ML Techniques for Music

Joel Diéguez
Universitat de Barcelona
jdieguvi15@alumnes.ub.edu

Clàudia Valverde
Universitat de Barcelona
clvalves7@alumnes.ub.edu

Abstract

Having an interest in machine learning (ML), we have observed a notable gap in the explanation of machine learning techniques within the music domain. Existing literature predominantly emphasizes the applicability of ML models in this domain rather than delving into their explainability. This lack of focus has resulted in a dispersed landscape of research in this field. Our paper aims to address this gap by specifically concentrating the explanation of ML techniques in the context of prompt-text Music Generation. We describe the structure of these systems, focusing on challenges at various stages. We also present different models in this field in chronological order, with a final emphasis on three state-of-the-art models Jukebox, Riffusion and MusicGen. Our hope is that readers will not only comprehend the framework in this field but will also grasp the current state of research models and the challenges they confront.

1 Introduction

In recent years, Machine Learning (ML) has experienced a breakthrough, capturing widespread attention and finding applications in various domains, such as text translation, image classification, Natural Language Processing (NLP), Computer Vision, and Quantitative trading. These areas are witnessing exponential growth, with new techniques emerging seemingly on a daily basis. Despite the prevalence of ML in these domains, its integration into the field of music has also marked significant progress.

Leveraging the capabilities of algorithms and data analysis, ML applications in music span a diverse range of functionalities, from composition and production to recommendation systems and genre classification.

Music Generation utilizes modern generative algorithms to extract implicit patterns in a piece of music based on rule constraints or a musical corpus. Its applications are diverse, extending from text-to-speech [19] to generating music conditioned on lyrics [**?**]and synthesizing audio from MIDI sequences [10]. The music production process typically involves five steps: 1) composition; 2) arrangement; 3) sound design; 4) mixing; and 5) mastering. Contemporary AI music generation covers various stages, inspiring creative ideas, assisting with song structure, and even engaging in fully autonomous composition. This provides creators with valuable tools for exploration and innovation across different aspects of music production.

Inspired by advancements in text-to-image generation, recent efforts explore generating audio from sequence-wide, high-level captions, such as "whistling with wind blowing". However, synthesizing high-quality and coherent audio encounters challenges, primarily due to the limited availability of paired audio-text data, a contrast to the image domain. Describing audio characteristics is inherently more complex than describing images, as audio involves temporal structures and capturing salient features is nuanced. Additionally, audio is structured along a temporal dimension which presents difficulties in creating sequence-wide captions, making the process more complex compared to the

35th Conference on Neural Information Processing Systems (NeurIPS 2021).

relatively straightforward image captioning approach. So this is one of the main challenges that audio/music generation faces.

2 Overview

In music, fundamental elements such as pitch, duration, intensity, and timbre combine to create melody, harmony, rhythm, and tone. In the realm of artificial intelligence composition, melody serves as a primary focus for automatic Music Generation systems (MGSs). Most melody generation algorithms aim to replicate specific styles, such as generating Western folk or free jazz. Harmony, audio generation and style conversion are also popular directions in intelligent composition.

The overall framework of the specific MGS is shown in Figure 1. In the process of generating music, the musical content needs to be encoded in digital form creating a representation that is taken as input to the algorithm, after which the intelligent algorithm is learnt and trained to eventually output different types of musical segments, such as melodies, chords, audio, style transitions, etc. Finally the evaluation of the model is done.

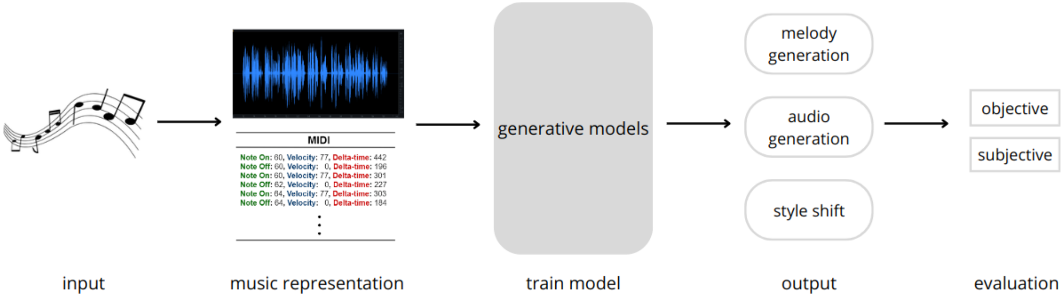
MIDI
Note On: 60, Velocity: 77, Delta-time: 442
Note Off: 60, Velocity: 0, Delta-time: 196
Note On: 60, Velocity: 77, Delta-time: 301
Note Off: 62, Velocity: 0, Delta-time: 227
Note On: 64, Velocity: 77, Delta-time: 303
Note Off: 64, Velocity: 0, Delta-time: 184
generative models
melody generation
audio generation
style shift
objective
subjective
input
music representation
train model
output
evaluation

Figure 1: Overview of a Music Generator System

3 Music Representation

The input to a music generative system is usually some representation of music, which has an important influence on the number of input and output nodes (variables) to the system, the correctness of training, and the quality of the generated content [2]. Music can be represented in two formats: audio or symbolic. Audio is a continuous signal whereas symbol is a discrete signal.

3.1 Audio Representation

Music can be represented in audio through two primary methods: a) Waveform [19] and b) Spectrogram [18]. The spectrogram is derived through the short-time Fourier transform of the original audio signal, preserving the original properties of the music waveform for the production of expressive music. However, this representation has limitations, as music fragments in the original audio domain are typically depicted as a continuous waveform $x \in [-1, 1]T$, with T representing the audio duration and a sampling frequency ranging from 16 kHz to 48 kHz. Short-term music fragments result in significant data generation. Therefore, modeling the raw audio can lead to a somewhat range-dependent system, making it challenging for intelligent algorithms to learn the high-level semantics of music, such as the Jukebox [6] which takes approximately nine hours to render one minute of music.

3.2 Symbolic Representation

Musical Instrument Digital Interface (MIDI) is a technical standard that describes a communication protocol, digital interface, and electrical connectors that connect a wide variety of electronic musical instruments, computers, and related audio devices for playing, editing, and recording music. MIDI facilitates the transmission of musical information, such as note sequences, velocity, pitch, and control signals, allowing for the creation and manipulation of music electronically. It has become a crucial tool in the music industry for composing, recording, and controlling various sound-producing devices, including synthesizers, drum machines, and computers, fostering interoperability among different musical instruments and equipment. For example, Music transformer [14], MusicVAE [23], etc. convert melody, drums, piano, etc. into MIDI events and use them as input to the training network to generate music fragments. However, MIDI events cannot effectively retain the concept of multiple tracks playing multiple notes at once [13].

There are other symbolic music representations such as Piano-roll which involves one-hot encoding of MIDI files and ABC notation that consists of encoding melodies to text.

3.3 Music Datasets

For deep neural network-based generative systems, selecting a suitable music dataset as the system's input is equally important than determining an appropriate musical representation. Some complete datasets exist such as Lakh MIDI Dataset [1], JSB-Chorales [20] have been widely used in the community to train different music generation algorithms, however we have identified two primary drawbacks. Firstly, in general, music datasets often lack diversity, potentially attributed to the inherent difficulty in extracting audio representations compared to text or image data. Many current datasets exhibit bias towards Western music or specific genres or instruments, possibly due to the technical challenges in obtaining comprehensive music representation. The second limitation pertains to restricted accessibility and copyright-related challenges, making it difficult to obtain large and comprehensive music datasets. While certain high-performing music generator models like MusicGen [3] utilize internal and private datasets, enhancing research in this field requires a concerted effort from the research community. This involves advocating for extensive international collaboration, ensuring cultural compliance, and promoting data transparency.

4 Models

After our study through the most renowned music generation tools, we can classify them into different categories:

Based on the type of problem they try to solve, we have:

- **Melody generation**: if they generate a simple melody for an instrument.
- **Arrangement generation**: if they are able to combine different instruments, having lead melodies and counterpoints.
- **Audio generation**: if they generate directly at audio level, using implicitly different instruments and styles. This method is more computationally complex as it involves identifying the different semantic structures from the raw audio.
- **Style transfer**: it uses different arrangements to transform a piece of audio into a particular style or instrument.

In this study we want to focus on the ones which generate audio.

Music generation tools can also be classified based on their input:

- **Parameter based**: these are models that require a human to specify certain characteristics in the form of parameters or settings to make the piece. Examples of these parameters could be the number of instruments, whether we want a male, female or robotic voice, the duration, the style or even the emotion it evokes.

  All music generation methods prior to neural networks belong to this category, and even some that already use them but still require manual parameters.
- **Prompt based**: similar to the image generation models we are used to nowadays, these models require information in the form of text describing the piece, it identifies the prompts requested and generates an audio accordingly.

  There are models that can take as input the lyrics of the song. Although there is the possibility for a model to create a song based on the style and sonority of the lyrics, the models we have encountered to date always require additional parameters, so we cannot include them in this category.
- **Visual based**: these models are based on visual inputs, such as images or video. They extract contextual information from them and generate an audio that matches.

This category fits a very important niche market of audio generation: copyright-free background music for videos.

However, the most interesting classification and the main part of this study is based on the different models or algorithms used.

In Figure 2, we show the year by year evolution of the most famous audio generation models. Note the distinction between models that use neural networks and those that do not.

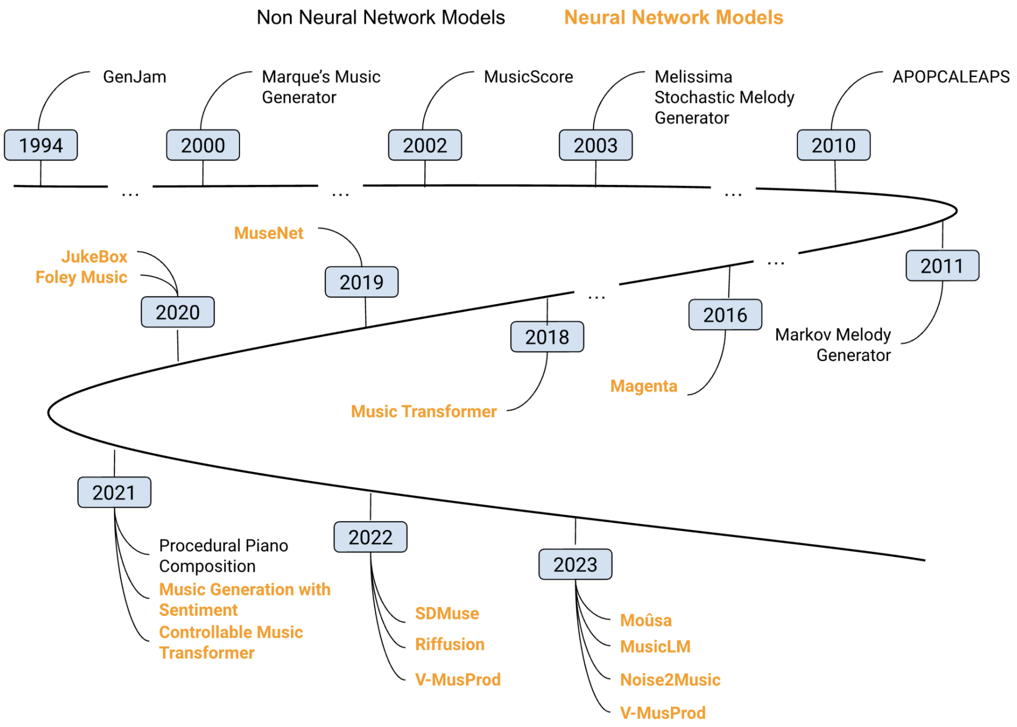

Figure 2: Cronology of Music Generators

4.1 Models prior to Neural Networks

On the one hand, we have the models prior to neural networks. There are 3 main ones:

4.1.1 Rule Based

These models are based on the rules of music theory. They include rules based on musical grammar, similarity, rhythm, etc. This allows to generate coherent music, however the output is very limited, all songs follow a similar form and cannot see the bigger picture. Moreover, the more laws we add, the more ambiguous and limited the result will be. One example of this arquitecture is Morpheus [12].

4.1.2 Markov Models

Markov models [21, 25] are mathematical models that are based on the probability of time series, for example the appearance of notes or chords, the transition between note and chord or a change of rhythm. Since music is innately a time series, these models work very well in generating new melodies. In addition, with different parameters, we can regulate the probability of interactions and adapt the music to our goal. An example of their good performance can be found in the work of Chi-Fang with their creation of Chinese folk music [15]. Unfortunately, when used to compose longer pieces, they tend to be repetitive and redundant.

4.1.3 Genetic Algorithms

Based on the laws of human evolution, these algorithms [7, 4] find the best musical combinations by evolving, mutating and combining results. These models require defining some way to evaluate whether the individuals are close to the objectives or not. Their biggest problem is that the results sound very "computer-generated" because they lack regularity and synchronicity.

4.2 Models after Neural Networks

On the other hand, we have neural networks. Once they were applied to audio generation, the results were much better and the research never looked back. Today these models dominate the state of the art in this field. Below, we have chosen three algorithms that represent the best performing models over the last few years.

The reason we have chosen these models is simple. The objective of this paper is to present the history and state of the art of audio generation so that future data scientists have a good basis for further research. These three models are good examples of different architectures that have been pillars for this field: VQ-VAE, Diffusion models and Transformers. Moreover, all of them are open source and have an open github to see and recreate their code.

4.2.1 JukeBox

This model [6] from OpenAi is based on the VQ-VAE architecture [22], i.e. Vector Quantised-Variational AutoEncoder. In its essence, it is a VAE enhanced with VQ i.e. adding Quantization. The objective of Quantization is to give a discrete and compact representation that at the same time allows to form a high fidelity reconstruction.

The structure is an auto-encoder with a discretization bottleneck. It consists of an Encoder, which encodes into a sequence of latent vectors, a bottleneck that quantizes each vector against a codebook, and a Decoder that decodes the embedded vectors back into the input space. The codebooks are reset when necessary to avoid collapse.

Observing the results, it was clear that the model focused too much on small frequencies and ignored the high frequencies, so it was divided into three layers, each one compressing the audio by a different factor. This estructute can be seen in Figure 3.

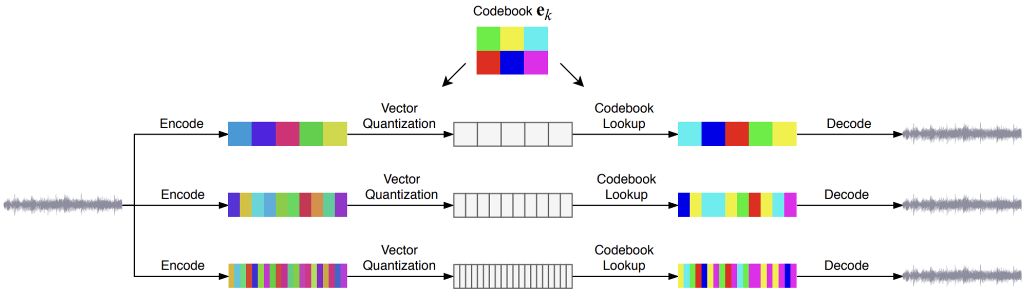

Figure 3: Graphical representation of Jukebox's architecture.

This architecture uses self-supervised training. Starting and ending with audio files.

As we have seen so far, it is parameter based. However, we can combine it with **MuLan** [16] . MuLan is a revolutionary encoder that allows to encode two types of data: audio and text, in the same output space.

Its structure is divided into two towers. The first one for audio, consisting of an M-ResNet-50 [17, 11] encoder, that is a residual-layered convolutional network usually used for images but applied to audio, and an M-AST [9], an audio spectrogram transformer. The second tower encodes the text using a BERT [5], a transformer for text encoding. To train, it uses a large-scale training dataset of (audio, text) pairs extracted from 50 million internet music videos.

Combining them, it is possible to apply the VQ-VAE architecture on the embeddings generated by MuLan, whether they are generated from audio or text. Moreover, this facilitates training, since we can train on audio files and compare the results with the original audios, and evaluate it on text.

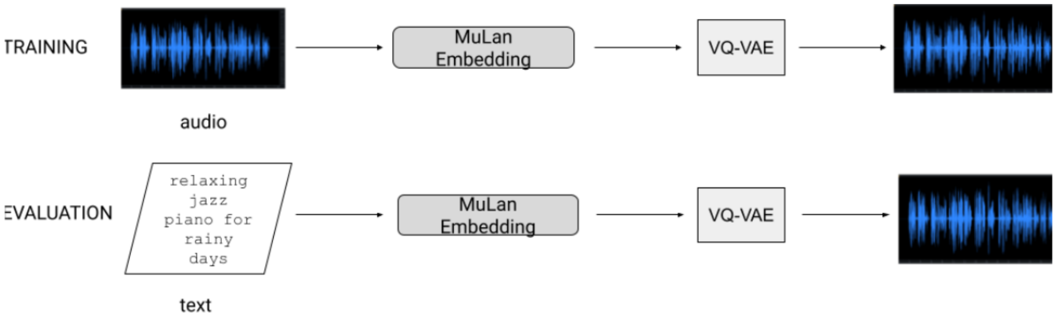

Figure 4: Training and evaluation pipeline of Jukebox using MuLan.

This eases the lack of public datasets with text-to-audio pairs.

4.2.2 Rifussion

This network [8] is based on the Stable Diffusion model [24], an open-source AI model that generates images from text. In fact, it was born by fine-tuning the Stable-Diffusion-v1-5 checkpoint to generate spectrogram images given text inputs, and then converted to audio. The architecture itself is composed by denoising autoencoders and a diffusion process to generate the data.

This is an example of the various current models that use architectures already trained for other complex and data-intensive problems, and are repurposed to generate audio. These structures are already used to deal with very complex data and strong relationships, so it is not difficult for them to comprehend audio.

Another example of this is **FlowGPT**, based on ChatGPT. This model became popular in 2023 for generating a song called *nostalgIA* in the style of *Bad Bunny* and *Bad Gyal*. This song raised a lot of controversy online between people who considered it a masterpiece better than their original songs and fully captured their essence and people who considered it a blatant plagiarism.

4.2.3 MusicGen

MusicGen [3] uses the transformer architecture and takes text or melodies to generate new songs. First, it uses multiple codebooks (patterns) to extract information from the data and a positional embedding to embed it. Then, we apply a multi-layer transformer decoder, where in each layer it uses casual self-attention. Next, we apply the conditioning with the given text/melody with a cross-attention block. Then, we apply a linear model with ReLU. After each step, the outputs are normalized and connected using residual layers. Finally, we will reconstruct the song applying the codebooks based on the results obtained.

This model is used in **AudioCraft** [3], an AI audio toolkit created by Meta. It is considered by many the best model to generate music nowadays.

5 Evaluation Methods

Similar to the challenges associated with retrieving data for AI music generators, evaluating their results presents another hurdle. The assessment process involves a combination of subjective and objective measures. Within the objective evaluation domain, various tools and metrics, including harmonic metrics, rhythm metrics, melodic metrics, and statistical analysis such as frequency of musical events, have been devised to offer quantifiable assessments. These measures fall short of perfectly modeling concepts in music that humans perceive, given the importance of human perception. This is where subjective evaluation comes into place, where it takes into account human perception and emotional response. This duality could be one of the reasons why its hard to achieve significant scalability in these models.

For this study we have decided to evaluate our three choice models with the following metrics:

- Possible input format.
- Number of parameters.
- Training time.
- Execution time.
- Subjective evaluation of the output.

6 Results

To compare the models, we have recreated their operation with our own prompts.

We have not trained any model from scratch as it would mamny require resources to which we do not have, but we have accessed the pre-trained versions and tested them.

Table 1 contains information we have collected from official sources or, in case the information was not public, we have tried to approximate it as best as possible.

There is a couple of notes we wanted to mention here. Even though Jukebox has 6 billion parameters in total, the VQ-VAE only has 2 million, the rest are the upsampler models. It is also notable that while the whole model takes approximately 6 weeks to train, the VQ-VAE only takes 3 days. Also, for Riffusion, there is no official paper so we could not find the exact numbers, we used an approximation based on the original Stable Diffusion model.

Table 2 shows the data about the testing results.

Table 1: General information about the models.

	Arquitecture	Developers	Training dataset	# Parameters
Jukebox	VQ-VAE	OpenAI	1.2 M songs	6 B
Riffusion	Diffusion model	StabilityAI	images and songs	890 M
MusicGen	Transformer	Meta AI	20k hours of music	300M, 1.5B or 3.3B

Table 2: Information about the models obtained through testing.

Name	Input	Execution Time	Time generated
Jukebox	prompts, parameters or audio	>9h	1min 20s
Riffusion	promts, lyricks or images	<10s	12s
MusicGen	promts or audio	$\sim$10min	2min

6.1 Subjective Evaluation

During our exploration of JukeBox, we noticed a significant variability in its performance. Remarkably, it operates at a high speed. When provided with audio input, the outcomes show limited variation. Likewise, when using text prompts, the generated outputs display diversity, with some instances yielding satisfying results and others revealing noticeable artifacts. Surprisingly, the music is notably clear. The genres are quite strict, possibly influenced by the interface being utilized. Interestingly, the lyrics are very clear, which might be attributed to fine-tuning, although this might indicate more difficulty in generalization.

Riffusion delivers high-quality music with minimal noise. However, the platform's inherent architecture imposes limitations on user control over the musical output. Another aspect for improvement is Riffusion's inability to generate longer compositions, restricting users to short tracks. Additionally, the looping feature in the short jams may introduce repetitiveness in the generated music. The Riffusion case is intriguing because, by itself, it has some problems generating intelligible human voices. However, when added as a lyric prompt, they give the best results we have seen among all other competitors, as they are fully understandable.

As the most recent addition to the field, MusicGen stands out as a cutting-edge controllable music generation model. In the realm of text-to-music generation, MusicGen has demonstrated superiority over its counterparts. It clearly shows high adderance to the text conditioning and generates original and good-sounding melodies. However, while text conditioning gives amazing results, the process becomes more complex when applied to audio conditioning, it needs a bit more help.

It's worth noting that the training dataset prominently features Western-style music, potentially leading to a lack of diversity in the music generated by MusicGen.

7 Conclusions

One of the primary challenges encountered during the preparation of this paper is the dispersed nature of literature and content within this field. Some papers predominantly focus on the application level, providing limited information on the general methods of music generation and lacking in-depth exploration of the theme. The unavailability of code and datasets from online music generators raises questions, primarily attributed to copyright issues or perhaps to the rapidly evolving nature of the field.

To address these challenges, we have presented a chronological overview of music generators, with a specific focus on JukeBox, Riffusion, and MusicGen. Through this analysis, we have obtained results and compared metrics related to their architecture and training procedures. Despite the overall positive outcomes with pre-trained weights, it's noteworthy that some models still exhibit anomalies in their ouputs.

References

[1] Thierry Bertin-Mahieux, Daniel PW Ellis, Brian Whitman, and Paul Lamere. The million song dataset. 2011.

[2] Jean-Pierre Briot, Gaëtan Hadjeres, and François-David Pachet. *Deep learning techniques for music generation*, volume 1. Springer, 2020.

[3] Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez. Simple and controllable music generation, 2023.

[4] Roberto De Prisco, Gianluca Zaccagnino, and Rocco Zaccagnino. Evocomposer: An evolutionary algorithm for 4-voice music compositions. *Evolutionary Computation*, 28:1–42, 10 2019.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[6] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. Jukebox: A generative model for music, 2020.

[7] Majid Farzaneh and Rahil Mahdian Toroghi. *Music Generation Using an Interactive Evolutionary Algorithm*, page 207–217. Springer International Publishing, December 2019.

[8] Seth* Forsgren and Hayk* Martiros. Riffusion - Stable diffusion for real-time music generation. 2022.

[9] Yuan Gong, Yu-An Chung, and James Glass. Ast: Audio spectrogram transformer, 2021.

[10] Curtis Hawthorne, Ian Simon, Adam Roberts, Neil Zeghidour, Josh Gardner, Ethan Manilow, and Jesse Engel. Multi-instrument music synthesis with spectrogram diffusion. *arXiv preprint arXiv:2206.05408*, 2022.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[12] Dorien Herremans and Elaine Chew. Morpheus: Generating structured music with constrained patterns and tension. *IEEE Transactions on Affective Computing*, 10(4):510–523, October 2019.

[13] Allen Huang and Raymond Wu. Deep learning for music. *arXiv preprint arXiv:1606.04930*, 2016.

[14] 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. *arXiv preprint arXiv:1809.04281*, 2018.

[15] Chih-Fang Huang, Yu-Shian Lian, Wei-Po Nien, and Wei-hua Chieng. Analyzing the perception of chinese melodic imagery and its application to automated composition. *Multimedia Tools and Applications*, 75, 07 2015.

[16] Qingqing Huang, Aren Jansen, Joonseok Lee, Ravi Ganti, Judith Yue Li, and Daniel P. W. Ellis. Mulan: A joint embedding of music audio and natural language, 2022.

[17] Ievgeniia Kuzminykh, Dan Shevchuk, Stavros Shiaeles, and Bogdan Ghita. *Audio Interval Retrieval Using Convolutional Neural Networks*, page 229–240. Springer International Publishing, 2020.

[18] Chien-Yu Lu, Min-Xin Xue, Chia-Che Chang, Che-Rung Lee, and Li Su. Play as you like: Timbre-enhanced multi-modal music style transfer. In *Proceedings of the aaai conference on artificial intelligence*, volume 33, pages 1061–1068, 2019.

[19] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[20] Omar Peracha. Js fake chorales: a synthetic dataset of polyphonic music with human annotation. *arXiv preprint arXiv:2107.10388*, 2021.

[21] Adhika Sigit Ramanto and Nur Ulfa Maulidevi. Markov chain based procedural music generator with user chosen mood compatibility. *International Journal of Asia Digital Art and Design*, 21:19–24, 2017.

[22] Ali Razavi, Aaron van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2, 2019.

[23] Adam Roberts, Jesse Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. A hierarchical latent vector model for learning long-term structure in music. In *International conference on machine learning*, pages 4364–4373. PMLR, 2018.

[24] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, June 2022.

[25] Duncan Williams, Victoria Hodge, Lina Gega, Damian Murphy, Peter Cowling, and Anders Drachen. Ai and automatic music generation for mindfulness. 01 2019.