IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 31 December 2024, accepted 15 January 2025, date of publication 20 January 2025, date of current version 29 January 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3531798

SURVEY

Music Generation Using Deep Learning and Generative AI: A Systematic Review

ROHAN MITRA, (Member, IEEE) AND IMRAN ZUALKERNAN, (Member, IEEE)
Computer Science and Engineering Department, American University of Sharjah, Sharjah, United Arab Emirates

Corresponding author: Imran Zualkernan (izualkernan@aus.edu)

This work was supported in part by the Open Access Program from the American University of Sharjah.

ABSTRACT This paper presents a systematic review of recent advances in music generation using deep learning techniques, categorizing the latest research in the field and identifying key contributions from various approaches. The study examines common data representations in music generation, including raw waveforms, spectrograms, and MIDI, alongside the most prominent deep learning architectures like Generative Adversarial Networks (GANs), Recurrent Neural Networks (RNNs), Variational Autoencoders (VAEs), and Transformer-based models. Through a comparative analysis, the paper highlights the strengths and limitations of these approaches. The findings suggest that GANs with spectrograms and RNNs with MIDI data are particularly effective for generating multi-track music, while autoregressive models like MusicGen and transformer models demonstrate superior performance in capturing long-term dependencies in music generation. Additionally, the paper underscores the emergence of diffusion models, which are gaining popularity for generating high-quality, complex music outputs. The major contribution of this review is the identification of the best-performing models for various music generation tasks and the provision of comprehensive insights into data representation methods, evaluation metrics, and future research directions.

INDEX TERMS Music generation, survey paper, GAN, LSTM, VAE, spectrograms, MIDI, RNNs, diffusion models, transformers, generative AI.

I. INTRODUCTION

Music has fascinated humans throughout time and new styles of music were developed as humans evolved and new technologies were developed. Generative Artificial Intelligence (AI) has emerged as a new field to automatically generate music using deep learning techniques. This paper introduces and examines the current state-of-the-art models in music generation using deep learning. Currently, there are no well-defined standards in music generation for widely accepted baselines, input formats used, or evaluation metrics. Papers like [1], [2], [3], and [4] have attempted to summarize the current advancements in the fields. However, each work has shortcomings. For example, [1] was first published in 2017 and updated in 2019, presenting older approaches to music generation. This is particularly problematic since the field of music generation with deep learning has evolved rapidly over the last 4 years, making this survey obsolete. Similarly, although [2] is comprehensive, the survey has a similar issue of being outdated, particularly with the rise of popularity and performance of transformer and diffusion models over the last 2 years. [3] reimplements a small subset of popular models and conducts comprehensive testing on the quality of music generated. Although this paper helps provide more standardized metrics and comparisons, it is far from a comprehensive survey paper on the topic. Lastly, [4] is a comprehensive survey, but does not include newer architectures like Diffusion, and does not provide a comparative analysis or attempt to segment the available literature into subfields and to explore each of them in detail. These shortcomings will be addressed in this paper.

The rest of the paper is organized as follows. The background of music generation is discussed next. This is followed by the survey methodology, followed by results of the survey.

The associate editor coordinating the review of this manuscript and approving it for publication was Jiachen Yang.

Subsequently, the discussion and comparison section, and ending with the conclusion of the survey and possible future work in the field of music generation using deep learning.

II. BACKGROUND

A. MUSIC THEORY

A list of common keywords and their descriptions relating to music theory is shown in Table 1. Table 1 shows the building blocks of any music including Note, Chords, etc. and some aspects of music quality including Harmony and Pitch.

TABLE 1. Keywords descriptions.

Keyword	Description
Note	A single sound represented by a single symbol
Chord	A set of notes, played at the same time
Bar	A single unit of time containing a given number of beats
Rhythm	A pattern of notes played
Melody	A linear succession of musical tones
Harmony	The process by which each sound is composed into whole units or compositions
Pitch	Property of sound based on their frequencies

B. MUSIC REPRESENTATIONS AND GENERATION TASKS

The three most common formats for generating music today are raw waveforms, spectrograms, and Musical Instrument Digital Interface (MIDI) [2]. Raw waveforms use the numerical values of the sound wave directly into a generative model [5]. Spectrograms are various types of intensity and frequency plots that lead to an image-like representation of the musical data [6] and are popular in audio recognition tasks. MIDI is a text-based representation of music. In addition to these commonly used representations of music there are other representations like PianoRoll [7] or ABC [8] representation not commonly seem in music generation.

TABLE 2. Music generation tasks per category.

Generation Category	Music Generation Task	Data Representation
Performance Generation	Single Instrument Generation	MIDI/Spectrograms
	Multi-Instrument Generation	Raw Waveforms
	Multi-Track Generation	MIDI
Score Generation	Score Generation	PianoRoll /MIDI
Audio Generation	Audio Generation	Spectrograms/Raw Waveforms/MIDI
	Song Continuation	Raw Audio

Table 2 shows a list of common music generation tasks, with the most used input and output data representations. Music generation can be done under several categories, the three most popular categories are: Score generation, performance generation, and audio generation. Table 2 shows the different types of music generation tasks that can be performed within each generation category and the common data representation method that is most used for each in the latest work done in that music generation task. As shown in Table 2, single instrument generation deals with models that are trained to generate music for a single instrument only (e.g., piano). Similarly, multi-instrument generation allows for an entire song to be generated with all its instruments (e.g., drums, piano, and guitar). Multi-track generation allows for generation of multiple tracks or songs that have the same overall theme. This could be thought of as songs that belong to the same album. Score and audio generation refer to generation of music scores like sheet music, which can be performed by a human, or generation of the final audio of the song respectively. Lastly, song continuation uses a part of an existing song and attempts to continue the rest of the song for the next few bars.

1) SCORE GENERATION

A music score is a special musical notation that can be used to represent a piece of music regardless of instrument. It contains notes, chords, and other information such as the key, musical expression, pitch, etc. This is the closest way to generating music to the way humans create music. This approach commonly uses the notes in a score to train a model to generate a music score that can then be performed by artists [9]. A popular way of doing this is based on the conversion of a music score to a Chromagram [1] which is a variation of a spectrogram which has been split into its individual notes. This helps convey both which notes are played, as well as their intensity.

2) PERFORMANCE GENERATION

Rather than generating a piece of music to be performed by musicians, deep learning can be used to directly generate a musical performance which can include single or multiple instruments. This approach goes one step further towards directly generating music to be played as opposed to score generation, but still uses intermediate representations of music like MIDI and ABC notation. MIDI is by far the most used notation due to its ability to easily be vectorized and this representation conveys more information than ABCs including music dynamics, timings, instruments, etc. Moreover, MIDI files can be more easily converted to higher quality music to be played back rather than ABC notation. Hence, the use of MIDI representation is most used by performance generation techniques.

3) AUDIO GENERATION

Audio Generation is the final tier of music generation that covers the generation of raw audio or spectrograms that can be played back without any processing. This approach allows all the information to be preserved (more than MIDI) since it uses the music (played in a studio) as input for training and can produce the same quality of music. The use of raw audio and spectrograms allows the data to truly represent the exact music and does not lose any information or expressiveness of the music as is the case with MIDI files that can still sound rather mechanical when played back. However, these

data representations lack an internal structure when compared with MIDI or music scores which can make it difficult for deep learning models to learn from.

There has been additional work done to move a step further from audio generation, to fusion, which generates the audio, lyrics, and video for a musical piece, but this work has not been covered in this paper. A similar conclusion holds for other music related tasks such as style transfer models and music inpainting. Although, some of the models explored in this paper can be adapted to perform style transfer, or music completion. More details about these other music generation tasks can be found in [2].

C. COMMON EVALUATION METRICS

One key issue in music generation is how to evaluate the quality of the music generated. In general, most papers in music generation use objective and subjective metrics to judge the quality of the output of a music generation task. Subjective metrics are usually based on a survey conducted with human participants, in which the music generated is played for each participant, and they are asked to rate the quality of the generated music using a set of metrics like realism, structure, and creativity. Some studies have compared the generated music to real music or generated music from other architectures, in which case the samples from each are randomly shuffled while presenting it to the surveyed.

Objective metrics are usually defined on a paper-by-paper basis, but most papers seem to agree to repurpose common image generation metrics like the Inception Score (IS) [10] and Frechet Inception Distance (FID) [11] used to evaluate vision-based generative models like the Generative Adversarial Networks (GANs). The Inception Score is based the Inception V3 model [12] which is a pre-trained image classification model and therefore perhaps not directly relevant for music generation. In the context of music generation, some approaches have built a genre classifier to simulate an image classifier and used that to generate a Pseudo–Inception Score [10]. This helps describe the distribution of the generated music. However, it has been widely agreed that the FID is more reliable, since it compares better the distribution of the generated data to a set of original data.

There have also been a set of other fairly used objective metrics such as the percentage of long notes in a song, and the range of pitches in a bar [13], etc. These metrics provide a better insight into the variance of generated notes and have been shown to be correlated to better human perception of music. Finally, overall, there is still a lack of standardized metrics, which is why many papers on music generation typically defined their own metrics.

D. TYPES OF GENERATIVE MODELS

The different types of generative models are described in this section. Additionally, Fig 1. summarizes the salient variations of each architecture, most of which are built upon in the music generation approaches discussed in this paper.

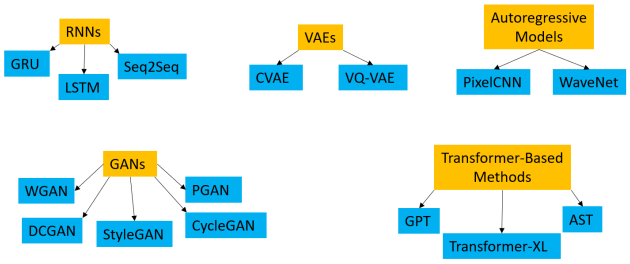

FIGURE 1. Salient variations of each architecture.

1) RECURRENT NEURAL NETWORKS

Recurrent Neural Networks are especially proficient at recognizing patterns in sequence data [14], and hence are highly applicable to music generation.

RNN Cells can take input sequences one element at a time and maintain an internal hidden state that considers relevant information from earlier time steps in the sequence. The hidden state is updated as each stage using the following equation (1) where x_t is the t^{th} element in the sequence and h_t is a hidden state.

$$h_t = f\,(x_t, h_{t-1}) \tag{1}$$

RNNs can be implemented using a recurrent layer, a Gated Recurrent Unit (GRU) or Long-Short Term Memory cell (LSTM).

RNNs aim to learn the parameters that maximize the likelihood of the target music sequences, hence using maximum likelihood estimation.

To generate new music, the RNN can be initialized with a seed sequence or a random symbol to start the song, and it can generate the subsequent symbols autoregressively. The generated symbol at each step becomes the input for the next step, and the process continues until the desired sequence length is reached.

2) AUTOREGRESSIVE MODELS

Autoregressive models were developed to predict time series data. They specialize in predicting probability of the next value in a sequence of values, assuming a fixed ordering for the sequence as shown in equation (2)

$$p\,(x) = \prod_{t=1}^{T} p(x_t|x_1, \ldots, x_{t-1}) \tag{2}$$

where p(x) is the probability distribution of the feature vector, and x_t is the t^{th} element in the sequence to be predicted, conditioned on all the previous elements in the sequence $x_1, \ldots, x_{t-1}$.

Any conditional distribution between the sequence of values is usually assumed to follow a Bernoulli distribution, to allow for the next value in the sequence to be based on all the previous values with different weights as shown in equation (3)

$$p_{\theta_i}(x_i|x_{<i}) = Bernoulli\,(f_i\,(x_1, \ldots, x_{i-1})) \tag{3}$$

where θ_i denotes the set of parameters used to specify the mean function $f_i : \{0, 1\}^{i-1} \rightarrow [0, 1]$.

These models often utilize deep learning architectures such as recurrent neural networks (RNNs) or transformers, but this approach has most popularly been applied with convolutional neural networks for music generation as will be described in the results section.

Generation of music using such an architecture can be achieved by simply beginning with a seed (a note or chord) and then generating the next symbol. The approach sequentially repeats this process using the sequence of symbols generated so far to predict the next symbol. However, this is very slow and expensive process for music generation which typically needs a sequence sampled at 44Hz or more. An approach to generate notes faster is described later in the paper.

3) VARIATIONAL AUTOENCODERS

A variational autoencoder is a generative deep learning model that uses an encoder-decoder architecture. In the context of music generation, the model can be trained by using the same music sequence as both input and output.

The encoder maps the music sequence to a probability distribution, which is usually Gaussian, hence learning the mean and standard deviation of the latent space distribution $z \sim N(\mu, \sigma)$.

The decoder performs the inverse mapping from the latent space back to the music sequence. The goal of the decoder is to map the latent vector back most accurately onto the music sequence provided as input during training.

During training, types of losses are used including reconstruction loss and regularization loss. The reconstruction loss is how different the generated output and the input sequences are and is usually measured using mean squared error or cross-entropy loss. On the other hand, the regularization loss is to ensure that the latent space that is generated has a prior distribution that is well behaved so that any latent vector can be sampled meaningfully.

To generate new music, a random vector ε from the latent space is chosen from a standard normal distribution and using the encoder's learnt μ and σ to map it to the latent space distribution using $z = \varepsilon * \mu + \sigma$. Now that there is a vector in the latent space, the decoder is used to convert that latent vector into a music sequence. The type of music generated will heavily depend on the learnt priors in the latent space based on what features it was able to capture from the input sequences during training.

4) NORMALIZING FLOW MODELS

Normalizing Flow Models are generative techniques that find several sequential bijective mappings that if applied sequentially, transform simple distribution (usually a Gaussian distribution) to more complex distributions (the dataset to be generated).

These models have a much more stable training process and converge much faster than using GANs or VAEs. This is because the model explicitly learns the data distribution $p(x)$ hence can use the log likelihood as its loss function.

The model learns a sequence of bijective functions $f_n(x)$ that transforms a point in the simple distribution, z_0, to a point in the dataset distribution, z_k. By sequentially applying the respective function as shown in equation (4)

$$z_i = f_i\left(z_{i-1}\right) \rightarrow z_{i-1} = f_i^{-1}\left(z_i\right) \tag{4}$$

where $z_{i-1} \sim p_{i-1}(z_{i-1})$ where $p_i(x)$ is the probability distribution after the i^{th} transformation.

Equation (5) shows the predicted probabilities.

$$p_i\left(z_i\right) = p_{i-1}(f_i^{-1}\left(z_i\right))|det\left[\left(df_i^{-1}\right)/\left(dz_i\right)\right]| \tag{5}$$

The transformations undertaken by the initial value of z through the entire sequence of functions f_i is called a normalizing flow.

Music generated from such models is a sequence of continuous vectors which may need some post processing to convert into discrete, playable representations like MIDI.

There are two types of such autoregressive flow models, Masked Autoregressive Flow (MAF) and Inverse Autoregressive Flow (IAF) which replace the forward (respectively backward) mapping with an autoregressive model. A popular autoregressive model in music generation called WaveNet, which will be explored in this paper, combines the use of IAF and MAF models to generate music.

5) GENERATIVE ADVERSARIAL NETWORKS

Generative Adversarial Networks consist of a generator and a discriminator that are trained in an adversarial fashion.

The generator is represented by a function $G(z, \theta_G)$, where z is a random noise vector and θ_G denotes the parameters of the generator.

The discriminator is represented by a function $D(x, \theta_D)$, where x represents a music sample, and θ_D denotes the parameters of the discriminator.

The generator hence aims to create realistic music samples $G(z)$ from random noise z, while the discriminator estimates a probability $D(x)$ to distinguish a real music sample from the generated music sample.

The entire model is trained using a combined loss function shown in equation (6)

$$\begin{aligned} min_G max_D\left(V\left(D, G\right)\right) = E\left[x \sim p_{data}\left(x\right)\right]\left[\log D\left(x\right)\right] \\ + E[z \sim p_z(z)][log(1 - D(G(z)))] \end{aligned} \tag{6}$$

Hence, the generator tries to minimize the value, while the discriminator tries to maximize the value, which leads to adversarial training.

Since the generator has now been trained to generate realistic music samples, it can now be given any random noise vector and use it to generate a music sample.

6) TRANSFORMER-BASED METHODS

Transformers also consist of an encoder and decoder architecture, but utilization a special mechanism called self-attention.

The input music sequences are embedded into continuous vectors which are passed to the encoder. The encoder passes the input sequence through several self-attention layers. Each self-attention layer captures relationships between different positions within a music sequence. It assigns weights to each position based on its relevance to other positions in the sequence, capturing dependencies and patterns across the sequence.

The decoder has a similar structure to the encoder but applies attention to the output sequence of the encoder. During training, it is trained to generate the output sequence shifted once to allow for autoregressive generation.

The transformer is trained using a log-likelihood loss is shown in equation (7)

$$Loss = -\sum log(p(y_i|y_1, \ldots, y_{i-1}, x)) \tag{7}$$

The output is generated by autoregressively by recursively predicting the next symbol. Each discrete symbol is predicted using a SoftMax layer and picking the maximum probability of each note which forms a sequence of musical notes.

III. SURVEY METHODOLOGY

The search terms used included "Music Generation Using Deep Learning", "Music Generation Using MIDI and GANs", "Music Generation Using MIDI and RNNs", "Music Generation Using MIDI and autoencoders", "Music Generation Using MIDI and reinforcement learning", "Music Generation Using Spectrograms and GANs", "Music Generation Using raw waveforms and GANs", "Music Generation Using raw waveforms and RNNs", and other such combinations of the categorizations of the field. The databases searched included arXiv, Google Scholar, Springer, IEEE Xplore, ResearchGate, and ScienceDirect.

The following inclusion criteria were used:

- The paper was published after 2020.
- The paper was commonly referred to as baselines in newer papers.
- The paper introduced a completely novel approach that was not seen before.
- The paper was foundational to the development of newer approaches that further improved the original idea.

The following exclusion criteria was used:

- The paper did not have quantitative or qualitative metrics, and only encourage the reader to judge the quality themselves.
- The approaches that did not use deep learning.
- Methods that were only applied to context of other sequence generation tasks (e.g., speech generation)

Once the papers were filtered, the following research questions were explored.

- What are the most popular data representational methods used in the music generation?
- What are the best deep learning architectures used for music generation?
- What are the different music generation tasks that have been explored?
- What are the most popular evaluation metrics for music generation?

IV. SURVEY RESULTS

Table 3 below shows the various models of music generation with respect to the architecture and the music representation. Each cell indicates the number of papers reviewed in that category. The categories identified in Table 3 will be used to discuss the papers included in the survey. There are also additional papers which do not directly fall into any one category.

TABLE 3. The categories of different deep learning methods for music generation.

Architecture	MIDI Data	Spectrograms	Raw Waveforms
AutoRegressive	-	-	3
GANs	3	3	2
RNNs	3	1	2
VAEs	1	-	1
Deep Reinforcement Learning	2	-	-
Diffusion Models	3	2	2
Transformer Models	-	-	2

A. AUTOREGRESSIVE MODELS

1) WAVENET

WaveNet was introduced by van den Oord et al. [15] as a novel raw audio generation model inspired by PixelCNN [16]. Like PixelCNN, WaveNet is an autoregressive model which means that it estimates the probability of raw signal at a particular time based on the sequence of signals that came at earlier timesteps. The model used causal convolutions to ensure that the estimated value of a signal does not depend on proceeding signals. To deal with the time complexity of applying causal convolutions, WaveNet introduced the concept of dilated convolutions to reduce computational complexity. WaveNet used SoftMax to model the conditional distributions. Since each raw audio signal is typically 16 bits and can have 65,536 probability values, WavNet used non-linear quantization to reduce the raw signal down to 256 values. For prediction, each sample generated was fed back into the model in an autoregressive manner to generate the subsequent sample recursively. Residual and skip connections were used to speed up convergence during training. Finally, the model can be conditioned by using binary representation of tags as input while training the model.

For the music generation task, Wavenet was trained on MagnaTagATune [17] with 200 hours of music audio and

a YouTube Piano dataset consisting of 60 hours of solo piano. Enlarging the receptive field was crucial to make the model generate more musical samples and the models did not enforce any long-term consistency and changed genres, instrumentation, volume, and sound quality every second despite using a few second long receptive fields. However, using conditional tags, WaveNet seems to be able to generate specific genre, etc. by encoding the conditioning. Lack of quantitative results on the music generation makes it difficult to easily compare WaveNet with other models.

2) MUSICGEN

Copet et al. [18] introduced MusicGen, a single-stage language model-based approach to generate high quality music. The authors use a codebook interleaving strategy to train the model which leads to efficient generation of music. The paper introduces a single model that can generate music unconditionally, based on text conditioning, and based on melody conditioning.

In essence, MusicGen is an autoregressive transformer-based decoder. The model has three primary components: Audio tokenization, codebook interleaving patterns, and conditioning. Audio tokenization is done using EnCodec [19], which is a convolutional autoencoder with a Residual Vector Quantization [20] latent space. This encodes an audio sample into a continuous vector which is then quantized based on the number of codebooks used and their size. This generates multiple codebooks, which need to be processed in parallel. This is done using codebook interleaving patterns. The interleaving patterns explored in this paper included exact flattened autoregressive decomposition, inexact autoregressive decomposition, and some arbitrary codebook interleaving patterns. The model was conditioned over both text and melody using a tokenized version of the text description or a chromogram of the input melody respectively.

The authors trained 300M, 1.5B, and 3.3B parameter models. They use the ''delay'' codebook interleaving pattern which converts 30 seconds of audio into 1500 autoregressive steps. The models are trained on custom datasets adapted from ShutterStock and Pond5 data collections and evaluated on MusicCaps dataset. The results are compared to Riffusion [21] and Mousai [22] which were retrained on the same dataset, and MusicLM [23] and Noise2Music [24] when possible. The authors use both quantitative and qualitative evaluation metrics. They use the Fréchet Audio Distance (FAD), the Kullback-Leiber Divergence (KL) and the CLAP score (CLAP) for quantitative evaluation and conduct a survey of 5 people for qualitative evaluation where the surveyed rated the music based on overall quality and relevance to text input.

MusicGen performed the best in terms of KL divergence, CLAP, and survey ratings compared to all the baselines. The model performed slightly worse than Noise2Music with regards to the FAD. Using conditioning of both text and melody lead to the best listener ratings. The authors also perform an ablation study based on the different codebook interleaving techniques and model scale to determine the best model.

Overall, MusicGen was able to introduce a new autoregressive approach based on a transformer model that can be conditioned with a mix of text and melody input to generate high quality samples. Moreover, despite using nonstandard metrics, they did provide a comprehensive comparison and even provide samples for readers to listen to from the proposed model and the baselines. Finally, this method does not allow more detailed control over the conditioning provided like other models that are explored later in this survey paper.

3) MUSICLM

Agostinelli et al. [23] introduced a novel approach to music generation named MusicLM, which uses a hierarchical sequence-to-sequence autoregressive model using a decoder-only Transformer. This approach used a special form to represent audio by combining three models: SoundStream [20], w2v-BERT [25], and MuLan [26]. SoundStream provides acoustic tokens, while w2vBERT provides semantic tokens, and MuLan's music embedding is used to represent the conditioning input. These input forms are combined with AudioLM [27] to generate music using two stages, semantic modelling and acoustic modelling.

The authors introduced a new dataset, MusicCaps with music and captions for each song to facilitate other research in the field. The authors use the Frechet Audio Distance (FAD), KL Divergence (KLD), and the MuLan Cycle Consistency (MCC) as metrics for the quantitative evaluation. The authors surveyed a group of people to only rate the closeness of the produced music to the prompt provided, and not on the quality or any other aspect of the audio. The authors compared the model to Mubert [28] and Riffusion [21]. MusicLM significantly outperformed user choice with more than twice the number of votes as the other models. Similarly, the model achieved better scores than both models in quantitative testing. The authors also performed testing to determine the usefulness of the semantic and acoustic tokens individually, as well as how much the model memorized from the given context. The authors concluded that both tokens were significant in improving the generated music and less than 1% of the generated music matched the context given.

Overall, the authors presented a unique way to represent the raw audio as a preprocessing step which was able to achieve better music generation for over 5-minute-long clips. However, the lack of human evaluation of quality of the music does not address the final goal of music generation that being for the music to be pleasurable to humans.

B. GANS WITH MIDI DATA

Using GANs to generate MIDI data is one of the most popular forms of music generation using deep learning. In particular, MIDINet [29] has been used as a baseline for a significant amount of research with MIDI data since its

publication. Another popular model often used as a baseline is MuseGAN [13]. These two GANs have inspired the development of A-Muze-Net [30] and MIDIPGAN [31] that successfully built on the previous models.

1) MIDINET

MIDINet [29] used a GAN with an input of MIDI strings to generated a series of MIDI notes and chords that represent a song. This model used a Deep Convolutional GAN architecture along with the use of a Conditioner CNN for both the generator and discriminator. This Conditioner CNN [32] uses a vector to encode knowledge from the previous runs and add that to multiple layers throughout the generator and discriminator. This scheme mimics the idea of a memory-based model like a RNN [33]. MIDINet converts MIDI representations into a matrix dimension consisting of number of notes per bar x number of timesteps per bar. This matrix is used as input to the model. The conditioning is applied based on the input from the previous bar to improve continuity and to ensure that subsequent bars follow the same pattern and melody.

A custom training dataset of pop music in MIDI format containing both the notes and chords was used. MIDINet evaluated two models, one to only generate melody (notes), and one to generate both melody and chords. A survey was conducted where people were asked to rate the generated music against MelodyRNN [34], and the survey takers ranked MIDINet better overall.

The stated advantage of this model was creativity (with lesser conditioning applied), and stability (with more conditioning, but a larger change of overfitting). Moreover, this model examined the data sequentially, minimizing the amount of memory needed to train and to make predictions. However, the paper does not provide of quantitative results for comparing to other models. Finally, the model could not distinguish between one long note and two short, repeated notes, due to the representation of MIDI as the matrix encoding used in MIDINet.

2) MUSEGAN

MuseGAN [13] is a novel GAN architecture that can perform multi-track sequence generation using MIDI input. MuseGAN used two basic ways to model multi-track interdependency and the temporal structure of music. Multi-track interdependency, as in Fig 2, is to ensure that the samples produced by the model stays consistent in each track, in terms of overall theme, whereas the temporal structure in each track is to ensure that each sample generated is not erratic in each bar, and that each bar is dependent on the previous bar. To deal with multi-track interdependency, the authors use a GAN with various architectures, mainly changing the number of generators and discriminators to generate multiple audio samples as seen in the Fig1 below. Moreover, to accurately model the temporal dependency of music, the authors propose the use of conditional generation, or generation from scratch as seen in Fig 3.

As seen in Fig 4, the input to MuseGAN, is split into of four inter-track time-independent vectors similar to [35]. For each track, the shared and private temporal structure generators, take the time-dependent vectors as input, and each of them output a series of latent vectors containing inter-track and intra-track temporal information. The output is concatenated with the time dependent vectors and sent to the bar generator, which generates the sequence.

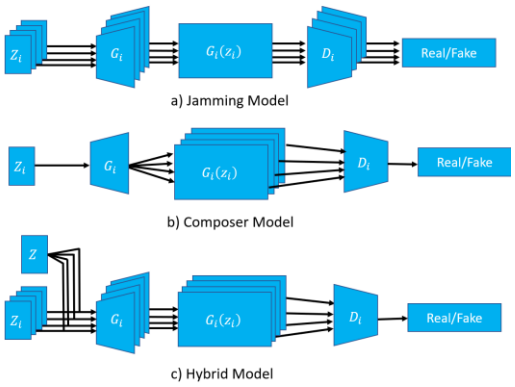

FIGURE 2. The three types of MuseGAN models.

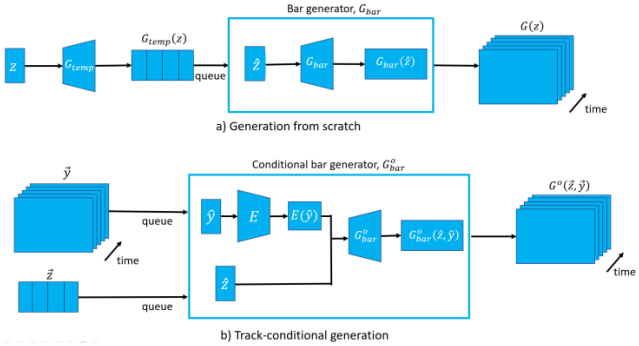

FIGURE 3. MuseGAN modelling temporal dependency.

The authors evaluated the model based on the ratio of empty bars generated, number of used pitch classes per bar, and the ratio of notes longer than three time-steps. Although these are rather uncommon metrics, they do provide an insight into the generated music. Overall, they concluded that using track-conditional generation with the Jamming model performed the best, with the least number of empty bars, the greatest number of used pitch classes and the highest percentage of long notes. Additionally, they conducted a survey of 144 people of varying backgrounds in music expertise to rate the generated music based on multiple metrics. They concluded that the hybrid model is preferred by all. However, for conditional generation, the pros preferred from scratch, while the non-pros preferred the jamming model.

Overall, the model performs very well and addresses both the issue of overall consistency across multiple track generation, as well as intra-track dependency to ensure the music generated follows a consistent theme. However, the metrics

used for evaluation, although useful to tell details about the tracks themselves, are non-standard, making it difficult to compare it to other music generation models, apart from the survey conducted at the end of the study.

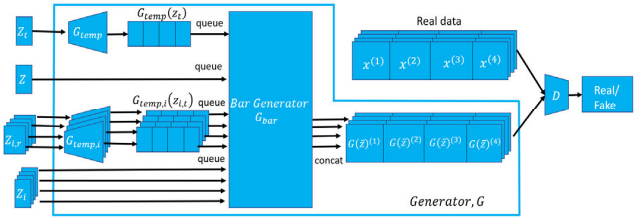

FIGURE 4. MuseGAN architecture.

3) MIDIPGAN

MidiPGAN [31], is the first paper to apply the Progressive GAN architecture [36], to music generation. The authors propose a method to convert MIDI input into the required format to apply a Progressive GAN, along with devising a suitable architecture for the PGAN to generate music.

As in Fig 5 below, the architecture of MidiPGAN is split into three stages, gathering and preparing input data, designing a model to learn to generate samples like this data, and a training process to enable the learning. As in Fig 5, at each stage, the data, both real and generated, in different resolutions are fed to the discriminator. This is done in accordance with PGAN [36].

For evaluation, the authors describe how the Inception Score was insufficient [37] and not suited to the problem at hand and propose using the Frechet Inception Distance (FID) instead [38], [39]. They reported an FID of 12.30, but did not compare it to any baselines, or any other models. Hence, it is hard to make sense of this score alone. However, the authors conducted a survey of 34 people, asking them to rate the Pleasingness, Creativity, Structure, and Realism. The authors also compare the results to similar works such as [13] and [29], but only considering a subset of other results obtained, since the surveys conducted weren't directly comparable. Overall, the people experienced in music rated MidiPGAN higher than inexperienced people, which is the opposite of the other baselines. Moreover, the creativity and structure were rated like the baselines, but MidiPGAN samples were rated lesser in realism than MIDINet.

In conclusion, this method performed well, but did not beat the baselines by a large amount. However, the paper did introduce a new approach that may be expanded upon and combined with other methods mentioned earlier to introduce the use of the Progressive GAN architecture. Moreover, MidiPGAN provided a thorough analysis of their survey results, which indicated a decent performance overall.

C. GANS WITH SPECTROGRAMS

GANs are commonly used with images, hence it makes intuitive sense to convert music generation into an image generation problem by representing music as an image. Hence, the use of spectrograms to represent image data based on its intensity and frequency has become popular. This section introduces three popular methods of using Spectrograms with GANs for music generation: Novel SpecGAN and MP3Net and using transfer learning of popular GANs such as StyleGAN.

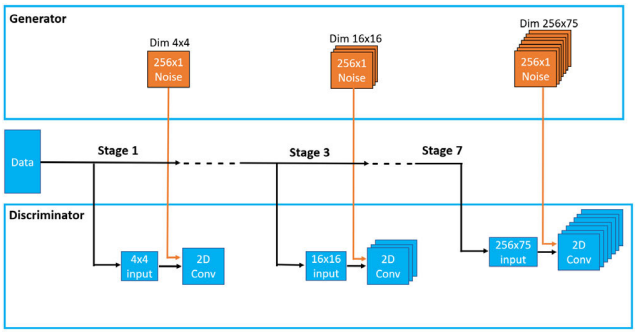

FIGURE 5. MidiPGAN architecture.

1) SPECGAN

SpecGAN [40] comes from a paper that introduced both SpecGAN and WaveGAN. SpecGAN is an approach to generate music using a GAN with spectrograms as input. Since spectrograms are essentially images, SpecGAN chose to employ a DCGAN architecture [41]. The algorithm involves converting the audio to images (spectrograms), which the GAN learns from, and then predicts. These generated spectrograms are then converted into audio for playback. SpecGAN has been evaluated on speech generation, drums, piano and bird sounds, all as one second clips.

SpecGAN was able to generate spoken digits 0-9, that was identifiable by humans during the survey they conducted. However, the authors did note that the other architecture proposed, WaveGAN, had higher quality audio. The authors used the Inception score and Nearest neighbor's metrics, with values of 3.71 and 0.8 respectively, which were both the lowest across all proposed methods and their variations. The low Inception Score indicates that the intra-dataset diversity is low, and the test data is not close to the train data, indicating poor performance. However, 66% of survey participants were able to identify the spoken digits correctly, despite giving it the lowest quality score of 1.9/5.

This method made use of well-known image generation techniques like DCGAN, by converting the audio to images. Moreover, this method provided a good overall score with human judges and converged quickly during training. However, inversion to audio takes time and is not very accurate [42]. Additionally, the amount of post and preprocessing introduces more uncertainty and increases the time complexity of the model. Furthermore, spectrograms can get very large, and hence using a Mel-Spectrogram might be better but may make inversion to audio more difficult.

2) MP3NET

MP3Net [43] uses a DCGAN architecture, with Modified Discrete Cosine Transform (MDCT) input. Each audio sample is passed through a MDCT transform to convert it to an image, which is then fed through a DCGAN where each subsequent layer in the generator increases the resolution along the time axis and adds a higher octave along the frequency axis. Since it uses the MDCT which stores all the in-phase data, phase generation is an integral part of the model.

MP3Net also used a Progressive GAN approach [36], but adapts it to fit with a WGAN [44] by adding a gradient penalty. Both real and data is passed through a ''psychoacoustic'' layer where gaussian noise is added.

The authors used 95 second and 5 second samples of 200+ hours of piano music. They rank the results against Jukebox [45], and although they do not provide any actual numerical values, they do claim to be orders of magnitude faster during training.

The 5s model had better piano timbre than the 95 second model, because of the additional feature depth of its shallow layers. The authors did notice a certain humming noise being generated as in [46], and used the suggestion from [46] to solve the issue, but saw a lack of variety of music being produced. The authors noted good, consistent tonality, and noted that the audio generated resembled harmonies of western music it was trained on, but the authors do not provide any quantitative results, compare to baselines, or even conduct a survey to mark the success of the algorithm.

3) TRANSFER LEARNING OF GANS

Papers like [47] and [48] explored the use of transfer learning of GANs, using popular GANs such as StyleGAN [49], StyleGAN2 [49], UNAGAN [50], etc. on audio datasets for music generation. They represent the audio as spectrograms, converting it into an image generation problem, and then employ transfer learning to generate the required images.

By combining the FreeSound Loop and Looperman datasets, the authors of [47] were able to create a large dataset of only drum music, used to benchmarks the three transfer learning models.

The authors use the Inception Score (IS), Fréchet Audio Distance (FAD), number of statistically different bins (NDB), and Jensen-Shannon divergence (JSD) as proposed in [51]. After testing, StyleGAN 2 performed the best, with an IS of 5.24±1.84, a FAD of 7.91, NDB of 0.09, and JSD of 0.59, beating StyleGAN and UNAGAN on the FreeSound dataset. However, UNAGAN did achieve an even lower FAD of 4.32, indicating that UNAGAN had generated audio closer to the input distribution.

D. GANS WITH RAW WAVEFORMS

To eliminate the preprocessing required to convert to and from a spectrogram, using the raw waveform directly became desirable. This section explores the two most prominent methods using GANs with raw waveforms: WaveGAN and Self Attention GAN.

1) WAVEGAN

WaveGAN [40], is from the same paper as SpecGAN in Section 3.2.1. It used raw waveforms as input, and adopted the principles of DCGAN in its construction. It used the same principles of DCGAN with 1D convolutions instead of 2D convolutions, to keep track of the sequence of frequencies as input. The exact set of modifications made is listed in [40].

Trained and tested the same way SpecGAN was, WaveGAN generates digits 0-9 that are intelligible to humans, and has superior quality to SpecGAN. Moreover, it achieved the best score amongst all variations proposed in the paper, with an inception score of 4.67 ± 0.01, and nearest neighbor distance of 0.8 to the training data. However, only 58% of the numbers spoken were identified correctly, but it received the highest quality score of 2.3/5. Hence, overall, the model did perform better, quantitatively, but still had issues generating numbers spoken correctly.

2) SELF-ATTENTION GANS

Tomaz Neves et al. [52] adopt the style of [53] and mimic a Self-Attention GAN (SAGAN) for audio generation, capable of fast and fully parallel generation. This is because they did not use an auto-regressive method in generation, improving speed of the generation process. Moreover, the GAN proposed used instrument conditioning which allows the user to choose which instrument the music sounds like it came from.

The SAGAN was trained on 21572 3-second clips of 8 different instruments. The authors also noted that data augmentation is useful in such training, as outlined in [54], and hence trained SAGAN with and without data augmentation, with the version with data augmentation achieving the best performance. The authors used the Inception Score (IS) and the Frechet Inception Distance (FID) to provide quantitative metrics, but since the problem was not an image recognition problem, they trained a simple convolutional network to classify the audio asper the 8 instruments, to use as a backbone to calculate the Inception Score. This CNN had a similar architecture as the discriminator and achieved 69% accuracy on the test set.

Overall, as shown in Fig 5, we notice the Inception score increasing over 3.5 through the epochs, and the FID decreasing to less than 100, in Fig7 below. We see SAGAN compared to WaveGAN, reaching a final IS of 3.76, only 0.7 less than WaveGAN's IS of 3.83, and an FID of 45.85 which is half of the FID of the WaveGAN (86.6).

The decrease in FID suggests that SAGAN was able to create music samples very similar to the training set. Moreover, the IS of 3.76 and 3.83 above may not be very accurate because of the accuracy of the CNN used as the backbone. Hence, with SAGAN's improved speed and lesser FID, we may conclude that it performs better than WaveGAN overall.

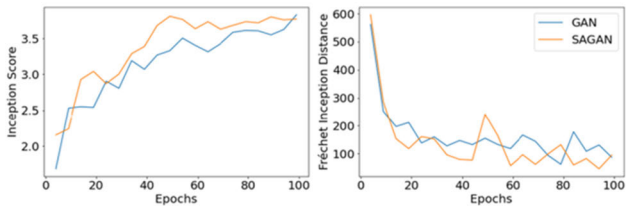

FIGURE 6. Comparison of SAGAN vs baseline.

E. RNN'S WITH MIDI DATA

Since it is essential for an entire piece of music to be coherent, models such as RNNs and their subsets, LSTMs, became a suitable alternative to GANs. This section describes three RNN based methods on MIDI data: MelodyRNN, A-Muze-Net, and LSTM Models that use MIDI input.

1) MELODYRNN

MelodyRNN [22] is a combination of Google's Magenta [55] and DeepMind's WaveNet. It used LSTM cells to predict music as a sequence of MIDI data represented as strings. MelodyRNN has high quality music generation via three types of MelodyRNN: Basic, LookBack and Attention. Each of the 3 types only differ by the input format. The Basic type represented the input by one-hot encoding, Lookback allows the model to remember the last two bars by using custom labels, and Attention uses a technique called ''attention'' to allow the model to easily access information already seen without referring to a previous RNN cell state. Regardless of the type of MelodyRNN, they all use MIDI input transformed differently.

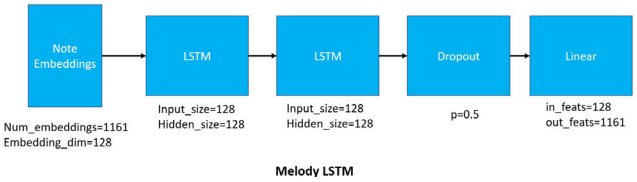

FIGURE 7. Right-hand (melody) LSTM model.

MelodyRNN was trained on a human voice dataset, and a piano music dataset with varying lengths from 30 seconds to 1-minute-long clips that were cropped to an unspecified fixed size. The authors only had 200 samples which are clearly not enough to train any neural network-based model, which the authors recognize.

The authors did not provide any quantitative results or conduct any survey, except compared the quality of the generated audio to Biaxial RNN [56] and WaveNet, by ear. This led to a very subjective conclusion that MelodyRNN had a decent output. Although, they provided a link to a website where readers can listen to the music themselves, this still does not allow us to quantitatively compare the generated audio to other baselines. An important fact to note is that the authors noticed that the music generated that were over 16s long, had repeated cycles of melodies.

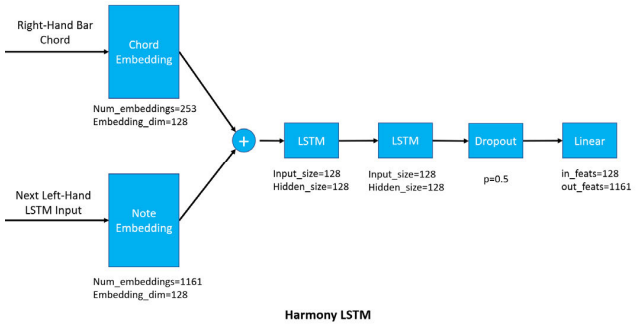

FIGURE 8. Left-hand (Harmony) LSTM model.

2) A-MUZE-NET

A-Muze-Net (AMN) [30] used a basic idea from piano theory, that the right hand of a pianist follows the Treble Clef, while the left hand follows the Bass Clef. This means that the right hand plays the melody, while the left hand plays the harmony. Inspired by [56] and [57], the authors exploit this idea by training two different models, one for each hand, and conditioning the model for the left hand, based on the right hand. This leads to the harmony being generated based on the melody, improving overall coherence of the generated audio. AMN uses the LSTM architecture for both models, as well as basic MIDI input.

The right-hand LSTM model (Melody LSTM) as in Fig 7 below, consisted of the MIDI embeddings as input, through two LSTM layers, producing probabilities for the next element in the sequence. On the other hand, the left-hand LSTM model (Harmony LSTM) as in Fig 8, takes in two inputs, the Right-hand LSTM Bar's Chord, and the next Left-hand LSTM input, combining them using a special embedding based on a method of deconstructing chords into its root chords, similar to finding the root of a word in NLP.

A-Muze-Net uses MuseGAN as their baseline, comparing it to using their right- and left-hand models independently, and then using them as described above. The authors use similar metrics to MuseGAN: The percentage of notes over a given length, the average amount of different pitches per bar, how much two tracks are aligned, and the percentage of notes that were out of scale. AMN outperforms MuseGAN with 100% notes above the min length, five more different pitches per bar, and a lower tonal distance. Additionally, they conducted a survey of 17 people, to rate their satisfaction of the songs from 1 to 5, with AMN achieving the highest score of 3.28 out of 5, compared to MuseGAN's 3.16.

Overall, AMN used a clever technique to improve the quality of generated music, and significantly beats the baseline MuseGAN using basic music theory to their advantage. However, this method is only suitable for the piano, and cannot be easily adapted to other instruments that do not clearly separate the harmony and melody at the same time

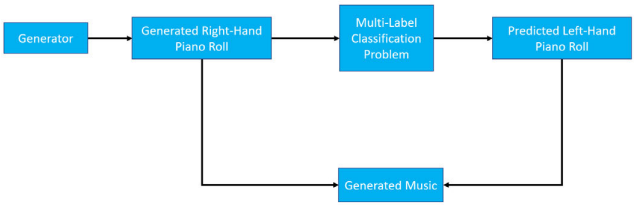

FIGURE 9. Generating the left-hand part by an ANN.

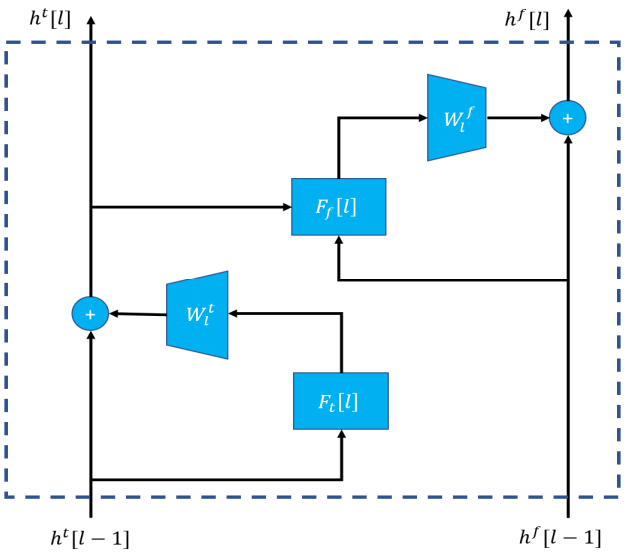

FIGURE 10. Mel-Net architecture. t indicates temporal, f indicates frequency based.

(e.g., one guitar can only play either one at any given time and requires both hands to play).

3) LSTM-BASED MIDI MODELS

Lyu et al. [58], examined different LSTM based architectures to generate dual-track piano music. They model the inter-dependency of the left and right hand, similar to [30], and compare their results to MuseGAN [13]. The authors explore using a Simple LSTM Encoder-Decoder Model, an Attention-Based-LSTM Encoder-Decoder Model, and a CNN+Attention-Based-LSTM Encoder-Decoder Model.

The authors model each track as played by either hand of a pianist, as seen in Fig 9. The generator generates the music for the right hand (melody) and then they pass that into a multi-layer perceptron (MLP) modelling it as a multi-label classification problem, to generate the corresponding left-hand harmony. The melody and harmony (right and left hands) are combined to give the final output.

The dataset used was a dataset of 8 different types of piano music, with over 100 samples. Using similar metrics as above, the authors use the number of used pitch classes per bar, the percentage of long notes, and User study ratings, where 10 people ranked the music on a scale of 1-5 of how natural it is. Overall, the Embedding Attention-LSTM Encoder-Decoder had the greatest number of pitches per bar (7.79) compared to 4.57 of MusGAN, indicating good variability, whereas the Pianoroll CNN+Attention-LSTM Encoder-Decoder had the most percentage of long notes (91.2%) while MuseGAN had only 64%. However, MuseGAN still had the highest score by survey takers, rating it 3.16 out of 5 on being natural, while both the proposed models were rated 2.4 and 2.7 respectively.

F. RNN'S WITH SPECTROGRAMS

Since RNNs work well with MIDI input, attempts have been made to use the ability of RNNs to capture long term dependency with spectrograms. We explore a popular model, Mel-Net in this category.

1) MEL-NET

Vasquez and Lewis [59] introduced a novel RNN based architecture that uses spectrograms to represent the input audio. The authors' primary goal was to introduce a model that can capture both local and global structure of any given audio while generating new audio.

Mel-Net used two types of stacks: Time Delayed Stack and Frequency Delayed Stack, similar to [29]. The Time Delayed Stack contains information summarized from all the previous spectrograms, while the Frequency Delayed Stack uses the preceding data obtained from the current spectrogram, while also using information obtained from the corresponding Time Delayed Stack to produce the next frame. This interdependency of the two stacks is shown in Fig 10 where the output of the current layer of the Time Delayed Stack is used as part of the Frequency Delayed Stack. Overall, Mel-Net uses three Time Delayed Stacks and one Frequency Delayed Stack, leading to a total of four RNNs in parallel.

The proposed method is tested on both music and speech generation, trained on the VoxCeleb2 and MAESTRO datasets respectively. Mel-Net is compared to WaveNet as a baseline for speech synthesis and music generation and compared to Wave2Midi2Wave for music generation. The metric used in the study is human evaluation, by asking the survyees to determine which audio clip had better long-term structure. Mel-Net outperformed WaveNet in both datasets with over 95% of people picking Mel-Net to have better long-term structure. Moreover, Mel-Net outperformed Wave2Midi2Wave in music generation, with 62.5% of survyees picking Mel-Net to have better long-term structure.

Overall, Mel-Net could capture long term dependency well. However, not much is said about the quality of the generated audio, and no other quantitative data has been provided to further compare the results to other models. Regardless, Mel-Net could achieve the goal of the study, to capture long term dependency and be used for a variety of audio generation tasks, not limited to just music generation.

G. RNN'S WITH RAW WAVEFORMS

With the success of RNNs with MIDI and Spectrograms as input, attempts to use raw waveforms directly to generalize its application from just music generation to audio

generation became desired. Hence, this section explores the use of LSTMs and RNNs in general with raw waveform input as a method of generating music.

1) LSTM MODELS

Inspired by work in [60] and [61], Kalingeri and Grandhe [62] explore the use of different LSTM based architectures on raw waveforms as input. The authors consider 5 different LSTM based architectures: Only LSTM Cells, Fully Connected LSTM, 2D Convolutional LSTM, 1D Convolutional LSTM, and Bilinear LSTM.

The first model uses only LSTM Cells between the input and output layers, the second model adds a fully connected layer with dropout before the LSTM cells, the convolutional models use 1D and 2D convolutions respectively, with pooling, before adding LSTM cells, while the Bilinear LSTM model uses 2 groups of LSTM layers, either in parallel or in series after a fully connected layer.

The models were all trained on 25 piano tunes, for 2000 epochs. The authors conducted a survey of 10 people to rate the music out of 5, concluding the best models to be: Bilinear LSTM (3.6/5) and 2D Convolutional LSTM (3.4/5).

This structure does not use any computationally expensive adversarial training, and only needs basic preprocessing, performing fairly well to humans, in fact better than most models examined so far. However, the lack of other quantitative results makes it hard to accurately compare the results with other models. On the contrary, an important point to note is that the approach here does not use the musical structure of the music in any way, which allowed the network to capture the overall structure and coherence of the music.

2) OTHER RNN MODELS

In music generation using RNNs, a persistent issue seems to be the lack of a global structure and theme to the generated audio. This is caused by the fact that most RNNs implemented only keep track of the previous note or bar, but this leads it to lack an overall structure that would be expected in music.

Using data on 12-bar blues from jazz musicians, papers like [63] attempted multiple tests to use LSTMs to 'look at the big picture' and capture the global structure of music. Eck and Schmidhuber [63] observed that LSTM models were able to learn and generate chords very easily, even when allowed to generate multiple chords at once, it was able to learn the theme of the blues relatively easily. Next, Eck and Schmidhuber trained the LSTM models to learn both melody and chords, where it was restricted to only one note per timestep but allowed multiple chords. The LSTM structure was able to learn the blues, learning and successfully replicating both the local and global structure of the song.

Similarly, papers like [64] explored other methods of representing the MIDI data to allow the models to capture the global theme. Sturm et al. tried using higher level notation known as ABC notation to generate new transcripts. The authors evaluate and discuss the results of the generated population, results of each sample, and its ability to aid in music composition. The authors use the ABC notation and attempt to train a model to predict each character, and then define a token that could be one or more characters and trains another model to predict them. The idea is that grouping characters together will force the model to focus on a larger group of tokens being placed together correctly, rather than worrying about each note individually. This forces the model to capture the theme of the music at a higher level.

H. VAE'S WITH MIDI DATA

Like GANs, Variational Auto-Encoders were used for music generation as well. In particular, the most popular of which is the one presented below that allowed for the most flexibility, by adjusting tonal tension of the generated music.

1) ADJUSTING TONAL TENSION

Guo et al. [65] investigated allowing more control over the generated music. In particular, Guo et al. usd Variational Auto Encoders to allow the user to adjust tonal tension, by incorporating two tonal measures based on the Spiral Array Tension theory [66], [67] to the proposed VAE. The proposed VAE could generate rhythm based on a seed value given to the model and adjust its pitch to be in line with the tonal tension specified by the user.

Guo et al. focussed on two primary methods of measuring tonal tension as in [68]: Cloud Diameter, and Tensile Strain. The Cloud Diameter represents the range of pitches in a single window or ''cloud'', while the Tensile Strain is measured as the distance between the geometric position of the key of the audio, and the geometric gravity point of all the pitches in the cloud.

The architecture of the VAE is depicted in Fig 11, illustrating the use of GRUs in both the encoder and decoder. A key point to note is that the decoder predicts the pitch and rhythm of the melody and bass separately, as well as the cloud diameter and tensile strain.

The VAE was trained on MIDI files of over 2000 pop songs, mostly Chinese pop songs. The results are evaluated with different models, using only tensile strain, only cloud diameter, or only beat positioning, as well as an amalgamation of them all. Overall, the model with the cloud diameter only, and the model with the combination of all the metrics performed the best, beating the baseline.

I. VAE'S WITH RAW WAVEFORMS

1) AMAE

Dieleman et al. [69] introduced a novel method for music generation focused on capturing long term dependencies and focused on generating realistic and human-sounding music. The authors noted that using raw waveforms allowed for some noise to be captured in the music generated that makes the audio sound more natural and less computer generated. Additionally, the authors mentioned how the current auto-regressive models for music generation work well to

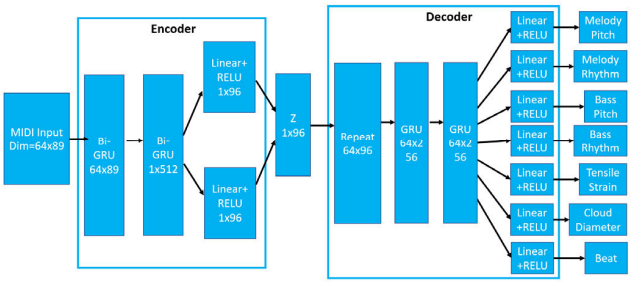

FIGURE 11. Architecture of VAE with tonal tension tuning.

capture local structure but still miss global coherence. Hence, they introduce a new architecture, the ArgMax Auto Encoder (AMAE) to tackle this issue.

The AMAE essentially includes up to three separate autoregressive models that were trained on different levels of abstraction, to allow the overall model to adhere to local and global consistency. Inspired by the efforts of WaveNet [15], the authors attempt to increase the size of the receptor fields by using autoregressive discrete autoencoders (ADAs), to help capture long term dependencies. Each autoregressive model is made into an autoencoder by attaching an encoder that learns the required conditioning based on the level of abstraction the authors decide. The autoregressive model is then used as a decoder in the autoencoder, taking input from the result of the encoder that modifies the input signal based on the learnt conditioning. Each of the autoregressive models mentioned use the WaveNet [15] architecture because it allows the user to define the size of the receptor fields.

The authors trained AMAE on 413 hours of piano music. The authors provided samples of generated audio for the readers to judge, but not providing many quantitative metrics. The only quantitative results provided were the negative log likelihood, comparing it to WaveNet and Vector quantisation variational autoencoders (VQ-VAE). AMAE performed much better than WaveNet (NLL of 0.695 and 1.151 respectively), while performing approximately the same as VQ-VAE (NLL of 0.682). Moreover, the authors also studied the predictability of the generated audio, noting that it is not very predictable locally, which provides a larger variety of sounds generated. The authors end by exploring using a different number of models within AMAE, as well as different ADA architectures.

Overall, the paper introduced a novel approach to music generation, which performed well, and converged quickly. Furthermore, providing the generated music to listen to allows for the reader to compare to other models by ear, and the negative log likelihood provides a somewhat comparable quantitative result. However, the lack of result details on the goals of the model – Natural sounding music capturing long term dependencies, does not provide much support to the authors claims. The use of a survey to rate these features would have helped give the readers a better view of the goals being achieved.

J. DEEP REINFORCEMENT LEARNING WITH MIDI DATA

With the idea of allowing the model to learn the principles behind music, and explore different possibilities, the method of using Deep Reinforcement Learning for music generation was explored. Since there is very little work done in this field, we explore the two most popular models, Bach2Bach, and RE-RL Tuner.

1) BACH2BACH

Kotecha [70] described the use of a deep reinforcement learning approach to train a probabilistic model such as the Bi-axial LSTM to generate music accurately. The author claim that such a learning technique encourages a greater global coherence and structure of the generated music and encourages the model to explore as it learns. Inspired by [64], Kotecha employs the Bi-axial LSTM model as the generator. Kotecha uses deep Q Learning [71]. It uses MIDI files of Bach's sheet music, as input, and allows Bi-axial LSTM to learn in the deep Q learning environment.

As in Fig 12, the weights learnt by the Bi-axial model were used to prime the weights for the 3 subnetworks used in the architecture – Target Q Network, Q Network, and the Reward function. Each of which are neural networks that provide the RL structure each of its necessary features – State, Action, Reward.

Overall, we see that the generative network – Bi-axial RNN performed better than or as good as the baseline Note RNN, while the results from the Q network were at par to its benchmark. The number of notes repeated were only 47.7%, with 55.7% of the melodies being unique and only 0.4% of notes not in the right key. Amidst other metrics, these salient metrics do indicate the music generated is of a decent quality. Moreover, the author has contacted a professional musician, who commented on the piece generated, but did not provide any direct metrics. The main comment was that the music lacked a deeper structure, and did not flow easily, but instead had frequent breaks, leading to rhythmic discontinuity.

Hence, even though this was a good attempt at approaching music generation as a RL problem, it did not perform as expected. However, the lack of comparison to any popular baseline or survey conducted does make it difficult to compare.

2) RE-RL TUNER

Liu et al. [72] had a similar idea of using RL to help teach the model a more globally coherent structure, but also tried to incorporate the ideas of music theory to help guide the model to better music generation. In particular, the authors split the data into its different scales and then applied the Latent Dirichlet Allocation (LDA) model [61] to determine possible underlying music principles.

As in Fig 13, we can see that the Q network and target Q network are DQN training tricks where the target Q network is a delayed copy of the Q network. Feature extraction is the LDA-based topic feature extractor and Reward is the

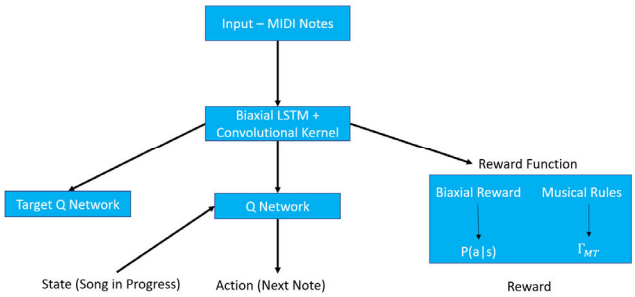

FIGURE 12. RL Method to train the network in Bach2Bach.

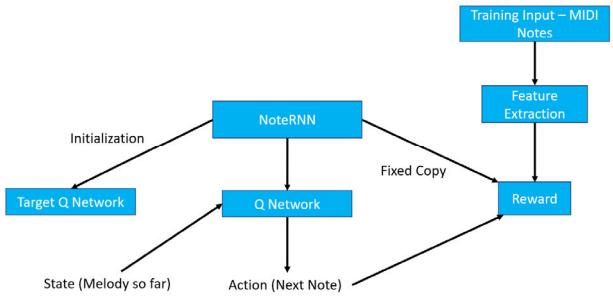

FIGURE 13. RL method for RE-RLTuner.

multi-scale modeling design. The reward is generated as in Fig 14, using a separate network to merge the results from LDA and one hot encoded vector of the subsequent note.

Training was done on the Nottingham dataset, with no further details provided. The model is trained using Note RNN as the LSTM model. The proposed model is then compared to RL Tuner and Re-RL Tuner. On average, the baseline LSTM model beat Re-RL Tuner, but not RL Tuner, based on the number of unique notes per bar and the average and maximum intervals of the bar. Re-RL Tuner achieved 2.90 average notes per bar, 17.5 being the maximum interval per bar with an average interval of 11 notes per bar. Moreover, the authors conducted a survey with 19 people, ranking LSTM model 3.21/5, RL Tuner as 3.16/5 and Re-RL Tuner as 3.58/5, successfully beating the baseline LSTM model.

K. COMBINING MIDI AND RAW WAVEFORMS

After examining the three kinds of input formats – MIDI, Spectrograms, and Raw Waveforms, we conclude that MIDI and Raw waveforms appear the most promising and are the most widely used methods across the different architectures. Manzelli et al. [73] noticed this trend and are working on a method to combine the two input methods.

MIDI input methods could capture long-term dependencies within the music, they still did not sound very realistic, while raw waveform input methods sound much more real, but lack structure and long-term coherence. Hence, an attempt at combining the benefits of either is needed. Manzelli et al. [73] present a novel method to do so, by building two models, one LSTM on MIDI and one on raw waveforms, but augmenting the latter to take as input the output of the LSTM model. The second model chosen is WaveNet [15], while the LSTM model is a biaxial LSTM. This is still a work in progress and is still awaiting an update.

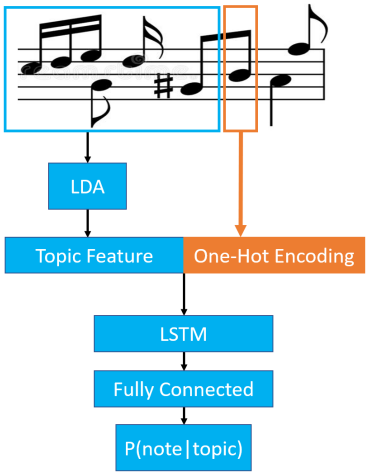

FIGURE 14. Reward network.

L. DIFFUSION MODELS WITH MIDI DATA

The use of diffusion models has become very popular lately in text and image generation. However, the applications of diffusion models in music generation is not widely explored, but is facing an increased interest lately. There are three popular papers in this field, based on Symbolic Music Generation, Simultaneous Music Generation and Separation, and DiffuseRoll.

1) SYMBOLIC MUSIC GENERATION

Mittal et al. [74] note the limitation of diffusion models to only work with continuous data, unlike MIDI data, and propose a method of converting the data into continuous values to use with a diffusion model.

The authors used the MusicVAE [75] autoencoder to obtain a mapping of the MIDI input in a latent space. This mapping is then passed through a transformer model to better capture the temporal information from the latent space mapping. This output from the transformer is then fed into the diffusion model trained with conditioning of noise from a sinusoidal wave. The authors explore the use of the proposed model with both unconditional generations, as well as infilling (predicting missing bars in the middle of a melody).

The model was trained on the Lakh MIDI Dataset with 99% of the data being used for training, and 1% being used for validation. The model was compared to an autoregressive transformer model with a mixture density output layer, made by the authors. The results of each model is also compared to a normal distribution, as well as an interpolation output. The metrics compared are the FID, the Maximum Mean Discrepancy (MMD), and the consistency and variance of the notes compared to the original song.

The diffusion model had a 99% pitch consistency, and 96% duration consistency for unconditional testing. Moreover, the model had a 97% pitch and duration consistency during infilling. The diffusion model significantly outperformed the baselines. Furthermore, the diffusion model obtains an FID of 0.0166, and MMD of 0.0018 which is very close to 0, indicating a very good performance. However, it is still comparable to the performance of the autoregressive transformer which obtained an FID of 0.0126 and MMD of 0.0012.

Overall, the approach presented is an interesting method of applying the diffusion model architecture to music generation using MIDI data. The paper uses nonstandard metrics, but does perform very well and successfully uses diffusion models to generate music. However, the lack of human judgment (the authors or an external survey) of the produced music still leaves room for confirmation of the quality of produced music.

2) SIMULTANEOUS MUSIC GENERATION AND SEPARATION

The paper by Mariani et al. [76] focuses on using diffusion based models for both generation and source separation by training it to learn the joint probability distribution of sources that are derived from the same context.

The primary method used is based on the statistical idea of modelling a prior distribution based on the training set, and then using this prior distribution to generate the audio required during inference. This prior distribution is learnt by using score-matching with a diffusion based generative model inspired by [77]. Using this method, the authors focus on three tasks: Total generation, partial generation, and source separation. Total generation refers to creation of the entire music without any input melody. Partial generation refers to generation of a melody that can accompany a given input, and source separation deals with trying to identify individual parts of the input music.

The proposed model was trained on the Slakh2100 dataset which is primarily for source separation. However, it does provide reasonable priors for music generation with the same model. The authors provide the code to allow the reader to replicate results, and briefly describe how to. Although the authors perform experiments on both music generation and source separation, this survey mainly focuses on music generation, and will evaluate this model on the same. The authors disregard the use of any quantitative metrics, and only provide generated audio samples for the reader to listen. The lack of any survey or further comment on the generated music makes it difficult to conclusively rank this method against others. However, more extensive testing is provided for source separation.

The primary merit of this paper is the ability to generate and decompose music using the same model. However, the lack of testing on the generated music does not allow any comparison to any other model, and the authors fail to compare it to any other baseline for music generation.

3) DIFFUSEROLL

DiffuseRoll [66] is based on using a diffusion model that can generate multi-track piano rolls which can be mapped into MIDI. Moreover, the paper introduces a color-coding scheme to easily encode information about the pitch, velocity, instrument, and position which allows for mapping from piano roll to discrete (MIDI) and vice-versa easily. Lastly, the paper also proposes a post-processing method to optimize the quality of the output melody.

DiffuseRoll was trained by converting MIDI into piano-roll data and training the diffusor on that. The output piano-roll is then converted back into MIDI. However, before conversion to MIDI, the output from the diffusor is filtered to remove notes that are too densely packed within a short time frame. This allows for post processing to improve the quality of the generated tracks both quantitatively and qualitatively.

The model was compared to MuseGAN quantitatively through accuracy, precision, recall, F1-score, Inception Score and Entropy. DiffuseRoll outperformed MuseGAN in all quantitative metrics and achieved a high entropy which indicates diversity in the generated notes, as well as a high inception score, which indicates high quality diverse tracks. Moreover, the authors qualitatively test their model by asking 20 experienced musicians to evaluate their generated music and rate it between 1 to 5, with 10 generated and 10 real piano rolls. The authors mention that the model did well with human listeners.

Overall, the paper presented a new diffusion model for music generation, as well as a technique to convert MIDI to and from piano-rolls. DiffuseRoll performing better than MuseGAN quantitatively proves its ability to generate music very well. However, the lack of results from the qualitative testing is questionable and doesn't allow for comparison to other methods.

M. DIFFUSION MODELS WITH SPECTROGRAMS

Diffusion models are naturally used with images, which is why it is intuitive to apply a diffusion model to a spectrogram representation of audio to be able to generate more spectrograms.

1) RIFFUSION

Riffusion [21] is an open-source project by Seth & Hayek that applies stable diffusion 1.5 to spectrograms without any training. This allows generation of music via spectrograms using the pretrained stable diffusion model and inherently allows for text conditioning. To allow for generation of longer tracks, the authors pick an initial spectrogram and re-run the model with different seeds and prompts to obtain several spectrograms. The authors interpolate the seeds and prompts within the latent space of the model to obtain smooth transitions between the generated spectrograms. This allows for an interesting audio that allows transferring between styles and genres.

Since this is a "hobby project" as stated by the authors, it does not have any metrics or testing conducted. However, the simple application of stable diffusion to this field is impactful and has been used as a baseline in other works.

2) MOUSAI

Schneider et al. [22] introduced a latent diffusion model that could generate music over one minute long using context that can also exceed 60 seconds. The overall generative model comprised of two independent models, Diffusion Magnitude-Autoencoding (DMAE), and latent text-to-audio diffusion. The first model compresses the conditioning audio waveform, while the second performs diffusion in the latent space based on the text embedding provided as prompt. Both models use a 1D U-Net model on the input spectrograms. The presented model can also be used with text conditioning using a pretrained language model to obtain the text embeddings.

Since the proposed model generates long context music through text descriptions, the authors compare it to Riffusion [21]. The evaluation method used in this paper is vastly different from other works in the field. The authors ask three surveyees to listen to the generated samples from both models given text prompts from 4 different genres, and then classify which genre each prompt belongs to. The results for this are then presented as a confusion matrix, showing the surveyees were able to correctly identify the genre of samples made by Mousai than Riffusion. However, no in-depth analysis of the results is provided. In contrast, the authors provide some qualitative descriptions of the generated music such as the sound quality by examining the spectrograms generated. Moreover, the generated music has a strong contextual relevance even over 1 minute apart, where the authors describe that rhythms, loops, and riffs are found in a song. The authors attribute this to the use of attention blocks in the U-Net.

Overall, the paper provided a latent diffusion model that uses spectrograms to generate long samples of music with textual context, as well as a diffusion magnitude autoencoder that can compress an input signal 64-fold. However, the lack of a proper survey and any quantitative metrics to evaluate the model makes it difficult to compare to other models easily, while the results provided remain subjective.

N. N. DIFFUSION MODELS WITH RAW WAVEFORMS

Diffusion models generally deal best with continuous data, which is why using raw data is an ideal scenario. Surprisingly, there is still relatively less work done in the field, possibly because of the lack of explicit structure in raw audio making it difficult to work with.

1) ERNIE-MUSIC

Despite raw waveforms being difficult to work with, [78] introduces a model called ERNIE-Music which deals with text to audio generation. The paper introduces a diffusion model that can generate raw audio based on input free-form text or text using a set of conditioning words.

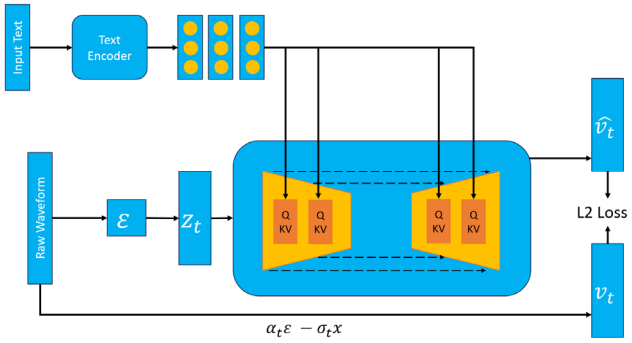

FIGURE 15. Diffusion with raw data.

The authors propose two models, an unconditional model and a conditional model. The unconditional model simply generates raw waveforms without any text input, while the conditional model incorporates the input text to provide conditioning to the generated audio. The conditional model is based on a U-Net architecture which is common in diffusion models. The U-Net architecture used is primarily based on stacked convolutional and self-attention blocks as in Fig 15. Fig 15 also illustrates the U-Net architecture with orange blocks, as well as the self-attention layers (QKV blocks), and skip connections (dashed lines). The text prompt and the raw waveform are both passed into the U-Net architecture, with the text used as conditioning. To use the text as conditioning, the model takes the noised latent and the text input and performs a fusing operation to obtain a text-aware embedding used in the diffusion process.

For experimentation, the authors assembled a dataset of text prompts and raw waveforms from the internet in a weakly supervised manner. They carefully use the comments and upvotes on available online music to generate this dataset. The authors acknowledge the lack of standardized metrics in the field to evaluate models, and hence perform a survey with 10 participants. The authors present score for both text-to-music relevance to examine the consistency of the text conditioning applied, as well as music quality which is the goal of any generative paper. Overall, the text to music relevance has an average score of 2.43/5 and the quality had an average score of 3.63/5. Hence, the authors achieve a great score on quality, but might need to further work on the incorporation of the provided text conditioning. However, this is the first model that converts text to raw audio using diffusors and is definitely an important step in the development in this field.

2) NOISE2MUSIC

Noise2Music [24] consists of a series of diffusion models used to generate music conditioned with text prompts. The text embeddings to be given to the diffusion models are obtained from the T5 encoder [79]. The authors explore the use of both raw audio and spectrograms to train the model and generate music. The authors use two models, one to generate an intermediate representation, and another cascade model

to use that intermediate representation and produce the final audio.

Using waveforms, the generator produces an audio output conditioned on the text prompt, while the cascader model generates the final waveform based on the generated audio and the original text prompt using cross attention. The spectrogram model has a generator that generates spectrograms with values between -1 and 1, followed by a vocoder model that generates audio directly based on the generated spectrogram. The authors use a custom dataset to train the model with 30 second clips and the corresponding text prompts. The authors use two quantitative metrics, Frechet Audio Distance (FAD) [80], and MuLan similarity score [26]. Noise2Music is compared to Riffusion [21], Mubert [28], and MusicLM [23], where the wavefrom model achieves the best overall scores across both metrics. The authors also had a qualitative analysis using 5 sources of music (3 baselines and 2 proposed models), and conduct pairwise comparisons to determine which one is closer to the provided text prompt. The qualitative analysis shows Noise2Music is comparable to MusicLM.

Overall, the paper introduces a new method of using diffusion models to generate music, exploring the use of both raw audio and spectrograms, with the raw audio model performing better. However, much like MusicLM, the qualitative analysis remains incomplete to accurately determine the overall quality of the generated music.

O. TRANSFORMER MODELS WITH RAW WAVEFORMS

With the increased popularity and great performance of transformers in various tasks, applying them in generation of music was apparent. Two salient works that use transformers with raw audio are the compound word transformer [81], and an improvement on it for multi-genre music generation [82].

1) COMPOUND WORD TRANSFORMER

The authors of [81] address the requirement of transformer models to take input of a sequence of tokens based on a finite predefined vocabulary. The primary contribution of this paper is the method of tokenization that makes a distinction between the different information encoded in a token, such as note, temporal, or metric tokens. This technique is used to convert a piece of music into compound words based on neighboring tokens. The authors also implement this technique on a transformer architecture and evaluate the music generated.

The authors form compound words by combining neighboring tokens in the same musical event (chords, notes, beats, etc.) into a single timestep based on the semantic meanings and similarity. They then pad these sequences to the same length. The authors use a multi-task learning approach to generate the corresponding embeddings, which predicts the next event in the sequence as well as predicting which compound word an event corresponds to. These embeddings are then converted into hypergraphs which are used to train a transformer model. The authors also acknowledge that the proposed method can be applied to other similar multi-token problems despite the paper introducing the method in the context of music generation.

The authors assembled a dataset of 1748 pop piano music of roughly 4 minutes each, and performed the tokenization into compound words, then using the transformer to generate similar audio. Quantitatively, the authors provide more performance metrics related to the time taken for training, inference, and memory usage than about the generated music. However, they do provide a metric on how well the melody and chords match the original distribution by using the longest common subsequence and cosine similarity respectively. Their best model, using the XL Transformer, achieved a melody match of 0.866 and a chord match of 0.800. Moreover, the authors also perform qualitative evaluation, where each of the 18 surveyees had to rate the fidelity, richness, humanness, correctness, structure, and an overall rating out of 5. Once again, the best performing model was the same, and achieved an overall score of 3.35/5.

Overall, this paper provides a strong basis for how raw audio can be preprocessed and made into a format easier for transformers to process. This process also increased the convergence rate of the model, which is one of the main achievements of the paper.

2) MULTI-GENRE MUSIC TRANSFORMER

Inspired by the Compound Word Transformer [81], the authors use the compound word embeddings to train a linear transformer through an adaptive learning approach. This was done on a multi-genre compound word dataset that the authors assembled using the preprocessing introduced in [82].

The authors use a linear transformer since it addresses long term sequence dependency which is a crucial aspect of music generation. Moreover, the use of an independent feed-forward head in the decoder incorporates additional token types to provide more context such as the genre or chord progression. The architecture is designed in a modular fashion which allows the user to define their own token sampling and embedding models. The authors encode each token separately and then concatenate them, which creates an adaptive token size, allowing for smaller embedding dimensions. Furthermore, the authors use a threshold based sampling method and adjusting hyper parameters in the CWT model encoding to allow and encourage more diverse chord progressions and avoid degeneration (producing only a small variance of notes).

The authors assembled a dataset from YouTube and performed piano transcription and beat tracking before converting the obtained tokens into compound words. The authors do not perform any qualitative evaluation, and only compare their training time, memory used, inference time, and average number of tokens per song to the CWT model. However, this shows that the model is significantly faster to train and obtains a similar performance in terms of number of tokens per song. Lastly, the authors do provide a sample piano roll of the original and generated songs, which shows

a decent variation in pitch and illustrates the fact that it can produce the entire song fairly similarly.

The paper shows an improvement to the original CWT model by allowing multi-genre generation through adaptive learning by allowing variation of the token size based on the genre and context. Although, the lack of quantitative and qualitative metrics provided make it hard to compare to other models.

P. TRANSFORMER MODELS WITH GUITAR TAB DATA

Most of the work done in music generation is based on MIDI data, but one paper explored the use of guitar tabs to generate music specifically for guitars. The authors of GTR-CTRL [83] train a Transformer XL model with conditioning to generate guitar tabs that are conditioned on instrumentation and genre.

The authors used the publicly available DadaGP dataset with 26,181 songs in token format. The authors used 3 types of models: Unconditional, instrumentational control, and genre control.

For the instrumentation model and genre models, the authors append a set of control tokens to the input tokens to the model which are used for conditioning. These could be control codes to control the key of the song, the starting note, and similar for the instrumentation model, and the name of the genre for the genre conditioning models.

The authors use highly non-standard metrics for evaluation. Quantitatively, they used pitch class histogram entropy and groove consistency to measure rhythm and pitch. They used prompted instrument presence and unprompted instrument presence to measure instrumentation. Lastly, they used GPBERT which is a classification model to identify genres from the DadaGP dataset to evaluate the genre generation, which achieved just over 90% accuracy, which indicates the genre conditioning worked well.

Further testing of pitch and rhythm metrics showed significant differences across genre and instrument conditioning. Testing of instrumentation showed the best instrumentation was achieved through a full prompt being supplied, while genre classification showed only folk music getting confused as rock music sometimes, but not to a large extent. The authors also performed qualitative analysis but only using their own observations and no survey.

The authors conclude by attempting to combine the instrumentation and genre models into a single model and note that this model performs a wider range of instrumentation when providing genre conditioning which was not observed in the genre model alone.

V. DISCUSSION AND COMPARISON

In the following discussion, the criteria to be used to define a good model is generating music that humans find pleasing and realistic. Hence, the user rating mentioned in most papers is an important metric, although other metrics do tell us more about the structure and distribution of the generated audio. This criterion has been chosen as a goal by most authors in the papers presented in section II-D, and we too believe it is a goal of all such music generation models.

Additionally, we provide a summary of the top three models based on several factors based on their applicability such as speed of convergence, consistency, creativity, etc in Fig. 16. Moreover, we also summarize the best models based on the different music generation tasks in Fig. 17.

A. BEST MODELS PER CATEGORY

The best models in each category can be seen in Table 4. Further details on the reasons for their choices are presented in this section.

MusicGen was the best performing autoregressive model beating MusicLM during evaluation in [18]. Moreover, it applied a newer method of using a LM to facilitate music generation over discrete inputs which is why it surpasses the relatively old approach provided by Wavenet.

MidiPGAN was the best GAN using MIDI Data, since it had a higher human rating than MuseGAN and MIDINet, while being highly rated for Realistic music generation. Moreover, it does not suffer from the issue of confusing short, consecutive notes as MIDINet does, and is faster than MuseGAN. Furthermore, MidiPGAN had an FID of 12.3, which means the generated music was not very close to the trained music, which indicates diversity. We can be sure this is good diversity and not poor performance, by noting that MidiPGAN received an average user rating of 3.34/5 which means the generated music was pleasant and realistic.

MP3Net was the best GAN with Spectrograms, since it uses a similar architecture to MidiPGAN, by following the ProGAN architecture, along with the DCGAN architecture. Additionally, MP3Net had higher quality music produced than SpecGAN. Although the music generated by MP3Net did not have much variety because of the checkerboard pattern issue, it still had better conversion quality from spectrograms to audio than SpecGAN. However, StyleGAN2 performed exceptionally well, but both MP3Net and StyleGAN2 did not provide similar metrics to successfully compare them convincingly. Since MP3Net also trains the fastest out of all presented models in the category, we choose MP3Net to be the best model in this category.

Self-Attention GAN (SAGAN) performed the best in GAN with Raw Waveforms. This is primarily because it is faster to train, needs less data than WaveGAN, allows the user to pick the instrument conditioning to apply, and the fact that the metrics presented in [52] convincingly beat WaveGAN as their baseline. However, it has a lower inception score then WaveGAN, indicating not much variety in the images generated.

A-Muze-Net (AMN) performs the best in RNNs with MIDI Data. This is primarily because AMN has a much higher user rating than LSTM MIDI Models. However, even though both A-Muze-Net and LSTM MIDI Models beat MuseGAN as their baselines, the fact that A-Muze-Net is made strictly for piano music, sets it aback. LSTM MIDI Models allow for better generalization by focusing on the

TABLE 4. Best model in each category.

Category	Model Name
Autoregressive Models	MusicGen
GANs with MIDI Data	MidiPGAN
GANs with Spectrograms	MP3Net
GANs with Raw Waveforms	Self-Attention GAN
RNNs with MIDI Data	A-Muze-Net
RNNs with Spectrograms	Mel-Net
RNNs with Raw Waveforms	LSTM Models
VAEs with MIDI Data	Adjusting Tonal Tension
VAEs with Raw Waveforms	AMAE
Deep RL with MIDI Data	RE-RL Tuner
Diffusion Models with MIDI Data	Symbolic Music Generation
Diffusion Models with Spectrograms	Mousai
Diffusion Models with Raw Waveforms	Noise2Music
Transformers with Raw Waveforms	Multi-Genre Music Transformer

melody and harmony being predicted separately from the same model, rather than building two separate models to model each hand of a pianist. On the contrary, since our primary evaluation criteria is that the user should like the generated music, AMN is ranked the highest. Additionally, we note that AMN generates 100% of its notes to be of a decent length, and has about 9.54 pitches per bar, which is a good range considering there are 8 notes in an octave.

Mel-Net is one of the few models that uses RNNs with Spectrograms, and is by far the most popular, and well performing. First, it performs the best in any audio generation task, not just music generation, which allows for better generalization to other use-cases. Moreover, the ability of Mel-Net to capture long term dependencies is likely to lead to more coherent music that follows the consistent theme. Furthermore, despite not assigning a score out of 5 by the survey takers, survey takers preferred Mel-Net over WaveNet and Wave2Midi2Wave models.

LSTM Models beat the other RNN models for many reasons. First, the fact that LSTMs can remember more data, allows it to capture a more holistic view of the music piece, and generate more globally coherent music, which is the reason it has the highest user rating amongst all models presented in this paper, of 3.6/5. Secondly, it only uses very basic preprocessing, which is not computationally expensive, improving the speed of the process. Lastly, not explicitly using any musical structure allowed it to learn the best structure and patterns in the training data, generalizing better.

Adjusting tonal tension-based models are the best amongst available VAE with MIDI input. This is because of the flexibility offered, as well as the high quality of music produced. However, the comments from the professional musician do identify a key idea that the music still does not sound very realistic overall.

AMAE being the only major paper using VAE with raw waveforms, is the best in the category. It was able to introduce new methods to model both long- and short-term dependencies in music and capture a more holistic view of the audio. Moreover, it was able to beat the results of WaveNet, and provides a stable and generalizable method of generating audio. However, the lack of a survey conducted makes it difficult to compare it easily to other models.

RE-RL Tuner is the best model amongst the models using Deep Reinforcement Learning approach with MIDI input. This is primarily because it has been shown to be better than the LSTM models (used as the baseline), and the high ratings by human survey takers of 3.58, which is one of the highest ratings across all models in this paper. Even though Bach2Bach had a decent quality of music produced, the poor feedback by the professional musician mentioning it did not sound very realistic, sets it aback.

Symbolic Music Generation introduced the use of diffusion models in music generation, and provided the best overall metrics compared to the other models. The paper boasts 99% pitch consistency, and 96% duration consistency for unconditional testing, and 97% pitch and duration consistency during infilling. These extremely high metrics, in conjunction with an FID of 0.0166, and MMD of 0.0018 indicates very high-quality music generated from the model. However, the lack of human evaluation still stands in the way of proper comparison.

Mousai outperformed Riffusion in the prompt adherence study conducted in [22], and was able to generate longer tracks with longer conditioning, effectively outperforming the simple application of stable diffusion to spectrograms as in Riffusion.

Noise2Music outperformed ERNIE-Music primarily with a high prompt relevance, and a low FAD which indicates that the generated audio has similar quality to the training samples, while generating longer samples of audio.

For transformers with raw waveforms, the Multi-Genre Music Transformer outperforms the baseline CWT since it is an improvement of the usage of compound word encoding with a transformer allowing for adaptive and faster learning.

B. BEST ARCHITECTURE

The best architecture amongst GANs, RNNs, VAE, RL, Transformers, Language Models, and Diffusors is not a simple distinction to make. However, on average, GANs performed the best with different input data types. All the models using a GAN had results very close to each other, such as MIDINet, MidiPGAN, MP3Net, StyleGAN2, etc. Moreover, the flexibility of being able to use any input type (MIDI, Spectrograms, or Raw Waveforms), allows GANs to be a better architecture on average. Although none of the highest two user rated models used GANs, the average user rating of models that are based on GANs are over 3.2/5, which stands to be a very good rating overall.

On the other hand, on average, most models with GANs are slower to train, solely due to the architecture used. However, this is not as crucial as the quality and user acceptance of the music generated, which is why GANs remain the best architecture amongst those considered.

Additionally, Language Models (LMs) performed exceptionally well, but could not be directly compared to GANs due to the lack of human surveys conducted during evaluation.

TABLE 5. Quantitative result comparison.

Model	Dataset – Number of samples	Dataset – Length of each sample	Inception Score	Frechet Inception Distance	% of long notes	Range of pitches in a bar	Survey takers' rating (/5)	Number of survey takers
WaveNet	260	29 sec	-	-	-	-	-	-
MusicGen	10,000	Full length	-	-	-	-	-	-
MusicLM	106,574	30 sec	-	-	-	-	-	-
MIDINet	1022	8 bars	-	-	-	-	3.30	21
MuseGAN	45,129	96 beats	-	-	52.5	4.89	3.16	144
MidiPGAN	-	75 notes	-	12.30	-	-	3.34	34
SpecGAN	1850	10 sec	3.71	-	-	-	2.8	300
MP3Net	200+hrs	95s and 5s	-	-	-	-	-	-
TL with GANs	23,983	-	5.24	7.91	-	-	-	-
WaveGAN	1850	10 sec	4.67	-	-	-	3.2	300
Self-Attention GAN	21572	3s	3.76	45.85	-	-	-	-
Melody RNN	200	30-60s	-	-	-	-	-	-
A-Muze-Net	243	-	-	-	100	9.54	3.28	17
LSTM MIDI Models	100+	-	-	-	91.2	7.79	2.7	10
Mel-Net	2000+	-	-	-	-	-	-	50
LSTM Models	25	-	-	-	-	-	3.6	10
RNN models	23623	7 bars	-	-	-	-	-	-
Adjusting Tonal Tension	2000	-	-	-	-	-	-	-
AMAE	413	-	-	-	-	-	-	-
Bach2Bach	100+	-	-	-	-	3.89	-	-
RE-RLTuner	100+	-	-	-	-	11	3.58	19
Symbolic Music Generation	988,893	64 bars	-	0.0166	-	-	-	-
Simultaneous Music Generation and Separation	2100 tracks	-	-	-	-	-	-	-
DiffuseRoll	-	-	-	-	-	-	-	20
Riffusion	-	-	-	-	-	-	-	-
Mousai	2500hrs	44sec	-	-	-	-	-	-
ERNIE-Music	4094 tracks	20 sec	-	-	-	-	3.63	10
Noise2Music	340,000 hrs	30 sec	-	-	-	-	-	-
Compound Word Transformer	1748 tracks	4 min	-	-	-	-	3.35	18
Multi-Genre Music Transformer	150 tracks	-	-	-	-	-	-	-
GTR-CTRL	26,181 tracks	-	-	-	-	-	-	-

The use of LMs emphasizes the structure and patterns in the data to be learnt and generated well as described by authors in [18] and [23], which is where GANs failed to perform well. Hence, LMs and GANs can both be considered the best architectures to use for music generation.

C. BEST INPUT METHOD

Overall, the best method of input is MIDI input, which allows the model to successfully capture the overall structure of the music, while being relatively fast to process compared to spectrograms, and lighter to train compared to raw waveforms. This offers a major advantage to allow training on larger, longer datasets so that the model can train for longer and pick up more patterns in the data. Moreover, the MIDI input offers flexibility to transform the data into other representations, as seen in [29] and [64]. Furthermore, it allows separation of chords and notes, to promote approaches that focus on either of them separately such as [30] and [58] which significantly improved overall performance. This flexibility further supports the use of MIDI in music generation.

However, this does limit the application of the work done to Music Generation only, and not generation of speech, or any other such audio generation problems that cannot be represented by MIDI, but the importance of such a wide range of applicability is debatable.

D. BEST CATEGORY

Across all the types of input and architectures, the overall best architectures were MIDI with LSTMs and Autoregressive models like MusicLM and MusicGen.

Autoregressive models include the relatively new approach of using language models on discretized representations of audio to generate music. This not only allows for higher quality generation, but using the representation used in MusicGen, this can be easily generalized to other forms of audio generation, allowing for a highly flexible model. In fact, MusicGen

TABLE 6. Qualitative result comparison.

Model	Pros	Cons
WaveNet	- Works well speech generation and TTS - Fast to train - Uses only convolutions - Can be conditioned to generate a particular genre/instrument	- Lack of quantitative music generation results - Could not capture long-term dependencies
MusicGen	- Single model for text and melody-conditioned generation - Single stage model with efficient code book interleaving - Beat both LM based and diffusion based models	- Lack of detailed control over adherence to conditioning
MusicLM	- High quality music consistent over several minutes - Uses popular models to build the architecture - Proposing a new music generation evaluation dataset MusicCaps	- Lack of melody conditioning - Misunderstands negations in prompts and does not always follow temporal ordering in prompts
MIDINet	- Use of conditioner CNN for memory - Creativity and Stable modes - Sequential processing - Beat MelodyRNN	- Cannot distinguish between 2 short, repeated notes vs 1 long note - Lack of quantitative results
MuseGAN	- Overall consistency and inter-track dependency considered - Models multi-track dependency well	- Slow to train - Needs a lot more data than other models
MidiPGAN	- ProGAN Architecture - Detailed survey conducted on Pleasingness, Creativity, Structure and Realism	- Didn't beat the baseline by much
SpecGAN	- Generate digits 0-9 spoken and humans could understand it - 0.8 nearest neighbor distance – close to the input distribution - Quick training convergence - Good score with human judges	- Quality not very good - Converting to and from spectrogram is lossy – Reduces quality
MP3Net	- Use ProGAN + DCGAN architecture - Orders of magnitude faster than Jukebox (baseline) in training	- Variety of music produced was poor - Has CNN checkboard pattern issue [24]
TL with GANs	- StyleGAN2 performed best even though it wasn't trained for music generation - Successfully modelled the problem as an image generation problem	- No survey performed – Hard to judge against our criteria
WaveGAN	- High quality (more than SpecGAN) - Had issues generating spoken numbers – Not necessarily important for music generation	- More data needed per track - Passed sequentially though 1DConv layers – Slow
Self-Attention GAN	- Fast and fully parallel music generation - User can pick which instrument they want to dominate the song - Data augmentation improved quality and robustness - Beat WaveGAN (baseline)	- Needs a lot of RAM – parallelism
Melody RNN	- 3 Models suggested – Basic, LookBack, and Attention - Better quality than Bi-axial RNN – by authors	- Not enough data used to train model - Music >16s long had cycles - Evaluation very subjective
A-Muze-Net	- Uses basic music theory to split melody and harmony - Out performs MuseGAN	- Only for piano music
LSTM MIDI Models	- Better than MuseGAN (baseline) - Split into melody and harmony like A-Muze-Net	- MuseGAN had higher human rating
Mel-Net	- Better than WaveNet (baseline) - Ability to capture long term dependencies - Generalizes to all audio generation tasks	- Lack of results regarding quality of audio generated - Generates high quality spectrograms to combat conversion loss – Slow to train
LSTM Models	- Basic preprocessing – not computationally expensive - Does not use musical structure, allowing it to capture the overall theme	- Trained on only 25 piano tunes – too few
RNN models	- LSTMs perform better than any other type of RNN method -Able to capture the long-term dependencies better - Use of higher-level notation improves generation	- No survey performed – Hard to judge against our criteria
Adjusting Tonal Tension	- Adjusts tonal tension based on user input - Introduces new parameters for losses – Cloud Diameter and Tensile Strain - Predicts melody and bass separately	- Nonstandard metrics and no comparisons made qualitatively
AMAE	- Able to capture long term dependencies well - Sounds more natural than regular autoregressive models	- Large model – needs a lot of RAM - Nonstandard metrics used - No metric about long term dependencies or natural sounding tone
Bach2Bach	- Models music generation as a RL problem, using LSTMs - High quality	- Professional musician said doesn't sound realistic - No comparisons
RE-RLTuner	- Better than Note RNN and LSTM MIDI Models (baseline) - Use of LDA to automatically detect music principles that govern a song - Good ratings by human surveyees	- Time consuming to train

TABLE 6. *(Continued.)* **Qualitative result comparison.**

Symbolic Music Generation	- Models generation with MIDI data for diffusion models - Iterative process allows for easy conditioning to generate a particular style of music - Best FID so far	- Not compared to other popular models - No human evaluation to verify the generated music
Simultaneous Music Generation and Separation	- Allows for both music generation and instrument separation in one model - Can be conditioned with text prompts - Extensive testing on separation of instruments	- No proper testing on generation of audio - No metrics or survey to compare to other models
DiffuseRoll	- New method to convert from piano-roll to MIDI and back - Beat MuseGAN	- No results shared, only conclusions stated - Can not easily compare to other models
Riffusion	- Easy application of pretrained stable diffusion v1.5 - Able to generate audio that interpolates between different prompts and styles by interpolating in the latent space of the model	- No metrics or testing conducted (hobby project)
Mousai	- Generate music over a minute long @ 48kHz with context over a minute long. - Generate audio in real time using novel efficient 1D U-Net architecture - Novel diffusion magnitude autoencoder for 64-fold compression.	- Use of magnitude spectrograms instead of mel-spectrograms – larger model and lower quality
ERNIE-Music	- Highest human rating - Direct conversion from text to waveforms - unique	- Low level of influence from the text conditioning
Noise2Music	- Models for both spectrogram and raw waveform generation - Better than MusicLM, Mubert, and Riffusion	- Survey conducted only on prompt relevance, and not quality
Compound Word Transformer (CWT)	- Introduces new encoding that allows usage of transformers - melody match of 0.866 and chord match of 0.800 during testing - Fast convergence	- A lot of preprocessing required
Multi-Genre Music Transformer	- Improves on CWT with even faster convergence - Trained on multiple genres	- Lack of quantitative and qualitative metrics
GTR-CTRL	- Use of guitar tabs for generation - Allows for both genre conditioning and instrument conditioning - Good scores achieved by metrics used	- Highly nonstandard metrics making it incomparable to other models

and MusicLM did better than several diffusion-based methods on both quality and adherence to conditioning, which makes them the top performers. Moreover, the use of autoregressive models easily models the interdependency between the note at each timestep and the preceding notes.

MIDI with LSTMs also performed well, particularly with models that beat GAN based models, such as A-Muze-Net and LSTM MIDI Models beating MuseGAN as baselines. Moreover, the ability of LSTMs to remember and make use of globally important data helps improve the overall theme and coherence of the song. On average, LSTM models have a user rating of roughly 3.3/5, ranking close to MIDI with GANs. Moreover, a major reason for LSTMs being ranked high is the fact that all the Deep Reinforcement Learning methods also use LSTM based models with MIDI input as their backbone. In fact, one of the highest user rated models using RL uses LSTM as its backbone for each of the networks within it (RE-RL Tuner with average user rating of 3.58/5). This further advocates the use of LSTMs with MIDI input due to their flexibility and ability to generalize to learn any Turing machine.

E. BEST MODEL OVERALL

Lastly, we shall evaluate the best model across all categories. In this case, two models outshine the rest: RE-RL Tuner [72], and MusicGen [18].

RE-RL Tuner is the best model across all, receiving the 2nd highest user rating of 3.58/5 (only 0.02 less than the highest). The reason RE-RL Tuner is chosen as the best model is the fact that it used LSTM based MIDI models as a backbone, which are one of the best performing models overall. In fact, it was able to beat LSTM MIDI Models in terms of user ratings, and overall quality of generated audio. Moreover, RE-RL Tuner was able to successfully capture the principles governing the music using LDA, which other models were unable to do successfully, and is the reason why it is better than ERNIE-Music despite receiving a slightly lower user rating, since it provides a higher level of explainability. RE-RL Tuner also allows for a larger scope for experimentation, and hence can be modified to allow for conditioning. Additionally, since RE-RL Tuner is a RL model, changing the backbone used, or using other methods to capture the theory behind the music, may allow for better results.

However, it is time consuming to train, which is its major drawback. Although, the improved ability to produce well structured, globally consistent music that is pleasant to humans majorly outweighs the training time required.

Moreover, MusicGen uses a relatively newer approach to music generation using a single language model (LM) to learn from discrete representations of music known as codebooks. Hence, like RE-RL Tuner, this too uses a discrete form of music input, which is the predominant performer across the

TABLE 7. Summarized model evaluations.

Model	Architecture	Contribution	Performance	Future Extensions
WaveNet	Autoregressive with causal and dilated convolutions; SoftMax for conditional distribution	Introduced dilated convolutions to handle long-range dependencies efficiently; used non-linear quantization to reduce signal representation	No quantitative results; generated music lacked long-term consistency but was genre-specific with conditioning	Improve long-term consistency and provide more robust evaluation metrics
MusicGen	Transformer-based autoregressive decoder with EnCodec for audio tokenization	Introduced codebook interleaving strategy; enabled text and melody conditioning; efficient autoregressive music generation	Best performance in KL divergence, CLAP, and survey ratings; slightly worse than Noise2Music in FAD	Allow more detailed control over the conditioning inputs for finer-grained music generation
MusicLM	Hierarchical sequence-to-sequence autoregressive model; decoder-only Transformer	Combined semantic and acoustic tokens for better representation; introduced MusicCaps dataset	Outperformed baselines in prompt relevance and quantitative metrics; lacked evaluation on audio quality	Incorporate human evaluation of music quality; explore better representations for pleasurable music generation
MIDINet	Deep Convolutional GAN with Conditioner CNN	Uses conditioning based on previous bars to improve melody and chord continuity; innovative MIDI representation to encode notes and chords.	Preferred over MelodyRNN in surveys; no quantitative comparisons; struggles with distinguishing note lengths due to encoding limitations.	Explore better MIDI encoding to resolve note-length issues; quantitative comparison with other models.
MuseGAN	GAN with multi-track and conditional generation	Addresses multi-track consistency and temporal dependency; hybrid track-conditional generation for better results.	Evaluated with unique metrics (e.g., empty bars, pitch classes); hybrid models preferred in surveys, but metrics make comparisons difficult.	Develop standard evaluation metrics; extend for more complex multi-instrument arrangements.
MidiPGAN	Progressive GAN	Adapts Progressive GAN for music generation; introduces Frechet Inception Distance (FID) for evaluation.	FID of 12.30; rated well in surveys but realism was rated lower than MIDINet; performs comparably to baselines with limited improvement.	Combine Progressive GANs with methods like conditional generation to improve realism and consistency.
SpecGAN	DCGAN using spectrograms	Converts audio to spectrograms for GAN-based generation; employs image-generation techniques for music generation.	Lowest Inception Score (3.71); generated intelligible spoken digits; faster training but lower quality compared to WaveGAN.	Use Mel-Spectrograms for better results; improve inversion accuracy to reduce time complexity.
MP3Net	DCGAN with Modified Discrete Cosine Transform (MDCT)	Adapts DCGAN with psychoacoustic layers; Progressive GAN approach for better audio timbre and tonality.	Orders of magnitude faster training than Jukebox; good piano timbre but limited variety; lacks quantitative benchmarks and surveys.	Improve diversity and expand training dataset to support more instruments and music styles.
Transfer Learning of GANs	Transfer learning on StyleGAN, StyleGAN2, UNAGAN	Benchmarks transfer learning of GANs on drum music; highlights the efficacy of StyleGAN2 and UNAGAN for specific metrics like Fréchet Audio Distance (FAD).	StyleGAN2 achieves best overall performance (IS 5.24, FAD 7.91); UNAGAN generates closer-to-distribution audio (FAD 4.32).	Apply transfer learning to broader music genres; integrate with non-spectrogram-based approaches for better phase modeling.
WaveGAN	DCGAN with 1D convolutions	Uses raw waveforms as input; adapted for sequence data generation while maintaining principles of DCGAN.	Best performance among SpecGAN variants; higher quality and better Inception Score; generates intelligible digits 0-9.	Enhance scalability for longer sequences and richer audio datasets; improve diversity and fidelity of generated waveforms.
Self-Attention GAN	GAN with Self-Attention	Fast parallel music generation with instrument conditioning and effective data augmentation for quality improvement.	Achieved FID of 45.85 (better than WaveGAN) and IS of 3.76 (comparable to WaveGAN's 3.83).	Address accuracy limitations in IS metric and explore diverse datasets or extensions for broader applicability.
Melody RNN	LSTM-based	Combines Magenta and WaveNet for high-quality MIDI sequence generation with three input types: Basic, LookBack, Attention.	Subjective comparison to Biaxial RNN and WaveNet; music >16s often repeats melodies.	Address dataset limitations, provide quantitative results, and explore improved methods to prevent repetitive melodies.

TABLE 7. *(Continued.)* **Summarized model evaluations.**

A-Muze-Net	Dual LSTM models for melody and harmony	Exploits piano theory to separately model melody and harmony, improving coherence of generated piano music.	Outperformed MuseGAN in most metrics; user satisfaction score of 3.28/5 (better than MuseGAN's 3.16).	Adapt architecture for instruments beyond piano and explore multi-instrument harmony generation.
LSTM MIDI Models	Various LSTM Encoder-Decoder architectures	Explores attention mechanisms and CNN+LSTM combinations to enhance dual-track piano music generation.	Embedding Attention-LSTM: high pitch variability (7.79 pitches/bar); CNN+Attention-LSTM: 91.2% long notes; MuseGAN: highest naturalness (3.16).	Improve naturalness score of proposed models and explore scalability for multi-track music generation.
Mel-Net	RNN-based with Time and Frequency Delayed Stacks	Captures both local and global audio structure effectively using spectrogram representation.	Outperformed WaveNet and Wave2Midi2Wave in long-term structure (95% and 62.5% user preference, respectively).	Quantify quality of generated audio, enhance evaluation metrics, and investigate domain-specific optimizations for broader use cases.
LSTM Models	LSTM-based architectures: Only LSTM, Fully Connected LSTM, 2D/1D Convolutional LSTM, Bilinear LSTM	Explored multiple LSTM variants for raw waveform input, capturing structure and coherence without using adversarial training.	Best models: Bilinear LSTM (3.6/5) and 2D Convolutional LSTM (3.4/5) rated by a survey of 10 people.	Explore quantitative evaluations and incorporate musical structure to improve global coherence.
RNN models	LSTM with melody-chord modeling and higher-level representations like ABC notation	Improved global structure in generated music through representation changes (e.g., ABC notation) and melody-chord models.	Successfully captured local and global themes in blues using melody and chord data, evaluated qualitatively.	Develop advanced tokenization methods and further evaluate global structure across other music genres.
Adjusting Tonal Tension	Variational Auto Encoder (GRUs in encoder-decoder)	Enabled user control over tonal tension using Spiral Array Tension measures: Cloud Diameter and Tensile Strain.	Models trained on 2000+ pop songs; best performance when combining all metrics or using cloud diameter only.	Expand dataset diversity, explore other tonal tension measures, and refine decoder predictions for more complex polyphonic music.
AMAE	ArgMax Auto Encoder with WaveNet-inspired autoregressive discrete autoencoders (ADAs)	Captured long-term dependencies and global coherence using multi-level autoregressive models for realistic music generation.	Outperformed WaveNet in NLL (0.695 vs 1.151) and matched VQ-VAE. Provided audio samples for qualitative judgment.	Incorporate detailed goal-specific metrics (e.g., surveys on naturalness) and explore varying ADA architectures and model configurations for broader applications.
Bach2Bach	Bi-axial LSTM trained using Deep Q Learning	Introduced reinforcement learning for music generation, enabling greater coherence and structured exploration	Performed better or as good as baseline Note RNN; had 55.7% unique melodies; noted rhythmic discontinuity in the output	Incorporate deeper structure and smooth rhythmic flow; compare with popular baselines for better evaluation
RE-RLTuner	Latent Dirichlet Allocation (LDA) for feature extraction, DQN for reinforcement learning	Combined reinforcement learning with music theory principles to enhance music generation	Beat baseline LSTM with higher survey ratings (3.58/5); showed unique note patterns and coherent intervals	Provide detailed training and performance analysis; expand human evaluation studies
Symbolic Music Generation	MusicVAE + Transformer + Diffusion Model	Adapted diffusion models for symbolic music data by mapping MIDI data into a continuous latent space	Achieved high consistency (99% pitch, 96% duration); competitive FID and MMD scores against autoregressive transformer	Include human judgment for evaluating generated music quality
Simultaneous Music Generation and Separation	Diffusion-based generative model	Unified music generation and source separation within a single model	No quantitative metrics provided; offers audio samples but lacks comparative analysis	Introduce comprehensive testing and comparison for music generation results
DiffuseRoll	Diffusion model with piano-roll representation and color-coding scheme	Enabled multi-track piano-roll generation with optimized post-processing	Outperformed MuseGAN quantitatively and received high ratings from musicians	Provide detailed qualitative results for broader comparison with other methods
Riffusion	Stable Diffusion 1.5 applied to spectrograms	Generates music from spectrograms using pretrained stable diffusion, enabling text conditioning; allows smooth style/genre transitions through seed interpolation.	No quantitative metrics provided; qualitative observations highlight smooth transitions and interesting audio output.	Incorporate rigorous evaluation metrics; refine for broader applicability beyond hobbyist use.
Mousai	Latent Diffusion Model with 1D U-Net	Introduced Diffusion Magnitude-Autoencoding (DMAE) and text-to-audio latent diffusion for long-context music generation.	Outperformed Riffusion in genre classification by human evaluators; qualitative insights on rhythm and context retention.	Include detailed quantitative metrics; broaden testing to improve comparison and validation.

TABLE 7. *(Continued.)* **Summarized model evaluations.**

ERNIE-Music	U-Net with convolution and self-attention blocks	First model to generate raw audio from text prompts using a diffusion model; supports conditional and unconditional generation.	Text-to-music relevance scored 2.43/5, and music quality scored 3.63/5 in a survey of 10 participants.	Improve text conditioning for higher relevance; explore additional datasets for robust training.
Noise2Music	Two-stage diffusion models (waveform and spectrogram)	Generates music using a generator and cascade models; explores raw audio and spectrograms, achieving strong results with raw audio.	Achieved best scores in FAD and MuLan similarity metrics; qualitatively comparable to MusicLM.	Refine qualitative evaluations; further explore improvements in spectrogram-based generation techniques.
Compound Word Transformer (CWT)	Transformer (XL)	Tokenization method distinguishing note, temporal, and metric tokens to create compound words for music generation.	Melody match: 0.866, Chord match: 0.800, Overall survey score: 3.35/5. Focus on time taken for training, inference, and memory usage.	Extend tokenization method to other multi-token problems. Improve qualitative metrics for music generation.
Multi-Genre Music Transformer	Linear Transformer	Extension of CWT model using compound word embeddings for multi-genre music generation with adaptive learning.	Faster training compared to CWT. Generates varied pitch and chord progressions. No quantitative or qualitative metrics provided.	Enhance modular architecture to support more genres and contexts. Include more evaluation metrics for performance comparison.
GTR-CTRL	Transformer XL	Generates guitar tabs conditioned on instrumentation and genre using control tokens.	Genre classification: 90% accuracy. Pitch class histogram entropy, groove consistency, and instrument presence tested.	Combine instrumentation and genre models into a single model for broader range of instrumentation and improved genre conditioning.

FIGURE 16. Category wise best models for application.

field. The method allows for both text and melody-based conditioning providing greater flexibility to the user to generate music or use a sample to augment. Additionally, testing showed it achieving a lower FAD than Riffusion, Mousai, and MusicLM, a lower KL Distance than all of those and Noise2Music and achieving a human rating of 85% on overall quality of music.

However, the only downside of MusicGen is the lack of fine-tuned control over the conditioning provided. However, this is superseded by the flexibility to condition using both text and melody in the same model.

VI. SUMMARY

All in all, we have successfully reviewed the best works in the field, under 12 categories based on the architecture used by the models and their input formats. We compared and concluded the best models in each category as in Table 4, along with the best architecture to be used, the best input

FIGURE 17. Music generation task wise best models.

format, the best category, and the best model across all models. The best architecture to be used are GANs, the best input format is MIDI, the best categories are GANs with MIDI and LSTMs with MIDI, the best model overall is RE-RL Tuner and MusicGen. The primary metric used to rank these were their user ratings (scores by human survey takers), followed by their other metrics and qualitative properties.

The quantitative results obtained can be found in Table 5, and the qualitative comparison can be found in Table 6 at the end of the paper. Additionally, Table 7 contains summarized evaluations of each of the models providing the architecture, primary contributions of the paper, performance, and future extensions. Based on the analysis, we also conclude on the top three models based on amount of creativity, consistency, speed, quality, and human rankings in Fig. 16. Moreover, Fig. 17 concludes on the best models by generation tasks.

VII. CONCLUSION, GAPS, AND FUTURE WORK

With the given research out there, we have a good understanding of basic music generation and have even moved on to generate music over just a few seconds in length to over a minute. However, we are nowhere close to achieving the level of music generation expected. All music generated by the approaches discussed in this paper still fail to add more of a human and creative touch, with lack of innovation in the generated sounds, as well as incorporating emotion into it. Approaches such as [72] attempt to begin grasping such humane features in music and looking at the music as a whole rather than just independent notes. We hope to see more such approaches develop in the field, but it still is yet to be addressed in more detail, with more realistic music being generated.

Future work in this field primarily details using methods like transformers, diffusion models, and language models, which perform exceedingly well in other fields such as image and text generation, to generate music as in sections 3O&P, 3L,M&N and 3A respectively. Moreover, methods like Reinforcement Learning, to encourage the model to explore and facilitate learning the principles behind music rather than hard coding them or not including them at all. Furthermore, we look forward to being able to combine data formats as in Section 3K, to leverage the level of detail and relationship represented in each data format, to allow the model to gain a better understanding of the structure of the music. Lastly, we hope to see established and unified evaluation metrics amidst all music generation models, to allow for better comparison and baselining between different approaches. All these advancements would allow us to achieve our end goal of generating realistic, pleasing music for humans.

ACKNOWLEDGMENT

This paper represents the opinions of the author(s) and does not mean to represent the position or opinions of the American University of Sharjah.

REFERENCES

[1] J.-P. Briot, G. Hadjeres, and F.-D. Pachet, "Deep learning techniques for music generation—A survey," Aug. 2017, *arXiv:1709.01620*.
[2] S. Ji, J. Luo, and X. Yang, "A comprehensive survey on deep music generation: Multi-level representations, algorithms, evaluations, and future directions," 2020, *arXiv:2011.06801*.
[3] Z. Yin, F. Reuben, S. Stepney, and T. Collins, "Deep learning's shallow gains: A comparative evaluation of algorithms for automatic music generation," *Mach. Learn.*, vol. 112, no. 5, pp. 1785–1822, May 2023, doi: 10.1007/s10994-023-06309-w.
[4] L. Moysis, L. A. Iliadis, S. P. Sotiroudis, A. D. Boursianis, M. S. Papadopoulou, K. D. Kokkinidis, C. Volos, P. Sarigiannidis, S. Nikolaidis, and S. K. Goudos, "Music deep learning: Deep learning methods for music signal processing—A review of the state-of-the-art," *IEEE Access*, vol. 11, pp. 17031–17052, 2023, doi: 10.1109/ACCESS.2023.3244620.
[5] R. Bittner, M. Müller, J. Nam, M. Krause, and Y. Özer, "Deep learning and knowledge integration for music audio analysis (dagstuhl seminar 22082)," *Dagstuhl Rep.*, vol. 12, no. 2, pp. 103–133, 2022.
[6] K. K. Jena, S. K. Bhoi, S. Mohapatra, and S. Bakshi, "A hybrid deep learning approach for classification of music genres using wavelet and spectrogram analysis," *Neural Comput. Appl.*, vol. 35, no. 15, pp. 11223–11248, May 2023, doi: 10.1007/s00521-023-08294-6.
[7] C. Walder, "Modelling symbolic music: Beyond the piano roll," in *Proc. 8th Asian Conf. Mach. Learn.*, Jan. 2016, pp. 174–189. Accessed: May 30, 2023. [Online]. Available: https://proceedings.mlr.press/v63/walder88.html
[8] C. Walshaw. *A Statistical Analysis of the ABC Music Notation Corpus: Exploring Duplication*. University of Greenwich. Accessed: May 30, 2023. [Online]. Available: http://www.cmpe.boun.edu.tr/fma2014/fma2014_proceedings.pdf
[9] D. Jeong, T. Kwon, Y. Kim, and J. Nam, "Graph neural network for music score data and modeling expressive piano performance," in *Proc. 36th Int. Conf. Mach. Learn.*, May 2019, pp. 3060–3070. Accessed: May 30, 2023. [Online]. Available: https://proceedings.mlr.press/v97/jeong19a.html
[10] S. Barratt and R. Sharma, "A note on the inception score," 2018, *arXiv:1801.01973*.

[11] A. Obukhov and M. Krasnyanskiy, "Quality assessment method for GAN based on modified metrics inception score and Fréchet inception distance," in *Software Engineering Perspectives in Intelligent Systems* (Advances in Intelligent Systems and Computing), R. Silhavy, P. Silhavy, and Z. Prokopova, Eds. Cham, Switzerland: Springer, 2020, pp. 102–114, doi: 10.1007/978-3-030-63322-6_8.

[12] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," Dec. 2015, *arXiv:1512.00567*.

[13] H. Dong, W.-Y. Hsiao, L.-C. Yang, and Y. Yang, "MuseGAN: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment," in *Proc. AAAI Conf. Artif. Intell.*, Apr. 2018, vol. 32, no. 1, pp. 34–41. Accessed: May 14, 2022. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/11312

[14] A. Nadeem, M. Naveed, M. Islam Satti, H. Afzal, T. Ahmad, and K.-I. Kim, "Depression detection based on hybrid deep learning SSCL framework using self-attention mechanism: An application to social networking data," *Sensors*, vol. 22, no. 24, p. 9775, Dec. 2022.

[15] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," Sep. 2016, *arXiv:1609.03499*.

[16] A. van den Oord, N. Kalchbrenner, L. Espeholt, K. kavukcuoglu, O. Vinyals, and A. Graves, "Conditional image generation with PixelCNN decoders," in *Advances in Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates, 2016. Accessed: May 30, 2023. [Online]. Available: https://papers.nips.cc/paper_files/paper/2016/hash/b1301141feffabac455e1f90a7de2054-Abstract.html

[17] *The MagnaTagATune Dataset | City University MIRG*. Accessed: May 30, 2023. [Online]. Available: https://mirg.city.ac.uk/codeapps/the-magnatagatune-dataset

[18] J. Copet, F. Kreuk, I. Gat, T. Remez, D. Kant, G. Synnaeve, Y. Adi, and A. Défossez, "Simple and controllable music generation," Jun. 2023, *arXiv:2306.05284*.

[19] A. Défossez, J. Copet, G. Synnaeve, and Y. Adi, "High fidelity neural audio compression," Oct. 2022, *arXiv:2210.13438*.

[20] N. Zeghidour, A. Luebs, A. Omran, J. Skoglund, and M. Tagliasacchi, "SoundStream: An end-to-end neural audio codec," *IEEE/ACM Trans. Audio, Speech, Language, Process.*, vol. 30, pp. 495–507, Nov. 2022, doi: 10.1109/TASLP.2021.3129994.

[21] S. Forsgren and H. Martiros. (2022). *Riffusion—Stable Diffusion for Real-Time Music Generation*. [Online]. Available: https://riffusion.com/about

[22] F. Schneider, O. Kamal, Z. Jin, and B. Schölkopf, "Moûsai: Text-to-music generation with long-context latent diffusion," 2023, *arXiv:2301.11757*.

[23] A. Agostinelli, T. I. Denk, Z. Borsos, J. Engel, M. Verzetti, A. Caillon, Q. Huang, A. Jansen, A. Roberts, M. Tagliasacchi, M. Sharifi, N. Zeghidour, and C. Frank, "MusicLM: Generating music from text," 2023, *arXiv:2301.11325*.

[24] Q. Huang, D. S. Park, T. Wang, T. I. Denk, A. Ly, N. Chen, Z. Zhang, Z. Zhang, J. Yu, C. Frank, J. Engel, Q. V. Le, W. Chan, Z. Chen, and W. Han, "Noise2Music: Text-conditioned music generation with diffusion models," 2023, *arXiv:2302.03917*.

[25] Y.-A. Chung, Y. Zhang, W. Han, C.-C. Chiu, J. Qin, R. Pang, and Y. Wu, "W2v-BERT: Combining contrastive learning and masked language modeling for self-supervised speech pre-training," Sep. 2021, *arXiv:2108.06209*.

[26] Q. Huang, A. Jansen, J. Lee, R. Ganti, J. Yue Li, and D. P. W. Ellis, "MuLan: A joint embedding of music audio and natural language," Aug. 2022, *arXiv:2208.12415*.

[27] Z. Borsos, R. Marinier, D. Vincent, E. Kharitonov, O. Pietquin, M. Sharifi, D. Roblek, O. Teboul, D. Grangier, M. Tagliasacchi, and N. Zeghidour, "AudioLM: A language modeling approach to audio generation," Sep. 2022, *arXiv:2209.03143*.

[28] Mubert-Text-to-Music. (Jun. 22, 2023). *Jupyter Notebook. Mubert: Music Powered by AI*. Accessed: Jun. 23, 2023. [Online]. Available: https://github.com/MubertAI/Mubert-Text-to-Music

[29] L.-C. Yang, S.-Y. Chou, and Y.-H. Yang, "MidiNet: A convolutional generative adversarial network for symbolic-domain music generation," Jul. 2017, *arXiv:1703.10847*.

[30] O. Goren, E. Nachmani, and L. Wolf. *A-Muze-Net: Music Generation by Composing the Harmony Based on the Generated Melody | SpringerLink*. Accessed: May 14, 2022. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-98358-1_44

[31] S. Walter, G. Mougeot, Y. Sun, L. Jiang, K.-M. Chao, and H. Cai, "MidiP-GAN: A progressive GAN approach to MIDI generation," in *Proc. IEEE 24th Int. Conf. Comput. Supported Cooperat. Work Design (CSCWD)*, May 2021, pp. 1166–1171, doi: 10.1109/CSCWD49262.2021.9437618.

[32] M. Mirza and S. Osindero, "Conditional generative adversarial nets," Nov. 2014, *arXiv:1411.1784*.

[33] L. Medsker and L. C. Jain, *Recurrent Neural Networks: Design and Applications*. Boca Raton, FL, USA: CRC Press, 1999.

[34] Q. Lou, "Music generation using neural networks," Stanford, CA, USA, 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:37905348

[35] O. Mogren, "C-RNN-GAN: Continuous recurrent neural networks with adversarial training," Nov. 2016, *arXiv:1611.09904*.

[36] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of GANs for improved quality, stability, and variation," Feb. 2017, *arXiv:1710.10196*.

[37] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training GANs," in *Advances in Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates, 2016. Accessed: May 14, 2022. [Online]. Available: https://proceedings.neurips.cc/paper/2016/hash/8a3363abe792db2d8761d6403605aeb7-Abstract.html

[38] K. Shmelkov, C. Schmid, and K. Alahari, "How good is my GAN?" presented at the Eur. Conf. Comput. Vis. (ECCV), 2018. Accessed: May 14, 2022. [Online]. Available: https://openaccess.thecvf.com/content_ECCV_2018/html/Konstantin_Shmelkov_How_good_is_ECCV_2018_paper.html

[39] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs trained by a two time-scale update rule converge to a local Nash equilibrium," in *Advances in Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates, 2017. Accessed: May 14, 2022. [Online]. Available: https://proceedings.neurips.cc/paper/2017/hash/8a1d694707eb0fefe65871369074926d-Abstract.html

[40] C. Donahue, J. McAuley, and M. Puckette, "Adversarial audio synthesis," Feb. 2018, *arXiv:1802.04208*.

[41] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," Jan. 2015, *arXiv:1511.06434*.

[42] R. Decorsière, P. L. Søndergaard, E. N. MacDonald, and T. Dau, "Inversion of auditory spectrograms, traditional spectrograms, and other envelope representations," *IEEE/ACM Trans. Audio, Speech, Language, Process.*, vol. 23, no. 1, pp. 46–56, Jan. 2015, doi: 10.1109/TASLP.2014.2367821.

[43] K. van den Broek, "MP3net: Coherent, minute-long music generation from raw audio with a simple convolutional GAN," Jan. 2021, *arXiv:2101.04785*.

[44] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," Dec. 2017, *arXiv:1701.07875*.

[45] P. Dhariwal, H. Jun, C. Payne, J. W. Kim, A. Radford, and I. Sutskever, "Jukebox: A generative model for music," Apr. 2020, *arXiv:2005.00341*.

[46] A. Odena, V. Dumoulin, and C. Olah, "Deconvolution and checkerboard artifacts," *Distill*, vol. 1, no. 10, p. e3, Oct. 2016, doi: 10.23915/distill.00003.

[47] T.-M. Hung, B.-Y. Chen, Y.-T. Yeh, and Y.-H. Yang, "A benchmarking initiative for audio-domain music generation using the freesound loop dataset," Aug. 2021, *arXiv:2108.01576*.

[48] A. Koretzky and N. Rajashekharappa, "A StyleGAN-2 inspired generative adversarial network for the PCA-controllable generation of drums samples for content-based retrieval," Splice, New York, NY, USA, Tech. Rep., 2021, p. 2.

[49] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," presented at the IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019. Accessed: May 14, 2022. [Online]. Available: https://openaccess.thecvf.com/content_CVPR_2019/html/Karras_A_Style-Based_Generator_Architecture_for_Generative_Adversarial_Networks_CVPR_2019_paper.html

[50] J.-Y. Liu, Y.-H. Chen, Y.-C. Yeh, and Y.-H. Yang, "Unconditional audio generation with generative adversarial networks and cycle regularization," in *Proc. Interspeech*, Oct. 2020, pp. 1997–2001, doi: 10.21437/Interspeech.2020-1137.

[51] E. Richardson and Y. Weiss, "On GANs and GMMs," in *Advances in Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates, 2018. Accessed: May 14, 2022. [Online]. Available: https://proceedings.neurips.cc/paper/2018/hash/0172d289da48c48de8c5ebf3de9f7ee1-Abstract.html

[52] P. L. T. Neves, J. Fornari, and J. B. Florindo, "Self-attention generative adversarial networks applied to conditional music generation," *Multimedia Tools Appl.*, vol. 81, no. 17, pp. 24419–24430, Mar. 2022, doi: 10.1007/s11042-022-12116-7.

[53] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," in *Proc. 36th Int. Conf. Mach. Learn.*, Jan. 2018, pp. 7354–7363. Accessed: May 14, 2022. [Online]. Available: https://proceedings.mlr.press/v97/zhang19d.html
[54] S. Zhao, Z. Liu, J. Lin, J.-Y. Zhu, and S. Han, "Differentiable augmentation for data-efficient GAN training," in *Advances in Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates, 2020, pp. 7559–7570. Accessed: May 14, 2022. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/55479c55ebd1efd3ff125f1337100388-Abstract.html
[55] *Magenta*. Accessed: May 14, 2022. [Online]. Available: https://magenta.tensorflow.org/
[56] D. D. Johnson, "Generating polyphonic music using tied parallel networks," in *Computational Intelligence in Music, Sound, Art and Design* (Lecture Notes in Computer Science), J. Correia, V. Ciesielski, and A. Liapis, Eds. Cham, Switzerland: Springer, 2017, pp. 128–143, doi: 10.1007/978-3-319-55750-2_9.
[57] Y. Yan, E. Lustig, J. VanderStel, and Z. Duan, "Part-invariant model for music generation and Harmonization," in *Proc. ISMIR*, Jan. 2018, pp. 204–210.
[58] S. Lyu, A. Zhang, and R. Song, "Dual-track music generation using deep learning," May 2020, *arXiv:2005.04353*.
[59] S. Vasquez and M. Lewis, "MelNet: A generative model for audio in the frequency domain," Jun. 2019, *arXiv:1906.01083*.
[60] A. Nayebi and M. Vitelli, "GRUV: Algorithmic music generation using recurrent neural networks," in *Deep Learning for Natural Language Processing*. Stanford, CA, USA, 2015, p. 6.
[61] A. Huang and R. Wu, "Deep learning for music," Jun. 2016, *arXiv:1606.04930*.
[62] V. Kalingeri and S. Grandhe, "Music generation with deep learning," Dec. 2016, *arXiv:1612.04928*.
[63] D. Eck and J. Schmidhuber, "A first look at music composition using LSTM recurrent neural networks," Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale, Lugano, Switzerland, Tech. Rep. IDSIA-07-02, 2002, p. 11.
[64] B. L. Sturm, J. Felipe Santos, O. Ben-Tal, and I. Korshunova, "Music transcription modelling and composition using deep learning," Apr. 2016, *arXiv:1604.08723*.
[65] R. Guo, I. Simpson, T. Magnusson, C. Kiefer, and D. Herremans, "A variational autoencoder for music generation controlled by tonal tension," Oct. 2020, *arXiv:2010.06230*.
[66] E. Chew, "The spiral array: An algorithm for determining key boundaries," in *Music and Artificial Intelligence* (Lecture Notes in Computer Science), C. Anagnostopoulou, M. Ferrand, and A. Smaill, Eds. Berlin, Germany: Springer, 2002, pp. 18–31, doi: 10.1007/3-540-45722-4_4.
[67] E. Chew, "Mathematical and computational modeling of tonality," in *International Series in Operations Research & Management Science*, vol. 204. Boston, MA, USA: Springer, 2014, doi: 10.1007/978-1-4614-9475-1.
[68] D. Herremans and E. Chew, "Tension ribbons: Quantifying and visualising tonal tension," in *Proc. 2nd Int. Conf. Technol. Music Notation Represent.*, Cambridge, U.K., vol. 2, May 2016, pp. 8–18.
[69] S. Dieleman, A. van den Oord, and K. Simonyan, "The challenge of realistic music generation: Modelling raw audio at scale," in *Advances in Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates, 2018. Accessed: May 30, 2023. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2018/hash/3e441eec3456b703a4fe741005f3981f-Abstract.html
[70] N. Kotecha, "Bach2Bach: Generating music using a deep reinforcement learning approach," Dec. 2018, *arXiv:1812.01060*.
[71] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing Atari with deep reinforcement learning," Dec. 2013, *arXiv:1312.5602*.
[72] H. Liu, X. Xie, R. Ruzi, L. Wang, and N. Yan, "RE-RLTuner: A topic-based music generation method," in *Proc. IEEE Int. Conf. Real-time Comput. Robot. (RCAR)*, Jul. 2021, pp. 1139–1142, doi: 10.1109/RCAR52367.2021.9517538.
[73] R. Manzelli, V. Thakkar, A. Siahkamari, and B. Kulis, "An end to end model for automatic music generation: Combining deep raw and symbolic audio networks," in *Proc. Musical Metacreation Workshop, 9th Int. Conf. Comput. Creativity*, 2018, p. 6.
[74] G. Mittal, J. Engel, C. Hawthorne, and I. Simon, "Symbolic music generation with diffusion models," Nov. 2021, *arXiv:2103.16091*.
[75] A. P. Roberts, J. Engel, C. Raffel, C. Hawthorne, and D. Eck, "A hierarchical latent vector model for learning long-term structure in music," in *Proc. 35th Int. Conf. Mach. Learn.*, Jan. 2018, pp. 4364–4373. Accessed: May 30, 2023. [Online]. Available: https://proceedings.mlr.press/v80/roberts18a.html
[76] G. Mariani, I. Tallini, E. Postolache, M. Mancusi, L. Cosmo, and E. Rodolà, "Multi-source diffusion models for simultaneous music generation and separation," Feb. 2023, *arXiv:2302.02257*.
[77] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," Dec. 2020, *arXiv:2006.11239*.
[78] P. Zhu, C. Pang, Y. Chai, L. Li, S. Wang, Y. Sun, H. Tian, and H. Wu, "ERNIE-music: Text-to-waveform music generation with diffusion models," Feb. 2023, *arXiv:2302.04456*.
[79] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *J. Mach. Learn. Res.*, vol. 21, no. 140, pp. 1–67, Jan. 2019.
[80] K. Kilgour, M. Zuluaga, D. Roblek, and M. Sharifi, "Fréchet audio distance: A metric for evaluating music enhancement algorithms," Jan. 2018, *arXiv:1812.08466*.
[81] W.-Y. Hsiao, J.-Y. Liu, Y.-C. Yeh, and Y. Yang, "Compound word transformer: Learning to compose full-song music over dynamic directed hypergraphs," in *Proc. AAAI Conf. Artif. Intell.*, May 2021, vol. 35, no. 1, pp. 178–186, doi: 10.1609/aaai.v35i1.16091.
[82] A. Kaushal Keshari, "Multi-genre music transformer—Composing full length musical piece," Jan. 2023, *arXiv:2301.02385*.
[83] P. Sarmento, A. Kumar, Y.-H. Chen, C. Carr, Z. Zukowski, and M. Barthet, "GTR-CTRL: Instrument and genre conditioning for guitar-focused music generation with transformers," in *Proc. 12th Int. Conf. Artif. Intell. Music, Sound, Art Design (EvoMUSART)*, Brno, Czech Republic. Berlin, Germany: Springer-Verlag, Apr. 2023, pp. 260–275, doi: 10.1007/978-3-031-29956-8_17.

ROHAN MITRA (Member, IEEE) received the double degree in computer science and mathematics (data science) from American University of Sharjah (AUS), United Arab Emirates, in 2023. He works as a Machine Learning Engineer and a part-time Research Assistant. Since 2020, he has been working as an Undergraduate Research Assistant in machine learning with the Department of Computer Science and Engineering, the Department of Mathematics, and the Department of Mass Communication. His research interests include artificial intelligence, deep learning, computer vision, and medical imaging. His awards and honors include his membership to the Eta-Kappa-Nu Honor Society, the Engineering Honors Society, and the Alpha Lambda Delta Honor Society at American University of Sharjah. Moreover, he has also received a 100% scholarship for the entirety of the double degree at AUS, as well as been awarded an Undergraduate Research Grant for further research.

IMRAN ZUALKERNAN (Member, IEEE) received the B.S. (Hons.) and Ph.D. degrees in computer science from the University of Minnesota, Minneapolis, in 1983 and 1991, respectively. He was an Assistant Professor at the Department of Computer and Electrical Engineering, The Pennsylvania State University, from 1992 to 1995. He was a Principal Design Engineer at AMCS Inc., Chanhassen, MN, USA, from 1995 to 1998. He was the Chief Executive Officer at Askari Information Systems, from 1998 to 2000, and the Chief Technology Officer at Knowledge Platform, Inc., Singapore, from 2000 to 2003. In 2003, he joined American University of Sharjah, United Arab Emirates, where he is currently working as a Professor and the Head of Computer Science and Engineering. He is the author or co-author of more than 230 peer-reviewed articles.

Open Access funding provided by 'American University of Sharjah' within the CRUI CARE Agreement