DeepDeadNet

Benjamin Wakefield
Northeastern University
wakefield.b@northeastern.edu

ABSTRACT

DeepDeadNet is a web application built with Python and Django, and leverages the Musika and Demucs libraries. It allows users to generate new music in the style of the Grateful Dead, and isolate or mix each instrument to their preference. The goal of the project is to provide an accessible graphical user interface to people who are interested in AI music generation, but are unfamiliar with command-line interfaces. Additionally, this project combines multiple neural networks into a single workflow, in order to demonstrate how they can be applied for a practical use case. Broadly, this project aims to explore the current potential of music generation and alteration by artificial intelligence.

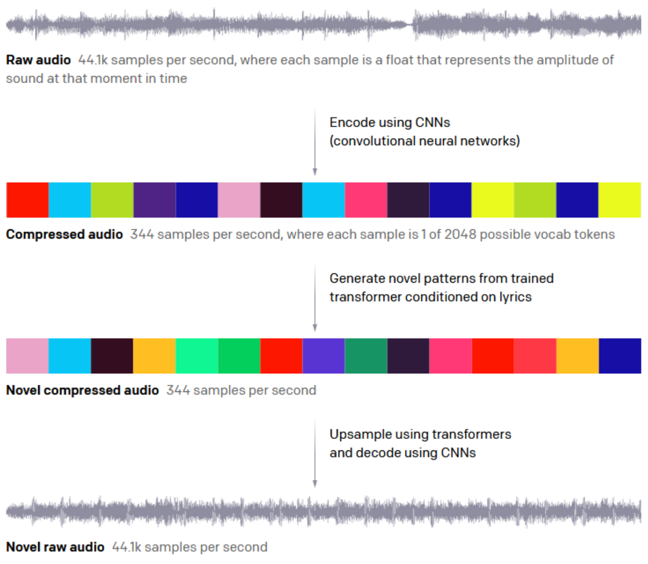
1. INTRODUCTION

Recent advancements in AI have made it possible to teach machines to recognize patterns in music and generate new compositions based on those patterns. By training an AI model on the Grateful Dead's music, we can create new pieces that capture the essence of the band's unique style. This project has the potential to push the boundaries of what AI can do in the realm of creative expression and could pave the way for new forms of music production. Additionally, it provides a unique listening experience for the Deadheads who just can't get enough!

2. BACKGROUND

Jukebox, MuseNet, Riffusion, and Musika are all AI music generation models that use deep neural networks. However, they use different types of neural network architectures and techniques to generate music, and have varying levels of usability.

2.1 Jukebox

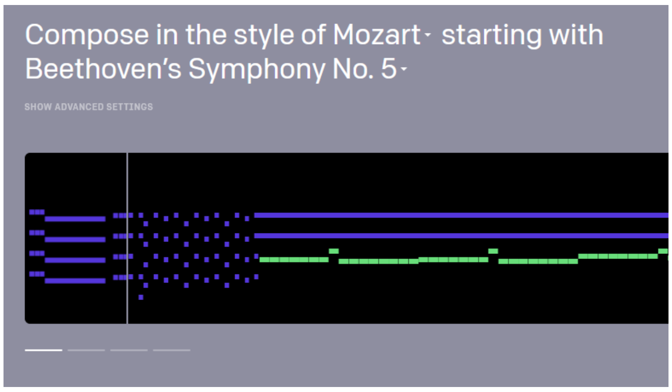

Raw audio 44.1k samples per second, where each sample is a float that represents the amplitude of sound at that moment in time
Encode using CNNs (convolutional neural networks)
Compressed audio 344 samples per second, where each sample is 1 of 2048 possible vocab tokens
Generate novel patterns from trained transformer conditioned on lyrics
Novel compressed audio 344 samples per second
Upsample using transformers and decode using CNNs
Novel raw audio 44.1k samples per second

OpenAI Jukebox[1] is an AI system developed by OpenAI that uses a combination of deep neural networks and natural language processing to generate music in a variety of styles and genres. The system is based on a large corpus of music that has been preprocessed and organized into a hierarchy of musical concepts, such as scales, chords, and melody lines. When a user inputs a musical prompt, such as a short melody or a genre, the system generates a sequence of musical events that follow the input and are consistent with the hierarchical structure of the music corpus.

OpenAI Jukebox is trained on a dataset of over a million songs, and its deep neural networks have billions of parameters. The system is capable of generating original compositions in a wide range of styles and genres, from classical and jazz to pop and hip-hop. The generated music can be customized by adjusting parameters such as the tempo, key, and instrumentation.

Although the results from Jukebox are stunning, the time required both to train a model and generate audio is excessive. It takes hundreds of hours to train a model, and generation requires a few hours for each second of audio, causing it to not be practical for this project.

2.2 MuseNet


Compose in the style of Mozart starting with Beethoven's Symphony No. 5

MuseNet[2] is an AI system developed by OpenAI that uses deep neural networks to generate music in a wide range of genres and styles. The system is based on a generative model architecture that uses a combination of hierarchical latent variables and autoregressive models to generate long sequences of musical events. MuseNet is trained on a massive dataset of MIDI files that cover a broad range of musical genres and styles, including classical, jazz, pop, and rock. The system can generate original compositions of varying lengths and complexity, and users can specify parameters such as the genre, tempo, and instrumentation to customize the output.

One of the key features of MuseNet is its ability to generate music that is both diverse and coherent. The system is able to learn and incorporate complex musical structures and patterns from the training data, enabling it to generate music that is musically consistent and coherent, while also being diverse and unpredictable.

Due to the fact that MuseNet is designed to process MIDI datasets, it would not be practical for this project. Although there is a limited amount of MIDI transpositions for Grateful Dead songs, the amount of raw audio available far outweighs this. However, a future project might involve training a MuseNet model on the guitar solos of Jerry Garcia (the lead guitarist of the Grateful Dead) by manually transcribing them to MIDI.

2.3 Riffusion

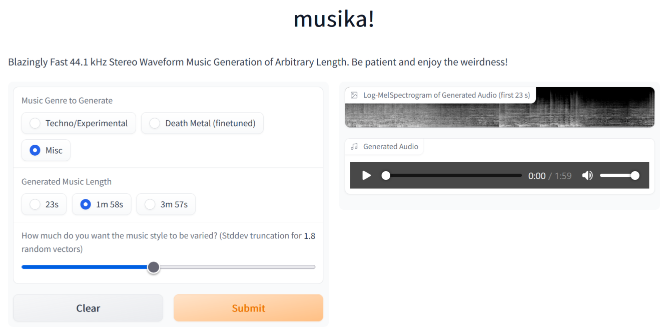
Riffusion[4] is an open-source music generation system developed by the Riffusion research group at the University of Montreal. The system uses a combination of deep neural networks and evolutionary algorithms to generate novel and creative musical sequences, such as melodies, chords, and rhythms.

The system is based on a generative model architecture that uses deep recurrent neural networks (RNNs) to learn the statistical patterns and relationships in a large corpus of MIDI data. The system also employs an evolutionary algorithm to guide the generation process, allowing the model to explore and discover new and creative musical sequences that are not present in the training data.

One of the unique features of Riffusion is its ability to generate music that is both diverse and coherent. The system can generate complex and intricate musical structures and patterns, while also maintaining a consistent musical style and coherence. The system is also highly customizable, allowing users to specify various parameters such as the style, genre, and complexity of the music.

Since Riffusion performs audio generation quickly, it is a reasonable choice for this project. However, there are currently no published methods for reliably training a Riffusion model, which presents a roadblock. Additionally, the fidelity of the audio produced by Riffusion is very low, since it relies on converting compressed images of spectrograms to audio. Nevertheless, as neural networks designed for processing images improve, and the resolution of the images is increased, Riffusion could yield very promising results.

2.4 Musika


musika!

Blazingly Fast 44.1 kHz Stereo Waveform Music Generation of Arbitrary Length. Be patient and enjoy the weirdness!

Music Genre to Generate: Techno/Experimental, Death Metal (finetuned) [✓ Misc]

Generated Music Length: 23s [✓ 1m 58s] 3m 57s

How much do you want the music style to be varied? (Stddev truncation for random vectors) 1.8

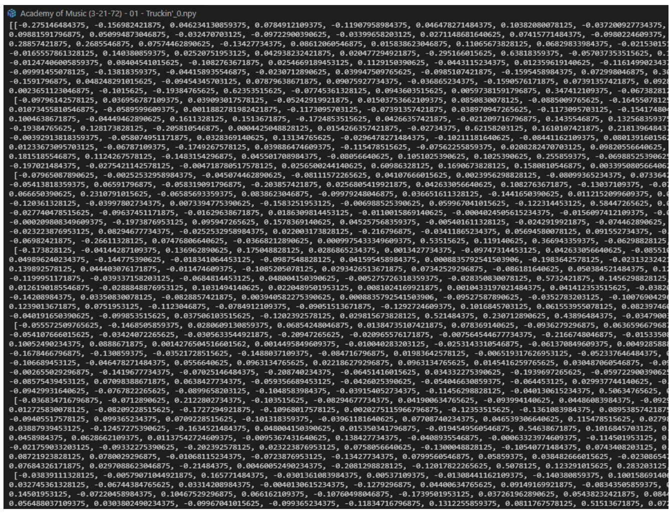
Musika [3] is a music generation system that can be trained on a single consumer GPU and allows for much faster than real-time generation of music of any length on a consumer CPU. The system uses a two-step process to generate music: first, it learns a compact representation of spectrogram magnitudes and phases with adversarial autoencoders, and then it trains a Generative Adversarial Network (GAN) on this representation for a particular music domain. A latent coordinate system enables generating arbitrarily long sequences of excerpts in parallel, while a global context vector allows the music to remain stylistically coherent through time. Musika is designed to be user-controllable, allowing for novel ways of composing or performing music. The system is evaluated quantitatively to assess the quality of the generated samples and is showcased for user control in piano and techno music generation.

Due to the incredibly fast speed of music generation, Musika was chosen as a foundation for DeepDeadNet. It also satisfies the requirement of being able to 'easily' train a neural network on a dataset of raw audio. Compared to the other techniques, Musika is the most suitable for the goal of this project. Although OpenAI's Jukebox yields much better results, it can take multiple hours to generate a single second of audio, while Musika is able to generate a few minutes of audio in a few seconds.

3. PROJECT DESCRIPTION

3.1 Data Consolidation

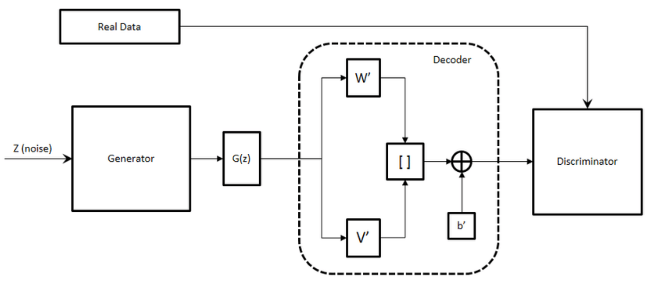
In order to train a Musika model, a huge quantity of input data is required. Thankfully, the live performances of the Grateful Dead have been archived extensively, and can be downloaded in bulk. However, the recordings are not all high-quality, and needed to be cleaned up. First, each individual audio file was normalized to a maximum peak of 0 decibels. This was done in order to prevent unnatural fluctuations in volume in the generated audio. Additionally, each track commonly contained audience noise at the beginning and end. To prevent this from bleeding into the models, each track was batch processed in order to remove the first and last ten seconds.


Academy of Music (3-21-72) - 01 - Truckin'_0.npy

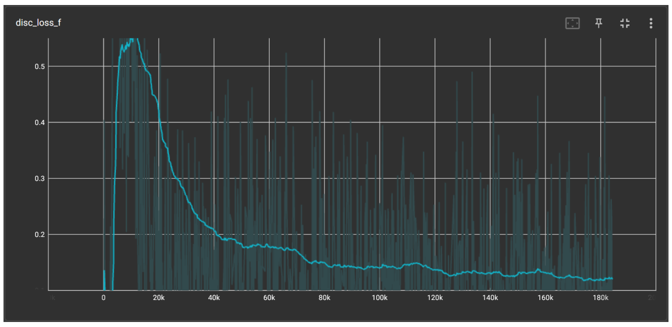
```
[[-0.275146484375, -0.156982421875, 0.046234130859375, 0.0784912109375, -0.11907958984375, 0.046478271484375, 0.10382080078125, -0.037200927734375,
0.09881591796875, 0.050994873046875, -0.032470703125, -0.09722900390625, -0.03399658203125, 0.0271148681640625, 0.07415771484375, -0.0980224609375,
0.28857421875, 0.2685546875, 0.07574462890625, -0.13427734375, 0.08612060546875, 0.015838623046875, 0.11065673828125, 0.06829833984375, -0.02153015
-0.0165557861328125, 0.140380859375, 0.02520751953125, 0.042938232421875, 0.020477294921875, -0.295166015625, 0.63818359375, -0.0570373535156 25, 0.1
-0.01247406005859375, 0.08404541015625, -0.1082763671875, 0.0254669189453125, 0.1129150390625, -0.0443115234375, 0.012359619140625, -0.1161499023437
-0.09991455078125, -0.13818359375, -0.044158935546875, -0.0230712890625, 0.039947509765625, -0.0985107421875, -0.1595458984375, 0.072998046875, 0.3
-0.1591796875, 0.048248291015625, -0.09454345703125, 0.07879638671875, 0.09075927734375, -0.036865234375, -0.1590576171875, 0.07391357421875, 0.092
0.0023651123046875, -0.1015625, -0.19384765625, 0.62353515625, -0.07745361328125, 0.0943603515625, 0.00597381591796875, 0.347412109375, -0.067382812
 [-0.09796142578125, 0.036956787109375, 0.039093017578125, -0.0524291992187 5, 0.0150375366210937 5, 0.0850830078125, -0.0885009765625, -0.1645507812
0.01073455810546875, -0.0589599609375, 0.0011882781982421875, -0.1173095703125, -0.07391357421875, 0.038970947265625, -0.1173095703125, -0.154174804
0.1004638671875, -0.04449462890625, 0.1611328125, 0.1513671875, -0.1724853515625, 0.04266357421875, -0.021209716796875, 0.1435546875, 0.13256835937
-0.19384765625, 0.128173828125, -0.205810546875, 0.0004425048828125, 0.0154266357421875, -0.02734375, 0.62158203125, 0.1610107421875, 0.21813964843
-0.00392913818359375, -0.05807495117187 5, 0.0328369140625, 0.13134765625, -0.0296478271484375, -0.1021118164062 5, -0.08441162109375, 0.0801391601562
0.01233673095703125, -0.06787109375, -0.1749267578125, 0.03988647460937 5, -0.115478515625, -0.0756225585937 5, 0.0208282470703125, 0.0982055664062 5,
0.1815185546875, 0.1124267578125, -0.1483154296875, 0.04550170898437 5, -0.08056640625, 0.1051025390625, 0.1025390625, 0.255859375, -0.06988525390625
-0.197021484375, -0.0275421142578125, -0.00471878051757812 5, 0.0256500244140625, 0.60986328125, 0.1690673828125, 0.1580810546875, 0.0033950805664062
 [-0.07965087890625, -0.0025253295898437 5, -0.04507446289062 5, -0.08111572265625, 0.0410766601562 5, 0.0023956298828125, -0.0809936523437 5, 0.0733642
-0.05413818359375, 0.06591796875, -0.058319091796875, -0.203857421875, 0.025680541992187 5, 0.042633056640625, 0.1082763671875, -0.13037109375, -0.07
0.066650390625, 0.2310791015625, -0.06585693359375, 0.0838623046875, -0.09979248046875, 0.0366516113281 25, -0.1441650390625, 0.011215209960937 5, 0.0
-0.120361328125, -0.03997802734375, 0.00733947753906 25, -0.1583251953125, -0.0069885253906 25, 0.05996704101562 5, -0.122314453125, 0.58447265625, 0.0
-0.027740478515625, -0.09637451171875, -0.01629638671875, 0.0186309814453125, -0.0110015869140625, -0.00040245056152343 75, -0.0156097412109375, -0.0
-0.00020980834960937 5, -0.1973876953125, 0.095947265625, 0.1578369140625, 0.045257568359375, -0.005401611328125, -0.0242919921875, -0.07446289062 5,
-0.0232238769531 25, 0.08294677734375, -0.0252532958984375, 0.0220031738281 25, -0.216796875, -0.03411865234375, 0.05694580078125, 0.091552734375, -0.
-0.06982421875, -0.26611328125, 0.07476806640625, -0.0366821289062 5, 0.000997543334960937 5, 0.53515625, 0.119140625, 0.366943359375, -0.06298828125,
 [-0.173828125, -0.04144287109375, 0.136962890625, 0.175048828125, 0.0286865234375, 0.0013427734375, -0.09747314453125, 0.042633056640625, -0.08551
0.049896240234375, -0.144775390625, -0.018341064453125, -0.0987548828125, 0.04159545898437 5, 0.00088357925415039 06, -0.1983642578125, -0.0231323242
0.139892578125, 0.04440307617187 5, -0.011474609375, -0.1085205078125, 0.0293426513671875, 0.07342529296875, -0.086181640625, 0.05038452148437 5, 0.12
-0.1199951171875, -0.0393371582031 25, -0.0684814453125, 0.0480041503906 25, -0.0052757263183593 75, -0.02835083007812 5, 0.5732421875, 0.1456298828125,
0.012619018554687 5, -0.028884887695312 5, 0.1031494140625, 0.0220489501953125, 0.0081024169921875, 0.0010433197021484375, 0.04141235351562 5, -0.0382
-0.14208984375, 0.03350830078125, -0.0828857421875, 0.00394058227539062 5, 0.00088357925415039 06, -0.09527587890625, -0.0352783203125, -0.10076904296
0.1239013671875, 0.0751953125, -0.1123046875, -0.0784912109375, -0.09051513671875, -0.1292724609375, 0.1016845703125, 0.06155395507812 5, 0.08239746
-0.04019165039062 5, -0.099853515625, 0.037506103515625, -0.1202392578125, 0.029815673828125, 0.521484375, 0.230712890625, 0.4389648437 5, -0.0347900
 [-0.05557250976562 5, -0.1468505859375, 0.028060913085937 5, 0.06854248046875, 0.0138473510742187 5, 0.078369140625, -0.0936279296875, 0.0636596679687
-0.0541076660156 25, -0.0342407226562 5, -0.0305633544921875, -0.20947265625, -0.020965576171875, -0.0075645446777343 75, -0.2166748046875, -0.0153350
0.100524902343 75, 0.0888671875, 0.00142765045166015 62, 0.0014495849609375, -0.01004028320312 5, -0.025314331054687 5, -0.06137084960937 5, 0.0049285888
-0.1678466796875, -0.130859375, -0.03521728515625, -0.1488037109375, -0.084716796875, 0.01983642578125, -0.006519317626953125, -0.0523376464843 75, 0
-0.106689453125, -0.046478271484375, 0.0556640625, 0.0963134765625, 0.022186279296875, 0.0963134765625, 0.014541625976562 5, 0.030487060546875, -0.09
-0.00265502929687 5, -0.1419677734375, -0.07025146484375, -0.208740234375, -0.0645141601562 5, 0.034332275390625, -0.1939697265625, -0.0597229003906 25
-0.08575439453125, 0.07098388671875, 0.0638427734375, -0.05935668945312 5, -0.0426025390625, -0.05404663085937 5, -0.064453125, 0.02993774414062 5, -0.
-0.09429931640625, -0.0767822265625, -0.0899658203125, -0.1048583984375, -0.03915405273437 5, -0.11456298828125, -0.04013061523437 5, 0.50634765625, 0
 [-0.036834716796875, -0.0712890625, 0.2122802734375, -0.103515625, -0.08294677734375, 0.041900634765625, -0.093994140625, 0.0448608398437 5, -0.092
0.012725830078125, -0.08209228515625, -0.1727294921875, -0.10968017578125, 0.00202751159667968 75, -0.12353515625, -0.1361083984375, 0.0895385742187
-0.09405517578125, 0.099365234375, 0.0709228515625, -0.101318359375, -0.0396118164062 5, 0.07708740234375, 0.04653930664062 5, 0.115478515625, 0.0279
0.03887939453125, -0.12457275390625, -0.1634521484375, 0.0480041503906 25, 0.01535034179687 5, -0.01945495605468 75, 0.54638671875, 0.101684570312 5, 0.
0.0458984375, 0.0628662109375, 0.01137542724609375, -0.00953674316406 25, 0.138427734375, -0.0408935546875, -0.00063323974609375, -0.114501953125, 0.
-0.021759033203125, -0.09332275390625, -0.202392578125, 0.0232238769531 25, 0.0758056640625, -0.1300048828125, -0.10540771484375, 0.074340820312 5, 0.
0.08721923828125, 0.0780029296875, -0.01068115234375, -0.0723876953125, -0.13427734375, 0.0799560546875, 0.05859375, 0.03848266601562 5, -0.02308654
0.07684326171875, 0.029708862304687 5, -0.21484375, 0.0046005249023437 5, -0.2081298828125, -0.12017822265625, 0.5078125, 0.123291015625, 0.283203125,
 [-0.03839111328125, -0.005790710449218 75, 0.165771484375, -0.0301361083984375, 0.00537109375, -0.013084411621093 75, -0.140380859375, 0.100158691406
0.032745361328125, -0.06744384765625, 0.03314208984375, -0.0040130615234375, -0.1279296875, 0.04400634765625, 0.09149169921875, -0.08343505859375, 0
0.14501953125, -0.07220458984375, 0.10467529296875, 0.066162109375, -0.10760498046875, -0.1739501953125, 0.037261962890625, 0.0543823242187 5, 0.084
0.056488037109375, 0.030380249023437 5, -0.09967041015625, -0.099365234375, -0.11834716796875, 0.1312255859375, 0.0811767578125, 0.51513671875, 0.07
```

The consolidation process yielded a dataset containing 68 GB of raw audio, which includes 771 songs across 35 different performances. In total, this resulted in 99 hours of audio that the models could be trained on, which was sufficient. However, before the learning could begin, the raw audio needed to be encoded into arrays of latent vectors using the NumPy library. Essentially, this compresses the amplitude values of the waveforms into latent vectors that the neural network can understand. The figure above shows the raw view of one of these arrays.

3.2 Training the Musika Models



Musika utilizes the Generative Adversarial Network (GAN) architecture, which is a machine learning technique that is well suited to audio. GAN models are trained using a two-part process, where a generator network and a discriminator network are trained simultaneously in an adversarial manner. The generator takes as input a random vector and produces fake samples, while the discriminator takes as input both real and fake samples and tries to distinguish between them. During training, the discriminator tries to correctly identify the real samples from the fake ones, while the generator tries to produce samples that are indistinguishable from the real ones. This creates a feedback loop where the generator learns to produce more realistic samples as the discriminator becomes better at distinguishing them. The training process continues until the generator produces samples that are similar enough to the real samples, or until a certain number of iterations or loss threshold is reached. GAN training can be challenging, as it involves finding the right balance between the generator and discriminator networks to prevent the generator from producing easily detectable fake samples, while still allowing it to generate diverse and realistic samples.

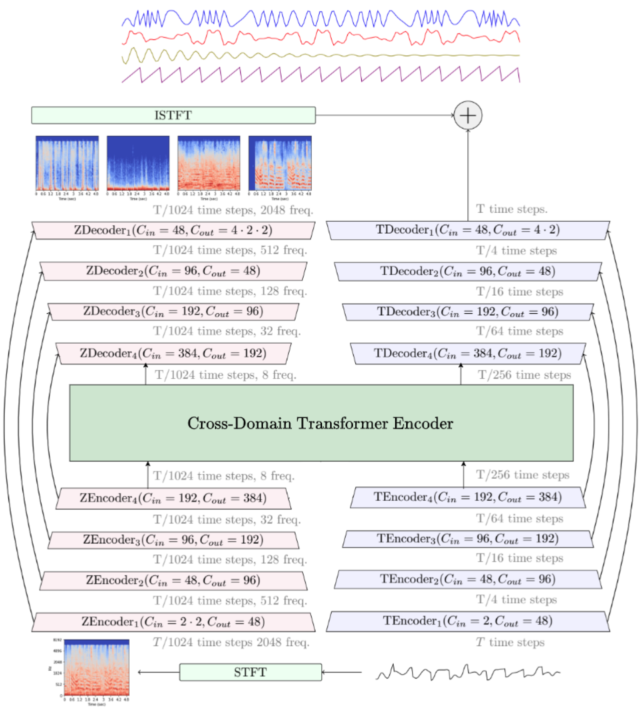


Multiple models were trained using the Musika library, with varying datasets and training parameters. The main model was trained on the previously mentioned dataset of Grateful Dead live recordings, spanning from 1968 to 1990, with 99 hours of audio in total. This model was trained from scratch (not continuing from an existing checkpoint), for 500k iterations at a low learning rate of 0.0005. The results are consistent with the example models provided in the Musika library. Pictured above is a graph of the discriminator loss values during training, which represent how closely the generated audio matches the provided dataset, with lower being more closely.

Additional models were finetuned from the Musika base model on specific albums, which contain about 2-3 hours of audio each. These albums include Cornell '77, Veneta '72, Winterland '73, and Nassau '90. Each of the finetuned models was trained for only 100k iterations, at a variable learning rate that changed based on the loss values from the discriminator. These models also yielded results in line with the example models in the Musika library, and started producing listenable audio much faster than the main model.

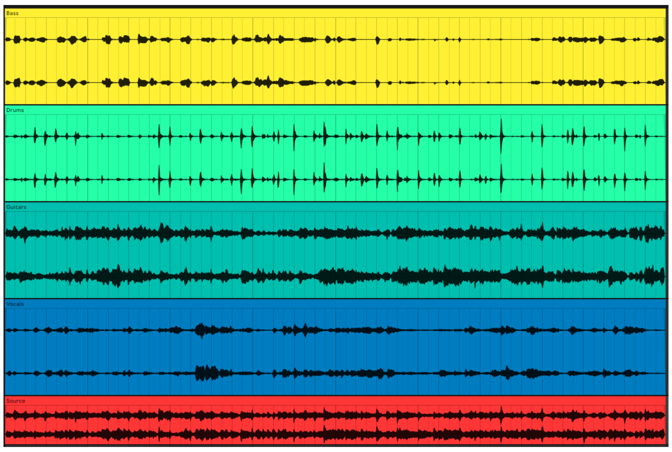
3.3 Demucs Source Separation

Separating individual instruments is a surprisingly complex problem in data analysis. Once individual tracks have been combined into a single mix, it is almost impossible for a computer to perfectly separate them out again, because some data is lost during the mixing process. A common analogy for this process is if you were to make a smoothie, give it to someone else who did not watch you make it, and then ask them what fruits they think you used. This is simple for one or two ingredients, but more than that would be very difficult.



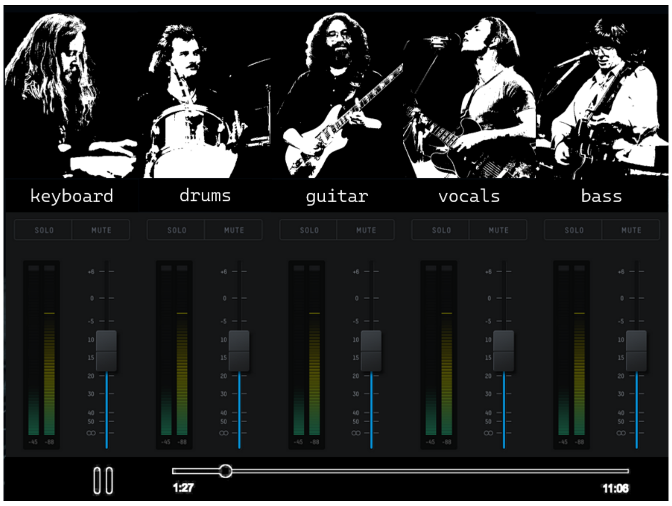
ISTFT
T/1024 time steps, 2048 freq.
T time steps.
$\text{ZDecoder}_1(C_{in} = 48, C_{out} = 4 \cdot 2 \cdot 2)$
$\text{TDecoder}_1(C_{in} = 48, C_{out} = 4 \cdot 2)$
T/1024 time steps, 512 freq.
T/4 time steps
$\text{ZDecoder}_2(C_{in} = 96, C_{out} = 48)$
$\text{TDecoder}_2(C_{in} = 96, C_{out} = 48)$
T/1024 time steps, 128 freq.
T/16 time steps
$\text{ZDecoder}_3(C_{in} = 192, C_{out} = 96)$
$\text{TDecoder}_3(C_{in} = 192, C_{out} = 96)$
T/1024 time steps, 32 freq.
T/64 time steps
$\text{ZDecoder}_4(C_{in} = 384, C_{out} = 192)$
$\text{TDecoder}_4(C_{in} = 384, C_{out} = 192)$
T/1024 time steps, 8 freq.
T/256 time steps
Cross-Domain Transformer Encoder
T/1024 time steps, 8 freq.
T/256 time steps
$\text{ZEncoder}_4(C_{in} = 192, C_{out} = 384)$
$\text{TEncoder}_4(C_{in} = 192, C_{out} = 384)$
T/1024 time steps, 32 freq.
T/64 time steps
$\text{ZEncoder}_3(C_{in} = 96, C_{out} = 192)$
$\text{TEncoder}_3(C_{in} = 96, C_{out} = 192)$
T/1024 time steps, 128 freq.
T/16 time steps
$\text{ZEncoder}_2(C_{in} = 48, C_{out} = 96)$
$\text{TEncoder}_2(C_{in} = 48, C_{out} = 96)$
T/1024 time steps, 512 freq.
T/4 time steps
$\text{ZEncoder}_1(C_{in} = 2 \cdot 2, C_{out} = 48)$
$\text{TEncoder}_1(C_{in} = 2, C_{out} = 48)$
T/1024 time steps 2048 freq.
T time steps
STFT

Demucs is a deep learning-based method for music source separation that can extract individual audio tracks from a mixed audio signal. It uses a neural network architecture based on a type of recurrent neural network called a Transformer, which has been shown to be effective for processing sequential data such as audio signals. The network is trained to predict the magnitude and phase of the individual sources from the mixed signal. Once the network is trained, it can be used to extract individual sources from new mixed signals in real-time. Demucs works by iteratively refining the predicted sources by passing them through the network multiple times. This allows the network to gradually separate the individual sources from the mixed signal. Demucs has been shown to outperform other state-of-the-art methods for music source separation, and can be used for a variety of applications, including music transcription, remixing, and analysis. Pictured above is a diagram of the Demucs architecture.



Originally, Spleeter was used in order to process the generated audio from the Musika models. However, the results contained a large quantity of artifacts. The Spleeter model struggled with differentiating between vocals and accompaniment, due to the less than ideal quality of the generated audio. Much better results were seen with Demucs, which is used in the final web application. Overall, Demucs is able to separate the bass, vocals, and percussion at much higher fidelity than Spleeter. Pictured above is the resultant stems from running Demucs on a track generated by the main Musika model.

3.4 DeepDeadNet



Putting all of these processes together results in DeepDeadNet, an application that can generate music in the style of the Grateful Dead in real-time. Users can select from the main model, or show-specific models, and generate audio of a specified length in that style. They can then adjust the volume of each instrument to their liking. Since the two models that serve as the basis for application (Musika and Demucs) have very low computational requirements, this application can run on a server with minimal costs. Most of the computational resources are allocated to the training process, which has already been completed, the results of which are stored in the models.

4. FUTURE WORK

DeepDeadNet was built with scalability in mind, and can be repurposed for any audio dataset. As music generation and processing models improve over time, they can be seamlessly updated in application, and it will become more useable. Additonally, since the processing occurs in real-time, an endless Grateful Dead radio could be created that leverages DeepDeadNet, allowing users to enjoy new music in the style of the Grateful Dead forever.

It is also worth mentioning that the audio produced by the Musika models is not high-fidelity, and it is rare to find a result that is truly listenable. However, the results could be sampled by producers for use in a larger project, as the spectral quality of the generated audio is quite unique. Additionally, due to the nature of the encoders used for Musika, the audio must be compressed heavily, which results in inevitable low fidelity. However, a trainable encoder is currently being developed, which will help alleviate this issue.

5. CONCLUSIONS

This project showed me how difficult it can be to train a neural network. Although it's tempting to allow the training simply run its course, manual adjustments at critical moments are often necessary to prevent the network from overfitting or generally spiralling out of control. There is also a certain element of randomness to training a neural network, which requires immense patience to overcome. On the other hand, this project served as a catalyst for my exploration of various neural network architectures, including those unrelated to audio. More specifically, I've been exploring Stable Diffusion for artwork generation (the basis for Riffusion), as well as the application of ChatGPT's API in video games for realistic character interactions. Moving forward, I will continue to seek opportunities in the world of AI, and am very excited to see how it will grow.

Overall, this project aimed to explore the current potential of music generation and enhancement by artificial intelligence. DeepDeadNet proves that current machine learning models are able to distinguish between not only different styles of music, but also different instruments, all in real-time. The AI revolution has only just begun, yielding amazing results in art and text generation, and it seems that music is following suit. But at this stage, nothing beats the real thing.

6. ACKNOWLEDGMENTS

Professor Victor Zappi
Professor Hubert Ho
Professor Mike Frengel
Professor Daniel Godfrey
Professor Brian Robison
Professor Ronald Smith
Kate, Ron, and Julie

7. REFERENCES

[1] P. Dhariwal, C. Jun, C. Payne, J. W. Kim, A. Radford, and I. Sutskever. Jukebox: A generative model for music, 2020.

[2] C. Hawthorne, E. Elsen, J. Song, D. Roberts, I. Simon, C. Raffel, J. Engel, and S. Oore. Neural generation of multi-track music with hierarchical variational autoencoders. In *Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

[3] M. Pasini and J. Schlüter. Musika! fast infinite waveform music generation, 2022.

[4] Z. Rasheed, S. Oore, J. Arsenault, A. Bergeron, M. McKinney, and G. Tzanetakis. Riffusion: Creative ai for music generation. In *Proceedings of the 21st International Society for Music Information Retrieval Conference (ISMIR)*, 2020.