Deep Philharmonics

Aditya Kumaran
akumaran6@gatech.edu

Kartik Sinha
kartik24@gatech.edu

Malav Patel
malav.patel@gatech.edu

Saahir Dhanani
saahir@gatech.edu

Abstract

In developing a multi-stage pipeline of stem isolation, music generation, and vocal cloning, we chained together four cutting-edge deep learning models to produce a novel musical artifact. We discovered that while our generative Transformer worked particularly well with piano-based instrumentals, it struggled to inference MIDIs without defined melodies. Furthermore, the isolation model generalizes poorly to genres outside of mainstream English music; the vocal cloning model also produced more promising inferences on rap acapella tracks than pop tracks, given their similarity to the spoken word.

1. Introduction

Our overarching objective within this project was to develop a product that is able to generate an entirely fresh piece of music by replacing the vocals of a given song with a user's voice based on a few training samples, and synthesizing an instrumental from a given track. This is not only appealing technically as a combination of various deep learning methodologies, but, if successful, also has the potential to disrupt the music industry by providing a new way for users to interact with and create music. Our implementation is concerned with crafting a pipeline from the existing methodologies of four different research papers, combining their individual use cases into a combined song generation pipeline. We begin with an isolation network, which will be able to separate an input track into raw vocals and instrumentals (drums and production), the latter of which we process as input into a music generation network, which will generate an additional phrase based on the input notes and key. Finally, we use our extracted vocals and process them to serve as input to a vocal cloning network, which allows us to generate a vocal lead for our generated instrumental. We combine the generated music and vocals to create a completely new piece.

The problem of music generation is conventionally tackled using RNNs and LSTMs, or even GANs, which are architecturally limited by two factors - they necessarily follow sequential processing, in that sentences must be processed word by word, and the fact that they utilize hidden states to retain past information. These kinds of Seq2Seq models are dependent on the Markov model, which means that each state is dependent only on the immediately previous state. Sequential processing means that these models can't be trained in parallel since the word encodings are required in the order that the sentence is formatted. Similarly, the Markov property of these models means that each word embedding only strongly impacts the meaning of the immediate next word, and its importance decreases exponentially[4]. While LSTMs do handle this problem relatively better than RNNs thanks to more thorough processing of prior hidden states and even bi-directional models, such workarounds don't make for a rigorous solution for longterm dependencies.

Conversely, Transformer models differ from the existing architecture in that they are non-sequential and process sentences as a unit rather than word by word, utilize positional embeddings as opposed to recurrences to encode information specific to a position of a given token in a sentence through learned weights, and implement self-attention, which is a novel architectural addition capable of computing similarity scores between words in a sentence. These features ensure that Transformers don't suffer from the same longterm dependency issues as outlined earlier, since they no longer rely on hidden states to remember stored information and instead process entire sentences together to alleviate the risk of losing dependency information. Positional embeddings and multi-head attention provide further insight into relationships between different words and maintain a rigorous understanding.

Regarding datasets, the PyTorch Music Transformer model [2] is derived from GPT-2 and is trained over the MAESTRO MIDI dataset, comprising of 200 hours of virtuosic piano performances captured with fine alignment between note labels and audio waveforms. The DeMUCS model [5] from Facebook Research is a model featuring

Hybrid Transformer-based source separation, trained over the MUSDB HQ dataset of 150 full length music tracks (10h duration) of different genres along with their isolated drums, bass, vocals and other stems. The VITS SoftVC Voice Conversion model is specifically designed to adopt the unique vocal characteristics of a selected voice. To train this model, we constructed a dataset using lecture recordings from this class, all of which were spoken by Professor Zsolt Kira. To create the dataset, we segmented the lecture videos into 10-second sections, which were then resampled to 44100Hz to ensure a consistent format across all data points. This process ensured that the model learned from the dataset effectively and produced high-quality results with a high level of accuracy and precision. After preparing the dataset, we trained the VITS SoftVC Voice Conversion model for approximately 200 epochs using 420 selected segments of Professor Zsolt Kira's voice. This training allowed the model to deeply learn the nuances and complexities of Professor Zsolt Kira's voice.

2. Approach

Pursuing our passion as audiophiles, we created an AI system that generates new music by remixing a given song, while preserving the original song's melody and lyrics. Our purpose was to complement the musical ideation and production process, while simultaneously providing new sources of inspiration for writers, a way to cure writer's block, and to bring novel music to the world.

To achieve this, we created a DAG (directed acyclic graph) workflow ML pipeline. First we take in an input song as an audio waveform and perform music source separation using a Wave U-net and Conv-Tasnet inspired transformer-based model. This gives us the song's stems–bass, drums, vocals, and "other", which contains the rest of the mix. We first pass in the vocals into our voice conversion model along with the trained weights on a target voice source (our exemplar for the purposes of our project is Professor Zsolt Kira). This gives us the vocals of the song with the same melody and lyrics, however it is now in the voice of the target source. Additionally, we give as input the remainder of the stems produced by Demucs to a music generation transformer model. However, this model takes in as input MIDI files. MIDI is a communications protocol to describe musical notes digitally and is a ubiquitous and industry standard format. This introduced an additional task for our ML pipeline to form—namely, to convert an audio file into its representative MIDI format. To perform this task, we employ Basic Pitch [1], a CNN-based model released by Spotify's Audio Intelligence Lab which performs polyphonic note transcription and is instrument agnostic. Thus, given a Demucs-generated audio file containing instrumental music, Basic Pitch converts it into a format of pitch-length note sequences (along with other musical information). Next, this MIDI file is supplied to our music generation transformer-based model, which produces more MIDI given the file as context. The produced music is completely AI-generated. Finally, this MIDI is played back in any digital instrument of choice and merged with the converted vocals to produce a new piece of music.

As our goal was to craft a multi-model pipeline, the key problem we anticipated was ensuring the various models interfaced well in the various data formats required by each model. Training time was also a significant concern, seeing that we were dealing with multiple models. In the essence of time, we used pre-trained weights for two of the four models (Basic Pitch and Demucs) and found them to work quite well both quantitatively and qualitatively with our dataset. We faced issues in interfacing the source-separation output with the generative model's input. After testing several different solutions and models, we found Basic Pitch's CNN-based architecture (trained on the notable Maestro dataset among others) to work quite well for this transcription task. However, because our pipeline entails a conversion from audio to midi, i.e. conversion from the waveform domain to the relatively highly discretized MIDI format domain, this process is inherently lossy. Although the resulting output is polyphonic, we lose various characteristics of sound, notably timbre. This loss of information implies a qualitatively weaker output from our music generation model.

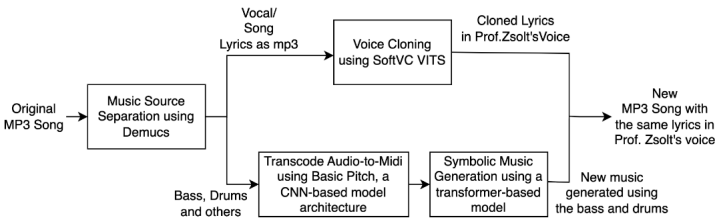

Figure 1. Modeled Pipeline

3. Experiments and Results

To evaluate our pipeline we performed a few analyses while training the model and then we evaluated our pipeline using a state-of-art pre-trained music-similarity model.

3.1. Training the pipeline

To train our pipeline, we started with training each of the individual components. For the Demucs model, we used the pre-trained model provided in the research paper (resource). The original experiments were done on 8 Nvidia V100 GPUs with 32GB of memory, using fp32 precision. They trained the model for 1200 epochs of 800 batches each over the MUSDB18-HQ dataset, completed with the 800 curated songs dataset presented in Section 4, sampled at 44.1kHz and stereophonic (reference). This is the first

step of our pipeline where a song is split into multiple categories. This model splits the song into vocals, bass, drums and other noise.

Then, to train the Voice Cloning model, we passed in the OMSCS video recordings (resource) of Prof. Zsolt Kira with some hyperparameter tunings. Based on our analysis, we observed that a majority of the model hyperparameters were optimally calibrated. Nevertheless, we discovered that modifying the batch size during training had a positive impact on the model's performance. Specifically, increasing the batch size enabled the model to achieve improved results by enabling greater computational efficiency and better utilization of available resources. In addition, we conducted manual evaluations of the model's performance at regular intervals of 50 epochs. This periodic assessment allowed us to gauge the quality of the model's performance and determine whether it was on track to meet our goals. By stopping the training process early when satisfactory results were obtained, we were able to significantly reduce training time while still achieving optimal results. We increased the segment_size parameter, which is the length of the audio and the tensor to be passed to the decoder, which eventually increased the training speed of the decoder by 10%. Additionally, the AdamW optimizer was used in this work because it has been shown to improve generalization performance and prevent overfitting in large-scale deep learning models. We also computed the loss using different functions and plotted the overall training loss. In figure 2, we used the Mel Spectrogram as our loss function. The Mel spectrogram is actually a common general representation for modeling audio. Along with being a better correlate with human pitch perception, it's also more compact than normal spectrograms. In figure 3, we used the KL-Divergence Loss to understand the training loss with the increasing epochs. In figure 4, we computed the overall loss with the number of epochs and found that increasing the number of epochs decreases the overall training loss for our model. Hence, we decided to re-train our model by tuning the segment_size, the number of epochs and the learning rate. This is the second part of our pipeline where the model is trained to convert the lyrics of the song in Prof. Zsolt Kira's voice.

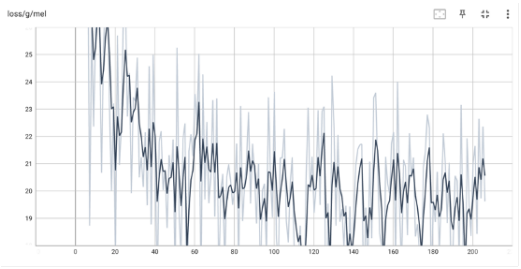

Figure 2. Mel Spectrogram Loss vs. the number of epochs

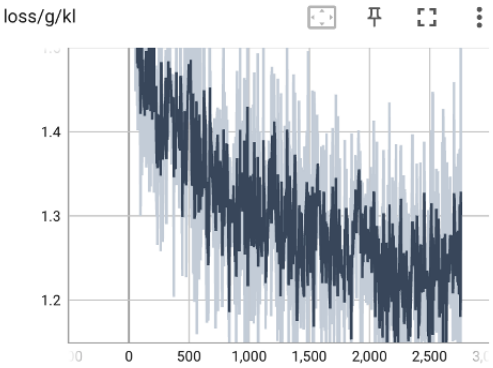

Figure 3. KL Divergence Loss vs. the number of epochs

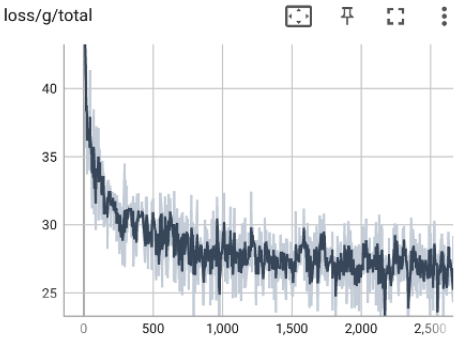

Figure 4. Total Loss vs. the number of epochs

Since the output of the Demucs model is a .mp3 file and the music generation transformer takes a midi file as its input, we used Basic Pitch (a python library) to convert the output of the Demucs model to the specific format. This library is used for Automatic Music Transcription using a lightweight neural network.

For the music generation transformer, the model generates polyphonic music of multiple tracks (instruments) like phrases consisting of bass, drums, guitar, piano, and strings tracks. We re-trained the model with tuned hyperparameters on the training data collected from Lakh Pianoroll Dataset.

We conducted multiple experiments with various learning rates and epochs as shown in figure 5, the best results were obtained when training the model for 19 epochs and the learning rate of 0.0007. This model was implemented as a deep CNN to generate a two-stream architecture and the results were combined via a mask generated by the foreground stream. We noticed that cross entropy had a faster learning rate and convergence than mean squared error with our dataset. This helped the neural network quickly correct its errors and reduce its loss. From figure 5 and figure 6, we

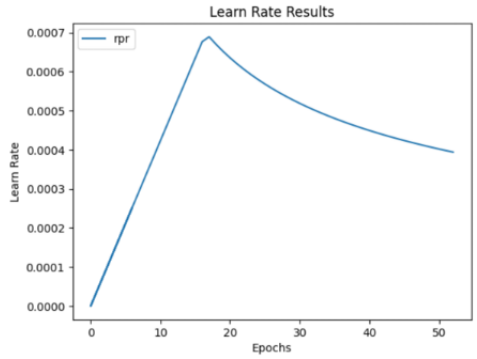

Figure 5. Learning Rate vs. the number of epochs

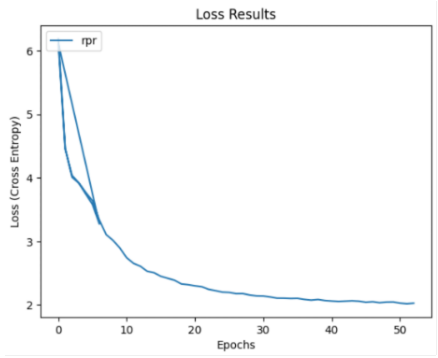

Figure 6. Cross Entropy Loss vs. the number of epochs

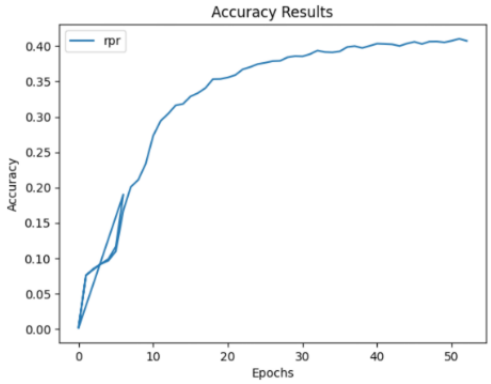

Figure 7. Accuracy vs. the number of epochs

can observe that as the number of epochs increases, the loss decreases and the training accuracy increases.

3.2. Testing our pipeline

To test our pipeline, we started with three music songs: one female Korean song, one female Hindi song, and one male Spanish song that were downloaded from YouTube. To evaluate our work, we passed each of these songs individually through our pipeline and compared the output with the input generated. To precisely assess our outputs, we used an algorithm that produces an estimate of the audio signal's frequency content to compose the magnitude of the function that calculates the windowed discrete-time Fourier transform for the given audio input. The algorithm includes computing cross-correlation in the spatial and frequency domain rather than comparing audio files directly.

In addition, we conducted experiments involving the manual adjustment of pitch in order to achieve a more realistic output voice. However, we discovered that this approach resulted in the entire song being in a uniform pitch, which failed to accurately represent the original song. Instead, we utilized the model's capability to automatically adjust the pitch of the output voice to match both Professor Zsolt Kira's natural speaking pitch and the pitch of the voice in the song. While this yielded significant improvements in the voice's realism, it also produced unnatural fluctuations in pitch. We predict that training the model for a longer period and with larger amounts of data would enable a more refined understanding of the natural pitch in Zsolt Kira's voice and allow for more effective and precise modifications of the pitch.

4. Evaluation

Original Songs	Generated Songs
This Love	This Love feat. Prof. Kira
Para Enamorarte	Para Enamorarte feat. Prof. Kira
KKKG	KKKG feat. Prof. Kira

Table 1. Results.

In Table 1 we have attached the links to the original songs that were inputted into our pipeline and the modified songs that were generated through the series of models. Contrasting their similarities and differences qualitatively highlights the performance of the pipeline.
The first song is a Korean song sung by a female singer.

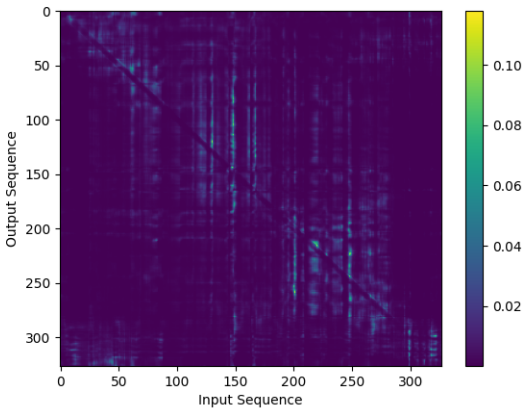

Figure 8. "This Love" - Correlation Matrix

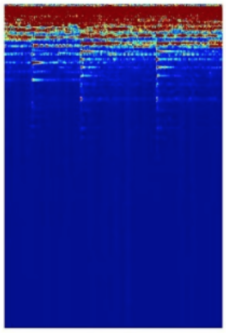

Figure 9. ”This Love” - Spectrogram

For every song, we generated a correlation matrix and a spectrogram to understand the correlation between the input sequence and the output sequence through our pipeline. For the ”The Love” song we found that the max absolute correlation is 1738.2269287109375.

The second song is a Spanish song sung by a male singer. We have generated two graphs: a frequency signal comparison and Power vs. Lag. The maximum absolute correlation value is 8578.802734375, indicating an extremely strong correlation relationship between the input and the output song.

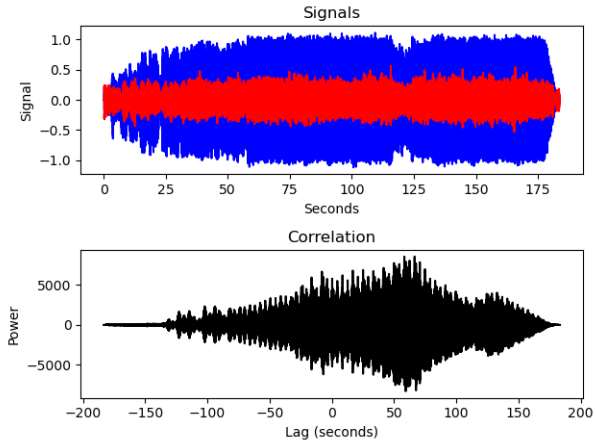

Figure 10. ”Para Enamorarte” - Max absolute correlation: 8578.80

The third song is a Hindi song sung by a female singer. As observed in figure 11, the max absolute correlation value is 1248.8504638671875 which indicates a strong positive between the input and output song.

5. Ethical Implications

The primary ethical considerations that emerge from the undertaking of our exploration are concentrated in the vocal cloning portion of our pipeline. Naturally, the ability

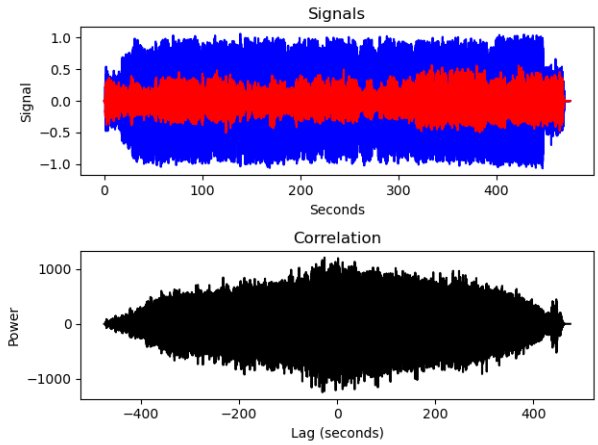

Figure 11. ”KKKG” - Max absolute correlation: 1248.85

to develop a network that can clone an individual’s voice and produce a synthetic clip that is able to be tailored to the user’s preference raises concerns regarding identity theft and evidence manipulation in a manner that is reminiscent of deepfake technology. It is important to receive explicit permission from any vocal subject whose recording is used to train such models over, or utilize specifically license-free voice samples.

A secondary concern that may arise from the success of such an endeavor is that musicians’ work and the study of music theory in general may become less valuable if networks can begin to generate competent music independently. While it is true that the value of music does lie in the personal and idealistic expression of human artists, there is evidence that synthetically generated art is equally capable of positive public reception, as has been the case with AI art. This is, of course, a far lesser concern given our reservations about the degree of potential success achievable in this research area as a whole in the near future, particularly after reflecting upon the output of our own pipeline. Nevertheless, the mitigation of such a concern is the greater responsibility of consumers of art in society as a whole in order to preserve the cultural importance and idealistic integrity of music to humanity.

6. Conclusion and Findings

We aimed to develop a novel musical artifact by creating a multi-stage pipeline that combined stem isolation, music generation, and vocal cloning. By leveraging the strengths of various deep learning models, we successfully crafted a pipeline that generated an entirely new piece of music using a user’s voice and a given track. Our results were fairly promising, however, there is still considerable room for improvement. In our pipeline, which follows a directed acyclic graph (DAG) structure, earlier elements exert a considerable

impact on the quality of the final output.

Primarily, this is attributed to our isolation model, from which we obtain vocals for the vocal cloning model and instrumental stems for generating a new backing track. Many popular songs feature multiple singers who harmonize or include backing vocals, along with effects such as reverb or echo. The vocal isolation model accurately isolates these vocal elements but lacks the ability to separate backing vocals, differentiate between multiple singers, or properly handle effects applied to vocals. Moreover, artifacts from other instruments may still be present in the isolated vocal track. Consequently, when this vocal track is input into the vocal cloning model, it attempts to replicate all elements, including artifacts, backing vocals, multiple voices, and effects. This results in the target voice sometimes producing peculiar noises, as it is utilized to replicate numerous elements simultaneously, some of which should not be replicated by a voice. Additionally, the cloning model's pitch adjustment efforts can sometimes lead to significant pitch shifts throughout the track. The vocal cloning model also faces challenges in generalizing to languages other than English, occasionally generating incoherent noises for otherwise normal lyrics in a different language. We hypothesize that this is due to the fact that the So-Vits architecture utilizes a text encoder, similar to the Vits project, that was mainly trained on English text [3]. Moreover, we currently simply merge the generated instrumentals and new vocals together; however, this combined output could also serve as input for another model, introducing a new learned aspect to our network and transforming a previously deterministic effort into a more dynamic and adaptive process.

Despite these technical limitations, we observed that combining these models in the pipeline produced results surpassing our initial expectations. Our work demonstrates progress in the realm of music generation, showcasing the potential for further advancements and refinement in generating high-quality, novel musical compositions using artificial intelligence. In future research, we aim to explore techniques and strategies that can enhance the integration between these models, ultimately improving the overall quality and coherence of the generated music.

7. Work Division

For general information about the delegation of work among team members, please see Table 2.

Aditya explored state-of-the-art music generation datasets and models including CLaMP, MuseGAN, and PyTorch Transformer; brainstormed pipeline DAG for model structure; coordinated submission documents for proposal and paper; performed testing, intrusive architecture tuning experiments, and retrained Music Generation Transformer Model portion of pipeline.

Kartik read various research papers in the field, notably Demucs, Wave U-Net, Conv-Tasnet, and Basic Pitch; worked with music-source separation models and transcription models; tested various open-source frameworks and implemented a cross-correlation (FFT convolution) similarity metric. Performed qualitative analysis of results.

Malav participated in exploration of optimized vocal isolation models; coordinated team meetings; tested various open-source frameworks and similarity metrics; corroborated quantitative analysis metrics to evaluate results.

Saahir brainstormed potential project outcomes, drawing from state-of-the-art audio-focused research; completed vocal isolation portion of pipeline; curated custom audio recording dataset and fine-tuned vocal cloning model.

For links to our codebase and detailed work, please refer to our Github: GitHub Repo.

Student Name	Contributed Aspects
Aditya Kumaran	Model Exploration, Coordination, Music Generation
Kartik Sinha	Literature Review, Testing, Metrics and Analysis
Malav Patel	Testing, Vocal Isolation, Metrics and Analysis
Saahir Dhanani	Model Exploration, Track Isolation, Vocal Cloning

Table 2. Contributions of team members.

References

[1] Rachel M. Bittner, Juan José Bosch, David Rubinstein, Gabriel Meseguer-Brocal, and Sebastian Ewert. A lightweight instrument-agnostic model for polyphonic note transcription and multipitch estimation, 2022. 2

[2] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Ian Simon, Curtis Hawthorne, Andrew M. Dai, Matthew D. Hoffman, Monica Dinculescu, and Douglas Eck. Music transformer, 2018. 1

[3] Jaehyeon Kim, Jungil Kong, and Juhee Son. Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech, 2021. 6

[4] James A. Michaelov, Megan D. Bardolph, Seana Coulson, and Benjamin K. Bergen. Different kinds of cognitive plausibility: why are transformers better than rnns at predicting n400 amplitude?, 2021. 1

[5] Simon Rouard, Francisco Massa, and Alexandre Défossez. Hybrid transformers for music source separation, 2022. 1