAD is almost the same as a model trained from scratch, it may not be worthwhile to start from the watermarked model.

## VI. Conclusion & Discussion

This work introduced a straightforward yet effective approach to watermarking audio language models. This is done via watermarking their training data in a way that is robust to the compression algorithm used to create tokens. It does not require modifications to the model architecture or the training process. Our method is the first to watermark at the latent level and is robust to changes in the decoding process. The main drawback of the current approach is that it requires training the model from scratch, which may be difficult for versioning large models or for adapting already-trained models. While there is a slight decrease in robustness to audio edits compared to post-hoc watermarking, this method allows for keeping the watermark in situations for which post-hoc watermarking is not suitable (open-sourcing...). In conclusion, our method can help trace content origin for open sourced models. It is not a standalone solution and should be complemented with measures like policies, education or monitoring.

## REFERENCES


[1] Felix Kreuk, Gabriel Synnaeve, Adam Polyak, Uriel Singer, Alexandre Défossez, Jade Copet, Devi Parikh, Yaniv Taigman, and Yossi Adi, "Audiogen: Textually guided audio generation," *arXiv preprint arXiv:2209.15352*, 2022.

[2] Haohe Liu, Zehua Chen, Yi Yuan, Xinhao Mei, Xubo Liu, Danilo Mandic, Wenwu Wang, and Mark D Plumbley, "AudioLDM: Text-to-audio generation with latent diffusion models," *Proceedings of the International Conference on Machine Learning*, 2023.

[3] Andrea Agostinelli, Timo I. Denk, Zalán Borsos, Jesse Engel, Mauro Verzetti, Antoine Caillon, Qingqing Huang, Aren Jansen, Adam Roberts, Marco Tagliasacchi, Matt Sharifi, Neil Zeghidour, and Christian Frank, "Musiclm: Generating music from text," 2023.

[4] Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez, "Simple and controllable music generation," *NeurIPS*, vol. 36, 2024.

[5] Chengyi Wang, Sanyuan Chen, Yu Wu, Ziqiang Zhang, Long Zhou, Shujie Liu, Zhuo Chen, Yanqing Liu, Huaming Wang, Jinyu Li, Lei He, Sheng Zhao, and Furu Wei, "Neural codec language models are zero-shot text to speech synthesizers," 2023.

[6] Apoorv Vyas, Bowen Shi, Matthew Le, Andros Tjandra, Yi-Chiao Wu, Baishan Guo, Jiemin Zhang, Xinyue Zhang, Robert Adkins, William Ngan, Jeff Wang, Ivan Cruz, Bapi Akula, Akinniyi Akinyemi, Brian Ellis, Rashel Moritz, Yael Yungster, Alice Rakotoarison, Liang Tan, Chris Summers, Carleigh Wood, Joshua Lane, Mary Williamson, and Wei-Ning Hsu, "Audiobox: Unified audio generation with natural language prompts," 2023.

[7] Zalán Borsos, Raphaël Marinier, Damien Vincent, Eugene Kharitonov, Olivier Pietquin, Matt Sharifi, Dominik Roblek, Olivier Teboul, David Grangier, Marco Tagliasacchi, and Neil Zeghidour, "Audiolm: a language modeling approach to audio generation," 2023.

[8] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer, "High-resolution image synthesis with latent diffusion models," 2021.

[9] OpenAI, "Gpt-4 technical report," 2023.

[10] Robin San Roman, Pierre Fernandez, Alexandre Défossez, Teddy Furon, Tuan Tran, and Hady Elsahar, "Proactive detection of voice cloning with localized watermarking," 2024.

[11] Guangyu Chen, Yu Wu, Shujie Liu, Tao Liu, Xiaoyong Du, and Furu Wei, "Wavmark: Watermarking for audio generation," 2024.

[12] Loïc Barrault, Yu-An Chung, Mariano Coria Meglioli, David Dale, Ning Dong, Mark Duppenthaler, Paul-Ambroise Duquenne, Brian Ellis, Hady Elsahar, Justin Haaheim, et al., "Seamless: Multilingual expressive and streaming speech translation," *arXiv preprint arXiv:2312.05187*, 2023.

[13] Pierre Fernandez, Guillaume Couairon, Hervé Jégou, Matthijs Douze, and Teddy Furon, "The stable signature: Rooting watermarks in latent diffusion models," 2023.

[14] Changhoon Kim, Kyle Min, Maitreya Patel, Sheng Cheng, and Yezhou Yang, "Wouaf: Weight modulation for user attribution and fingerprinting in text-to-image diffusion models," *arXiv preprint arXiv:2306.04744*, 2023.

[15] Kundan Kumar, Rithesh Kumar, Thibault de Boissiere, Lucas Gestin, Wei Zhen Teoh, Jose Sotelo, Alexandre de Brebisson, Yoshua Bengio, and Aaron Courville, "Melgan: Generative adversarial networks for conditional waveform synthesis," 2019.

[16] Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae, "Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis," 2020.

[17] Robin San Roman, Yossi Adi, Antoine Deleforge, Romain Serizel, Gabriel Synnaeve, and Alexandre Défossez, "From discrete tokens to high-fidelity audio using multi-band diffusion," *NeurIPS*, vol. 36, 2024.

[18] Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi, "High fidelity neural audio compression," *arXiv preprint arXiv:2210.13438*, 2022.

[19] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[20] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al., "Improving language understanding by generative pre-training," 2018.

[21] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al., "Language models are few-shot learners," *NeurIPS*, vol. 33, pp. 1877–1901, 2020.

[22] Eugene Kharitonov, Ann Lee, Adam Polyak, Yossi Adi, Jade Copet, Kushal Lakhotia, Tu-Anh Nguyen, Morgane Rivière, Abdelrahman Mohamed, Emmanuel Dupoux, et al., "Text-free prosody-aware generative spoken language modeling," *arXiv preprint arXiv:2109.03264*, 2021.

[23] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi, "Soundstream: An end-to-end neural audio codec," 2021.

[24] Rithesh Kumar, Prem Seetharaman, Alejandro Luebs, Ishaan Kumar, and Kundan Kumar, "High-fidelity audio compression with improved rvqgan," 2023.

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin, "Attention is all you need," *NeurIPS*, vol. 30, 2017.

[26] Matthew Le, Apoorv Vyas, Bowen Shi, Brian Karrer, Leda Sari, Rashel Moritz, Mary Williamson, Vimal Manohar, Yossi Adi, Jay Mahadeokar, and Wei-Ning Hsu, "Voicebox: Text-guided multilingual universal speech generation at scale," 2023.

[27] Kai Shen, Zeqian Ju, Xu Tan, Yanqing Liu, Yichong Leng, Lei He, Tao Qin, Sheng Zhao, and Jiang Bian, "Naturalspeech 2: Latent diffusion models are natural and zero-shot speech and singing synthesizers," 2023.

[28] Or Tal, Alon Ziv, Itai Gat, Felix Kreuk, and Yossi Adi, "Joint audio and symbolic conditioning for temporally controlled text-to-music generation," 2024.

[29] Zach Evans, CJ Carr, Josiah Taylor, Scott H. Hawley, and Jordi Pons, "Fast timing-conditioned latent audio diffusion," 2024.

[30] Wen-Nung Lie and Li-Chun Chang, "Robust and high-quality time-domain audio watermarking based on low-frequency amplitude modification," *IEEE transactions on multimedia*, vol. 8, no. 1, pp. 46–59, 2006.

[31] Nima Khademi Kalantari, Mohammad Ali Akhaee, Seyed Mohammad Ahadi, and Hamidreza Amindavar, "Robust multiplicative patchwork method for audio watermarking," *IEEE Transactions on Audio, speech, and language processing*, vol. 17, no. 6, pp. 1133–1141, 2009.

[32] Chang Liu, Jie Zhang, Han Fang, Zehua Ma, Weiming Zhang, and Nenghai Yu, "Dear: A deep-learning-based audio re-recording resilient watermarking," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023, vol. 37, pp. 13201–13209.

[33] Kosta Pavlović, Slavko Kovačević, Igor Djurović, and Adam Wojciechowski, "Robust speech watermarking by a jointly trained embedder and detector using a dnn," *Digital Signal Processing*, vol. 122, pp. 103381, 2022.

[34] Patrick O'Reilly, Zeyu Jin, Jiaqi Su, and Bryan Pardo, "Maskmark: Robust neuralwatermarking for real and synthetic speech," in *ICASSP*. IEEE, 2024, pp. 4650–4654.

[35] Ning Yu, Vladislav Skripniuk, Dingfan Chen, Larry S Davis, and Mario Fritz, "Responsible disclosure of generative models using scalable fingerprinting," in *International Conference on Learning Representations*, 2021.

[36] Lauri Juvela and Xin Wang, "Collaborative watermarking for adversarial speech synthesis," *arXiv preprint arXiv:2309.15224*, 2023.

[37] Ning Yu, Vladislav Skripniuk, Sahar Abdelnabi, and Mario Fritz, "Artificial fingerprinting for generative models: Rooting deepfake attribution in training data," in *Proceedings of the IEEE/CVF International conference on computer vision*, 2021, pp. 14448–14457.

[38] Yunqing Zhao, Tianyu Pang, Chao Du, Xiao Yang, Ngai-Man Cheung, and Min Lin, "A recipe for watermarking diffusion models," *arXiv preprint arXiv:2303.10137*, 2023.

[39] Chenchen Gu, Xiang Lisa Li, Percy Liang, and Tatsunori Hashimoto, "On the learnability of watermarks for language models," *arXiv preprint arXiv:2312.04469*, 2023.

[40] Tom Sander, Pierre Fernandez, Alain Durmus, Matthijs Douze, and Teddy Furon, "Watermarking makes language models radioactive," *arXiv preprint arXiv:2402.14904*, 2024.

[41] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, Thomas Scialom, et al., "Llama 2: Open foundation and fine-tuned chat models," 2023.

[42] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro, "Diffwave: A versatile diffusion model for audio synthesis," 2021.

[43] Kevin Kilgour, Mauricio Zuluaga, Dominik Roblek, and Matthew Sharifi, "Fr\'echet audio distance: A metric for evaluating music enhancement algorithms," *arXiv preprint arXiv:1812.08466*, 2018.