**CSTC 210 Final Project: AI Hip Hop Music Video Generator Prototype**

**Introduction**

Music videos are maybe the ideal music visualizers—carefully crafted to complement audio with interesting, narrative-driven visuals that move to the rhythm and mood of the song. However, traditional music videos are static, pre-produced works that cannot respond to music in real time. This project aims to bridge that gap by developing a model capable of generating artificial music videos dynamically from live audio input. At its core, the project leverages RAVE (Realtime Audio Variational autoEncoder) as a feature extractor for audio to provide a compact latent representation of the sound.

Music videos are downloaded and segmented into individual frames and their corresponding audio samples. A custom deconvolutional decoder model is then trained to predict image frames from these audio samples. It would be difficult to do this with raw waveform data, so that's where RAVE comes in: by pretraining RAVE on five hours of hip hop mixes, I ensure it learns a robust latent representation of the genre’s rhythmic and melodic structures. The pretrained RAVE encoder transforms audio samples into latent vectors, which are then passed to the custom decoder to generate corresponding 256x256 image frames. The full pipeline can be summarized as: audio sample → pretrained RAVE model (encoder portion) → latent vector → custom deconvolutional decoder → 256x256 image frame.

As an audio visualization enthusiast, I am interested in how the visual representation of sound can make people feel and enhance the listening experience. A lot of that depends on how visual and auditory art interacts with culture. Inspired by the ideas in *Neural Synthesis as a Methodology for Art-Anthropology in Contemporary Music*, this project positions itself at the intersection of art and cultural inquiry (Dyer). By using neural synthesis to generate dynamic visuals from hip hop music, the model offers a reimagining of the genre’s aesthetic and cultural elements. The abstract outputs, though not literal representations of the music videos, serve as a form of artistic anthropology, exploring how AI can reinterpret the relationship between sound and image reflective of cultural narratives (Dyer). While the outputs are still experimental, the pipeline represents a step toward creating a system capable of generating culturally compelling, real-time music videos driven purely by audio input.

**Data and Preprocessing**

The data for this project are music videos downloaded from YouTube. First, I downloaded about five hours of hip hop mixes, including videos from DJ Noize and DJ Boat. I also downloaded the videos for *Alright* by Kendrick Lamar and *All My Life* by Lil Durk featuring J Cole. I selected these videos, because *Alright* has highly dynamic and stark, culturally relevant visuals—however, the video is all black and white. So, I selected *All My Life* to supplement the former with a more colorful, emotive example. This combination was vaguely intended to capture the visual and thematic range of hip hop. After training RAVE on the High-Performance Computing Cluster, I preprocessed each of the two music videos, splitting them into thousands of frames and their corresponding audio encodings from RAVE.

In line with the principles outlined in *Data Feminism*, this project acknowledges the power dynamics inherent in dataset creation (D’Ignazio, Ch. 6). The choice of these two music videos reflects an effort to engage with the cultural richness of hip hop, but it also raises ethical questions about how AI interacts with such works. By transforming these videos into training data, there is a risk of detaching their aesthetic and cultural significance from their original context. This highlights the need to critically consider how AI systems reframe cultural narratives and whose voices are represented—or overlooked—in the process.

**Methodology**

To handle RAVE’s sequential output, I prepended a stateful RNN layer to the decoder architecture. This modification allowed the model to process variable-length sequences over time and encode temporal dependencies, with the tentative goal of capturing longer-term trends in the music and producing more visually cohesive outputs. The RNN layer was implemented as a Gated Recurrent Unit (GRU) with a persistent hidden state across training batches. However, this addition introduced unexpected challenges: the outputs shifted from earlier, shadowy, human-like forms to abstract, amorphous moving balls of light. This suggested that the RNN either failed to integrate temporal information effectively or overfit to certain patterns in the data.

Following the addition of the RNN layer, the decoder architecture includes a sequence of fully connected linear layers followed by a series of deconvolutional layers. These layers decompress and expand the input latent features into a 256x256 image. I chose this size, since it is large enough to retain the important elements of the videos, like people and faces, but small enough to ensure a reasonable training time.

Loss function experimentation further shaped the methodology. Initially, I used mean-squared error (MSE), which provided pixel-level feedback but proved too granular for capturing higher-level features essential to image quality. To address this, I introduced perceptual loss using VGG16, a pretrained convolutional neural network that evaluates image similarity in a feature space. While this approach reduced numerical error, it destabilized training due to conflicts in input normalization. The perceptual loss required images normalized to [0, 1], whereas the pipeline was optimized for [-1, 1] normalization. Ultimately, under time constraints, I reverted to MSE with [-1, 1] normalization, which offered more stable, though less nuanced, results.

I trained the decoder on the High-Performance Computing Cluster on two music videos: *Alright* by Kendrick Lamar and *All My Life* by Lil Durk featuring J. Cole. The training process involved 60 epochs on the first video, followed by reloading the model and training for an additional 60 epochs on the second video. While this sequential training approach allowed the model to process both datasets, it was suboptimal; a simultaneous training strategy would likely have enabled the model to generalize better across both videos.

As for testing the model, the demo portion of the pipeline offers multiple options for generating visualizations with the model. There are options to recreate a video using only its audio track, reproduce a video based on preprocessed latent vectors, and generate visuals in real time from live audio input. However, I did not use cross-validation during training because the primary goal of this project was not to achieve precise, quantifiable performance metrics but to qualitatively evaluate the artistic interest of the outputs. The nature of this work prioritizes subjective assessment over strict model validation, as in these early stages the primary focus is on exploring the creative potential of the pipeline.

**Results**

The final outputs consist of 256x256 image frames generated from audio input, displayed as video sequences. Early training produced promising results, with shadowy, human-like forms emerging from the decoder. These outputs suggested that the model was beginning to learn meaningful visual patterns from the dataset. However, as the training progressed and architectural changes were introduced—particularly the addition of the RNN layer—these forms gave way to abstract, amorphous shapes resembling moving balls of light. While less visually detailed, these later outputs still exhibit a reactive quality in that they dynamically shift in response to the rhythm and energy of the input music. This improvisational aspect—where the model reimagines novel audio input in unexpected ways—aligns with the idea that we want our computers to improvise not just to mimic human creativity, but to push beyond it, offering new perspectives that emerge from the interplay of data, architecture, and process (Lewis 129). Such improvisations run the risk of unintelligibility, and indeed, the model’s amorphous results are difficult to interpret (Lewis 129). When run on new audio input from the microphone, the system represents a human-computer improvisation, as the computer improvises along with the user to produce visuals inspired by his/her sounds (Lewis 129).

**Reflection**

Looking back on the roadblocks I faced during this project, each one taught me something important about balancing complexity and stability in generative models. Adding the RNN layer, while intended to capture temporal trends in the audio, ended up making the visuals less coherent, showing that adding complexity doesn’t always make a system better. Experimenting with loss functions was another challenge—perceptual loss had potential to improve the outputs, but issues with normalization made it impractical, so I had to go back to using mean-squared error. Training the decoder on two music videos separately also caused problems, as the model seemed to “forget” patterns from the first video when I switched to the second. These challenges showed me how important it is to keep things simple and make sure all parts of the pipeline work well together. While frustrating at times, these roadblocks helped me learn a lot and gave me ideas for how to improve the project in the future.

**Future Work**

It’s important to note that time constraints limited my ability to fully experiment with and refine many of my ideas. With additional time, I could focus on fleshing out effective implementations of the RNN layer, perceptual loss, and simultaneous training. Perhaps I could try different solutions for temporal cohesion, like averaging latent features over time, instead of using an RNN. Once I figure out how to properly employ perceptual loss, I could experiment with hybrid loss functions that combine perceptual loss and mean-squared error to balance high-level feature extraction with pixel-by-pixel alignment. Another avenue would be expanding the dataset to include more music videos, since the two videos I used do not fully represent the diverse range of styles and sounds in hip hop. This would encourage the model to generalize better. I could incorporate different genres of music or even movies and TV shows to make a more general-purpose visualizer. Part of my original grand plan was to build a frontend website that would run the exported models with JavaScript in the browser to visualize the user’s microphone input; this would be a great way to enhance the user experience and provide a platform to showcase the project, but time constraints prevented me from getting that far.

A further downside to this project is that the final output does not clearly reflect the process and work required for its generation. The technical achievements of this project are hidden in the code. Perhaps in the future, the internal process could be visualized alongside the output, following the principle of Show Your Work from *Data Feminism* (D’Ignazio, Ch. 7).

**Works Cited**

D’Ignazio, Catherine, and Lauren F. Klein. *Data Feminism*. The MIT Press, 2023.

Dyer, Mark. “Neural synthesis as a methodology for art-anthropology in Contemporary Music.” *Organised Sound*, vol. 27, no. 2, Aug. 2022, pp. 219–226, https://doi.org/10.1017/s1355771822000371.

Lewis, George E. “Why do we want our computers to improvise?” *Oxford Handbooks Online*, 5 Feb. 2018, https://doi.org/10.1093/oxfordhb/9780190226992.013.29.