HarmonyCloak

Gaslighting Generative Audio Models

Silas Wang ; NYU GenAudio

Agenda

1. Issue(s) at Large
   a.) Automation
   b.) Privacy
   c.) The Arts
   d.) Threat Model
2. Preliminaries
   a.) Psychoacoustics
   b.) AI Overview
   c.) Adversarial Machine Learning
3. HarmonyCloak
   a.) Design
   b.) Harmony Cloak & MIDI/Audio
   c.) Optimization
   d.) Evaluation

Slides, Notes, References:

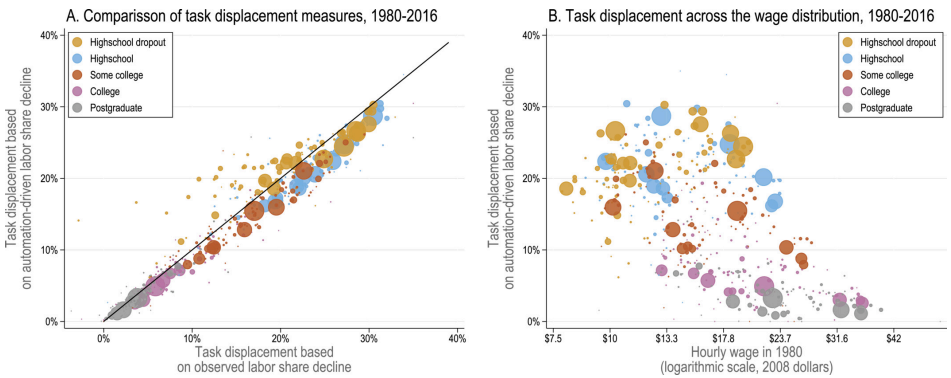
tinyurl.com/genaudio-harmonycloak

Issue(s) at Large

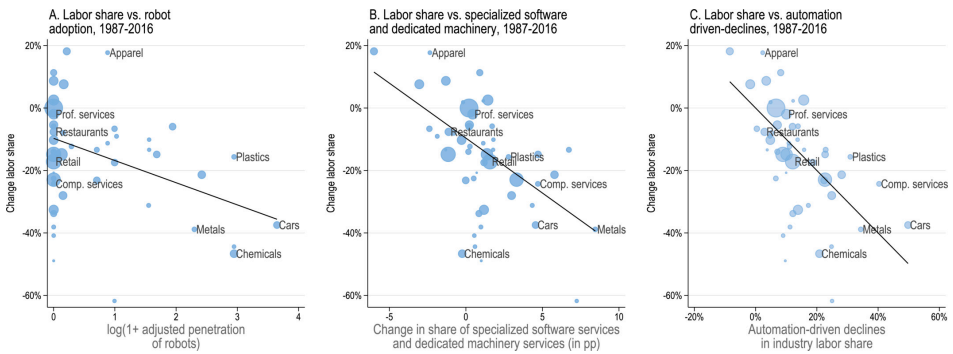
Issue(s) at Large

technology is advancing fast enough to outpace our measures for the displaced

Automation



Task Displacement by Education



Labor share & Automation

Self Driving Trucks

Millionaires' Row, Cleveland Ohio

Privacy

Government investment(s) in Surveillance

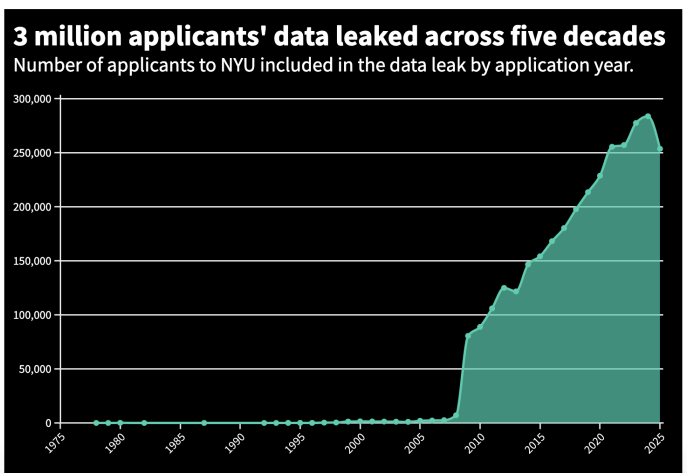
AI-driven Surveillance False Positives

NYU hit with 10 class action lawsuits following data breach

The plaintiffs are accusing the university of failing to protect personal information after a hacker leaked over 3 million applicants' data on NYU's homepage.

Dharma Niles and Krish Dev

April 1, 2025

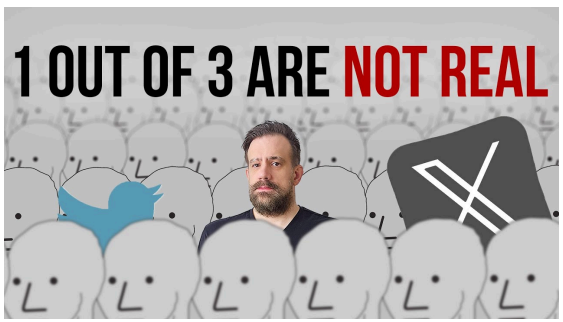


Poorly managing sensitive data

… and Misinformation

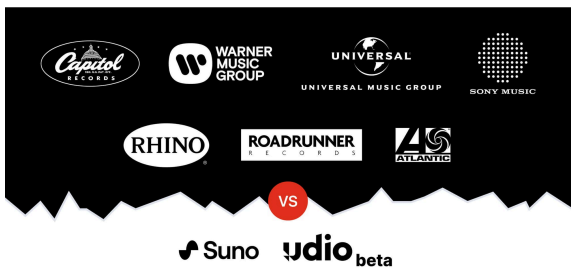
AI Voice Deepfake Scams

This Deepfake Scandal

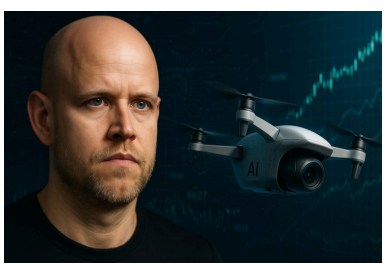


Dead Internet Theory

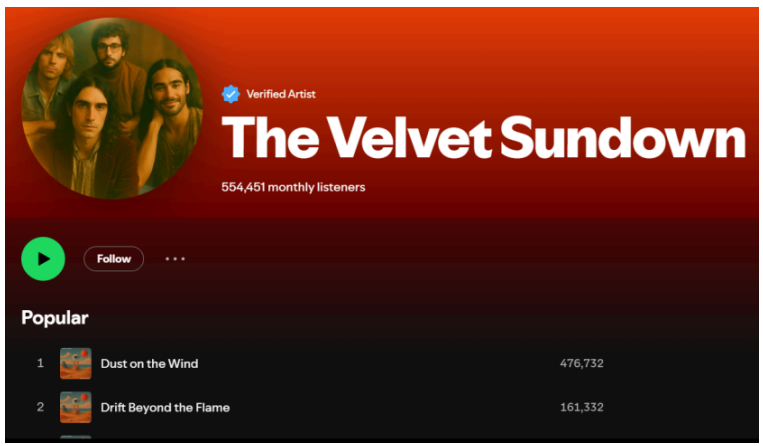
The Arts



Everybody's Favorite Music Lawsuit



Spotify Drone Investments



Everybody's Favorite Human Band

Threat Model

[1]in a black box setting, this is not the case. In a white box setting, this is.

Threat Model

Attacker's Capability:

- Scrape music from the internet to train Generative AI models
- Copyright infringement, musicians do not get paid

[1]in a black box setting, this is not the case. In a white box setting, this is.

Threat Model

Attacker's Capability:

- Scrape music from the internet to train Generative AI models
- Copyright infringement, musicians do not get paid

Defender's Objective:

- Share music that Generative AI can't scrape
- Must be high-fidelity

[1] in a black box setting, this is not the case. In a white box setting, this is.

Threat Model

Attacker's Capability:

- Scrape music from the internet to train Generative AI models
- Copyright infringement, musicians do not get paid

Defender's Objective:

- Share music that Generative AI can't scrape
- Must be high-fidelity

Defender's Capabilities:

- Computer
- Can write and share songs
- Knowledge of the generative model[1]

[1] in a black box setting, this is not the case. In a white box setting, this is.

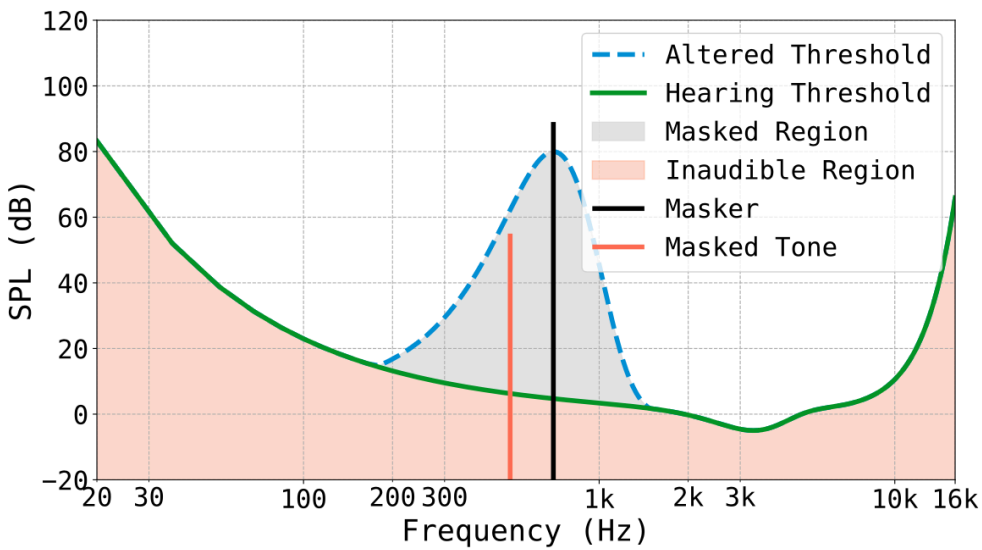
Preliminaries

Psychoacoustics

Psychoacoustics

Human hearing has a lower limit; a softest sound we can possibly hear. This **absolute hearing threshold** can be described mathematically for any frequency f:

$$T_H(f) = 3.64 \cdot \left(\frac{f}{1000}\right)^{-0.8} - 6.5 \cdot e^{-0.6\left(\frac{f}{1000}-3.3\right)^2}$$



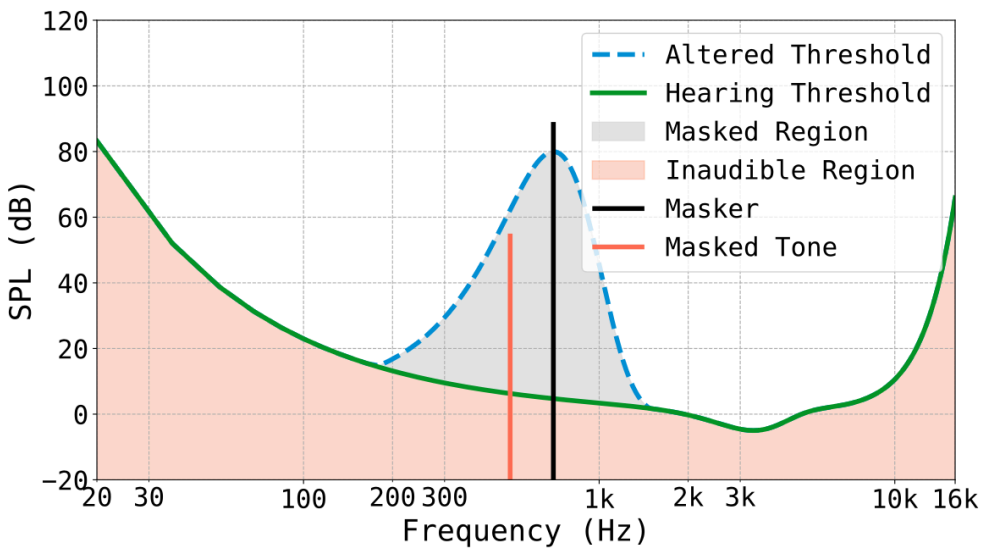
Psychoacoustics

Human hearing has a lower limit; a softest sound we can possibly hear. This **absolute hearing threshold** can be described mathematically for any frequency f:

$$T_H(f) = 3.64 \cdot \left(\frac{f}{1000}\right)^{-0.8} - 6.5 \cdot e^{-0.6\left(\frac{f}{1000}-3.3\right)^2}$$

Additionally, given two sounds A and B with similar frequency ($f_A \approx f_B$), we say that A **masks** B if we can hear A but not B.

- **Temporal Masking** can also happen if A and B are not played at the same time.



Generative AI Overview

We have devised multiple different ways to create **Generative AI**; Artificial Intelligence that can tailor output responses based on user input.

Generative AI Overview

We have devised multiple different ways to create **Generative AI**; Artificial Intelligence that can tailor output responses based on user input.

- GAN (*University of Montreal, 2014*): Two models work in parallel towards a unified goal.
  1. **Generator:** Trained to generate a target output
  2. **Discriminator:** Trained to determine if (1)'s target output is good or bad

Generative AI Overview

We have devised multiple different ways to create **Generative AI**; Artificial Intelligence that can tailor output responses based on user input.

- GAN (*University of Montreal, 2014*): Two models work in parallel towards a unified goal.
  1. **Generator:** Trained to generate a target output
  2. **Discriminator:** Trained to determine if (1)'s target output is good or bad
- Diffusion Models (*Stanford & UC Berkley, 2015*): Increasingly add or subtract noise given a prompt to achieve a desired output.

Generative AI Overview

We have devised multiple different ways to create **Generative AI**; Artificial Intelligence that can tailor output responses based on user input.

- GAN (*University of Montreal, 2014*): Two models work in parallel towards a unified goal.
  1. **Generator:** Trained to generate a target output
  2. **Discriminator:** Trained to determine if (1)'s target output is good or bad
- Diffusion Models (*Stanford & UC Berkley, 2015*): Increasingly add or subtract noise given a prompt to achieve a desired output.
- Transformers (*Google, 2017*): Prominent in NLP, each part of the input is given a fixed weight with respect to its relevance to the other parts of the input, which the model can then use to predict / generate a response. The T in ChatGPT stands for Transformer.

Generative AI Overview

We have devised multiple different ways to create **Generative AI**; Artificial Intelligence that can tailor output responses based on user input.

- GAN (*University of Montreal, 2014*): Two models work in parallel towards a unified goal.
  1. **Generator:** Trained to generate a target output
  2. **Discriminator:** Trained to determine if (1)'s target output is good or bad
- Diffusion Models (*Stanford & UC Berkley, 2015*): Increasingly add or subtract noise given a prompt to achieve a desired output.
- Transformers (*Google, 2017*): Prominent in NLP, each part of the input is given a fixed weight with respect to its relevance to the other parts of the input, which the model can then use to predict / generate a response. The T in ChatGPT stands for Transformer.

In order to protect our audio, we must gaslight the clanker's learning process (described above).

Generative Audio

Some prominent Generative AI Architectures with respect to audio use **transformers** or **GANs**:

Generative Audio

Some prominent Generative AI Architectures with respect to audio use **transformers** or **GANs**:

- MIDINet & MuseGAN: Convolutional Neural Network GAN, trained on tons of “piano rolls”; outputs MIDI

Generative Audio

Some prominent Generative AI Architectures with respect to audio use **transformers** or **GANs**:

- MIDINet & MuseGAN: Convolutional Neural Network GAN, trained on tons of “piano rolls”; outputs MIDI
- MusicBERT: Extracts characteristics using a Transformer-based model.

Generative Audio

Some prominent Generative AI Architectures with respect to audio use **transformers** or **GANs**:

- MIDINet & MuseGAN: Convolutional Neural Network GAN, trained on tons of “piano rolls”; outputs MIDI
- MusicBERT: Extracts characteristics using a Transformer-based model.
- MusicLM: Transformer-based model that generates audio from text (Consistent, Proprietary, Google)

Generative Audio

Some prominent Generative AI Architectures with respect to audio use **transformers** or **GANs**:

- MIDINet & MuseGAN: Convolutional Neural Network GAN, trained on tons of “piano rolls”; outputs MIDI
- MusicBERT: Extracts characteristics using a Transformer-based model.
- MusicLM: Transformer-based model that generates audio from text (Consistent, Proprietary, Google)
- MusicGen: Transformer-based model that generates audio from text (Flexible, Open Source, Meta)

Adversarial Attacks & Machine Learning

To gaslight a generative model, we must make it think that something else is right.

Adversarial Attacks & Machine Learning

To gaslight a generative model, we must make it think that something else is right.

We can achieve this using some kind of *error-maximizing noise*

Adversarial Attacks & Machine Learning

To gaslight a generative model, we must make it think that something else is right.

We can achieve this using some kind of *error-maximizing noise*

- Throw off the model's predictions $\Longrightarrow$ gaslighting

Adversarial Attacks & Machine Learning

To gaslight a generative model, we must make it think that something else is right.

We can achieve this using some kind of *error-maximizing noise*

- Throw off the model's predictions $\Longrightarrow$ gaslighting
- Examples include Adversarial Attacks and Data Poisoning

Adversarial Attacks & Machine Learning

To gaslight a generative model, we must make it think that something else is right.

We can achieve this using some kind of *error-maximizing noise*

- Throw off the model's predictions $\Longrightarrow$ gaslighting
- Examples include Adversarial Attacks and Data Poisoning
- Useful for testing Robustness of a ML Model pause

Adversarial Attacks & Machine Learning

To gaslight a generative model, we must make it think that something else is right.

We can achieve this using some kind of *error-maximizing noise*

- Throw off the model's predictions $\Longrightarrow$ gaslighting
- Examples include Adversarial Attacks and Data Poisoning
- Useful for testing Robustness of a ML Model pause

However, just because a model's output is horribly inaccurate does not mean that it cannot continue to learn off of the given data.

Adversarial Attacks & Machine Learning

To gaslight a generative model, we must make it think that something else is right.

We can achieve this using some kind of *error-maximizing noise*

- Throw off the model's predictions $\Longrightarrow$ gaslighting
- Examples include Adversarial Attacks and Data Poisoning
- Useful for testing Robustness of a ML Model pause

However, just because a model's output is horribly inaccurate does not mean that it cannot continue to learn off of the given data.

Unlearnable Example: utilizes *error-minimizing noise*

- Developed for HarmonyCloak
- Trick the model into thinking "there is nothing left to learn"

Adversarial Attacks & Machine Learning

To gaslight a generative model, we must make it think that something else is right.

We can achieve this using some kind of *error-maximizing noise*

- Throw off the model's predictions $\Longrightarrow$ gaslighting
- Examples include Adversarial Attacks and Data Poisoning
- Useful for testing Robustness of a ML Model pause

However, just because a model's output is horribly inaccurate does not mean that it cannot continue to learn off of the given data.

Unlearnable Example: utilizes *error-minimizing noise*

- Developed for HarmonyCloak
- Trick the model into thinking "there is nothing left to learn" $\Longrightarrow$ $\boxed{\text{gaslighting \& data protection}}$

HarmonyCloak

Design

HarmonyCloak must produce noise that does not degrade audio fidelity and effectively gaslights Generative Audio models.

Design

HarmonyCloak must produce noise that does not degrade audio fidelity and effectively gaslights Generative Audio models. So we must fulfill these four requirements for HarmonyCloak to properly generate “unlearnable examples” (UEs):

Design

HarmonyCloak must produce noise that does not degrade audio fidelity and effectively gaslights Generative Audio models. So we must fulfill these four requirements for HarmonyCloak to properly generate “unlearnable examples” (UEs):

1. The noise must be masked by the song

Design

HarmonyCloak must produce noise that does not degrade audio fidelity and effectively gaslights Generative Audio models. So we must fulfill these four requirements for HarmonyCloak to properly generate “unlearnable examples” (UEs):

1. The noise must be masked by the song
2. The noise must adapt *dynamically* to the song’s structure

Design

HarmonyCloak must produce noise that does not degrade audio fidelity and effectively gaslights Generative Audio models. So we must fulfill these four requirements for HarmonyCloak to properly generate “unlearnable examples” (UEs):

1. The noise must be masked by the song
2. The noise must adapt *dynamically* to the song’s structure
3. The noise must adapt *timbrically* to the song’s instruments

Design

HarmonyCloak must produce noise that does not degrade audio fidelity and effectively gaslights Generative Audio models. So we must fulfill these four requirements for HarmonyCloak to properly generate “unlearnable examples” (UEs):

1. The noise must be masked by the song
2. The noise must adapt *dynamically* to the song’s structure
3. The noise must adapt *timbrically* to the song’s instruments
4. The noise must not degrade under compression algorithms

Creating Unlearnable Examples

Two possible scenarios: the white box and black box.

Creating Unlearnable Examples

Two possible scenarios: the white box and black box.

In a **white box** setting (we know what model they're using):

1. Split the song into small chunks (windowing)

Creating Unlearnable Examples

Two possible scenarios: the white box and black box.

In a **white box** setting (we know what model they're using):

1. Split the song into small chunks (windowing)
2. Generate the noise for each small chunk
3. Enjoy a cup of hot chocolate and complete your ACM Final Project

Creating Unlearnable Examples

Two possible scenarios: the white box and black box.

In a **white box** setting (we know what model they're using):

1. Split the song into small chunks (windowing)
2. Generate the noise for each small chunk
3. Enjoy a cup of hot chocolate and complete your ACM Final Project

In a **black box** setting (we don't know what model they're using):

1. Split the song into small chunks (windowing)

Creating Unlearnable Examples

Two possible scenarios: the white box and black box.

In a **white box** setting (we know what model they're using):

1. Split the song into small chunks (windowing)
2. Generate the noise for each small chunk
3. Enjoy a cup of hot chocolate and complete your ACM Final Project

In a **black box** setting (we don't know what model they're using):

1. Split the song into small chunks (windowing)
2. Figure out which model is used via Meta-Learning
   - Generate noise and run tests for multiple models
   - fine-tune each noise injection technique, pick the most optimal one (Meta-testing)

Creating Unlearnable Examples

Two possible scenarios: the white box and black box.

In a **white box** setting (we know what model they're using):

1. Split the song into small chunks (windowing)
2. Generate the noise for each small chunk
3. Enjoy a cup of hot chocolate and complete your ACM Final Project

In a **black box** setting (we don't know what model they're using):

1. Split the song into small chunks (windowing)
2. Figure out which model is used via Meta-Learning
   - Generate noise and run tests for multiple models
   - fine-tune each noise injection technique, pick the most optimal one (Meta-testing)
3. Enjoy a cup of hot chocolate and read your DST Chapters

HarmonyCloak with MIDI

Given MIDI as input:

HarmonyCloak with MIDI

Given MIDI as input:

1. Split the MIDI file into small chunks / windows

HarmonyCloak with MIDI

Given MIDI as input:

1. Split the MIDI file into small chunks / windows
2. *determine what the spectrogram of the MIDI instrument looks like*

HarmonyCloak with MIDI

Given MIDI as input:

1. Split the MIDI file into small chunks / windows
2. *determine what the spectrogram of the MIDI instrument looks like*
3. Generate noise for each chunks / windows for each local amplitude peak.

HarmonyCloak with MIDI

Given MIDI as input:

1. Split the MIDI file into small chunks / windows
2. *determine what the spectrogram of the MIDI instrument looks like*
3. Generate noise for each chunks / windows for each local amplitude peak.
   - No louder than the masking noise, no softer than the absolute hearing threshold.

HarmonyCloak with MIDI

Given MIDI as input:

1. Split the MIDI file into small chunks / windows
2. *determine what the spectrogram of the MIDI instrument looks like*
3. Generate noise for each chunks / windows for each local amplitude peak.
   - No louder than the masking noise, no softer than the absolute hearing threshold.
4. Enjoy a cup of hot chocolate and get a good night's rest.

HarmonyCloak with Audio

HarmonyCloak with Audio

Given Audio as Input:

1. Split the audio file into small windows

HarmonyCloak with Audio

Given Audio as Input:

1. Split the audio file into small windows
2. *Conduct a STFT to obtain dominant frequencies in dB SPL*

HarmonyCloak with Audio

Given Audio as Input:

1. Split the audio file into small windows
2. *Conduct a STFT to obtain dominant frequencies in dB SPL*
3. *After finding the relative dB SPL*, introduce the noise component

HarmonyCloak with Audio

Given Audio as Input:

1. Split the audio file into small windows
2. *Conduct a STFT to obtain dominant frequencies in dB SPL*
3. *After finding the relative dB SPL*, introduce the noise component
   - No louder than the masking noise, no softer than the absolute hearing threshold.

HarmonyCloak with Audio

Given Audio as Input:

1. Split the audio file into small windows
2. *Conduct a STFT to obtain dominant frequencies in dB SPL*
3. *After finding the relative dB SPL*, introduce the noise component
   - No louder than the masking noise, no softer than the absolute hearing threshold.
4. *FFT your results back into the Time Domain*

HarmonyCloak with Audio

Given Audio as Input:

1. Split the audio file into small windows
2. *Conduct a STFT to obtain dominant frequencies in dB SPL*
3. *After finding the relative dB SPL*, introduce the noise component
   - No louder than the masking noise, no softer than the absolute hearing threshold.
4. *FFT your results back into the Time Domain*
5. Enjoy a cup of hot chocolate and work on your GenAudio Projects

Loss Function and Optimizations

When creating Generative Models, we want to *optimize* our results such that the output of said is close enough to the target output that we envision it to achieve.

Loss Function and Optimizations

When creating Generative Models, we want to *optimize* our results such that the output of said is close enough to the target output that we envision it to achieve. We exploit this to do our gaslighting:

Loss Function and Optimizations

When creating Generative Models, we want to *optimize* our results such that the output of said is close enough to the target output that we envision it to achieve. We exploit this to do our gaslighting:

$$\min_{\theta} \mathbb{E}\Big[\min_{\delta} \mathcal{L}_{gen}(f(x+\delta))\Big] + \alpha \sum_{m=1}^{M} w^m \parallel \delta^m \parallel_2 \text{ (generate the noise)}$$

such that

$$\mathcal{H}(T_H) \leq \delta^m \leq x^m \forall m \in \{1, 2, ...M\}$$

where

$$\mathcal{H}\Big(T_{H(v)}\Big) = 127 \cdot 10^{\frac{1}{4}\log_{10}\left(T_{H(v)}\right) - L_U + 94} \text{ (constrain the magnitude, given note velocity fo for audior MIDI)}$$

$$\mathcal{H}\Big(T_{H(v_t^m)}\Big) \leq \delta_t^m \leq \mathcal{M}(x_t^m), \forall t, m \text{ (windowize it with a sigmoid function for audio)}$$

Loss Function and Optimizations

When creating Generative Models, we want to *optimize* our results such that the output of said is close enough to the target output that we envision it to achieve. We exploit this to do our gaslighting:

$$\min_{\theta} \mathbb{E}\Big[\min_{\delta} \mathcal{L}_{gen}(f(x+\delta))\Big] + \alpha \sum_{m=1}^{M} w^m \parallel \delta^m \parallel_2 \text{ (generate the noise)}$$

such that

$$\mathcal{H}(T_H) \leq \delta^m \leq x^m \forall m \in \{1, 2, ...M\}$$

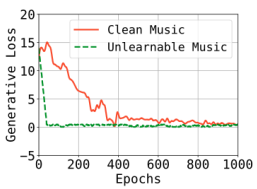
where

$$\mathcal{H}\Big(T_{H(v)}\Big) = 127 \cdot 10^{\frac{1}{4}\log_{10}\left(T_{H(v)}\right) - L_U + 94} \text{ (constrain the magnitude, given note velocity fo for audior MIDI)}$$

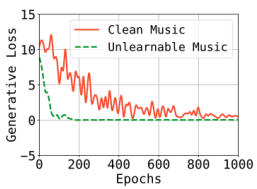
$$\mathcal{H}\Big(T_{H(v_t^m)}\Big) \leq \delta_t^m \leq \mathcal{M}(x_t^m), \forall t, m \text{ (windowize it with a sigmoid function for audio)}$$

in other words, this is how we mathematically describe the high level stuff we have been building up to

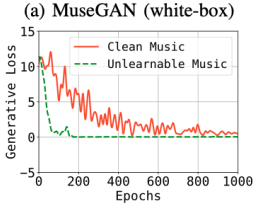
Evaluation



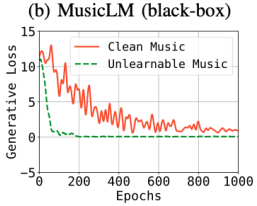
(a) MuseGAN (white-box)



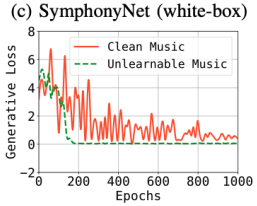
(b) MusicLM (black-box)



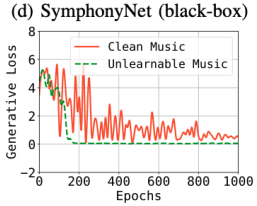
(c) SymphonyNet (white-box)



(d) SymphonyNet (black-box)



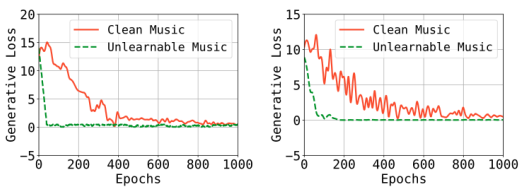
(e) MusicLM (white-box)



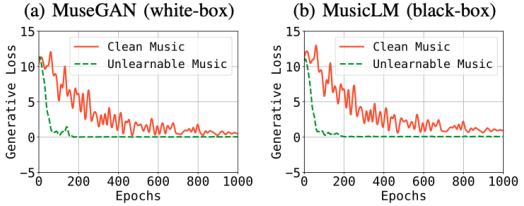
(f) MusicLM (black-box)

Settings	Music	Model	Avg. User Rating			
			H↑	P↑	N↑	OR↑
	Clean Training Music	-	4.17	4.81	4.19	4.45
	Generated Music (Trained on Clean Music)	MuseGAN	4.11	4.05	4.44	4.17
		SymphonyNet	4.44	4.53	4.20	4.11
		MusicLM	4.55	4.49	4.29	4.31
White-box	Unlearnable Training Music	-	4.11	4.72	4.01	4.32
	Generated Music (Trained on Unlearnable Music)	MuseGAN	2.34	3.10	1.20	2.40
		SymphonyNet	2.20	2.94	2.25	2.22
		MusicLM	3.22	3.10	2.29	2.31
Black-box	Unlearnable Training Music	-	4.03	4.45	3.98	4.12
	Generated Music (Trained on Unlearnable Music)	MuseGAN	2.44	3.78	1.94	2.88
		SymphonyNet	2.20	2.94	2.45	2.22
		MusicLM	3.41	3.04	2.78	2.34

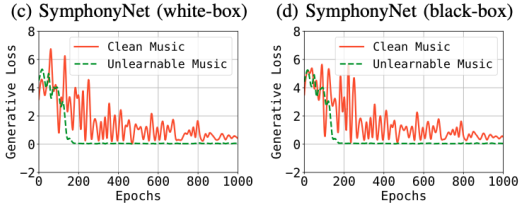
Evaluation



(a) MuseGAN (white-box) (b) MusicLM (black-box)



(c) SymphonyNet (white-box) (d) SymphonyNet (black-box)



(e) MusicLM (white-box) (f) MusicLM (black-box)

Settings	Music	Model	Avg. User Rating			
			H↑	P↑	N↑	OR↑
	Clean Training Music	-	4.17	4.81	4.19	4.45
	Generated Music (Trained on Clean Music)	MuseGAN	4.11	4.05	4.44	4.17
		SymphonyNet	4.44	4.53	4.20	4.11
		MusicLM	4.55	4.49	4.29	4.31
White-box	Unlearnable Training Music	-	4.11	4.72	4.01	4.32
	Generated Music (Trained on Unlearnable Music)	MuseGAN	2.34	3.10	1.20	2.40
		SymphonyNet	2.20	2.94	2.25	2.22
		MusicLM	3.22	3.10	2.29	2.31
Black-box	Unlearnable Training Music	-	4.03	4.45	3.98	4.12
	Generated Music (Trained on Unlearnable Music)	MuseGAN	2.44	3.78	1.94	2.88
		SymphonyNet	2.20	2.94	2.45	2.22
		MusicLM	3.41	3.04	2.78	2.34

https://mosis.eecs.utk.edu/harmonycloak.html

thanks