HarmonyCloak

Presentation

- agenda:
- Issues with data
  - disrespect of art within industries
  - disrespect of privacy
  - lack of ownership
  - surveillance
  - lack of any reason to care
- Watermarking in AI overview
- HarmonyCloak

Notes

Issues at Large

Watermarking in Artificial Intelligence

HarmonyCloak

Introduction:

- DALL-E, Chat-GPT, etc. has revolutionized various domains.
- Music generative AI brings concern of unauthorized exploitation of coprighted music
  - can lead to financial loss and undermine artistic integrity
- we need safeguards for this shit so generative AI does not continue to pose a threat to artists

Prior research in AI Data Protection:

- many different attempts to make data samples unlearnable.
  - error-minimizing noise into data (degrading performance of models trained on said data)
  - traditional data-encryption and differential privacy methods
  - *unlearnable examples (UEs)* do not compromise data quality for normal usage and also fucks with AI models
- most of these assume whitebox setting (all parameters are available)
- distance-maximizing noise to substantially shift image examples within feature space (good idea; but data isn't 100% unlearnable)

Challenges

1. Generative models aim to learn and replicate complex patterns and structures on their own.
   - generative models operate differently from classification models so they need specialized techniques based on generative model characteristics
2. Lack of clear ground truth for un-learnability
   - how do we assess how un-learnable something is?
3. Music is complicated! How do we generate unlearnable music whilst preserving its many different properties?

HarmonyCloak

- defensive framework to render unlearnable examples in music so musicians can safeguard their own work without help from others.
  - focuses on instrumental music, varied rhythmic structures, instrumentation.
- incorporate *error-minimizing* noise into music training samples
  - protective measure to prevent useful data extraction.
- modifications remain imperceptible to human ear through *time-dependent* optimization constraints

- account for Music Production process (translatable across various audio formats like mp3, wav, etc.)

TLDR:
- They claim HarmonyCloak is the first to use imperceptible error-minimizing noise to music that does not affect perceptual quality
- They claim that framework is beyond typical L_p-norm based / psychoacoustic hiding based methods
- Black-box and White-box approaches have both been developed to generate inperceptible noises (widely applicable)
- Extensive experiments have been done.

Preliminaries

Unlearnable Examples Models learn when there is a gap between its current understanding and the data it encounters.
- quantified by loss generated from each data sample.
- UE(s) conventionally for classification models are based on how each data sample can be modified so classification loss is close to 0
- *zero loss* gaslights model: nothing to learn from examples that are given.

Given data input x with label y, we can generate *error-minimizing noise* δ by solving

$$\arg\min_\theta \mathbb{E}_{x,y}[\min_\delta(\mathcal{L}(f(x+\delta),y)] \quad \text{s.t} \quad \|\delta\|_p \leq \varepsilon$$

- f is model, $\mathcal{L}$ is cross-entropy loss, noise magnitude bounded by $\|\delta\|_p$.
- to put it in context for generative models, we have to change things (this is for classification models)

Comparison of NOise-Based Attacks/Defenses
- *Error-maximizing noise* / adversarial attacks maximize prediction errors in AI during testing phase
  - Adversarial Training allows to enhance model robustness by integrating bad examples (min-max optimization problem)
- *Data poisoning attacks* degrade model's performance by tampering with training data
- Backdoor attacks are subset of data poisoning attacks that embed triggers in training data that mislead the model to incorrectly respond to trigger patterns.
- *Error-maximizing noise* on training samples for data poisoning is really effective, but applying error-maximizing noise to training samples doesn't stop a model from learning.
- *Error-minimizing noise* approach for unlearnable examples; error-minimizing noise through min-min optimization process;
  - gaslight model into thinking there is nothing that can be learned.

Representation of Music Signals
- MIDI for music composition, specifying which notes should be played. Also Audio exists.
- Some generative models like Google MuLan and MusicLM use audio directly to generate sound rather than just MIDI.
  - lead to high-fidelity music generation and timbre subtleties that MIDI doesn't always replicate.
- HarmonyCloak needs to accomodate for both MIDI and Audio
- MIDI is matrix with values ranging from 0 to 1, each element represents presence or absence of notes at different time steps.
  - e.g multi track piano roll for M tracks can be represented as tensor $x \in [0,1]^{R\times S\times M}$. R is time steps in a bar, S is number of available note candidates.

- Raw audio interpretation can either be converted from MIDI or represnted as $x \in [-1, 1]^N$ for N samples.

Aspects of Music
- research exists on audio machine learning attacks for imperceptible noise injection, but primary target of these papers is speech data to fuck with speech recognition systems.
- Employ L_p norm or psychoacoustic hiding properties to constrain noise
  - may not be directly applicable for music.
- music is multi-voice / polyphonic; extra layer on top of acoustic and perceptual complexities.
  - [39] provides human-in-the-loop attack to create adversarial music to evade copyright detection, but requires a lot of human effort and effectiveness depends on music complexity.
- No existing research considers audio streaming technology alongside the other aspects of music.

To ensure Perceptual quality of music stays the same when injecting noise, must demand more nuanced and low-effort approach.

Psychoacoustic Modeling
- Sounds falling below certain psychoacoustic hearing thresholds (absolute hearing threshold, etc.) become imperceptible.

Hearing threshold:

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

Frequency masking: loud sounds overwhelm softer sounds
- critical bandwidth around masker frequency; dropoff exists
- masking can also happen when masker and maskee sounds aren't played simultaneously (temporal masking)
- louder sounds can mask quieter ones that occur before (pre-masking) or after (post-masking) occurrence.

MP3 Lossy Compression - combines audio compression tech and psychoacoustic fuckshit for optimal balance between audio fidelity and file size. Good for music distribution.

Masking to noise ratio (MNR) can be calculated, enables encoder to allocate fewer bits to masked regions (compressing audio to smaller file size)
- HarmonyCloak has to work on mp3s.

Deep Music Generation
- Early attempts: RNN / Recurrent Neural Networks
- Recent advancements: GAN / Generative Adversarial Networks and Transformers.
  - MIDINet and MuseGAN use CNN-based GAN for music generation
  - SeqGAN uses RNN GAN framework (generative model is based off a stochastic policy in reinforcement learning)
- Transformers have gotten to be really effective, so we use NLP and Computer Vision for Music Generation, each with encoding strats for capturing complex audio information
- MusicBERT uses OctupleMIDI encoding for granular MIDI event representation, SymphonyNet uses Multitrack multiinstrument MMP Repetable encoding for more complex source material.
- MusicLM: large foundation model for high fidelity music based on textural descriptions
  - can translate complex prompts into music compositions.

  –3 key models: SoundStream for acoustic tokens, w2v-BERT for semantic tokens, MuLan for conditioning

- 2 stages of MusicLM: Semantic modeling stage (learns to map MuLan audio tokens to Semantic tokens)
  - acoustic modeling stage where tokens are predicted based on previously generated tokens and input text.
- MusicGen: auto-regressive Transformer model operating over EnCodec tokenizer. Codebooks sampled at 50Hz

Threat Model & UEs for Gen. AI
- Attacker's capabilities (AI companies / model owners): can scrape music data from Internet or Music streaming platforms to train GenAI models. Leads to copyright infringements, harms musicians.
  - probably has advantages and capabilities (unrestricted access to training dataset and model params)
  - Ability to perform adaptive attack strats for learning the unlearnable
- Defender's Objectives (you, the musician) would like to render your piece inaudible to Generative Audio models.
  - become big challenge for AI company to learn from music.
  - concurrently wants to ensure UEs resemble original music whilst indistinguishable for non-human listeners.
- Defender's Capabilities: Defender has some kind of computer
- Defender can convert compositions into mp3 for distribution and full access to all project files and songs
- Defender has no access to model / training data (in a white bo setting, this would be different. defender must know how to use generative model to train on music data)
- in a black box setting, defender must operate without knowledge of what generative model is used

UEs for Generative AI
- Generative models like Transformers and GANs behave differently

Unlearnable Examples from Autoregressive Models
- For autoregressive models like Transformers, we attempt to minimize negative log liklihood (NLL) of target sequence given predicted / partially generated sequence.
  - This is core for stuff like MusicLM and SymphonyNet. To reduce information that can be extracted by the models, introduce perturbation δ to trick the model into thinking the current sequence is flawless.

For sequence $X_{1:N}$ we use the *min-min* objective function to achieve this:

$$\min_{\theta} \min_{\delta} -\log\left(\Pi_{t=1}^{T} f\left(x_t | x_{t-1} + \delta_{t-1}, ...x_{t-p} + \delta_{t-p} : \theta\right)\right)$$

- x_t is predicted vbalue in sequence at given time t.
- p is autoregressive order

after optimizing for optimal noise, unlearnable data ($x + \delta$) can be used to train autoregressive model. *the result is that the model will no longer generalize on samples since noise loses the learning capability*

Unlearnable Examples for GAN-based Generative Models
- Generative Adversarial Networks (MuseGAN) are different than autoregressives since they capture statistical properties of training data and generate samples resembling the training distribution.
- GAN contains two parts:

1. Generator network (G). Aims to minimize loss through creating more realistic data
2. Discriminator network (D). Aims to maximize loss by figuring out how to better discern real from fake data.

Minimize generator loss by adding δ imperceivable noise onto sample to discourage discriminator from properly classifying sample:

$$\min_G \max_D [\Big(\min_\delta \mathbb{E}_{x+\delta}[\log D(x+\delta)] + \mathbb{E}_z[\log(1-D(G(z)))]\Big]$$

- x is a data sample
- z is a random noise input
- After determining optimal noise and feeding in unlearnable data $x+\delta$ into GAN, the loss reaches a minimum value during initial training rounds, suppressing informative knowledge of data.

HarmonyCloak *Methodologies*

Design Rationale
Four key aspects to tailor objective functions for creating unlearnable music examples:
- Concealing noiss with Music via Frequency Masking
  - Noise should be subtle and within critical bandwidth of the masking music tunes (frequencies must be close to musical notes)
- Dynamic Variations and Temporal Masking
  - Dynamic range and volume changes $\Longrightarrow$ noise must adapt over time and be close to the masking music, especially when they can't be played together.
- Track-Specific Noise Tailoring
  - defensive noises must be tailored for each instrumental (tempo, frequency range, timbre, etc. must be considered)
- Remaining Effectiveness under Lossy Compression
  - Noise must survive compression algorithms. So noise must not surpass dominant musical tones

Constrained Optimization Problem
- We can use this to solve for optimal noise.

Given one bar of multi track music sample $x = [x^1, x^2, ...x^M]$ for M tracks, imperceptible defensive noise is injected into each individual track $\boldsymbol{\delta} = [\delta_1, \delta_2, ...\delta_M]$, giving us $x^i + \delta_i$ for every track.

So we get the bilevel optimization problem:

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

$$\text{s.t}$$

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

- $f(\cdot)$ is the generative model
- $\mathcal{L}_{gen}(\cdot)$ is generative loss of model
- T_H is absolute human hearing threshold
- α is scaling coefficient
- w^m is $1-$ ratio of note velocity of track m to cumulative note velocity of all tracks
  - for balancing noise intensity added to each instrumental track
- $\mathcal{H}(\cdot)$ converts dB SPL threshold to note velocity

$$\mathcal{H}\Big(T_{H(v)}\Big) = 127 \cdot 10^{\frac{1}{4}\log_{10}\left(T_{H(v)}\right) - L_U + 94}$$

- L_U is magnitude of lienar transfer function normalized at 1kHz.
- inner minimization tries to find noise that minimizes overall generative loss on unlearnable music
- outer minimization adjusts model parameters θ (or the generator if GAN) to minimize generative model loss
- regularizer term attempts to minimizes overall amplitude of added noise through reducing L_2 norm of noise.
  - constraints in the $\mathcal{H}(T_H)$ equation above makes sure that noise gets constrained in the masked regions, sandwiched between the music amplitude and the threshold for hearing.

Use Projected Gradient Descent (PGD) as first-order optimization method to solve constrained inner minimization problem.
- Apply Stochastic Gradient Descent (SGD) for outer minimizastion

Window based Strategy for Temporal Dynamics
- Divide the music bar x into short non-overlapping windows during optimization. Each has length l_w.
  - for each window x_t, find frequency v_t^m of dominant musical note for every track (track m) based on cumulative musical note velocities, set defensive noise δ_t^m for window t and track m to same frequency v_t^m.
  - if there is no dominant musical note, no noise will be introduced. Noise is introduced uniquely for each instrumental track, so the strategy hinges upon creating defensive noises for each track.

Apply to windows using a `sigmoid()` function to constrain δ_t^m to the following range for all t windows m tracks:

$$\mathcal{H}\left(T_{H(v_t^m)}\right) \leq \delta_t^m \leq \mathcal{M}(x_t^m), \forall t, m$$

- v_t^m is dominant frequency, $\mathcal{M}(\cdot)$ is velocity of dominant musical note at bar x_t^m

Black-Box Setting
- Defender does not know which generative model is used to train music data, nor access to model for generating music
  - to make error-minimizing noise applicable to generative models, the cross-model transferability needs to be improved

Meta-Learning: strategy for tackling new tasks through *learning to learn.*
- Model first learns knowledge and finds associations / mappings amongst multiple training tasks during the meta-training phase.
- then adapts to unseen task with few examples through fine tuning (meta-testing phase)

Two-step iterative method: to generalize Unlearable Examples:
- Randomly sample $S_1, ...S_Q$ from a collection of generative models
- meta-testing and meta training

Meta-Training:
- Segment music into windows, initialize defensive noise δ_t^m for each window t and track m
- Use first $Q-1$ models to simulate white-box scenario to generate unlearnable music.

Bi-level optimization becomes the following:

$$\min_{\theta} \mathbb{E}[\min_{\delta} \left(\sum_{q=1}^{Q-1} \mathcal{L}_{gen}^{q}(f(x+\delta\}) \right] + \alpha \sum_{m=1}^{M} w^m \parallel \alpha^m \|_2$$

$$\text{s.t}$$

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

- for $q \in S$
- multiple gradients from selected models means we take average of gradients by dividing it by total number of models to ensure balanced optimization. This gives us

$$\nabla_x \mathcal{L}_{gen}(f(x+\delta)) = \frac{1}{Q} \sum_{q=1}^{Q-1} \mathcal{L}_{gen}^{q}(f(x+\delta))$$

Meta-Testing
- done to refine noise for adapting to unseen model.
- fine-tune defensive noise for last sampled model S_Q for generalizing by placing it in a black-box setting
- Divergence Loss (wtf is this) quantifies disparity between distribution of target model's output and distribution of clean music
  - simply put, how effective we are minimizing learning for the model.

We craft defensive noise with optimization objectives
- iteratively adjust perturbation (deviations) to minimize model's loss while maintaining imperceptibility for humans.
- iterative optimization process + divergence loss in black-box setting refines defensive noise

HarmonyCloak for Wave Audio
- Previously were talking about how to use it with MIDI, but we can adapt the system for wave-based formats.

Window the audio file, lengths of roughly 10ms, lengths of roughly 10ms. Compute the STFT to obtain dominant frequencies $\left(T_{H(v_t)}\right)$ in dB SPL. COmpute magnitude $M(v_t)$ by computing the FFT and calculating the relative dB-SPL with

$$\mathbb{M}_t = -20 \cdot \log\left(\frac{M(v_t)}{20^{-6}}\right)$$

introduce the noise δ_t for dominant frequency within each window. Magnitude must be constrained within the following range for all windows t:

$$T_{H(v_t)} \leq \delta_t \leq \mathbb{M}_t$$

- for multi-track wave based audio, operations are applied to each track individaully.

Evaluation
Experimental Setup

Evlauation Metrics:
1. effectiveness (training loss and perceptual quality of music produced by generative models)
2. perceptual quality (generated UEs should be enjoyable to listen to, defensive noises should have minimal impact on music quality)

Intra track and Inter track metrics for music:
- Empty Bars (EB) Ratio: percentage of empty bars in generated music

- Used Pitch Classes (UPC) per Bar: number of unique pitches within each bar of generated music [0, 12].
- Qualified Notes (QN) Ratio: Quantifies percentage of qualified notes in Generated Music
  - qualified note has a duration of at least 3 time steps. provides insight into level of fragmentation or coherence in music.
- Drum Pattern (DP): ratio of notes that conform to 8-16 beat patterns. Indicates adherence to established rhythmic patterns
- Tonal Distance (TD): tonal distance between pair of tracks. Larger TD implies waeker inter-track harmonic relations
  - helps assess level of harmonicity or dissonance, ranges from 0 to 5, 0 for harmonic, 5 for weak inter-track harmonic relation
- Frechet Audio Distance (FAD): statistics on a set of reconstructed music clips to statistics on a set of studio-recorded music
  - reference-free audio quality metric, correlates well with human perception. Low FAD is better.

TD and FAD to measure perceptual quality

Eb, UPC, QN, DP to compare real music with generated samples temporally (effectiveness of generator)

- similar distributions of real and generated samples $\implies$ temporal domain metrics should also exhibit proximity wtf does this mean