MEF UNIVERSITY

MUSIC GENERATION USING DEEP LEARNING TECHNIQUES

Capstone Project

Kutay Akalın

İSTANBUL, 2021

MEF UNIVERSITY

MUSIC GENERATION USING DEEP LEARNING TECHNIQUES

Capstone Project

Kutay Akalın

Advisor: Asst. Prof. Dr. Evren Güney

İSTANBUL, 2021

MEF UNIVERSITY

Name of the project: MUSIC GENERATION USING DEEP LEARNING TECHNIQUES
Name/Last Name of the Student: Kutay Akalın
Date of Thesis Defense: 25/01/2021

I hereby state that the graduation project prepared by Kutay Akalın has been completed under my supervision. I accept this work as a "Graduation Project".

25/01/2021
Asst. Prof. Dr Evren Güney

I hereby state that I have examined this graduation project by Kutay Akalın which is accepted by his supervisor. This work is acceptable as a graduation project and the student is eligible to take the graduation project examination.

25/01/2021

Prof. Dr. Özgür Özlük

Director
of
Big Data Analytics Program

We hereby state that we have held the graduation examination of _____ and agree that the student has satisfied all requirements.

THE EXAMINATION COMMITTEE

Committee Member	Signature
1. Asst. Prof. Evren Güney	……..
2. ……..	……..

Academic Honesty Pledge

I promise not to collaborate with anyone, not to seek or accept any outside help, and not to give any help to others.

I understand that all resources in print or on the web must be explicitly cited.

In keeping with MEF University's ideals, I pledge that this work is my own and that I have neither given nor received inappropriate assistance in preparing it.

Name	Date	Signature
Kutay Akalın	25/01/2021	

EXECUTIVE SUMMARY

MUSIC GENERATION USING DEEP LEARNING TECHNIQUES

Kutay Akalın

Advisor: Asst. Prof. Dr. Evren Güney

JANUARY, 2021, 24 pages

This project aims to generate songs using the Jukebox model and its architecture. Jukebox's Vector Quantized Variational AutoEncoder (VQ-VAE) architecture is state-of-the-art deep generative model used for music generation and gives an outstanding result. For this purpose, different Elvis Presley songs were analyzed in audio domain using various Music Information Retrieval (MIR) methods. The top level of the Jukebox model was retrained with these songs in order to increase the quality of the songs that will be produced in the style of Elvis Presley. After that, 3 new samples were generated using the first six seconds of Elvis Presley - Jailhouse Rock as the input signal. At the end, these new songs were analyzed and compared.

Key Words: Audio Signal Processing, MIR, Music Generation, Deep Learning, Generative Models, Sound Analysis, Vector Quantized Variational AutoEncoder

ÖZET

DERİN ÖĞRENME TEKNİKLERİ İLE MÜZİK ÜRETİMİ

Kutay Akalın

Proje Danışmanı: Dr. Öğretim Üyesi Evren Güney

OCAK, 2021,24 sayfa

Bu projede, Jukebox modelini ve mimarisini kullanarak şarkılar üretmek amaçlanmıştır. Jukebox'ın Vektör Nicemlenmiş Varyasyonel Otokodlayıcılar (VQ-VAE) mimarisi, müzik üretimi için kullanılan en güncel derin öğrenme modellerinden biridir ve oldukça başarılı sonuçlar vermektedir. Bu amaç için, çeşitli Müzik Bilgi Erişimi (MIR) yöntemlerini kullanarak dalga formatındaki farklı Elvis Presley şarkılarını analiz edildi. Elvis Presley tarzında üretilecek şarkıların kalitesini arttırmak için Jukebox modelinin en üst katmanı bu şarkılar ile yeniden eğitildi. Bundan sonra, giriş sinyali olarak Elvis Presley - Jailhouse Rock parçasının ilk altı saniyesi kullanılarak 3 yeni şarkı oluşturuldu. Üretilen bu yeni şarkılar analiz edildi ve karşılaştırıldı.

Anahtar Kelimeler: Ses Sinyali İşleme, MIR, Müzik Üretimi, Derin Öğrenme, Generatif Modeller, Ses Analizi, Vektör Nicemlenmiş Varyasyonel Otokodlayıcılar

TABLE OF CONTENTS

TABLE OF FIGURES

1. INTRODUCTION

Music Generation is one of most interesting cases in the Art & AI field. There are various approaches in this era with using different Deep Learning Methodologies, but it still keeps its mystery.

1.1. Musical Representation

Commonly, there are two main choices of music representation used in these approaches: Audio and Symbolic. This also corresponds to continuous and discrete variables relatively. Selecting representation also changes types of techniques for possible processing and transformation on music. Thus, selecting representation changes all methodology and architecture in music generation approach. Most of the researchers selected the symbolic representation. This is because in the symbolic representation, feature dimension is much lower than the audio domain. Analysing and processing symbolic data requires less computational cost, thus the solution of the problem becomes easier and faster. However, due to lower-dimensional feature space, it is considered that using symbolic representation causes loss of information that can be extracted from music and generating songs with less information can reduce the quality of song which will generated. Examples of audio and symbolic representation shown in Figure-1 and Figure-2 respectively.

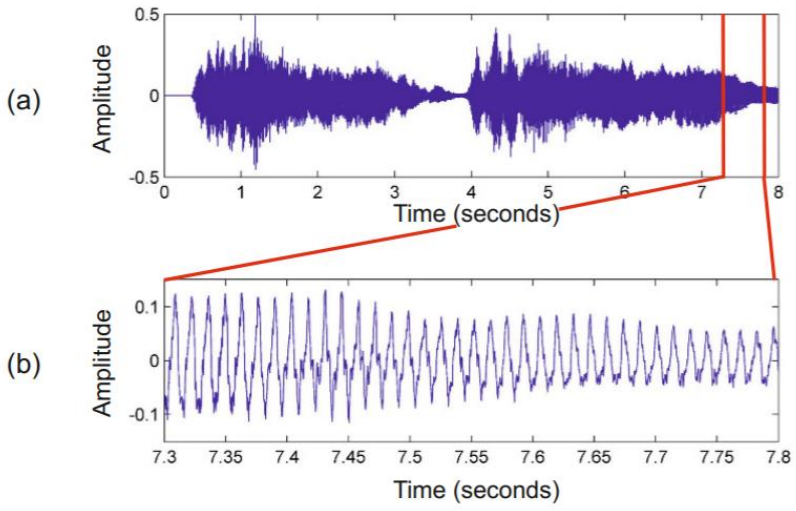

Figure 1: Example of the Waveform (Audio) Representation (Müller, 2016)

```
2,   96, Note_on,   0, 60, 90
2, 192, Note_off, 0, 60,  0
2, 192, Note_on,   0, 62, 90
2, 288, Note_off, 0, 62,  0
2, 288, Note_on,   0, 64, 90
2, 384, Note_off, 0, 64,  0
```

Figure 2: Example of the MIDI (Symbolic) Representation (Briot et al., 2020)

Most commonly used deep learning techniques for music generation are LSTM, RNNs and GANs in the literature.

1.2. Literature Review

Kotecha and Young (2018) created a LSTM neural network architecture that analyses the musical structure and generates new polyphonic music aligned with musical rules. This network is able to recall past information to project in the future. MIDI (musical instrument digital interface) audio files were used because of 2 reasons: MIDI files can contain the metadata like timestamps and MIDI is a common digital representation which allows access to freely and widely available data. The model uses a two-layered LSTM RNN architecture with recurrent connections along the note axis. One LSTM is positioned on the time axis and another LSTM is positioned on the note axis. It is called "bi-axial" configuration. Kotecha and Young (2018) explained the model training phase as "The network is trained to model the conditional probability distribution of the notes played in a given time step, conditioned on the notes in previous time steps. The output of the network can be read as at time step t, the probability of playing a note at time step t, conditioned on prior note choices. Therefore, the model is maximizing the log-likelihood of each training sequence under the conditional distribution." Finally, Kotecha and Young (2018) found that training time is highly correlated with quantitative results. Increase in the variability and complexity of music data affects the train time and decreases the probability of generating decent music (Kotecha & Young, 2018).

In the work of Engel et al. (2017), generative models implemented in the audio processing. Firstly, they described the autoencoder model of the WaveNet-style that states an autoregressive decoder on temporal codes which learned from the raw audio waveform. After that, Engel et al. (2017) improved the quantitative performance of the WaveNet autoencoder over a well-tuned spectral autoencoder baseline using NSynth dataset. WaveNet

auto encoder that learns temporal hidden codes to capture longer term structure effectively without external conditioning and Nsynth dataset is a large-scale dataset for exploring neural audio synthesis of musical notes. Engel et al. (2017) modified the architecture of the WaveNet model to take audio waveform as an input, produce embeddings, shift and feed into decoder to reproduce waveform. To get the best results, Engel et al. (2017) used a large FFT size (1024) relative to the hop size (256) and ran the algorithm for 1000 iterations. The result shows that WaveNet reconstructions have much more quality than baseline. However, pitch quality is not good as original audio. Thus, they classified the pitches using linear pitch classifier and found that pitches are most frequently mistaken for those one octave apart. Also, due to the memory constraints, the trained model is unable to fully capture global context (Engel et al., 2017).

Engel et al. (2019) introduce GANSynth, adversarial network audio synthesizer. As mentioned before, Autoregressive models like WaveNet have slow iteration sampling and lack global latent structure. On the contrary, GANs can handle these lacks but they are bad in generating locally coherent audio waveforms. Engel et al. (2019) overcome this problem by modelling log magnitudes and instantaneous frequencies with sufficient frequency resolution in the spectral domain. They used NSynth dataset on this study and used STFT with 256 stride and 1024 frame size, which resulted 75% frame overlap and 513 frequency bins. 3 different variants used in the model: phase as phase angle, IF as instantaneous frequency, IF-Mel as instantaneous frequencies to a Mel frequency scale without dimensional reduction. Engel et al. (2019) selected the baseline as WaveGAN to compare results and defined the evaluation metrics as Human Evaluation, Number of Statistically Different Bins (NDB), Inception Score (IS), Pitch Accuracy (PA) and Pitch Entropy (PE) and Fréchet Inception Distance (FID). The results show that all of the high-resolution models generate instances classified with similar accuracy to the real data but IF-Mel+ H model has the highest score which represents better perceptual quality to participants and best overall quantitative metric results (Engel et al., 2019).

2. PROJECT DEFINITION

In this project, my goal is to generate music using tracks in the audio representation. I will evaluate these generated songs using human evaluation metrics. As I mentioned before, using frequency domain audio allows us to extract much more information from audios and increase the quality of the generated songs. For this purpose, firstly, understanding the Music Information Retrieval (MIR) methods is essential for analysing and processing waveform sounds.

2.1. Sound Analysis

To analyse signals like sounds in digital platforms, understanding basic signal processing topics are essential. In this subsection, main audio signal processing steps, MIR methods and musical features will be explained.

2.1.1. Fourier Transform

Comparing the signal with sinusoids of varying frequencies is the core concept of Fourier analysis (measured in Hz). It is possible to think of any other sinusoid or pure note as an oscillation. For representation, we use frequency parameter $\omega \in R$ with a magnitude coefficient $d\omega \in R \geq 0$ (Müller, 2016).

Fourier Transform is an effective method that can transform any time domain signal, periodic or non-periodic, into a function of frequency domain. We can also reverse this transaction using inverse Fourier transform. The formula of Fourier transform given below:

$$F(\omega) = \int_{-\infty}^{+\infty} f(t) e^{-i\omega t} dt \quad (1)$$

In the Equation 1, $F(w)$ denotes the Fourier Transform of the given angular frequency (w) at time t. Angular frequency can be calculated by multiplying 2, π and frequency (f) of the continuous signal $f(t)$. Fourier Transform of the note C4 (261.6 Hz) is shown in the Figure-3.

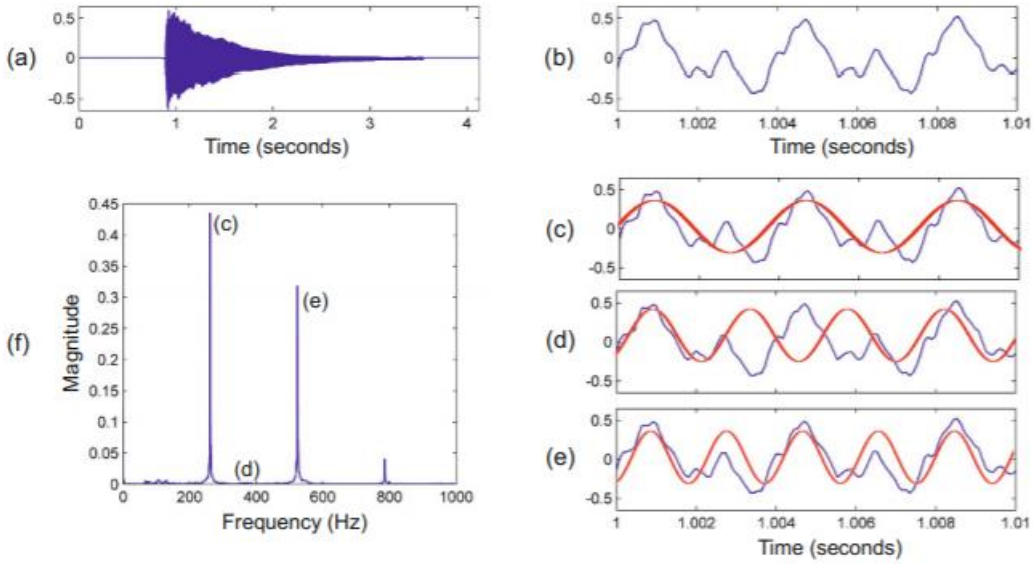

Figure 3: (a) Waveform of a note C4 (261.6 Hz) played on a piano. (b) Zoom into a 10-ms section starting at time position t = 1 sec. (c–e) Comparison of the waveform with sinusoids of various frequencies ω. (f) Magnitude coefficients dω in dependence on the frequency ω (Müller, 2016).

2.1.2. Discrete Fourier Transform (DFT)

Only a limited number of parameters can be stored and processed using digital platform. Therefore, for interpretation, analog signals must be transformed into discrete form. The Discrete Fourier Transform or DFT is the transformation that deals with a finite discrete-time signal and a finite or discrete number of frequencies (Müller, 2016). DFT of a signal is shown in Figure-4.

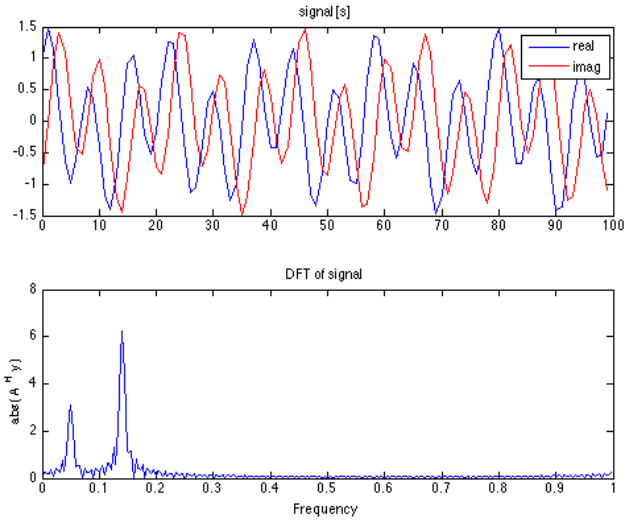

Figure 4: Example of DFT

2.1.3. Short Time Fourier Transform (STFT)

According to Kehtarnavaz (2008), Short-time Fourier transform (STFT) is a "sequence of Fourier transforms of a windowed signal. STFT provides the time-localized frequency information for situations in which frequency components of a signal vary over time, whereas the standard Fourier transform provides the frequency information averaged over the entire signal time interval." STFT is reversible, so, the main signal can be recaptured using the Inverse STFT function. Transformation of the different windows are shown in the Figure-5.

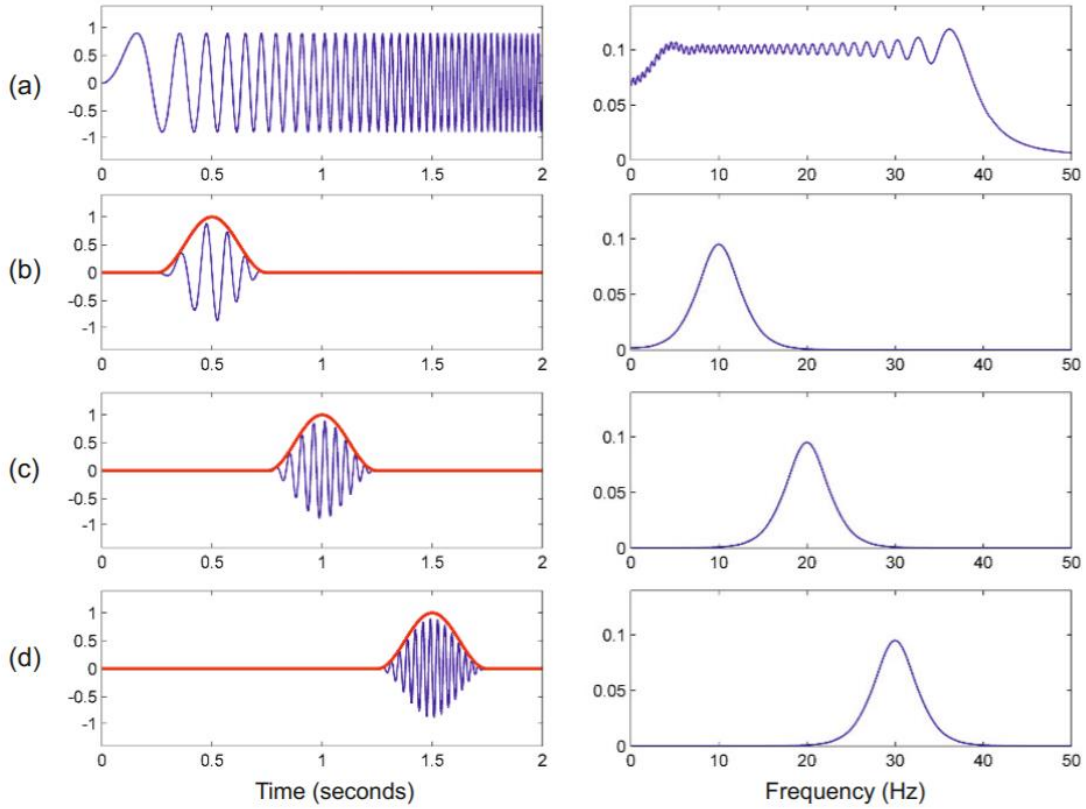

Figure 5: Chirp signal and windowed versions along with their magnitude Fourier transforms. (a) Original signal. (b) Window centered at t = 0.5. (c) Window centered at t = 1.0. (d) Window centered at t = 1.5 (Müller, 2016).

2.1.4. Mel-Frequency Cepstral Coefficients (MFCCs)

Mel-Frequency Cepstral Coefficients (MFCCs), initially designed for automatic speech recognition, are also used in the sense of timbre-based structure analysis (Müller, 2016). People can hear lower frequencies better than high frequencies. MFCC converts the conventional frequency to Mel Scale. For this reason, the difference in the human perception

of sound is taken into account. In the audio analysis, generally, 12 or 13 Mel frequency coefficients are considered as features. Mel scale can be calculated using the given formula below:

$$\text{Mel}\,(f) = 2595 * \log\left(1 + \frac{f}{700}\right) \quad (2)$$

2.2. Model & Architecture

There are many different algorithms in the literature that analyse certain aspects of music and generate only that particular aspect of the music. Huang et al. (2019) tried to generate piano sheet music using NADE with the use of blocked-Gibbs sampling. Blaauw and Bonada (2017) synthesised a singer voice based on modified version of WaveNet architecture using parametric vocoder. Wu et al. (2019) generated symbolic music using 3 different LTSM model as subnetworks to capture long-term structure. Also, some generative models were applied raw audio domain to produce piano pieces. Oord et al. (2016) introduced a deep learning model for generating raw audio waveforms called WaveNet, which is used as the base model in most research. Yamamoto et al. (2020) proposed Parallel WaveGAN, non-autoregressive WaveNet model that is equipped to capture the time-frequency distribution of the realistic speech waveform by jointly optimizing multi-resolution spectrograph and adversarial loss functions. Vasquez & Lewis (2019) proposed MelNet, generative model for spectrogram domain.

However, all these models analyse the certain aspect of music like instrument, timbre, human voice and melody. Using songs in audio domain allows the extract a lot information from the songs but also make the problem challenging due to the increase in the feature dimension.

2.2.1. Model

Dhariwal et al. (2020) proposed the Jukebox model that generates music with lyrics in the raw audio domain. Generative models are good at capturing the salient aspects of the data and generating new instances indistinguishable from the actual data. Dhariwal et al. (2020) demonstrated that they can create a single device capable of producing diverse high-fidelity music in the raw audio domain, with long-range coherence lasting several minutes.

This model is actual state-of-the-art in the music generative models. Using raw audio domain enables the capturing all information which can be extracted from the music and merging all aspects of music in a single model is an actual achievement for music generation concept. Thus, I selected the Jukebox model for doing experiments and generating new music using its architecture.

2.2.2. Architecture

Dhariwal et al. (2020) used a hierarchical Vector Quantized Variational AutoEncoder (VQ-VAE) architecture which proposed by Ravazi et al. (2019). Ravazi et al. (2019) used VQ-VAE model for image generation. Dhariwal et al. (2020) revised this model to make it compatible with the audio application. VQ-VAE includes an encoder that maps observations to a series of separate latent variables, and a decoder that reconstructs the observations from these separate variables. These encoder and decoder use same codebook (Ravazi et al., 2019).

This architecture revised for the music generation to compress audio into a separate space, with a loss function built to preserve the greatest amount of musical data while increasing compression degree. After the compressed space created, Dhariwal et al. (2020) used autoregressive Sparse Transformer (Child et al., 2019) trained using maximum-likelihood estimation. Mainly, Dhariwal et al. (2020) used 3 level abstraction and they use residual networks at each level consisting of non-causal 1-D dilated convolutions in the WaveNet style, interleaved with downsampling and upsampling of 1-D convolutions. After that, they used autoregressive Sparse Transformer to train over this compressed space with maximum-likelihood estimation and also trained autoregressive upsamplers to reconstruct the missing information at each compression stage (Dhariwal et al., 2020). Architecture of VQ-VAE model is shown in Figure-6.

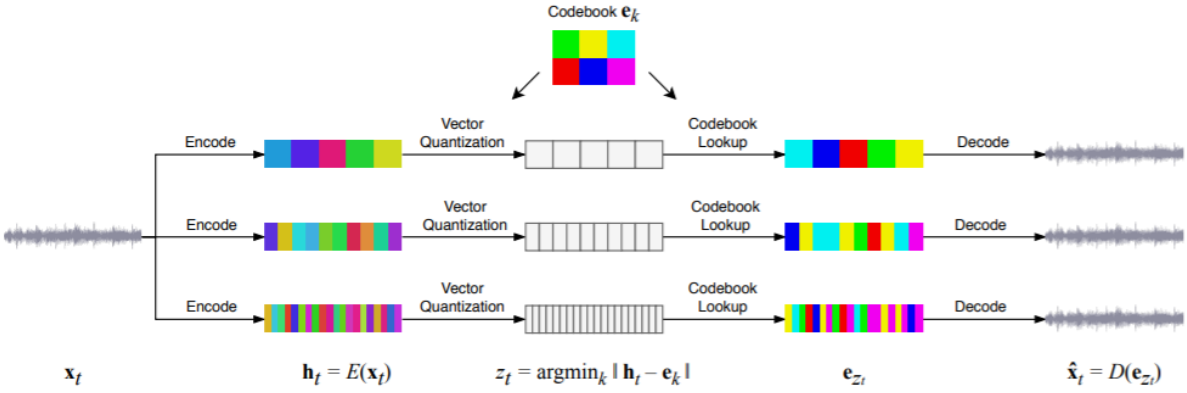

Figure 6: Architecture of VQ-VAE (Dhariwal et al., 2020)

3. ABOUT THE DATA

I obtained 20 different Elvis Presley songs to retrain pre-trained Jukebox VQ-VAE architecture. This pre-trained model is already trained by wide range of music genres and artist styles. Thus, retraining the just top level of the VQ-VAE model will be enough to increase quality of generated songs in the style of Elvis Presley.

3.1. Feature Extraction

As I mentioned earlier, in the beginning of sound analysis, features of the music are needed to be extracted. For this purpose, I used a Python Package called Librosa. Librosa is designed for signal processing and analysing with creating Music Information Retrieval (MIR) framework. As an example, the analysis and features of the Elvis Presley - Jailhouse Rock are displayed in this section. Figure-7 and Figure-8 display the waveform of the song.

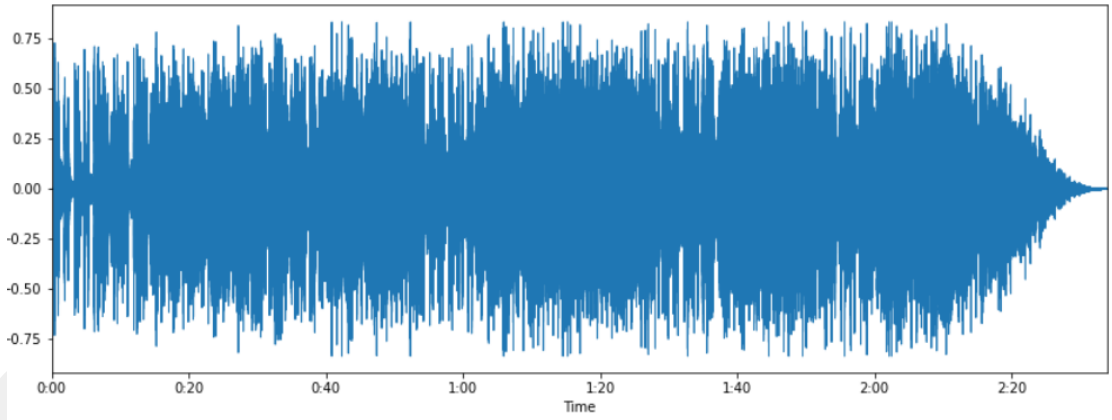

Figure 7: Waveform of the song

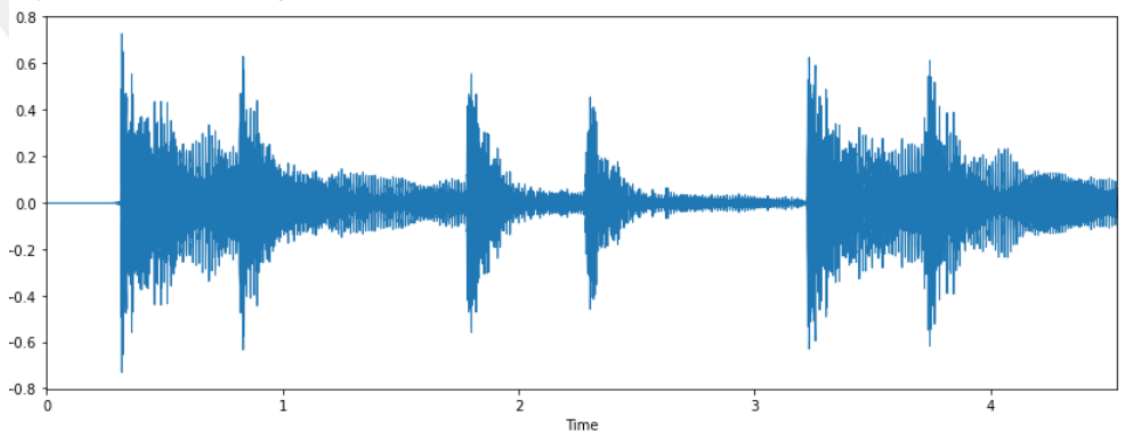

Figure 8: Waveform at the beginning of the song

A spectrogram is a graphical representation of the signal intensity or "loudness" of a signal over time at different frequencies in a given waveform. We can see different energy levels for different frequencies. Also, we can analyse the change in energy level over time (Chauhan, 2020). Spectrogram of the Elvis Presley – Jailhouse Rock song is shown in the Figure-9 and Figure-10.

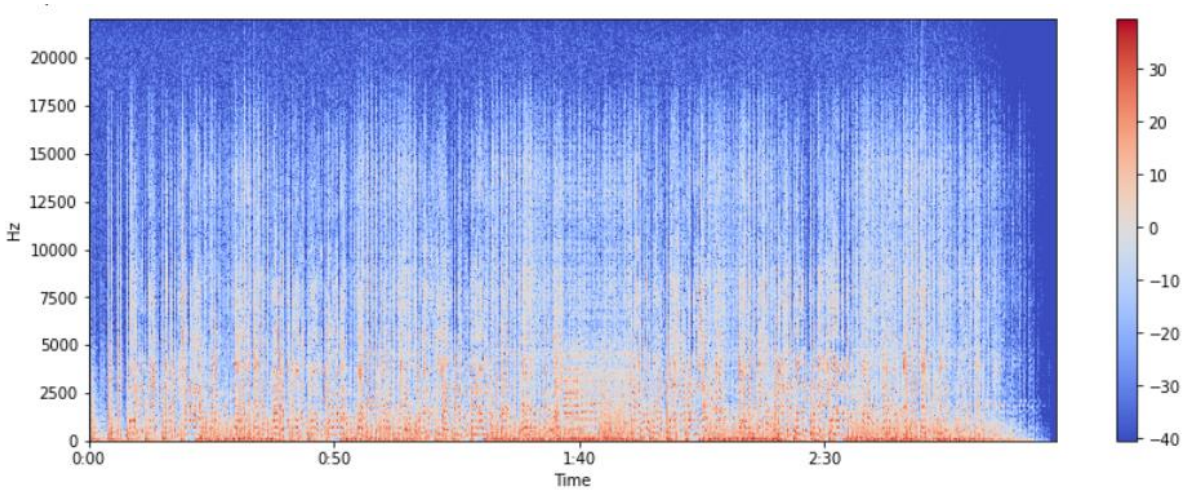

Figure 9: Spectrogram of the song

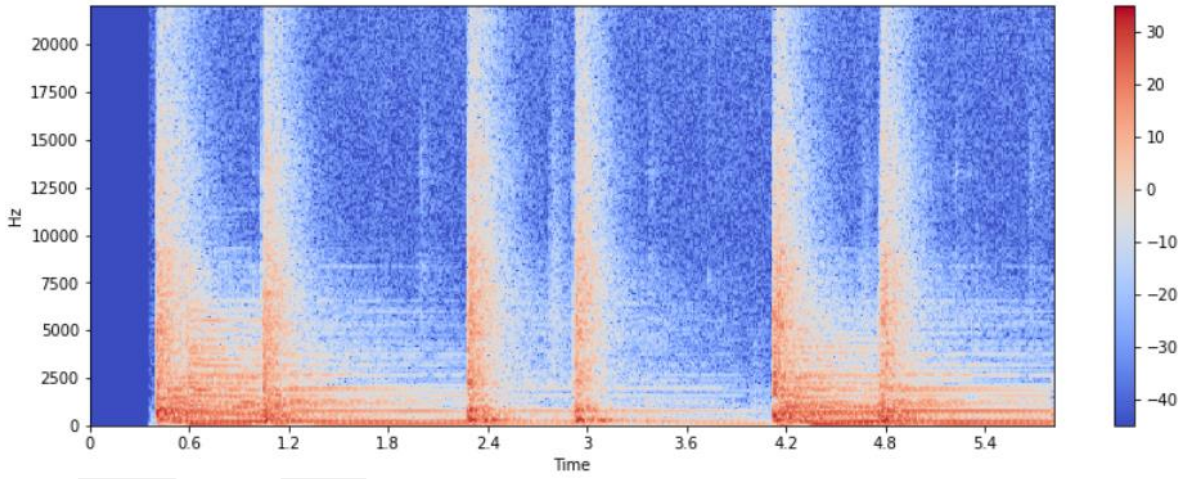

Figure 10: Spectrogram at the beginning of the song

The Mel scale is defined as perceived frequency of a pure sound by humans. Humans are more sensitive to understand small changes in melody at low frequencies than high frequencies. So, Mel scale converts the frequencies to match with human hearing characteristics. Mel frequency cepstral coefficients (MFCCs) models the characteristics of the human voice (Chauhan, 2020). MFCC values of the song is shown in the Figure-11 and Figure-12.

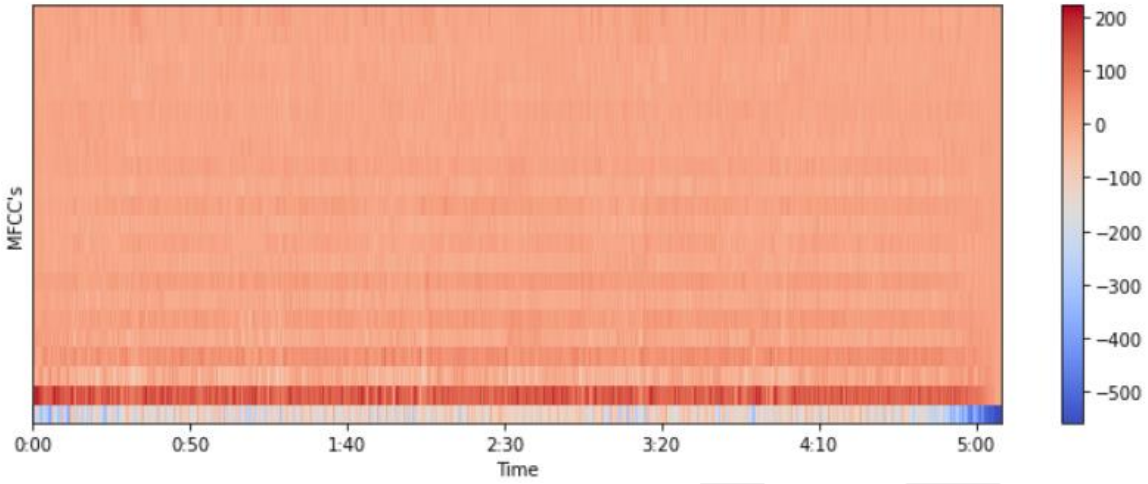

Figure 11: MFCC Spectrogram of the song

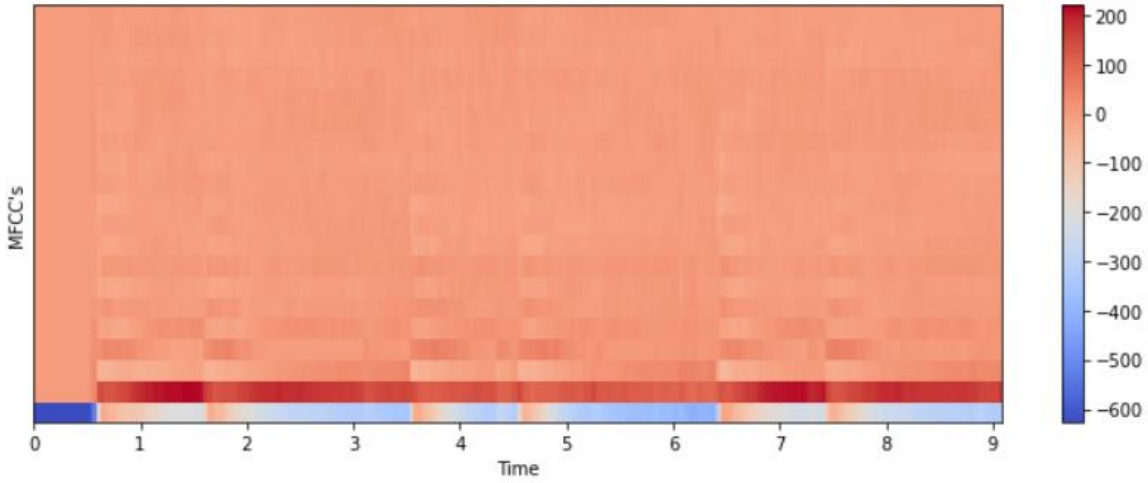

Figure 12: MFCC at the beginning of the song

Chromagram displays the pitch classes at the given time t with the heatmap representation. Chromagram of the Elvis Presley - Jailhouse Rock song is shown in the Figure-13 and Figure-14.

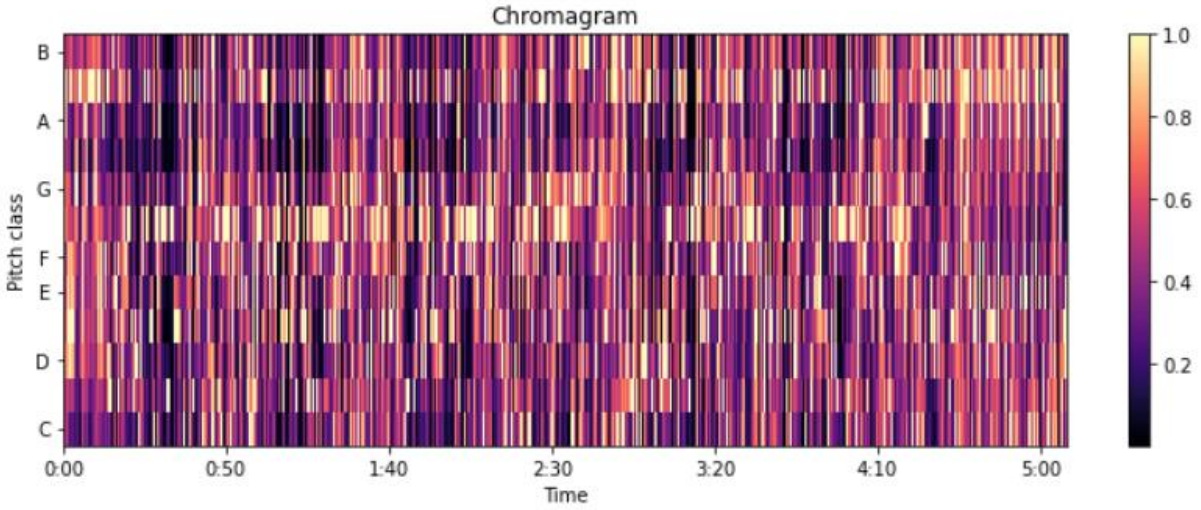

Figure 13: Chromagram of the song

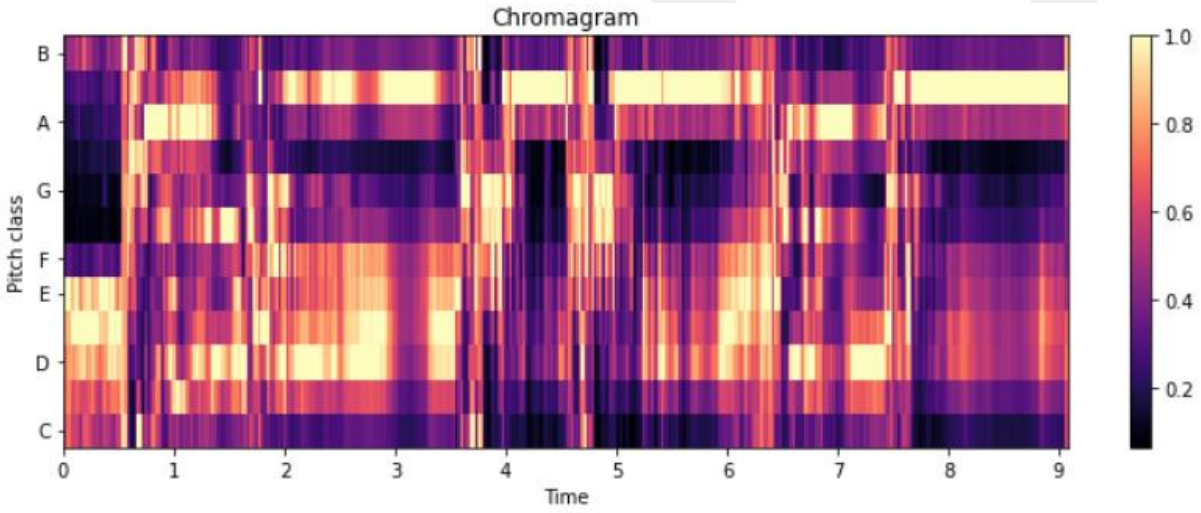

Figure 14: Chromagram at the beginning of the song

4. METHODOLOGY

To generate music using Jukebox architecture, multiple powerful GPU's are needed. For this purpose, I decided to use Google Colab for both the retraining and sampling phase. Google Colab is a free python environment for developing projects and also, it supports powerful GPUs and TPUs.

Firstly, all tracks in our data are uploaded to Google Drive and analysed. Analysis result of one song is defined in the About the Data Section.

4.1. Retraining the Top Level of VQ-VAE Architecture

Jukebox's VQ-VAE architecture is already trained with 1.2 million songs paired with metadata. Dhariwal et al (2020) trained this VQ-VAE architecture with 32bit and 44.1 kHz raw audio. The metadata contains artist, album, genre, and year of the release for each song. Besides that, they increase the data by arbitrarily alternating the right and left channels to create mono channel audio.

After this training is done, priors need to be learned to generate samples over the compressed space. This prior is a problem of autoregressive modelling in the discrete token space provided by the VQ-VAE architecture. To solve this problem, Dhariwal et al. (2020) proposed Scalable Transformer, a simplified version of Transformers which is easier to implement and scale. Besides that, for the upsamplers, they included conditioning data from the upper level codes to the autoregressive transformers. For this, they used a deep residual WaveNet followed by an upsampling of phase convolution and a layer norm, and applied the output to the embedding of the current stage as additional positional information (Dhariwal et al., 2020).

In the first model, Dhariwal et al. (2020) provided artist and genre labels to make the generative model more controllable. This conditioning reduces the entropy of the audio generation. That means, the model can generate songs in better quality with specifying artist and genre. In addition, at training time, they applied a timing signal to each section. The overall length of the piece, the start time of the particular sample and the fraction of the song that has elapsed are part of this signal. This makes it possible for the model to learn audio

patterns that rely on the overall structure, such as voice or instrumental introductions and cheering at the end of a piece (Dhariwal et al., 2020).

Due to the limitation on computational power, I tried to retrain top-level prior of this model using 20 different Elvis Presley songs. This model is already trained by some Elvis Presley tracks. But my aim is to retrain this model to analyse and understand Elvis Presley style better and generate high quality songs in this style. For this purpose, I used 'train.py' file with following arguments:

- --hps = vqvae, small_prior, all_fp16, cpu_ema
- --name = pretrained_vqvae_small_prior
- --sample_length = 1048576
- --bs = 4
- --audio_files_dir = {Elvis_Presley_Songs}
- --labels=False
- --levels=3
- --level=2
- --weight_decay=0.01
- --save_iters=1000

Training phase took more than 10 hours, although I used a relatively small dataset. After the training is finished, I added this new model into the 'make_models.py' file to use in sampling phase.

4.2. Sampling using New Model

To generate new samples using trained new model, I used Google Colab with GPU runtime. Primarily, Jukebox model gives us 3 main options for selecting the sampling mode: Ancestral, Windowed and Primed sampling. In the Ancestral mode, model takes s genre, artist, timing, and lyrics condition to generate random samples. In the Windowed mode, sample continuations again and again in each step, using the sense of overlapping windows of previous codes. In the Primed sampling, by translating it to VQ-VAE codes and sampling the corresponding codes at each step, the model will produce continuations of a given audio signal (Dhariwal et al., 2020).

After necessary libraries are imported, I chose the Primed sampling to generate samples. For this purpose, I selected the Elvis Presley- Jailhouse Rock track as the audio signal. I defined the `prompt_length_in_second` parameter equal to the 6. That means, model will pick up from 6 seconds of given audio signal and start generation. Then, I defined the model and its hyperparameters as follow:

- -- sr = 44100
- -- n_samples = 3
- -- chunk_size = 16
- -- max_batch_size = 3
- -- levels = 3
- -- hop_fraction = [.5,.5,.125]

We can increase the chunk size, maximum batch size and sample number. However, due to the limited access of GPU memory, Google Colab does not allow us to increase these hyperparameters and generate more samples with more batch size.

Total sample length defined as 50. That means, our model generates 50 seconds of musical sample starting with 6 seconds of given audio signal. After these parameters are defined, we need to specify artist and genre metadata. In this project, lyric conditioning is not used. Thus, our model generates samples without any lyrics. Artist metadata defined as "Elvis Presley" and for the first experiment, Genre metadata defined as "Pop Rock". Finally, sampling temperature is defined as 0.98. This value makes the Jukebox model generate more random. Increase in the sampling temperature limits this randomness.

After all these parameters are defined, we can start to generate samples. Firstly, top level of (referred as level 2) samples are generated. Then, first upsampling (referred as level 1) and second upsampling (referred as level 0) are followed. In each step, raw audios are decoded and saved into the corresponding level folder. Our final, fully completed samples are in the level 0 folder after all upsampling is done. This sampling takes more than 13 hours using Google Colab GPU's.

5. RESULTS

After sampling for 2 levels and upsampling for the final level is done, 3 different songs are generated based on the first 6 seconds of Elvis Presley- Jailhouse Rock. Due to time and computer power limitation, I decided to generate songs in less than one minute. Thus, the total length of new generated tracks becomes 50 seconds.

As I mentioned before, we have 3 different generated levels for each 3 audios. Level 1 and Level 2 is defined as the sampled version of new sounds. Level 0 is defined for the upsampled version of these audios and it represents the final output. Spectrogram of the generated one audio (called item_1) for each level are displayed in the Figure-15, Figure-16 and Figure-17.

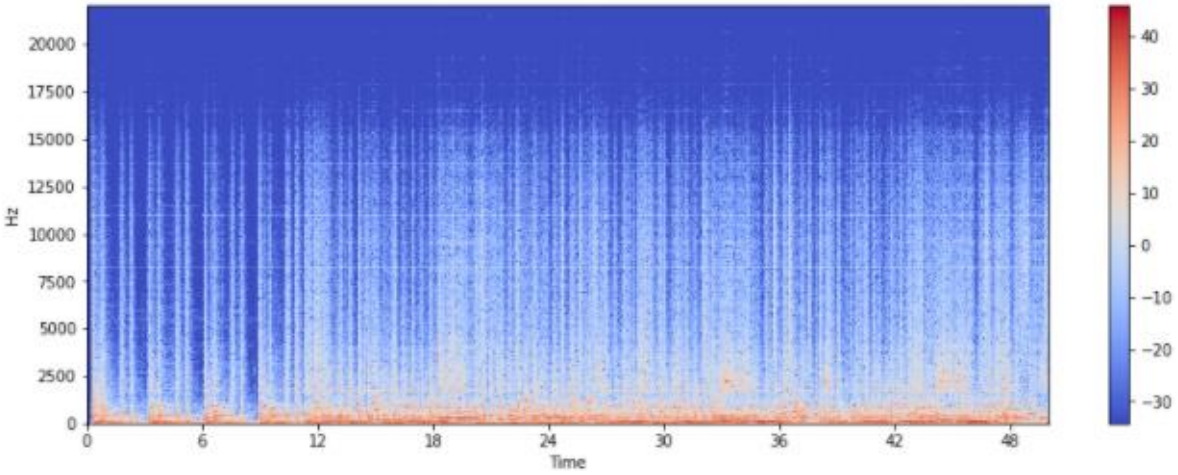

Figure 15: Spectrogram of item_1 in Level 2

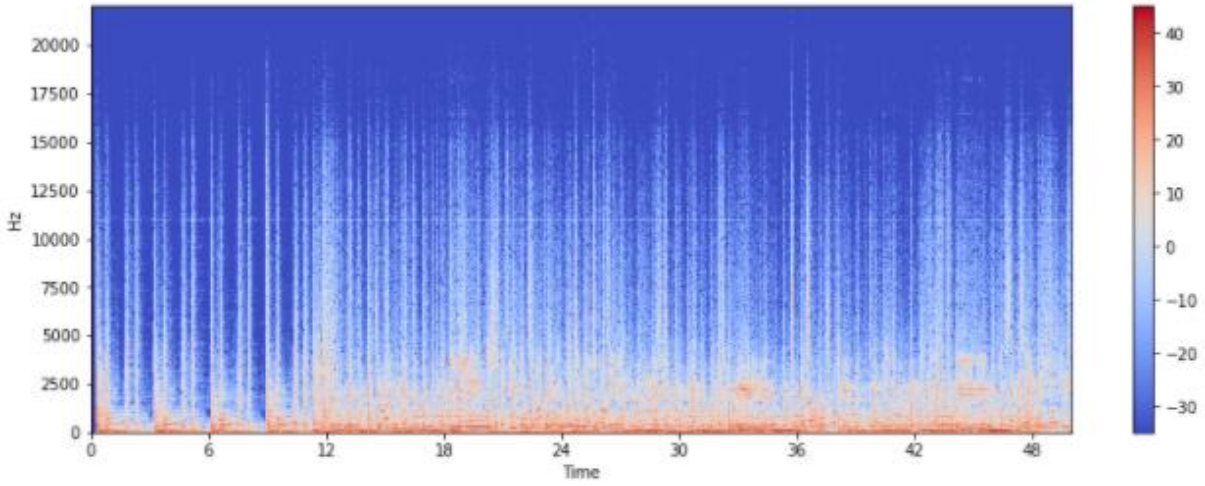

Figure 16: Spectrogram of item_1 in Level 1

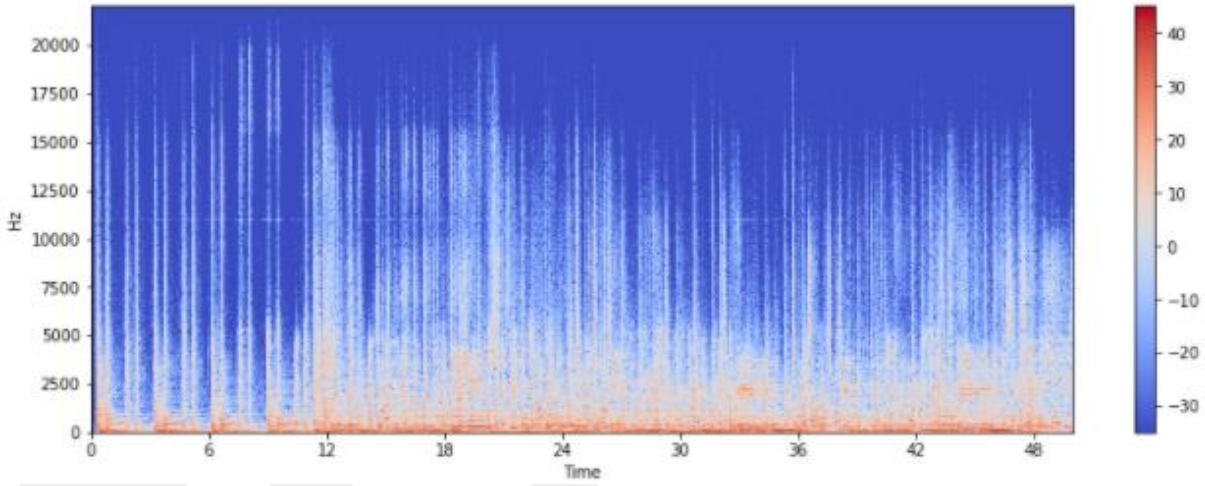

Figure 17: Spectrogram of item_1 in Level 0

We see that the frequency density of the sample decreases from Level 2 to level 0. Our model generates much more noisy samples in the first levels.

First 6 MFCC density of generated song in each level by time are shown in the Figure-18, Figure-19 and Figure-20.

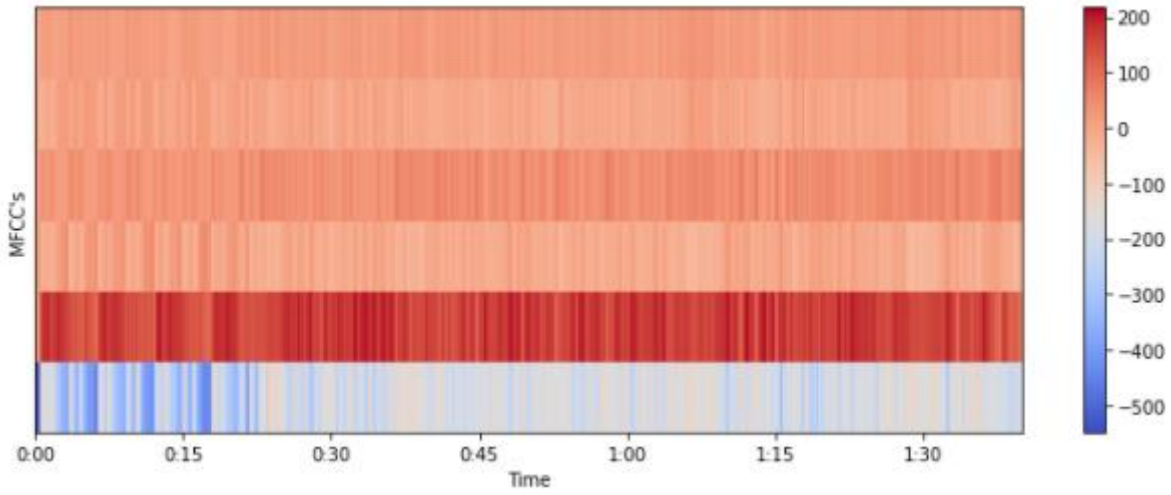

Figure 18: MFCC Spectrogram of item_1 in Level 2

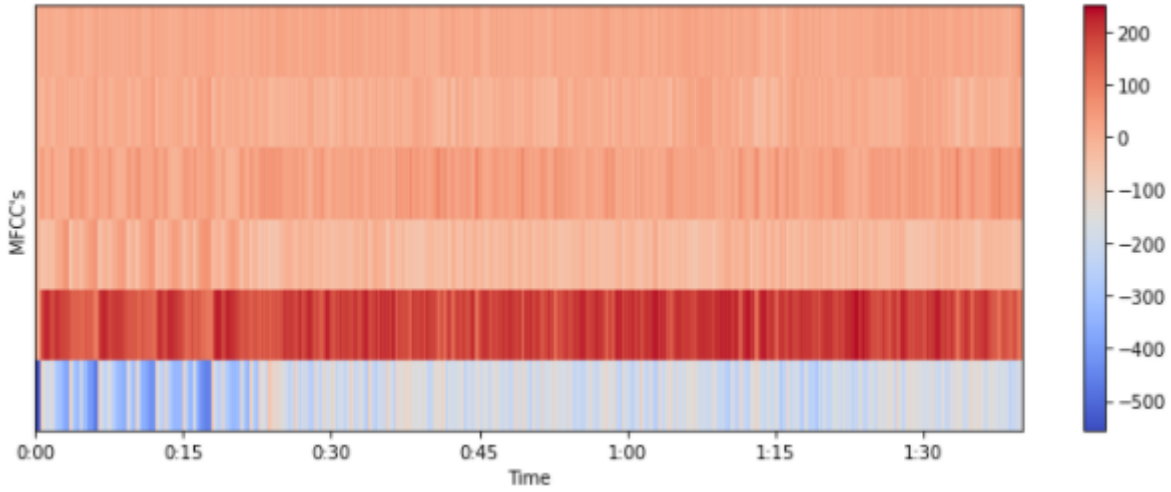

Figure 19: MFCC Spectrogram of item_1 in Level 1

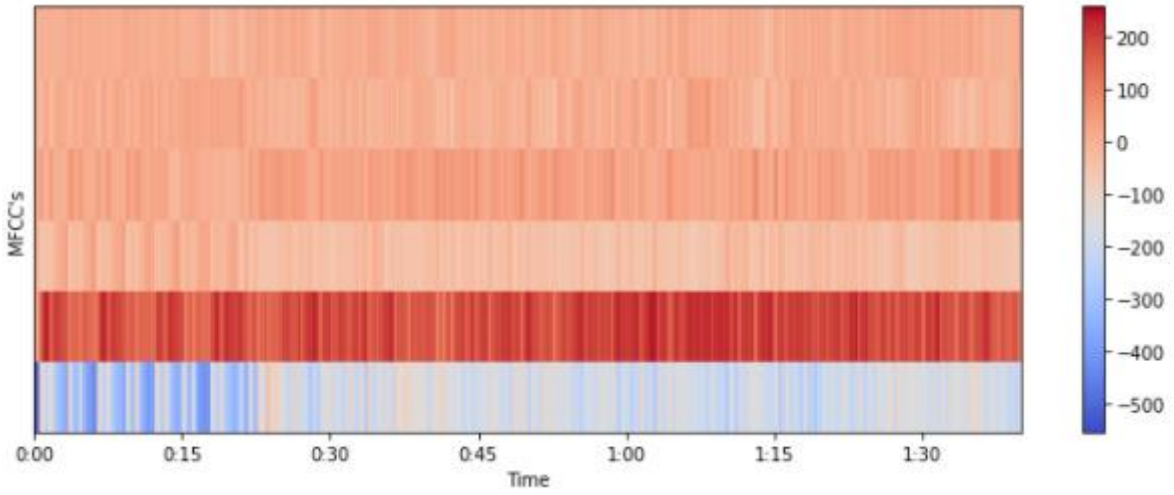

Figure 20: MFCC Spectrogram of item_1 in Level 0

We see that MFCC density is also decreasing through the levels. These levels are hierarchically ordered. For instance, Level 0 represents the upsampled version of Level 1. Each level contains 3 different sampled songs called; item_0, item_1 and item_2. Top level prior (Level 0) captures the long-range structure of music like human voices and melodies. Middle and bottom-up sampling priors (Level 2 and Level 1) captures the local musical structure like timbre (Dhariwal et al., 2020).

Evaluating the generated songs is another issue. Because, creativity and art content have no specific objective function to calculate quality and as I mentioned before, art can not be defined as a problem to be solved. To evaluate AI generated music, metrics should be heuristic and these metrics still have different shortcomings in many ways. Dhariwal et al. (2020) mentioned this problem with the following sentence: “Because everyone experiences music differently, it is generally tricky and not very meaningful to evaluate samples by the mean opinion score or FID-like metrics.”. Thus, they evaluated results manually and I decided to use the same approach to evaluate 3 generated songs.

Jukebox model can capture a wide range of musical information. However, in the downsampling and upsampling phase, VQ-VAE model add undesirable noise to the output. This noise is heard on all 3 generated songs. Also, we can hear local musical coherence in the in generated songs. These songs maintain the similar harmonies and textures during the sample length. But we cannot hear the long-term musical patterns. That means, we cannot

hear any repeated choruses or melodies in the entire generated songs. Dhariwal et al. (2020) linked this problem to lack of context information on the top-level upsampler.

These generated songs are not as interesting as human generated songs. In the human composed melodies, we hear memorable melodies in choruses and these melodies are generally repeated. In our samples, we cannot hear this type of melodies. Also, we can not hear any prior and consequent sequence like in the human generated song. That means, these generated songs are still far from taking place in the music industry marketplace.

Generated songs have nothing in common with original trained songs. I used ancestral mode to generate these samples. That means, the Jukebox model starts to create these songs using the first 6 seconds of given existing songs. Thus, the beginning of each sample has similarity with the given audio signal. However, after 25 seconds, different coherence, melody and rhythm are heard in each song. So, we can conclude that generated songs are different from each other based on their musical features like harmony, melody, texture and rhythm.

6. CONCLUSION

In this project, I briefly explained the musical feature extraction techniques using Music Information Retrieval methods and Deep Learning Techniques for Music Generation concept. I analysed different attributes of sounds in audio domain, retrained the Jukebox model architecture with different Elvis Presley songs and generated 3 different song samples in Elvis Presley style.

Recent developments in the deep learning era allow us to use these techniques in different fields like art. While Music Generation still keeps its mystery, there are various approaches to generate music in the last 5 years using different deep learning architectures in both audio and symbolic domains. I chose the audio domain representation to extract different aspects of music and prevent information loss. I used the Jukebox model because it has a state-of-art architecture to extract lots of information from sounds and can generate high quality, longer than a minute, diverse songs with various low and mid-level attributes like timbre, pitch, loudness and melody. However, these generated songs are still behind to take place in the music marketplace. There is still a major difference between human and AI generated songs. But, every year, the quality of AI-generated songs is increasing. Therefore, it would not be wrong to predict that in the next few years, music created with artificial intelligence will catch and surpass the human level. In my opinion, we will start listening to more AI-generated songs in our daily life soon.

Main two problem of Jukebox model are noise and lack of larger music structures like repetitive chorus and melodies in the generates songs. In the future work, these problems can be focused on. But, VQ-VAE architecture shows its performance on generating diverse long-range coherence harmonies and rhythms. Also, lyrics fit well with the certain piece of songs. Hereby, to generate high quality songs in audio domain, this architecture can be used and taken as basis for improvement.

REFERENCES

[1] Müller, M. (2016). *Fundamentals of music processing: audio, analysis, algorithms, applications*. Springer. doi: 10.1007/978-3-319-21945-5

[2] Briot, J.-P., Hadjeres G., & Pachet F. (2020). *Deep learning techniques for music generation*. Cham, Switzerland: Springer. doi: 10.1007/978-3-319-70163-9

[3] Kotecha, N., & Young, P. (2018). Generating Music using an LSTM Network. *ArXiv, abs/1804.07300.*

[4] Engel, J., Resnick, C., Roberts A., Dieleman, S., Eck, D., Simonyan, K., & Norouzi, M. (2017). Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders. *ArXiv, abs/1704.01279*.

[5] Engel, J., Agrawal, K. K., Chen, S., Donahue, C., Gulrajani, I. & Roberts A. (2019). GANSynth: Adversarial Neural Audio Synthesis. *ArXiv, abs/1902.08710*.

[6] Dhariwal, P., Jun H., Payne, C., Kim, J., Radford A. & Sutskever I. (2020). Jukebox: A Generative Model for Music. *ArXiv, abs/2005.00341*.

[7] Chauhan, N. (2020). Audio Data Analysis Using Deep Learning with Python (Part 1). Retrieved from https://www.kdnuggets.com/2020/02/audio-data-analysis-deep-learning-python-part-1.html

[8] Razavi, A., van den Oord, A., & Vinyals, O. (2019). Generating Diverse High-Fidelity Images with VQ-VAE-2. *ArXiv, abs/ 1906.00446.*

[9] Huang, C. A., Cooijmans, T., Roberts, A., Courville, A. C., & Eck, D. (2019). Counterpoint by convolution. *ArXiv, abs/ 1903.07227*

[10] Blaauw, M. & Bonada, J. (2017). A Neural Parametric Singing Synthesizer. *ArXiv, abs/ 1704.03809*

[11] Wu, J., Hu, C., Wang, Y., Hu, X., & Zhu, J. (2019). A Hierarchical Recurrent Neural Network for Symbolic Melody Generation. *ArXiv, abs/ 1712.05274*

[12] Oord, A. v. d., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., & Kavukcuoglu, K. (2016). WaveNet: A Generative Model for Raw Audio. *ArXiv, abs/ 1609.03499*

[13] Yamamoto, R., Song, E., & Kim, J.-M. (2020). Parallel WaveGAN: A Fast Waveform Generation Model Based on Generative Adversarial Networks With Multi-Resolution Spectrogram. *ArXiv, abs/ 1910.11480*

[14] Vasquez, S., & Lewis, M. (2019). MelNet: A Generative Model for Audio in the Frequency Domain. *ArXiv, abs/ 1910.11480*

[15] Child, R., Gray, S., Radford, A., & Sutskever, I. (2019). Generating Long Sequences with Sparse Transformers. *ArXiv, abs/ 1904.10509*

[16] Kehtarnavaz, N. (2008). *Digital Signal Processing System Design, Second Edition: LabVIEW-Based Hybrid Programming.* Academic Press, Inc. doi: 10.1016/B978-0-12-374490-6.X0001-3