Exploring Music Generation with Variational Auto Encoders and Uncompressed Audio

Tayyib Chohan
tayyibchohan@gmail.com

Julian Wong
jw0ng01@student.ubc.ca

Abstract

Our project investigates using Variational Auto Encoders for the task of music generation. Other relevant research primarily works with the MIDI file format which is a representation of how instruments are played. For this project, we investigated using some of the same architectures that have been successful, using the MIDI format with the WAV format. WAV is an uncompressed representation of the waveform. In comparison to MIDI, WAV can represent sound that does not come from an instrument such as speech. As a result, WAV can represent more sounds than MIDI but requires much more data. We found that working with WAV is very expensive in terms of compute and memory making it difficult to build models. Additionally, there are other questions like how to measure the reconstruction loss of a waveform.

1 Introduction

Generative models have become increasingly relevant in recent years as available compute, data, and research developments have improved the capabilities of these models. Generative models can do many complex tasks like sampling photos (Ramesh et al. 2021), videos (Brooks et al. 2024) and audio (Dhariwal et al. 2020). These models allow people to create media with their current skills, improving accessibility to creative outlets like music and lowering the barrier to entry. Generative models also offer new avenues for expression and exploration in art.

For our project, we wanted to learn more about generative models for music. We focused on implementing a Variational Auto Encoder (VAE) as we know this is an architecture that has worked using MIDI (Jiang et al. 2020). Additionally, we both have limited experience with VAE's as we worked on implementing components of a VAE in an assignment for another course.

Currently, there is limited research on using VAEs with audio files in the WAV format as many papers focus on using audio in MIDI (Jiang et al. 2020). WAV is an uncompressed audio format that represents music as a sequence of amplitudes. On the other hand, MIDI files do not contain actual audio waveforms; instead, they contain instructions or commands that describe musical events, such as note-on, note-off, pitch, velocity, duration, and other performance parameters. We wanted to try using the WAV format as it describes the sound we hear as opposed to how the instruments are played.

This project would best fit the description of the coding project as we are using an architecture similar to other implementations of a VAE for audio generation. Our goal is not to make anything fundamentally new but to learn more about VAEs, how to work with audio, and hopefully demonstrate how VAE can be used to generate WAV samples that sound like the training data to some degree. Additionally, this project fits well with the content of CPSC 440 as we discussed VAEs in class.

For this assignment, we used a GTX 1070Ti to train, test and sample from our models.

Course project for UBC CPSC 440/550, April 2024.

2 Related Work

Variational Auto Encoders (VAE) are a generative model architecture originally proposed for the purpose of image generation (Kingma and Welling 2022). Methods including Variational Auto Encoders (VAE), Generative Adversarial Networks (GAN), diffusion models, and transformers have been proposed for music generation (Choi et al. 2020), many of which include architectures featuring combinations and variants of these techniques (Han et al. 2023). Wu and Yang (2022) additionally augmented their data to hold more relevant musical information using a customized format called Revamped MIDI (REMI). We can see that many of these papers and architectures take advantage of MIDI and other musical representations despite not directly representing the audio waveform.

3 Description and Justification

3.1 Variational Auto-Encoders

Variational Auto Encoders is a stochastic variational inference and learning algorithm. Kingma and Welling (2022) demonstrated how VAEs can be used to learn a latent space representation of images, and that we can sample from latent space and pass the samples through the decoder to produce images that look like the training data.

Since we have seen that VAEs can work for MIDI (Jiang et al. 2020) we thought that VAEs could be used for generating WAV files as well.

3.2 Pytorch

We decided to use Pytorch to build our models as we have used it for other assignments. Additionally, Pytorch handles a lot of the implementation of common computation blocks allowing us to quickly try ideas and not need to worry about incorrect implementation of the computation block and their gradients. Finally, Pytorch has audio libraries for common audio processing tasks like loading WAV files into a tensor, saving tensors as WAV files, and changing the sample rate.

3.3 Dataset

We used the Kaggle MusicNet dataset for training our model. This dataset includes 330 freely-licensed classical music recordings in MIDI and WAV.

3.4 Data Processing

The sample rate of a WAV file refers to the number of samples of audio carried per second, measured in Hertz (Hz). It determines the fidelity of the audio reproduction. In simpler terms, it represents samples per second of audio waveform. For our VAE we did not want to handle training on multiple sample rates or producing samples with different sample rates. Additionally, decreasing the sample rate reduces the size of the training data for the same duration of training examples (there is less data per second of music). This allows us to use less memory for storing the training data and decreases the size of our model for the same length of training data and samples produced. For these reasons, our dataloader resamples all the examples to the same sample rate and we save all of our generated samples with the same sample rate. Through trial and error, we found that the lowest sample rate most audio players will play is 3000 Hz, which is the value we used.

VAE's have a fixed input and output size. To fit this constraint our dataloader randomly samples a different constant-size subsection of every training sample each time the example is loaded. For example, if we have a song in our training data that is 30 seconds long and our VAE takes as input 5 second sections of music, on each epoch when the song from our training data is loaded into memory we randomly select a 5 second window from the 30 second song.

3.5 Reconstruction Loss

VAEs use the evidence lower bound or ELBO as a metric to measure how well the VAE is performing (Kingma and Welling 2022). The higher the ELBO the better the VAE is performing. In our models, we used the negative ELBO as our loss function.

The negative ELBO is the sum of two parts: The reconstruction loss and the KL-Divergence. The reconstruction loss is a measure of how well the VAE can rebuild the training image, and the KL-Divergence is a measure of how different the learned latent space is from the prior belief of the distribution of the latent space (higher KL-Divergence indicates a larger difference in distributions). For audio, we thought that mean squared error MSE would be a good measure of the reconstruction as we would want to reduce the difference between the amplitudes of the waveforms. There may be better ways to measure how well the VAE rebuilds a waveform (some metrics like FID for images (Heusel et al. 2018)). Some further work to be done would be to investigate the reconstruction loss further.

4 Experiments

4.1 VAE Transformers

Initially, we attempted to create a transformer-based VAE as described by Jiang et al. (2020). Their implementation made use of the MIDI format which is much more compact than an equivalent WAV file. This means that our model was large for the duration of the input. Additionally the computation and memory scales quadratically with the length of the sequence. Running our model on a GTX 1070Ti, we ran out of memory using a duration of one second. We attempted to remedy this by decreasing the sample rate of the audio. We lowered the sample rate to 100 samples per second from the default value of 44100 samples per second, this deteriorated the quality of the input. A valid WAV has a sample rate of greater than or equal to 3000, so our data was not playable. Additionally, we reduced other hyperparameters of the model to decrease its size. After making these changes, we could not store the model in memory. Since this model did not seem feasible with our available hardware we decided to explore more compact architectures.

4.2 Convolutional VAE

Our next attempt at a valid architecture was to use convolutional layers in a Variational Auto Encoder similar to the original VAE paper (Kingma and Welling 2022). We could now fit a model with a sequence length of one second into our memory. This resulted in a quickly decaying training loss followed by a plateau shown in Figure 1. We could not improve the performance by adjusting the learning rate and other hyperparameters. Samples from this model were completely silent A. The VAE we made used an isotropic multivariate Gaussian approximation and assumed the ground truth distribution was also multivariate Gaussian for our ELBO. Meaning our covariance matrix was represented by the equation $\sigma^2\mathbf{I} = \Sigma$ where we learn the scaler σ. We tried loosening the assumption so that the covariance matrix is represented by a learned diagonal matrix. Despite these changes, we saw no improvements in training accuracy or sample quality.

We tried to improve this model by following Yuehan (n.d.)'s example of using an image representation of the audio called a Mel Spectrogram and then applying two-dimensional convolutions to the data. This representation transforms the audio data using short-time Fourier transforms and other signal processing techniques to visually represent the audio as an image. We were able to create the Mel Spectrograms but converting back to a waveform significantly deteriorated the quality as converting to a Mel Spectrogram is a lossy conversion. Additionally, Mel Spectrograms required math beyond our current scope of understanding so we abandoned the idea.

4.3 Fully Connected VAEs

The final architecture we tried was a VAE with fully connected instead of convolutional layers. The training loss for the model is depicted in Figure 2. As shown, the training loss quickly decreases initially but has very stochastic behaviour afterwards. Despite this, we were able to produce samples from our model that sounded like there may be instruments being played over noise A We attempted to improve the model by increasing the number of layers in both the encoder and decoder. As we scaled the model from one to fifteen hidden layers we noticed little to no change in the behaviour of the training loss. We did not scale the model beyond fifteen hidden layers because of memory constraints. Since the model was very stochastic we attempted a large range of learning rates but could not improve the model's loss from being stochastic.

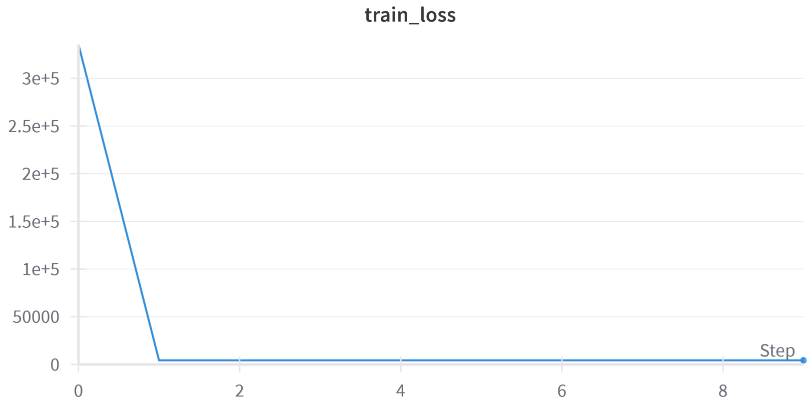

Figure 1: Training loss for VAE with a 1D Convolutional Layer

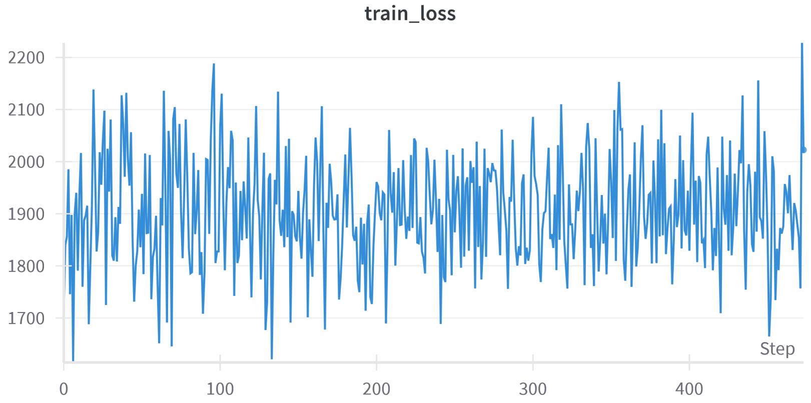

Figure 2: Training loss for Deep Fully Connected VAE

5 Discussion

Our exploration into music generation with Variational Auto Encoders (VAEs) and uncompressed audio has provided valuable insights into the challenges and potential of generating music with VAEs. We encountered several challenges highlighting the complexity of working with WAV files in VAEs.

One major challenge was the sheer complexity of working with uncompressed WAV audio data. Unlike MIDI, which represents musical events in a concise format, WAV files encode raw audio waveforms, making them significantly larger and more computationally demanding to process. This complexity posed difficulties in model architecture design, training, and optimization. We found that traditional architectures, such as convolutional and fully connected VAEs, struggled to effectively capture the nuances of audio data, resulting in poor sample quality and training instability.

Furthermore, our experiments revealed the importance of understanding the intricacies of music representation and signal processing. While we attempted to leverage techniques such as mel spectrograms to simplify the audio data, we encountered challenges translating these representations back into meaningful audio samples. This highlighted our need for a deeper understanding of signal processing techniques and their application in music generation tasks.

Despite these challenges, our exploration lays the groundwork for future research in this area. By addressing the limitations identified in our experiments, such as improving model architectures, incorporating advanced signal processing techniques, and leveraging larger computational resources, future endeavours in music generation with VAEs can overcome the issues we encountered and achieve more compelling results.

6 Future Work

There are a range of possible directions one could explore for future work. Firstly increasing computational resources would allow larger more intricate models to be used that could better fit the large format of WAV files. Another area we could explore further techniques to refine the representation of the data. This could be similar to representing the data as a Mel Spectrogram or embedding other information into the input (Wu and Yang 2022). Extending from this the visual nature of spectrograms and other representations could lend themselves to transfer learning (Zhuang et al. 2020). A future study could explore the fine-tuning of a computer vision model in the hope of transfer learning occurring. Another avenue for improvement is attempting to incorporate domain-specific knowledge into the model. We may be able to find a better loss function than mean squared error using music theory or signal processing to better quantify the performance of our model or build models that can more efficiently summarise patterns in the WAV format.

Acknowledgments

We heavily referenced our previous assignment on implementing a VAE. We used our implementation of components and the stub code from the assignment (Liao et al. 2024).

References

Brooks, Tim, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh (2024). "Video generation models as world simulators." URL: https://openai.com/research/video-generation-models-as-world-simulators.

Choi, Kristy, Curtis Hawthorne, Ian Simon, Monica Dinculescu, and Jesse Engel (2020). *Encoding Musical Style with Transformer Autoencoders*. URL: https://openreview.net/forum?id=Hkg9HgBYwH.

Dhariwal, Prafulla, Heewoo Jun, Christine McLeavey Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever (2020). "Jukebox." URL: https://openai.com/research/jukebox.

Han, Bing, Junyu Dai, Weituo Hao, Xinyan He, Dong Guo, Jitong Chen, Yuxuan Wang, Yanmin Qian, and Xuchen Song (2023). *InstructME: An Instruction Guided Music Edit And Remix Framework with Latent Diffusion Models*. arXiv: 2308.14360 [cs.SD].

Heusel, Martin, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter (2018). *GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium*. arXiv: 1706.08500 [cs.LG].

Jiang, Junyan, Gus G. Xia, Dave B. Carlton, Chris N. Anderson, and Ryan H. Miyakawa (2020). "Transformer VAE: A Hierarchical Model for Structure-Aware and Interpretable Music Representation Learning." *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 516–520. DOI: 10.1109/ICASSP40776.2020.9054554.

Kingma, Diederik P and Max Welling (2022). *Auto-Encoding Variational Bayes*. arXiv: 1312.6114 [stat.ML].

Liao, Renjie, Qi Yan, Muchen Li, Jiahe Liu, and Qihang Zhang (2024). *CPEN 455*.

Ramesh, Aditya, Mikhail Pavlov, Gabriel Goh, Scott Gray, Mark Chen, Rewon Child, Vedant Misra, Pamela Mishkin, Gretchen Krueger, Sandhini Agarwal, and Ilya Sutskever (2021). "DALLE: Creating images from text." URL: https://openai.com/research/dall-e.

Wu, Shih-Lun and Yi-Hsuan Yang (2022). *MuseMorphose: Full-Song and Fine-Grained Piano Music Style Transfer with One Transformer VAE*. arXiv: 2105.04090 [cs.SD].

Yuehan (n.d.). *Introduction to Variational Autoencoders (VAEs) in AI Music Generation — yuehan-z.medium.com*. https://yuehan-z.medium.com/introduction-to-vaes-in-ai-music-generation-d8e0cfc2245b. [Accessed 27-04-2024].

Zhuang, Fuzhen, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He (2020). *A Comprehensive Survey on Transfer Learning*. arXiv: 1911.02685 [cs.LG].

A Supplementary material

Links:

Audio Sample Clip Examples
Source Code