A Survey on Deep Learning for Symbolic Music Generation: Representations, Algorithms, Evaluations, and Challenges

SHULEI JI, XINYU YANG, and JING LUO, Xi'an Jiaotong University

Significant progress has been made in symbolic music generation with the help of deep learning techniques. However, the tasks covered by symbolic music generation have not been well summarized, and the evolution of generative models for the specific music generation task has not been illustrated systematically. This paper attempts to provide a task-oriented survey of symbolic music generation based on deep learning techniques, covering most of the currently popular music generation tasks. The distinct models under the same task are set forth briefly and strung according to their motivations, basically in chronological order. Moreover, we summarize the common datasets suitable for various tasks, discuss the music representations and the evaluation methods, highlight current challenges in symbolic music generation, and finally point out potential future research directions.

CCS Concepts: • **Applied computing** → **Sound and music computing;** • **Computing methodologies** → **Artificial intelligence**;

Additional Key Words and Phrases: Symbolic music generation, task-oriented survey, deep learning, symbolic music representations, music evaluation methods

ACM Reference format:
Shulei Ji, Xinyu Yang, and Jing Luo. 2023. A Survey on Deep Learning for Symbolic Music Generation: Representations, Algorithms, Evaluations, and Challenges. *ACM Comput. Surv.* 56, 1, Article 7 (August 2023), 39 pages.
https://doi.org/10.1145/3597493

1 INTRODUCTION

Music is widely regarded as a universal language. Automatic music generation has a long history that has been evolving for over 60 years. During this period, different methods, including but not limited to grammar rules, probability models, and neural networks, have been utilized in numerous studies on various datasets aiming at different generation tasks. With the upsurge of deep learning, automatic music generation has become a very active area. Some methods have generated promising results and have been applied to human-computer interaction composition tools and commercial composition software.

Symbolic music generation refers to representing music as a sequence of symbols, followed by feature learning and generative modeling. Since musical scores are highly symbolic and abstract, symbolic music generation is also considered score generation. The studies on score generation

Authors' address: S. Ji, X. Yang (corresponding author), and J. Luo, School of Computer Science and Technology, Xi'an Jiaotong University, No. 28, Xianning West Road, Xi'an, Shaanxi, 710049, China, emails: taylorji@stu.xjtu.edu.cn, yxyphd@mail.xjtu.edu.cn, luojingl@stu.xjtu.edu.cn.

0360-0300/2023/08-ART7 $15.00
https://doi.org/10.1145/3597493

have been the most abundant and have lasted the longest, evolving from simple melody generation to polyphonic generation, accompaniment arrangement, style transfer, interactive generation, music inpainting, and so on. Furthermore, some performance characteristics can also be symbolized to endow music with greater expressiveness and vividness. In contrast, music audio synthesis methods model the continuous audio signals directly.

Human composers typically follow a creative process that proceeds from abstract to concrete, from the whole to the details, e.g., creating a melody based on harmony or motivation. When simulating with computers, the music creation process is first decomposed into some subtasks, each solved individually and then connected to create a complete musical piece. Currently, most research on automatic music generation aims at tackling specific subtasks, while producing a complete and satisfactory song at a time by computers remains challenging.

Several papers have reviewed the field of automatic music generation. Nierhaus' book [1] provided a wide range of methods for algorithmic composition, including early **artificial intelligence (AI)** techniques. Fernández and Vico [2] comprehensively introduced algorithmic composition using AI methods, focusing on the origin of algorithmic composition and various **machine learning (ML)** methods. Liu and Ting [3] divided composition into three parts (melody, accompaniment, and musical form) and provided an overview of computational intelligence techniques used in automatic music generation, focusing on evolutionary algorithms and traditional neural networks. Herremans et al. [4] reviewed different music generation systems from the perspective of functional taxonomy. Briot et al. [5, 6] comprehensively surveyed deep music generation, introduced distinct deep learning network architectures, and highlighted some challenges and directions. Briot [7] presented a reduced version of [5] with additional supplements, introducing some conceptual frameworks to analyze the concepts and dimensions involved in music generation and many effective deep learning-based systems. Widmer et al. [8] and Kirke et al. [9] conducted extensive and detailed investigations in computational expressive music performance.

However, most surveys classified the research on automatic music generation from the perspective of network architectures or algorithms, with some illustrating different music generation tasks using the same network architecture. These surveys may be clear enough in introducing methods, but researchers working on a specific music generation subtask may prefer to know more about work related to their task. The taxonomy of [3] and [4] based on the essential elements of music composition is relatively coarse-grained from the perspective of tasks. Moreover, these reviews do not clearly distinguish between the symbolic and the audio, with some involving audio elements such as timbre. In contrast, this paper has a more fine-grained division for symbolic music generation to help researchers quickly understand a task within this area, as shown in Figure 1.

According to whether the generation has condition, control, performance characteristic, or interaction with people, symbolic music generation is divided into five categories: generation from scratch, conditional generation, controllable generation, performance generation, and interactive generation. Notably, both generation from scratch and conditional generation can be controlled, performance generation covers the first three types of generation tasks, and interactive generation often involves conditional and controllable generation. Each generation task is further divided into finer-grained subtasks. We summarize various methods under the same generation subtask so that researchers can quickly grasp the current situation and learn from other methods, which brings great convenience for follow-up studies. One thing that needs to be emphasized again is that this paper focuses on research using deep learning techniques.

1.1 History

There were many attempts to automate the music composition process long before the artificial neural networks era. A famous early example is the Musical Dice Game, where small fragments of

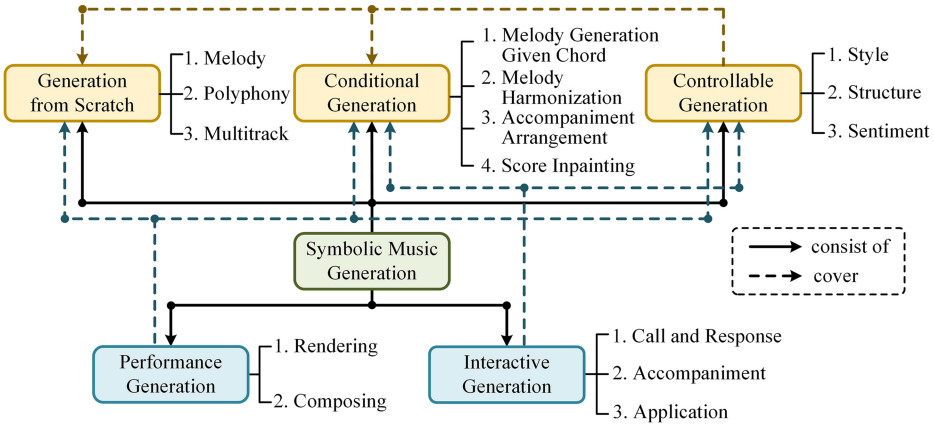

Fig. 1. Task-oriented taxonomy of symbolic music generation.

music are randomly re-ordered by rolling a dice to create a musical piece [4]. The composer John Cage continued the idea of chance-inspired composition. He created Atlas Eclipticalis by randomly placing translucent paper on a star chart and tracing the stars as notes [10]. Since the first computer was invented, computer-generated music has been appearing. To our knowledge, the first computer-generated music was born in 1957 through sound synthesis software developed by Bell Labs [5]. In the same year, the first musical score "The Illiac Suite" was created by computer with the help of stochastic models and rules [13]. During this period, Pinkerton's work [11] attempted to generate melodies with the computer, while Brooks et al. [12] established a Markov transition model based on a small music corpus. In 1964, the composer Koenig implemented the PROJECT1 algorithm, using serial composition and other techniques (such as Markov chains) to automatically generate music [14].

Since then, research on algorithmic composition [1] has emerged in an endless stream. Traditional automatic music generation algorithms are typically divided into the following three categories. The first category is rule-based music generation, which involves designing specific rules or grammars for algorithmic composition. The drawback of this method is that custom rules need to be created for different types of music, and the composition process often requires consistent intervention and fine-tuning during post-processing. The second category is the probability model, including the Markov Model and **Hidden Markov Models (HMM)**. Farbood and Schöner [15] were the first to use HMMs for algorithmic composition and trained a second-order HMM to generate Palestrina-style first-species counterpoints. However, such models can only produce subsequences in the original corpus. Additionally, these models lack memory, so the conditional probability may fluctuate significantly due to the difference in input data each time. The third category is neural networks that learn musical features. Todd [16] was the first to use **recurrent neural network (RNN)** to generate monophonic music note by note by predicting note pitches and durations. However, the early RNN has problems such as vanishing gradients, which makes it difficult to generate musical structures with long-term consistency.

Apart from the methods mentioned above, some studies adopted other techniques to compose music automatically and achieved promising results, like evolutionary algorithms (such as genetic algorithms), intelligent agents, and random fields. For example, GenJam [17] is a genetic algorithm-based model that creates jazz solos on a given chord progression. Cope's **Experiments in Musical Intelligence (EMI)** program [18], aided by the searching agent, successfully composed music in the styles of many famous composers such as Chopin, Bach, and Rachmaninoff. Lavrenko and

Pickens [19] adopted random fields to model polyphonic music. However, evolutionary algorithms for automatic music generation possess some deficiencies. First, the music generation process is too abstract to follow, making it challenging to extract valuable music ideas. Second, setting the fitness function is difficult [20]. Moreover, these algorithms attempt to minimize the search space using simple music representations, resulting in a loss of music quality [21].

1.2 Deep Symbolic Music Generation

Deep learning algorithms have become the mainstream method for automatic music generation. Deep symbolic music generation takes discrete tokens as input and uses deep neural networks to generate music automatically. This section will review the contributions of deep learning network architectures in symbolic music generation in recent years, but we do not deny the value of non-deep learning methods.

RNN is a valid model for learning sequential data, which is the first neural network architecture for music generation. As early as 1989, Todd [16] used RNN to generate monophonic melodies for the first time. However, due to the vanishing gradient problem, it is difficult for RNNs to store long historical information about sequences. To solve this problem, Hochreiter and Schmidhuber [22] designed a special RNN architecture—**long short-term memory (LSTM)**, to help the network memorize and retrieve information in the sequence. In 2002, Eck and Schmidhuber [23] used LSTM to create music for the first time, i.e., improvising blues music with good rhythm and reasonable structure based on a short recording. Boulanger-Lewandowski et al. [24] proposed the RNN-RBM model in 2012, which was superior to the traditional polyphonic music generation model but still had difficulty capturing the music structure with long-term dependency. In 2016, the Magenta team of Google Brain proposed Melody RNN models [25], further improving the ability of RNNs to learn long-term structures. Later, Hadjeres and Nielsen [26] proposed Anticipation-RNN that allows for imposing user-defined position constraints. Johnson [27] proposed TP-LSTM-NADE and BALSTM, which utilized a set of parallel, tied-weight RNNs to compose polyphonic music while preserving the translation invariance of the dataset.

With the ongoing advancement of deep learning technology, powerful deep generative models such as **variational auto-encoder (VAE), generative adversarial network (GAN)**, and Transformer have gradually emerged. The MusicVAE [28] proposed by Roberts et al. is a hierarchical VAE that can capture the long-term structure of polyphonic music and has eminent interpolation and reconstruction performance. Yang et al. [29] and Dong et al. [30] demonstrated the ability of **convolutional neural network (CNN)**-based GANs in music generation. Specifically, Yang et al. [29] proposed a GAN-based model called MidiNet to generate melody one bar (measure) after another and devised a novel conditional mechanism to generate the current bar conditioned on chords. To the best of our knowledge, the MuseGAN model proposed by Dong et al. [30] is the first model that can generate multi-track polyphonic music. Yu et al. [31] successfully applied RNN-based GAN to music generation for the first time by combining reinforcement learning. Huang et al. [32] were the first to use Transformer to create music with a longer-term structure successfully. Donahue et al. [33] adopted Transformer to generate multi-instrument music and proposed a pretraining technology based on transfer learning. Huang et al. [34] exploited the language model Transformer XL as the sequence model to generate popular piano music.

Computer composition and computer-aided composition have been very active fields of commercial software development with a broad spectrum of applications. For instance, PG Music's Band-in-a-Box[1] and Technimo's iReal Pro[2] can generate multi-instrument music based on

[1] https://www.pgmusic.com/.
[2] https://irealpro.com/.

user-specified chord scales. Google's Magenta project[3] is one of the most excellent research projects focusing on automatic music generation. It has developed multiple AI-based tools to assist musicians in being more creative [35], such as Melody Mixer, Beat Blender, and so on. Furthermore, some startups focusing on using AI to create music have emerged over the past few years. Jukedeck (acquired by Bytedance in 2019) and Amper Music[4] concentrate on creating royalty-free soundtracks for content creators such as video producers. Hexachords' Orb Composer[5] provides a more complex environment for creating music, making it more suitable for music professionals. Aiva Technologies designed and developed AIVA[6] to create music for movies, businesses, games, and TV shows. These are just a few of the many startups venturing into this uncharted territory. What all of these companies have in common is to encourage cooperation between humans and machines. All the programs are not developed to replace humans but to make non-musicians more creative and musicians more effective.

2 SYMBOLIC MUSIC REPRESENTATION

Symbolic music representations are usually discrete sequences containing musical elements such as pitch and duration.

2.1 Event

Musical instrument digital interface (MIDI) [36] is an industry-standard protocol that enables interoperability among various electronic instruments, software, and devices. It connects products from different companies, including digital instruments, computers, iPads, smartphones, and so on. MIDI carries the performance data and control information of notes. Therefore, many researchers symbolize music with MIDI-like events (such as Note On, Note Off, Time-Shift, etc.); we call this event-based music representation. The events can be serially arranged into a long sequence as individual tokens or the events of the same type can be combined into a compound word to reduce the sequence length. Conveniently, Fradet et al. [157] have developed a Python package named MidiTok to encode MIDI files into sequences of tokens to be used with sequential deep learning models. MidiTok features the most known MIDI tokenization methods [34, 40, 42, 81, 113], which will be introduced in the follow-up.

2.1.1 Individual Token. By ordering the note events according to the timestamps and the parts/tracks, music is often processed as 1D sequence data. However, the 1D sequence discards many aspects of the relationship among notes. For instance, it is difficult to indicate that the previous note is still playing at the beginning of the next note [38]. It is usually sufficient for score encoding to use the Note On, Note Off, and Pitch events, while performance encoding needs more events, such as Velocity and Tempo. Table 1 lists common events in some studies of symbolic music generation [34, 37, 65].

The simple MIDI-like representation [113] is very effective for capturing the pitch values of notes, but it has inherent limitations. Firstly, it cannot explicitly indicate the beginning of a bar or a beat in modeling the rhythm structure. Secondly, generating music with a stable rhythm using only Time Shift events is non-trivial, as the tokens need to be merged or split when adding or deleting notes. Meanwhile, the Note On and Note Off tokens need to be changed together, making the model inefficient [39]. Thirdly, the model finds it difficult to learn that the Note On and Note Off must appear in pairs, thus generating some hanging Note On without the corresponding Note Off.

[3] https://magenta.tensorflow.org/.
[4] http://www.ampermusic.com/.
[5] http://www.orb-composer.com/.
[6] http://www.aiva.ai.

Table 1. Musical Events

Event Type	Definition
Note On	indicates the start of a note, one for each pitch
Pitch	indicates the pitch value of a note
Note Off	indicates the end of a note, one for each pitch
Note Duration	inferred from the time gap between a Note On and the corresponding Note Off, by accumulating the Time Shift events in between
Time Shift	shifts the current time forward by the corresponding number of quantized time steps
Position/Sub-beat	points to different discrete locations in a bar
Bar	marks the bar lines
Piece Start	marks the start of a piece
Chord	one for each chord
Program/Instrument	sets the MIDI program number at the beginning of each track
Track	marks each track
Time Signature	marks the change of time signature
Note Velocity	sets the velocity for subsequent note-on events
Tempo	accounts for local changes in tempo

Fourthly, MIDI cannot explicitly express concepts of quarter notes or rests but can only represent the duration of note playing or absence. To address the above-mentioned problems, Huang et al. [34] proposed **REMI (REvamped MIDI-derived events)**, which used Note Duration instead of the Note Off event to promote the modeling of note rhythm and replace Time Shift events with Bar and Position tokens to provide a clear metric grid to simulate music. Bar indicates the beginning of a bar, and Position reveals certain positions in a bar. They also introduced a set of Tempo events to allow local tempo changes for each beat, providing expressive tempo freedom. In REMI's time grid, each Tempo is preceded by a Position event.

2.1.2 Compound Word. The 1D event-based representation treats each event as an individual token, making the representation sequence too long to capture sufficient information for song-level tasks. The average song length in REMI [34] encoding is 15,679, and even the Transformer struggles to cope with such long sequences [40]. Therefore, some studies group neighboring tokens into a compound word using different strategies, greatly shortening the sequence length. This kind of representation has grown in popularity with the widespread use of transformers in automatic music generation.

Before the development of compound word representation (CP), some studies attempted to represent several note events as a tuple [39, 41]. The **MUlti-track MIDI representation (MuMIDI)** [81] groups tokens of each note's attributes (i.e., pitch, duration, velocity) and includes other symbols such as bar, position, track, chord, and meta information. MuMIDI facilitates simultaneous multi-track generation in a single sequence and explicitly models the dependency of the notes from different tracks, which outperforms REMI [34] in modeling harmony. As different types of (musical) tokens may have different properties, Hsiao et al. [42] highlight the role of token types and group consecutive and related tokens into "compound words" (refer to Figure 2(c) in [42]), which greatly reduces the length of the token sequences. The compound words are partitioned into the note family N and the metric family M to facilitate modeling. The note family includes [pitch], [duration], and [velocity] events; the metric family contains [bar], [beat], [chord], and [tempo] events; and the [ignore] token is used to fill the missing token types per time step. Under compound word modeling, multiple tokens of various types are predicted at once in a single time

step. Later, Zeng et al. [40] designed a new music coding method called OctupleMIDI that encodes each note into a tuple with eight elements representing eight note characteristics, i.e., time signature, tempo, bar, position, instrument, pitch, duration, and velocity. OctupleMIDI greatly reduces the length of the note sequence, which is four times shorter than REMI and two times shorter than CP. Moreover, it supports changeable time signatures and tempos, which makes it simpler and more universal than previous coding methods. Similarly, Dong et al. [162] also encoded each musical event as a tuple of six variables (i.e., type, beat, position, pitch, duration, and instrument) to significantly shorten the sequence length of multitrack music.

2.2 Piano Roll

The piano roll representation is inspired by the automatic piano, which represents note events as a $h * w$ real-valued matrix, where h indicates the pitch range, which usually contains two additional entries to represent rest and note continuation, respectively. The value of w represents the number of time steps, which depends on the beat resolution. Multi-track music can be represented by multiple matrices. Typically, the piano roll is binary, meaning each position in the matrix only indicates whether the note is playing. But sometimes, its value has other meanings. For example, in DeepJ [43], a velocity value is placed at the playing position of a note. Furthermore, in [44], each position stores a list containing three note attributes: play, articulate, and dynamic. The piano roll expresses the music score as a 2D matrix that can be regarded as an image, so some call it the image-like representation [34]. With the help of pretty_midi and Pypianoroll Python toolkits, MIDI files can be easily converted into piano roll representations.

The piano roll representation is easy to understand and has been used widely. However, compared with the MIDI-like representation, it lacks note-off information, which makes it difficult to distinguish between long notes and continuous repeated short notes (like a quarter note and two eighth notes). There are several ways to solve this problem [7]: (1) introduce hold/replay as the dual representation of notes [43]; (2) mark the beginning or the ending of the note [16, 45]; and/or (3) use the special hold symbol '_' to replace the note continuation [46]. Furthermore, Angioloni et al. [47] proposed a new piano roll representation, the Conlon piano roll (PR^c), which can explicitly represent the duration. Another issue of piano roll representation is that it quantifies time with a specific beat resolution, eliminating the possibility of the model learning complicated rhythms and expressive timings [35]. Additionally, the larger temporal resolution brings a longer sequence length, which may lead to a scarcity of long-term structure in the generated music.

2.3 Sequence

The sequence-based representation exploits one or several sequences to respectively represent musical elements such as pitch, duration, and chord. Different sequences usually have the same length, equal to the number of notes or the product of the beat number and the beat resolution. For instance, Genchel et al. [48] represented music as pitch, duration, chord, and bar position sequences. DeepBach [46] adopts several lists to describe Bach's four-part chorales, including the four voice lists and two metadata lists (rhythm and fermatas). However, they regard the 16th note as the smallest time unit, which makes it impossible to encode triplets. To overcome this limitation, Pati et al. [49] proposed to divide each beat into six uneven ticks to realize the coding of triplets while only increasing the sequence length by 1.5 times. Note that a special symbol is usually used to encode note continuation, e.g., "hold" and "_".

2.4 Text

The most common text representation for music is the ABC notation. It consists of two parts: header and body. The header contains information such as the reference number, title, time

signature, note length, and key. The body includes notes, bar lines, and the like. Each note is encoded into a token where the pitch class of a note is encoded into its corresponding English letter with the octave symbol and duration symbol, rest is represented by 'z', and the bars are separated by '|'. ABC notation is quite compact but can only represent monophonic melodies. For a more detailed description of ABC, see [50]. Sturm et al. [51] converted ABC text into a token vocabulary, where each token is composed of one or more characters for the following seven types (with examples in parentheses): meter ("M:3/4"), key ("K:Cmaj"), measure (":|" and "|1"), pitch ("C" and "^c'"), grouping ("(3"), duration ("2" and "/2"), and transcription ("<s>" and "<\s>").

2.5 Graph

Of particular interest, some researchers represent the music score as a graph. Jeong et al. [38] represented the score as a unique form of graph $G = (V, E)$ for the first time, where V and E represent nodes and edges, respectively. Each note is regarded as a node in the graph, and adjacent notes are connected as different types of edges according to their musical relationship in the score. They defined six types of edges: next, rest, set, sustain, voice, and slur; considered forward and backward directions as different types of edges; and added a self-connection to every note. In addition, Chuan and Herremans [52] extended the tonnetz, a graphical representation used by music theorists and musicologists, to transform music into a sequence of 2D tonnetz matrices and graphically encoded the musical relationship between pitches. Specifically, each node in the extended tonnetz network represents a pitch, and the nodes on the same horizontal line follow the circle-of-fifth ordering. The pitches in a matrix column preserve the interval of major third. The number of rows in the tonnetz matrix is determined by the range of pitches. Compared with the piano roll, the music generated by tonnetz has better pitch stability and more repetitive patterns.

3 TASK-ORIENTED SYMBOLIC MUSIC GENERATION

This section reviews previous work on deep symbolic music generation from the perspective of tasks and compares different deep learning methods under the same tasks basically in chronological order. Most but not all of the tasks in symbolic music generation are covered in this section, since some tasks have not been extensively studied in the literature. The related studies are divided into five categories: generating music from scratch, conditional generation with musical context constraints, controllable generation that allows for controlling the structure or attributes (e.g., style, emotion) of generated music, performance generation that adds performance characteristics to the score generation, and interactive generation with its applications. Note that we do not provide detailed explanations of specific deep network architectures (such as RNN, CNN, etc.), as they are not the focus of this paper.

3.1 Generation from Scratch

Generation from scratch refers to generating music (such as melody, polyphony, and multitrack music) without conditions. Some popular methods are shown in Table 2.

3.1.1 Melody. Melody is a series of notes organized by specific pitch variation rules and rhythm relations. RNN is the most commonly used and simplest model for generating melody. One of the most well-known symbolic melody generation methods is the Melody RNN model [25] developed by Google Brain's Magenta project. This model comprises a baseline RNN model, called the basic RNN, and two RNN model variants, namely the lookback RNN and the attention RNN, which are designed to generate longer-term music structures. However, RNN-based models can only generate music sequences from left to right, which are far from interactive and creative applications. To

Table 2. Methods for Generating Symbolic Music from Scratch

Task	Method	Year	Music Representation	Model Architecture	Description	Length of generated music
Melody	Melody RNN [25]	2016	Event	RNN	Lookback RNN; Attention RNN	Variable-length
	Anticipation-RNN [26]	2017	Sequence	RNN	User-defined positional constraints	Variable-length
	MusicVAE [28]	2018	Piano roll	RNN-based VAE	Long-term music structure; Hierarchical decoder	16 bars
	[53]	2021	Sequence	RNN-based EC^2-VAE	Long-term music representations; Hierarchical contextual constraints.	8 bars
	GLSR-VAE [54]	2017	Sequence	RNN-based VAE	Continuous attributes; Geodesic latent space regularization	2 bars
	[158]	2021	Sequence	β-VAE and AR-VAE	Continuous-valued attributes; Latent space disentanglement and regularization	1 bar
	SeqGAN [31]	2017	Sequence	RNN-CNN-based GAN	Reinforcement learning; Monte Carlo search	12.8s
	RL Tuner [56]	2017	Piano roll	RNN	Reinforcement learning; DQN algorithm	32 notes (16th)
Polyphony	DeepBach [46]	2016	Sequence	RNN	User-defined constraints	Variable-length
	[27]	2017	Piano roll	RNN	Transposition-invariance; Tied parallel LSTM-NADE; Bi-axial LSTM	Variable-length
	[132]	2018	Piano roll	CNN & RNN-based VAE	Bar-level CNN feature learning	8, 16 or 32 bars
	[59]	2019	Piano roll	RNN-based VAE	Latent normalizing flows; Non-autoregressive generation	Variable-length
	PianoTree VAE [39]	2020	Event	RNN-based VAE	Tree structured musical syntax	8 beats
	[161]	2022	Piano roll	VAE	Using only linear layers; bit-rate reduction	1 bar
	C-RNN-GAN [60]	2016	Event	RNN-based GAN	Continuous sequential data	Variable-length
	Music Transformer [32]	2019	Event	Transformer	Relative attention mechanism; Compelling long-term structure	Minute-long
	MuseNet [61]	2019	Event	Transformer	Same general-purpose unsupervised technology as GPT-2	4-minute long
	[41]	2020	Event	Transformer	Adversarial learning; Novel computation of reward for each generated step (REGS)	Up to 1024 tokens
Multitrack Music	MuseGAN [30]	2018	Piano roll	CNN-based GAN	WGAN-GP; Multi-track interdependency and temporal structure	4 bars
	[63]	2018	Piano roll	CNN-based GAN	Binary neurons for binary-valued piano roll	4 bars
	DMB-GAN [64]	2019	Piano roll	GAN	Self-attention mechanism; Duel Multi-branches GAN architecture	4 bars
	[65]	2018	Event	RNN-based VAE	An extension of MusicVAE; Latent space manipulations	1 bar
	MNGANs [66]	2018	Piano roll	CNN-based GAN	Optimizing musicality and novelty alternately	4 bars
	LakhNES [33]	2019	Event	Transformer-XL	Cross-domain pre-training	Variable-length
	MMM [149]	2020	Event	Transformer	Concatenating time-ordered sequence of each track into a single sequence	4 or 8 bars
	MMT [162]	2023	Event	Transformer	New multitrack representation; New measure for musical self-attention	Up to 1,024 codes

solve this issue, Hadjeres and Nielsen [26] proposed Anticipation-RNN, which not only has the advantages of the RNN models but also allows the enforcement of user-defined position constraints.

Generative models like VAE and GAN have also been used for automatic music composition and combined with CNN and RNN to produce numerous variants. To address the problem that the recurrent VAE model has difficulty modeling the sequence with long-term structure, Roberts et al. [28] proposed the MusicVAE model, which employs a hierarchical decoder that generates each subsequence independently using the latent variables generated by the encoder. This model avoids the "posterior collapse" problem in recurrent VAE and has better sampling, interpolation, and reconstruction performance. However, this model is usually hard to train unless there is a large amount of data. Therefore, Wei and Xia [53] proposed a new model based on EC^2-VAE to learn long-term, phrase-level symbolic music representation through contextual constraints, which is the first model to achieve the disentanglement of long-term representations. Additionally, Hadjeres et al. [54] proposed GLSR-VAE to consider the continuous attributes of modeling data, which contains a **geodesic latent space regularization (GLSR)** to relate variations in the latent space to variations of the attributes. This model is the first specifically designed for continuous data attributes, although it requires differentiable calculations on data attributes and careful fine-tuning of hyperparameters. Also, for encoding continuous-valued attributes, Pati and Lerch [158] structured the latent space of VAE by using an attribute regularization loss to enforce a monotonic relationship between the attribute values and the latent code of the respective regularized dimension. Contrary to GLSR-VAE [54], the loss formulation proposed by Pati and Lerch [158] is agnostic to how the attributes are computed or obtained and only has two hyperparameters.

The vanilla GAN has limitations in generating discrete tokens, as the discrete output of the generator makes it difficult to transfer the gradient update of the discriminator to the generator, and the discriminator can only evaluate an entire sequence. To tackle these issues, Yu et al. [31] proposed SeqGAN, a sequence generation framework combined with **reinforcement learning (RL)**, which is the first work to extend GAN to generate discrete sequences. Although SeqGAN has proven its performance on several sequence generation tasks (such as poetry and music), it suffers from the problem of mode collapse [55]. Jacques et al. [56] proposed RL Tuner, a novel sequential learning method by combining ML and RL training for music generation. They utilized pre-trained RNN to supply part of the reward value and refined the sequence predictor with the imposed reward functions. In addition, the RL Tuner can explicitly correct the undesirable behavior of RNN, which could be helpful in a broad range of applications. Jacques et al. [57] further proposed Sequence Tutor, a general version of the RL Tuner, for sequence generation tasks, not just music generation.

3.1.2 Polyphony. A polyphony is composed of two or more independent melodies, which are combined organically to flow and unfold in a coordinated manner. Polyphonic music has sequential patterns between timesteps and harmonic intervals among simultaneous notes [44]. Therefore, generating polyphonic music is more complicated than generating simple melodies.

It may be certain that the first work with comprehensive consideration of polyphony is the study of Boulanger-Lewandowski et al. [24]. They proposed the RNN-RBM model to learn the probability rules of harmony and rhythm from polyphony scores with various complexity. To model polyphonic music, especially the four-part pieces, Hadjeres et al. [46] proposed an LSTM-based model called DeepBach, which does not sample from left to right, but models each part separately and permits users to impose unary constraints. However, RNN models don't have transposition-invariance, meaning the network learns the absolute relationship among notes rather than the relative relationship. Accordingly, Johnson [27] proposed two network architectures, TP-LSTM-NADE and BALSTM, with transposition-invariance that utilized a set of parallel, tied-weight

recurrent networks for the prediction and composition of polyphonic music. Although the BALSTM model proposed by Johnson possesses translation-invariance, it cannot maintain the consistency of output styles when training with multiple styles, leading to style shifts in one music piece. The DeepJ model [43] introduced in Section 3.3.1 solved this issue by adding style conditions at each layer.

In 2015, Fabius et al. [58] first proposed the VAE and applied it to generate polyphonic game songs. Since then, some follow-up studies on VAE-based polyphony generation have emerged. Combining the strengths of CNN and VAE, Koh et al. [132] proposed a convolutional-variational RNN-based encoder-decoder architecture to generate polyphonic music, where the CNN captures bar-level music structure, and the variational RNN generates novel music sequences bay by bar. Ziegler and Rush [59] studied the normalized flow in the discrete environment within the VAE framework. The flow modeled the continuous representation of discrete data through the prior model and realized the generation of polyphonic music without an autoregressive likelihood. Wang et al. [39] proposed a novel tree-structure extension model called PianoTree VAE to generate polyphonic counterpoint in the context of music representation learning. Additionally, Dubnov et al. [161] constructed a vanilla polyphonic VAE using only linear layers and then applied the bit-rate reduction to latent representations for studying temporal structure at different reductions. In contrast, GAN has not been applied much in generating polyphonic music from scratch. Mogren [60] proposed a novel generative adversarial model named C-RNN-GAN based on continuous sequential data to generate polyphonic music. Unlike the symbolic music representation commonly adopted in RNN models, this method trained a highly flexible and expressive model with fully continuous sequential data, including tone lengths, frequencies, intensities, and timing.

Transformer-based models have been widely applied in polyphonic music generation. Huang et al. [32] combined the relative attention mechanism into the Transformer and proposed the Music Transformer, which can generate long-term music structures of 2,000 tokens scale, generate continuations that coherently elaborate on a given motif, and generate accompaniments given the melody. It is the first successful application of Transformer to generate music with long-term structures. Payne [61] created MuseNet based on GPT-2, which can generate 4-minute music with ten different instruments combining various styles. MuseNet has full attention in the context of 4,096 tokens, which may be one of the reasons why it can remember the medium- and long-term structure within segments. Both Music Transformer and MuseNet are based on decoder-only transformers. Although these models using the teacher-forcing training strategy can produce some good pieces, in most cases, the generated music pieces are of poor musicality and structure. Therefore, Zhang [41] proposed a novel adversarial Transformer to generate music pieces with high musicality. The generation of long sequences is guided by adversarial objectives, which offer a powerful regularization to force the Transformer to focus on global and local structure learning.

3.1.3 Multitrack Music. Multitrack music belongs to polyphonic music, which usually consists of multiple interdependent tracks with their own temporal dynamics and instruments.

The MuseGAN model proposed by Dong et al. [30] is one of the most well-known models, which aims at generating multitrack polyphonic music with harmonic and rhythmic structure, multitrack interdependence, and temporal structure. However, the music generated by MuseGAN has multiple shortcomings, e.g., the number of qualified notes generated is much less than that of the training data, and the generated music contains many fragmented notes. Many follow-up studies have been conducted since the introduction of MuseGAN. In view of the problem that existing models need operations such as hard threshing or Bernoulli sampling to obtain binary piano rolls, Dong and Yang [63] utilized binary neurons to generate binary piano rolls directly based on MuseGAN. This method improved the ratio of qualified notes and reduced the number

of fragmented notes in the generated music compared with MuseGAN. The above GAN-based multitrack music generation methods use convolution to extract features, which cannot effectively learn temporal features. Consequently, Guan et al. [64] proposed a **Dual Multi-Branches GAN architecture (DMB-GAN)** with the self-attention mechanism, which can extract temporal and spatial features simultaneously. The DMB-GAN generates about twice as many effective notes as MuseGAN while training each batch in 1/5 the time of MuseGAN.

Unlike the fixed quintet generated by MuseGAN, Simon et al. [65] proposed an extension of the MusicVAE model, which can generate multitrack music with arbitrary instruments. The system can generate music from scratch and manipulate existing music through latent space. However, the model can only represent an individual bar and has difficulty generating the long-term structure. Adding chord conditions can alleviate this issue to some extent. Different from most existing work focusing on the imitation of musicality, Chen et al. [66] emphasized the imitation of artistic creation, i.e., novelty. They proposed the **Musicality-Novelty Generative Adversarial Nets (MNGANs)** for algorithmic composition, which is the first GAN aiming at music novelty. Two adversarial networks using the same generator alternately optimize the musicality and novelty of music. Since novelty is hard to define and quantify, they put forward a new model called novelty game to maximize the minimal distance between machine-created and any human-created music samples in the novelty space, while all human-composed music samples are far away from each other.

Donahue et al. [33] proposed a Transformer-based model LakhNES to generate multi-instrument music. They mainly presented a pre-training technology based on transfer learning, i.e., pre-train the model on one dataset (Lakh MIDI) and then fine-tune the model to another (NES-MDB) to improve the model performance. Unlike the constraint of fixed-length music in MuseGAN [30], LakhNES can generate arbitrary-length sequences. Ens and Pasquier [149] proposed a generative system named **Multi-Track Music Machine (MMM)** based on Transformer, where each track is represented as a time-ordered sequence and multiple tracks are concatenated into a single sequence. MMM can achieve track-level or bar-level inpainting and can impose control over track instrumentation and note density. Moreover, Dong et al. [162] proposed **Multitrack Music Transformer (MMT)**, a decoder-only transformer model with multi-dimensional input and output spaces to generate multitrack music. MMT utilizes a new multitrack representation (introduced in Section 2.1.2) to generate longer sequences at a faster inference speed than MMM [149].

3.2 Conditional Generation

Apart from generating music from scratch, another method conditions the generation process of partial music on the given music context (melody, chord, etc.), which is called conditional generation. Table 3 summarizes some representative methods for conditional music generation tasks.

3.2.1 Melody Generation Given Chord. A chord is a harmonic collection of arbitrary pitches composed of three or more notes that are playing at the same time. A series of ordered chords is called the chord progression. The melody generation systems usually ignore any forms of explicit harmonic background, which is a crucial guide for note selection [48]. Motivated by this, some researchers try to generate melodies given chord information, hoping to bring better musical structure to the generated melodies.

Companies providing music services, such as Amper[4], usually perform arpeggios on basic chords to generate melody. Yang et al. [29] proposed a CNN-based GAN model named MidiNet to generate melodies one bar after another, where they fed chord progressions into the inversed version of the generator to generate the conditions for regulating the generated melody of the current bar. Given the role of chord progressions in guiding melody in pop songs, Zhu et al. [67]

Table 3. Methods for Conditional Music Generation

Task	Method	Year	Music Representation	Model Architecture	Description	Length of generated music
Melody Generation Given Chord	MidiNet [29]	2017	Piano roll	CNN-base GAN	Generating music one bar after another; Conditioner CNN	8 bars
	CRMCG [67]	2018	Sequence	RNN	Encoder-decoder framework	Variable-length
	JazzGAN [68]	2018	Piano roll or Event	RNN-based GAN	Improvising jazz melodies over chord progressions; Harmonic bricks for phrase segmentation	Variable-length
	BebopNet [70]	2020	Sequence	RNN	Monophonic harmony-constrained jazz improvisations; Personalized generation	Variable-length
	MINGUS [71]	2021	Sequence	Transformer	Two parallel transformers for pitch and duration, respectively	Up to 35 tokens
Melody Harmonization	[72]	2017	Sequence	RNN	BiLSTM; A chord per bar	Up to 16 bars
	MTHarmonizer [73]	2021	Sequence	RNN	Chord functions; A chord every half bar	4-32 bars
	[74]	2021	Sequence	RNN	Orderless NADE; Chord balancing; Blocked Gibbs sampling	Variable-length
	SurpriseNet [76]	2021	Sequence	RNN-based VAE	Conditional VAE; Surprise contour	Variable-length
	AutoHarmonizer [77]	2021	Sequence	RNN	Controllable harmonic rhythm	Variable-length
Accompaniment Arrangement	[80]	2018	Piano roll	GAN	Recurrent CNN model; three new symbolic-domain harmonic features	8 bars
	MICA [67]	2018	Sequence	RNN	Attention cell and MLP cell	Variable-length
	PopMAG [81]	2020	Event	Transformer	MuMIDI representation; Extra-long context	Up to 32 bars
Music Inpainting	Coconet [75]	2017	Piano roll	CNN	Orderless NADE; Blocked Gibbs Sampling	128 notes (16th)
	[82]	2018	Event	Transformer	Virtuosic piano performances; NoteTuple representation [167]	16 or 32 notes
	ES-Net [83]	2020	Piano roll	U-Net	Edit events; Note by note control	8 bars
	InpaintNet [49]	2019	Sequence	RNN-based VAE	Traversing latent space	2-6 bars
	Music SketchNet [84]	2020	Sequence	RNN-based VAE	SketchVAE, SketchInpainter and SketchConnector; User-specified pitch and rhythm snippets	4 bars
	CLSM [85]	2021	Sequence	Transformer & LSTM	Contextual latent space model; Positional constraints; Interpolations or variations	Up to 4 bars
	[86]	2021	Event	XLNet	A novel relative bar coding; Look-ahead onset prediction	Up to 128 notes
	[87]	2022	Event	Transformer	Masked pretraining and finetuning; Track level and bar level controls	Up to 16 bars

proposed XiaoIce Band, an end-to-end melody and arrangement generation framework for song generation, which includes a **Chord based Rhythm and Melody Cross-Generation Model (CRMCG)** to generate melody given chord progression. The CRMCG model generates better results than the Melody RNN [25] and the MidiNet [29]. To alleviate the problems in the jazz generation, such as frequent and diverse key changes, unconventional and off-beat rhythms, as well as flexibility with off-chord notes, Trieu and Keller [68] proposed a jazz melody improvisation method over the chord progression. The results show that the music generated by JazzGAN is more popular than the music generated by Magenta's ImprovRNN [69]. Hakimi et al. [70] adopted BebopNet, a music language model trained on a corpus of jazz improvisations by Bebop giants,

to achieve the monophonic, harmony-constrained jazz improvisations, which was the first work to generate personalized jazz solos using deep learning techniques. Madaghiele et al. [71] proposed a Transformer-based Seq2Seq model MINGUS to generate monophonic jazz melodies conditioned on other musical features such as harmonic structure and rhythmic properties. Experiments show that MINGUS performs better than the BebopNet in generating jazz melodies with harmonic consistency.

3.2.2 Melody Harmonization. Melody harmonization refers to generating harmonic accompaniments for given melodies [73], where chords play a crucial role. Lim et al. [72] used a two-layer BiLSTM and a fully connected layer to learn the correspondence between melody and chord sequence pairs, which achieved better performance than the HMM model in both quantitative and qualitative evaluation. But the model enforces that only one chord can be generated per measure, and the chord type must be the triad. Yeh et al. [73] pointed out two main defects of this model: the excessive use of common chords and the non-congruent phrasing between the melody and chords. To solve these issues, they proposed an extended BiLSTM model called MTHarmonizer, which predicts both the chord labels and the chord functions. Compared with the distribution of chord labels, the distribution of chord functions is relatively balanced, which is easier for model learning. Besides, since chord labels and functions are interdependent, adding a chord function as a target will inform the model of which chord labels share the same function and may thus be interchangeable. Sun et al. [74] took the melody and partially masked chord sequences as the input of a BiLSTM-based network to learn the missing parts of chord sequences. Inspired by Coconet [75], they used blocked Gibbs sampling in the inference stage. This method can generate 96 kinds of chords at most (including partial triads and sevenths) and stipulates that there are up to two chords per bar.

Interestingly, Chen et al. [76] proposed a CVAE-based user-controllable framework to harmonize a given melody based on surprise contours, where the surprise contour is derived from the negative log probability of the Markov chain. To our best knowledge, this is the first VAE-based melody harmonization work. One limitation of previous melody harmonization models is the fixed harmonic rhythm. To improve this deficiency, Wu et al. [77] proposed AutoHarmonizer to achieve harmonic rhythm-controllable melody harmonization, which can generate denser or sparser chord progressions through a new sampling method. The RNN-based AutoHarmonizer consists of two parts: a harmonic rhythm model that provides coarse-grained chord onset information and a chord model that generates specific chords based on the given melody and the corresponding harmonic rhythm sequence. Melody harmonization encourages human-computer interaction, which is especially suitable for novices [78]. Some human-computer interaction interfaces and tools have been developed for this task, such as MySong [79].

3.2.3 Accompaniment Arrangement. Arranging is the art of giving an existing melody musical variety. Liu and Yang [80] defined the lead sheet arrangement as a process that takes as input a lead sheet and generates as output piano rolls of multiple instruments to accompany the melody. They implemented the lead sheet arrangement with a novel conditional generation model based on convolutional GAN for the first time. Considering the harmony of music arrangement, Zhu et al. [67] proposed a **Multi-Instrument Co-Arrangement model (MICA)** to generate multi-track accompaniment for melody, which contains two cooperate cells, the attention cell and MLP cell, to integrate the information of other tracks when generating one of the tracks. The results demonstrated that the harmony among multiple tracks was improved significantly. Ren et al. [81] proposed a transformer-based model for pop music accompaniment generation called PopMAG, which generated five instrumental accompaniment tracks under the conditions of chords and melodies. PopMAG is superior to MuseGAN [30] in both subjective and objective evaluation.

The Music Transformer [32] mentioned before can also generate accompaniments conditioned on melodies and chords.

3.2.4 Music Inpainting. Early music generation models mostly assume that music is generated continuously, meaning the current music segment only depends on the previous music segments. This method is inconsistent with typical human composition practice, which is usually iterative and non-sequential. The chronological generation paradigm limits the degree of interactivity. Once generated, it is impossible to adjust the specific parts to meet the user's aesthetic sensibilities or compositional requirements. Music inpainting refers to completing the missing information in a piece of music, which is more in line with the human creative process. Pati et al. [49] define the music score inpainting problem as: given the past music context C_p and the future music context C_f, an inpainted sequence C_i is generated to connect C_p and C_f in a musically meaningful manner.

The previously mentioned Anticipation-RNNs [26] can selectively regenerate the notes at specific positions. One of the **Markov Chain Monte Carlo (MCMC)** methods, Gibbs sampling [159, 160], seems beneficial for music inpainting and has been widely used in this task. For instance, the generation in DeepBach [46] is performed using Gibbs sampling, which can complete Bach's four-part chorales by resampling voices or notes. Huang et al. [75] trained a CNN-based model called Coconet to complete the partial music score, which is an instance of orderless NADE, and used blocked Gibbs sampling as an analogue to rewriting. However, the piano roll-based CNN models can only complete fixed-length spans. Inspired by Coconet, Ippolito et al. [82] trained a Transformer to infill the missing part of the MIDI transcription of performed piano music. Composers can utilize this infilling technique to rewrite parts of music or gradually morph one piece into another through iterative Gibbs sampling. Similar to Coconet, this method can only complete a fixed number of tokens. The ES-Net proposed by Chi et al. [83] can not only add notes to complete the music score but also delete wrong notes, allowing for finer, note-by-note control in human and AI collaborative composition. The experimental results showed that ES-Net produces better results than orderless NADE and Gibbs sampling.

Learning the latent space of VAE enables diversity in music inpainting. Pati et al. [49] proposed a VAE-based score inpainting model named InpaintNet, which uses latent embeddings to learn complex trajectories in the latent space. The learned embeddings are sent to the decoder to generate multiple music bars connecting two music clips. This method allows users to change the musical context to achieve interactive music generation. Inspired by the work of sketching and patching in computer vision, Chen et al. [84] proposed a neural network framework named Music SketchNet for music completion, which focuses on filling missing measures in incomplete monophonic music according to the surrounding context and user-specified music ideas (like pitch and rhythm). The experimental results demonstrated that the generated melody follows the constraints of users, and Music SketchNet is superior to the previous InpaintNet, both objectively and subjectively. One common disadvantage of InpaintNet and Music SketchNet is that the bar-level VAE restricts the missing parts to start and end exactly where the bar begins and ends. Akama et al. [85] proposed a **contextual latent space model (CLSM)** based on CVAE to generate subsequences with a sense of direction in the generative space, such as exploring variants of the subsequence. A new mask strategy is proposed to adapt to music inpainting tasks, and the normalized flows are used to map between prior network outputs and latent variables to learn complex conditional priors.

The methods already mentioned can only complete a short segment with a fixed number of notes or a fixed time span. Recently, Chang et al. [86] proposed a new score completion model based on XLNet, which can complete a variable-length sequence (up to 128 notes) within different time spans. A new music-specific positional encoding—relative bar encoding is proposed to better inform the model where notes are located. Guo et al. [87] also proposed a transformer-based music completion framework, enabling the generated music to have stronger stylistic similarity with the

original music by adding additional control tokens. Compared to previous studies that could only control track density, this method provides users with more control, including key, bar tensile strain, bar cloud diameter, track density, track polyphony, and track occupation. The model has been available for interactive generation in a Google Colab notebook.

3.3 Controllable Generation

Controllable generation refers to imposing extra control (such as style, emotion, and musical structure) in the automatic music creation process, allowing more human intervention to enhance human-computer interaction. Some representative methods are shown in Table 4.

3.3.1 Style. Style is one of the most representative characteristics of music. Previous studies have attempted to generate music with a specific musical style. The stylistic consistency of the generated music is either brought about by the dataset itself, such as Bach's four-part chorales, or learned by models that can differentiate stylistic characteristics. Music style transfer is a special task that will be introduced in the latter.

Conditioning each layer of BALSTM on style, DeepJ [43] model can create polyphonic music according to a specific music style or a mixture of multiple composer styles. Zhou et al. [88] collected a new Beatles dataset from the Internet and proposed RNN-based BandNet to generate multi-instrument music in the Beatles' style. Zhu et al. [89] extended the MICA of the XiaoIce Band to a **Multi-Style Multi-Instrument Co-Arrangement Model (MSMICA)**. The MSMICA employed adversarial training to learn music style, which can maintain the harmony of the generated music and control the style for better application. In addition, Choi et al. [90] used a transformer autoencoder to aggregate encodings of the input data across time to obtain a global representation of style from a given performance. They showed that combining this global embedding with other temporally distributed embeddings could better control the separate aspects of performance style and melody. The experimental results demonstrated that this conditional model generated performance like the input style and generated accompaniment that conformed to the given performance style for the melody.

Style transfer

Gatys et al. [91] introduced style transfer in the context of neural networks, which usually refers to preserving the explicit content features of an image and exerting explicit style features of another image. In accordance with this, music style transfer is to generate novel music by separating and recombining the content and style of different music pieces.

The main difficulty of the music style transfer task is the lack of aligned data, so most studies adopt unsupervised learning frameworks to solve this task. Brunner et al. [92] proposed a VAE-based model called MIDI-VAE, which can realize the style transfer of multi-instrument polyphonic music by automatically changing pitch, dynamics, and instrument. MIDI-VAE contains a style classifier that forces the model to encode the style information in the shared latent space, thus manipulating the existing songs and changing their styles effectively. It is the first successful attempt to perform unaligned style transfer on complete musical works. Lu and Su [93] studied the style transfer of homophonic music that contains a predominant melody part and an accompaniment part. They only considered the style transfer of the accompaniment part. An adapted DeepBach model [46] was employed to model the temporal information, combined with the autoregressive model WaveNet [94] to model the pitch information. Exploiting Gibbs sampling on the generative model to modify the accompaniment part, the model can transfer arbitrary homophonic music scores into the styles of Bach's four-part chorales and jazz. Wu and Yang [95] proposed a Transformer VAE model called MuseMorphose to achieve song-level (32-bar)

Table 4. Methods for Controllable Music Generation

Controlled Aspect	Method	Year	Music Representation	Model Architecture	Description	Length of generated music
Style	DeepJ [43]	2018	Piano roll	RNN	A specific mixture of composer styles	Variable-length
	BandNet [88]	2019	Event	RNN	The style of Beatles; Templated-base method to generate structured music	4-16 bars
	MSMICA [89]	2020	Sequence	RNN	Extension of MICA [67] with style control; Reinforcement learning similar to SeqGAN [31]	Variable-length
	[90]	2020	Event	Transformer	Obtaining a global style representation of style from a given performance	Up to 2048 tokens
	MIDI-VAE [92]	2018	Piano roll	RNN-based VAE	Unaligned style transfer; parallel VAE with a shared latent space and a style classifier	1 bar
	[93]	2018	Piano roll	LSTM & CNN	Style transfer of accompaniment in homophonic music	Variable-length
	MuseMorphose [95]	2021	Event	Transformer VAE	Style transfer of long musical pieces; User specified bar-level musical attributes	Up to 32 bars
	[96]	2018	Event	GAN	Style transfer with cycle consistency loss	4 bars
	ChordGAN [163]	2021	Piano roll	CNN-based GAN	Stylistic rendering of chords into notes; Chroma feature extraction;	16 bars
Structure	StructureNet [98]	2018	Sequence	RNN	Structure-tagging algorithm for creating structure dataset; Duration repeat and duration-interval repeat	16 bars
	SSMGAN [100]	2019	Event	RNN-CNN-based VAE GAN	Representing self-repetition with self-similarity; Low dimensional measure embeddings	Variable-length
	MusicFrameworks [101]	2021	Event	Transformer & RNN & CNN	Multi-level repetition and structure; Multi-step generative process	Full-song-length
	HAT [102]	2021	Event	Transformer	Harmony (texture and form)-aware learning; Hierarchical structure-enhanced module	Full-song-length
	PopMNet [103]	2020	Sequence	RNN & CNN	Bar-level pairwise relations; Representing structure using Directed acyclic graph; Two stage manner	Variable-length
	MELONS [104]	2022	Event	Transformer	Graph representation of music structure; Eight types of bar-level relations; Multi-step generation	Full-song-length
	Museformer [164]	2022	Event	Transformer	Music repetition structures via bar-level similarity statistic; Fine- and coarse-grained attention	Full-song-length
Sentiment	[105]	2019	Event	LSTM	VGMIDI dataset with valence-arousal (VA) annotation; Sentiment unit within mLSTM	Variable-length
	[44]	2019	Piano roll	LSTM	Global condition of emotional vectors; Four quadrants of VA emotional space as four emotions	8 bars
	Bardo Composer [106]	2020	Event	Transformer	Language model with Stochastic Bi-Objective Beam Search; GPT-2 emotion classifier	Variable-length
	[107]	2021	Event	RNN & Transformer	Calculating valence of chord progression; Seq2seq; Translating conditions into musical events	4-32 bars
	Music FaderNets [108]	2020	Event	GM-VAE	Arousal transfer; Inferring high-level features from latent low-level representations	Up to 100 tokens

fine-grained musical style transfer, where the user can specify bar-level musical attributes, including rhythmic intensity and polyphony.

The GAN model with multiple generators and cycle consistency loss has achieved great success in image style transfer. Therefore, Brunner et al. [96] adopted the previous image style transfer model named CycleGAN to realize symbol music genre transfer. Since no aligned data is required, the model can be easily retrained, which is also the first application of GAN in symbolic music style transfer. Compared with MIDI-VAE [92], the CycleGAN-based model does not limit the number of notes to be played simultaneously, thus making the music sound richer. However, a pre-trained CNN is required to distinguish between content and style features. Another application of GAN for style transfer is the ChordGAN proposed by Lu and Dubnov [163]. This model took as input the extracted chroma features to preserve content features like chords and harmonies and then stylistically rendered chords within the chromagram into notes. However, ChordGAN did not capture long-term structure in music and needed to be trained separately for each genre.

3.3.2 Structure. The structure is the key aspect of music composed by humans, which plays a vital role in giving the music a sense of overall coherence and intentionality. It appears in a piece of music as a collection of musical patterns, pattern variation, literal or motivic repetition, and transformation of musical segments that appeared earlier in the same piece [98]. Existing automatic music generation methods have attempted to incorporate structure into the generated music from two aspects. One is to combine the generated music segments based on a pre-specified structure template. However, forcing the music to match a specific template may distort the musicality. The other is to condition music generation on extra structure input, which typically involves a two-stage process of generating structure followed by music generation. Nevertheless, generating well-structured musical pieces remains a challenging task.

Combining music through structural templates is a straightforward post-processing method. The BandNet [88] mentioned before is a template-based method to generate music with repetitive structure. Concretely, they generated music clips of varying lengths for each part of a predefined song structure template (such as AABA) and then combined the generated segments into a complete song. In order to induce musical structure, Medeot et al. [98] proposed an RNN model called StructureNet. It learns structures from a dataset containing structural elements and their occurrence statistics, which is created from existing melody datasets using a structure-tagging algorithm. This model exists in the music structure space and can work with any music generation probability model after training. Jhamtani and Berg-Kirkpatrick [100] combined GAN with VAE and proposed a music generation method centered on self-repetition. They constructed a self-similar matrix by calculating the similarity between bars to represent the self-repetition in music. The problem with generating controllable music directly using GAN or VAE is that the high-dimensional continuous vectors have limited interpretability, making it difficult for users to control. And modeling full-length music with a simple fixed-length representation is non-trivial. Therefore, Dai et al. [101] proposed MusicFrameworks, a hierarchical music structure representation and a multi-step generation process that creates full-length melodies under the guidance of long-term repetition structure, melody contour, rhythm constraints, and so on. By decomposing music generation into sub-problems, MusicFrameworks captures repetition structures at multiple levels, allows simpler models, and requires less data. Considering the relationship between harmony and musical structure that chords are closely integrated with the spatial structure—texture and chord progressions promote the temporal structure—form, Zhang et al. [102] proposed a hierarchical music transformer to generate pop music with structure enhancement through harmony-aware learning. They exploit structure adaptively from the music and interact on music tokens at multiple levels. It is the first work to generate pop music of the entire song length by learning form and texture jointly bridged by harmony.

Moreover, some studies attempted to explore the relationships between music bars. Wu et al. [103] proposed PopMNet for the sake of generating pop music melodies with good organizational structures. Melody structure is defined by pairwise relations between bars in the melody, including repetition and sequence. Repetition refers to the repetition of a segment, and sequence means to repeat a segment with the same rhythm and different pitches. They exploited a directed acyclic graph to represent melody structure and an adjacency matrix to represent structure diagram. A two-stage generation process conditioned melody generation on the generated structure and the chord progression. The evaluation results demonstrated that compared with other models (such as Attention RNN [25], MidiNet [29], Music Transformer [32], etc.), the music generated by PopMNet has clearer structures and higher scores in human evaluations. However, the enumeration of the relationships between bars in PopMNet is incomplete. The music structures include both repetitions and flexible developments. Zou et al. [104] further proposed a music structure diagram representation including eight types of bar-level relationships and a transformer-based melody generation framework MELONS, which can generate the whole song with a given 8-bar melody theme without labeling the structure information. Additionally, by calculating the similarity between each pair of bars, Yu et al. [164] introduced the structure-related bars that are expected to be repeated in the current bar to be generated. Then they proposed a novel transformer named Museformer with fine- and coarse-grained attention to facilitate music structure modeling and long sequence modeling.

3.3.3 Sentiment. As German philosopher Hagel said [44], "Music is the art of mood. It is straightly directed against mood." The deep learning models have shown promising results in automatic music creation. However, controlling the sentiment of the generated music is still a challenge with insufficient research. There is a broad research space for the sentiment-controllable music generation that needs further exploration in the future.

Ferreira and Whitehead [105] proposed an mLSTM-based deep generative model to generate polyphonic music with given emotions for the first time, where the mLSTM controls the emotion of the generated music by optimizing the weight of specific neurons responsible for emotional signals. Zhao et al. [44] extended the BALSTM network [27] to generate polyphonic music with specific emotions. They first divided four basic emotions through Russell's two-dimensional **valence-arousal (VA)** emotion space and then used emotion vectors as global conditions to train the network to generate emotion-controllable music. Ferreira et al. [106] presented a system called Bardo Composer to generate background music for tabletop role-playing games. They utilized the VA emotional model to classify the speech text into four emotion categories. Then **stochastic bi-objective beam search (SBBS)** and a neural model were used to generate music clips that convey the desired emotions. The experimental results demonstrated that the subjects accurately recognized the emotions of the generated music. Makris et al. [107] proposed a new method to manually calculate the valence of chords in the lead sheet. Then they developed a seq2seq architecture based on LSTM or Transformer to condition the generation of lead sheets on the encoded valence, phrases, and time signatures. Additionally, Tan and Herremans [108] applied the proposed Music FaderNets to the style transfer tasks of arousal states with the help of learned high-level feature representations.

3.4 Performance Generation

A complete definition of performance is given in [109], relating the liveliness of a score to "the artist's understanding of the structure and 'meaning' of a piece of music, and his/her (conscious or unconscious) expression of this understanding via expressive performance." The music generated by score generation methods adheres strictly to the metrical grid written on the score, resulting in a lack of expressive performance in most of the generated pieces. Many researchers have begun to

Table 5. Methods for Performance Generation

Method	Year	Category		Music Representation	Model Architecture	Description	Length of generated music
		Render	Compose				
[110]	2019	√		Event	Conditional Variational RNN	Generating performance data conditioned on music score and interpretation sequence	Variable-length
VirtuosoNet [111]	2019	√		Sequence	RNN & VAE	Hierarchical Attention network (HAN); A multi-scale approach	Variable-length
[38]	2019	√		Graph	GNN & HAN	Representing six edge types in musical score using graph; A multi-scale approach	Variable-length
Performance RNN [112]	2017		√	Event	RNN	Generating polyphonic music with expressive timing and dynamics	30s
[113]	2018		√	Event	RNN	Jointly predicting notes and their expressive and dynamics	30s
Pop Music Transformer [34]	2020		√	Event	Transformer-XL	REMI for modeling beat-bar-phrase hierarchical structure in music	Variable-length
Compound Word Transformer [42]	2021		√	Event	Transformer	Compound words; Different feed-forward heads for tokens of different types	Full-song-length
[114]	2021		√	Event	Transformer-GAN	Gumbel-Softmax; WGAN, RSGAN, PPO-GAN	128 tokens
GrooVAE [115]	2019		√	Sequence	RNN-VIB	Seq2seq; Humanization, Drum Infilling and Tap2drum	Up to 16 bars

explore expressive performance generation. Here, we divide the performance generation studies into two categories: one is rendering expressive performance, which refers to adding performance characteristics to a given score; the other is composing expressive performance, which means modeling score and performance simultaneously to create novel expressive music. Some popular performance generation methods are shown in Table 5.

3.4.1 Rendering. Rendering expressive performance refers to endowing a given score with performance features. In this case, there is no need to re-model the pitches and durations that have already been provided in the score but to generate expressive performance features for the notes, such as timing, dynamics, and so on.

The mapping between score and performance is not a bijection because distinct musicians will inject their own styles when performing the same music. Rather than establishing a direct mapping between score and performance by controlling specific music concepts (like dynamics), Maezawa et al. [110] proposed a music performance rendering method that can explicitly model the performance with interpretation variations of a given music segment. The trained model can automatically generate expressive piano performances based on the music score and interpretation sequence. Jeong et al. [111] modeled the piano performance based on a hierarchical attention network and VAE, which mainly imitated the pianists' expressive controls for tempo, dynamics, articulations, and pedaling. The model used a multi-scale approach to predict the performance characteristics and keep the musical expressions consistent over long-term sections. Specifically, the tempo and dynamics at the bar level are predicted first and then fine-tuned at the note level. After that, Jeong et al. [38] further adopted the **graph neural network (GNN)** model to represent music scores as a unique form of graphs and applied it to render expressive piano performance for the music score, which is the first attempt to adopt GNN to learn the note representation in western music.

3.4.2 Composing. Composing expressive performance refers to creating new music with expressive performance characteristics. Compared with rendering existing works, its advantages are more embodied in improvisation. Because the piano performance is easier to quantify and the piano datasets are easier to obtain, the current generated performance is mainly limited to the piano performance. A few researchers also have attempted to compose drum performances.

Simon and Oore [112] proposed Performance RNN to compose polyphonic music with expressive timing and dynamics. The performance generated by this model lacks overall consistency, but local features (phrasing within a 1- or 2-second time window) are pretty expressive. Similarly, Oore et al. [113] utilized the LSTM-based model to jointly predict notes and their expressive timing and dynamics. To capture expressiveness, they chose a temporal representation based on absolute time intervals between events, rounded to 8ms. Compared with the previous models, Oore et al. produced more human-like performances with this new representation. But this representation caused already lengthy musical pieces to be on average 3-4 times longer [167].

The Music Transformer proposed by Huang et al. [32] achieved expressive piano performance modeling and is superior to the performance RNN [112]. Based on Transformer-XL, Huang and Yang [34] proposed Pop Music Transformer for modeling popular piano performances. They improved the way music scores transform into events and proposed a new event set called "REMI" (introduced in 2.1.1). This event set provides a metrical context for modeling the rhythm patterns, explicitly establishes a harmonic structure to make chord progression controllable, and facilitates coordinating the different tracks of a music piece. The results implied that music generated by Pop Music Transformer has more reasonable rhythm structures than Music Transformer [32]. Hsiao et al. [42] further improved the REMI representation by grouping adjacent tokens through the expansion compression technique and converting a piece of music into a compound word sequence. This method greatly reduced the length of the token sequence and created expressive piano pop music of full length (up to 10k tokens) for the first time. The training convergence speed of this method is 5-10 times faster than that of the Pop Music Transformer without losing the generative quality. Muhamed et al. [114] combined the adversarial loss derived from GAN to improve the performance of the Transformer in generating long-term consistent music sequences. The discriminative objective of the pre-trained discriminator SpanBERT can improve the fidelity of the generative model. Experimental results show that Transformer-GAN outperforms Music Transformer trained only by maximining likelihood.

Apart from the piano performance generation, Gillick et al. [115] automatically composed drum performances using the seq2seq models and the recurrent **variable information bottleneck (VIB)** model. They presented a class of models called GrooVAE, which are used to generate and control the expressive drum performance. Furthermore, they designed three specific applications using these models: Humanization, generating performance for a given drum pattern, and realizing the style transfer of drum performance (groove transfer) by disentangling performance characteristics from the score, thus performing the same drum pattern in different ways; Tap2drum, coming up with drums that match the sense of groove on another instrument (or just tapping on a table); Infilling, completing or modifying the drum beat by generating or replacing the part for the desired instrument.

3.5 Interactive Generation

Thus far, machine-generated music has been hardly comparable to human creation. So, the time is not ripe for machines to drive the whole music generation process. Interactive music generation refers to humans and machines cooperating in some way to create music together. According to [116], interactive music generation can be divided into two categories: call and response and accompaniment . For call and response, the machine learns to play a solo alternately with the

Table 6. Methods for Interactive Generation

Method	Year	Category		Music Representation	Model Architecture	Description	Length of generated music
		Call and response	Accompaniment				
[117]	2017	√		Piano roll	Agnostic	Learning semantic embeddings of musical input; Combination of generation and unit selection	Variable-length
[119]	2019	√		Event	RNN	A hybrid of improvisation with pitches generated by models and rhythms provided by human	Variable-length
The Bach Doodle [78]	2019		√	Piano roll	CNN	Using Coconet [75] for melody harmonization (4-voice) in the style of Bach	2 bars
RL-duet [120]	2020		√	Sequence	Attention-guided RNN-CNN	Reinforcement learning; Learning rewards from training data considering both the inter-part and intra-part harmonization of human and machine inputs	Variable-length

human in real-time, i.e., machine and human play in turn; accompaniment generation requires the machine to adapt and support human performance. We list several studies on interactive music generation in Table 6.

3.5.1 Call and Response. To study the music interaction scenario of call and response, Bretan et al. [117] proposed a system that employs a deep autoencoder to learn the semantic embedding. The system reconstructs the transformed embeddings by combining generation and unit selection to produce appropriate musical responses. During the live demonstration, the human plays the MIDI keyboard, and the computer generates the response. There have been several systems (such as AI Duet [118]) that perform human-computer cooperation in the configuration of call and response. However, most of the deep music generation models run in "offline" mode with almost no limitation on processing time. Integrating these models into real-time structured performances is a challenge. Moreover, these models are often agnostic to the specific style of the performer, making them impractical for live performances. To solve the real-time compatibility and style personalization problems, Castro [119] proposed a software system integrating off-the-shelf music generation models into improvisation. Specifically, he exploited the melody generated by Melody RNN [25] and the rhythm provided by human performers to improvise in the manner of call and response.

3.5.2 Accompaniment. The Bach Doodle created by Huang et al. [78] allows users to create their own melodies and then generate accompaniment in Bach's style using the improved Coconet model. However, the system cannot generate accompaniment in real time. To solve this problem, Jiang et al. [120] proposed the first deep reinforcement learning algorithm called RL-duet for online accompaniment generation, which can realize real-time interaction with human input without any delay. The key to this algorithm is an effective reward model learned from the training data that considers both the inter-part and intra-part compatibility of the generated notes from the horizontal and vertical perspectives.

3.5.3 Application. It is encouraging that some automatic music generation techniques have been gradually applied to the practical environment. Most applications are interactive, i.e., humans and AI work together to create music. For instance, Donahue et al. [121] designed Piano Genie, an intelligent instrument with which users can generate piano music in real time by playing on eight buttons. Louie et al. [122] developed a Coconet-based **Cococo (collaborative co-creation)**,

a music editor web interface for novice-AI co-creation. It augments the standard generative music interface through a set of AI-steering tools that allow users to express music intention better. Qiu et al. [123] developed an interactive music composition system named Mind Band, which takes expressions, images, and humming as input and outputs music that matches the input emotions.

Google's Magenta project[3] is one of the most excellent research projects focusing on deep music generation. It has developed multiple AI-based tools to assist musicians in being more creative [35]. The MusicVAE [28] proposed by Google Magenta has been used to create many tools: Melody Mixer and Beat Blender allow to generate interpolation between two melodies or drumbeats; Latent Loops creates a two-dimensional grid among four melodies and allows the users to draw a path in the space to create complex melodies. Besides, Muzic[7] started by Microsoft Research Asia is also a research project on AI music that empowers music understanding and generation with deep learning and artificial intelligence. The previously mentioned PopMAG [81] and Museformer [164] are derived from the Muzic project.

4 DATASETS

So far, a variety of music datasets have been born in the field of automatic music generation, covering music genres such as classical, folk, pop, and so on. For deep learning algorithms driven by a great quantity of data, the quality of the generated music is closely related to the selection of datasets. This section summarizes the common datasets in previous studies from the perspective of the storage forms of music, hoping to bring some convenience to future researchers. The brief introductions and methods for obtaining access to music datasets are shown in Table 7.

4.1 MIDI

As introduced in Section 2, MIDI is a descriptive "music language" describing the music information to be performed in bytes, such as what instrument to use, what note to start with, and what note to end at a specific time. The pretty_midi Python toolkit contains practical functions and classes for parsing, modifying, and processing MIDI data, through which users can easily read various note information contained in MIDI.

Music21 [150] is an object-oriented toolkit for analyzing, searching, and converting music of symbolic form. **J. S. Bach's (JSB)** four-part chorales dataset (402 chorales) can be directly obtained from the music21 Python package. However, this data set is small and lacks expressive information.

Ferreira and Whitehead [105] created a new music dataset VGMIDI in symbolic format with sentiment notation, which contains 95 labelled MIDI piano pieces (966 phrases of four bars) from video game soundtracks and 728 non-labelled pieces. All of them vary in length from 26 seconds to 3 minutes. The labelled music pieces are annotated by 30 human subjects according to a valence-arousal (dimensional) emotion model [166]. The sentiment of each piece is then extracted by summarizing the 30 annotations and mapping the valence axis to sentiment. For the concrete operation of emotion annotation, please refer to [105].

The **Lakh MIDI Dataset (LMD)** is the largest symbolic music corpus to date, including 176,581 unique MIDI files, of which 45,129 files have been matched and aligned with the entries in the **Million Song Dataset (MSD)** [151]. The LMD includes the following formats: (1) 176,581 MIDI files with duplicate data removed, and each file is named according to its MD5 checksum (called "LMD full"); (2) the subset of 45,129 files (called "LMD matched") that match items in the MSD; (3) All LMD-matched files are aligned with the 7Digital preview MP3s in the MSD (called "LMD aligned"). In addition, Dong et al. [30] transformed the MIDI files of LMD into multi-track piano rolls and named the resulting dataset the **Lakh Pianoroll Dataset (LPD)**.

[7] https://github.com/microsoft/muzic.

Table 7. Datasets Summary

Format	Name	Modality			Genre	Size (files)	Access
		Score	Performance	Audio			
MIDI	JSB Chorales dataset	√			Bach	402	Music21 toolkit [150]
	VGMIDI	√	√		Video game	823	https://github.com/lucasnfe/vgmidi
	Lakh MIDI Dataset	√	√		Multiple	176,581	http://colinraffel.com/projects/lmd/
	e-Piano Competition Dataset	√	√		Classical	~1,400	http://www.piano-e-competition.com
	ADL Piano MIDI dataset	√	√		Multiple	11,086	https://github.com/lucasnfe/adl-piano-midi
	POP909	√	√		Pop	909	https://github.com/music-x-lab/POP909-Dataset
	GiantMIDI-Piano	√	√		Classical	10,854	https://github.com/bytedance/GiantMIDI-Piano
	BitMidi	√			Multiple	113,244	https://bitmidi.com/
	Classical Archives	√			Classical	20,801	https://www.classicalarchives.com/
	FreeMidi	√			Multiple	25,860	https://freemidi.org/
MusicXML	TheoryTab Dataset	√			Multiple	16,000	https://www.hooktheory.com/theorytab
	Hooktheory Lead Sheet dataset	√			Multiple	11,329	Derived from [73]
	Wikifonia	√			Multiple	2,252	http://marg.snu.ac.kr/chord_generation/ (CSV format)
Text	Nottingham Music Dataset	√			Western folk	Over 1,000	abc.sourceforge.net/NMD/
	Henrik Norbeck's ABC tune	√			Ireland, Sweden, and others	Over 3,000	http://www.norbeck.nu/abc/
Multimodality	MusicNet	√	√	√	Classical	330	https://zenodo.org/record/5120004#.YpbPI-5ByUl
	MAESTRO	√	√	√	Classical	1,282	https://g.co/magenta/maestro-dataset
	Piano-Midi	√	√	√	Classical	332	www.piano-midi.de/
	Groove MIDI Dataset	√	√	√	Drum	1,150	https://magenta.tensorflow.org/datasets/groove
	ASAP	√	√	√	Classical	1,068	https://github.com/fosfrancesco/asap-dataset
	EMOPIA	√	√	√	Multiple	1,087	https://zenodo.org/record/5090631

The e-piano junior competition dataset is a collection of professional pianists' solo piano performances, which provides a substantial amount of expressive performance MIDI (~1,400) of professional pianists. Most of them are late romantic works, such as Chopin and Liszt, as well as some Mozart sonatas. Since this dataset provides high-quality piano performance data in MIDI format, including the fine control of timing and dynamics by different performers, it is widely used in performance generation. However, it does not contain the corresponding music score of the pieces.

The **Augmented Design Lab (ADL)** piano MIDI dataset is based on LMD. In LMD, there are many versions of the same song, and only one is reserved for each song in the ADL dataset. Later, Ferreira et al. [106] extracted from the LMD only the tracks with instruments from the "piano family" (MIDI program number 1–8). This process generated a total of 9,021 unique piano MIDI files, which are mainly rock and classical music. To increase the diversity of the genre (like jazz), they added another 2,065 files obtained from public resources on the Internet. All the files in the collection are de-duped according to MD5 checksums, and the final dataset has 11,086 pieces.

Wang et al. [155] proposed a novel dataset named POP909 for music arrangement generation, which contains multiple versions of piano arrangements of 909 popular songs created by professional musicians. The total length of POP909 is about 60 hours, including 462 artists, and the release of all songs spans around 60 years. The main body of the dataset contains vocal (main) melody,

secondary melody or main instrument melody (bridge) in MIDI format, and piano accompaniment of each song, which are aligned with the original audio file. Beat, chord, and key annotations are provided; each note contains expressive dynamics.

The GiantMIDI-Piano [152] released by ByteDance is the world's largest classical piano dataset, including MIDI files from 10,854 music works by 2,784 composers, with a total duration of 1,237 hours. In terms of data scale, the total duration of different music pieces in the dataset is 14 times that of Google's MAESTRO dataset. To construct the dataset, researchers have developed and open-sourced a high-resolution piano transcription system, which is used to convert all audio into MIDI files. The MIDI files include the onset, dynamics, and pedal information of notes.

Besides, BitMidi provides 113,244 MIDI files curated by volunteers worldwide; Classical Archives is the largest classical music website, including the largest collection of free classical music MIDI files (20,801); FreeMidi comprises more than 25,860 MIDI files of assorted genres.

4.2 MusicXML

MusicXML aims to completely represent western music symbols, such as rest, slur, beam, barline, key and time signature, articulation, ornament markings, and so on, which are not included in MIDI format. MusicXML is usually employed to store the lead sheet, a form of musical symbol utilized to specify the essential elements of a song, like melodies, lyrics, chords, repetition marks, and so on. Lead sheets rarely involve information about instruments or accompaniments and are the most basic form of music.

TheoryTab is an online music theory forum, and the **TheoryTab Dataset (TTD)** contains 16K lead sheet segments stored in XML format. Yeh et al. [73] collected a new dataset called the **Hooktheory Lead Sheet Dataset (HLSD)** from the TheoryTab. Each lead sheet contains high-quality, human-transcribed melodies and their corresponding chord progressions. The chords are specified by both literal chord symbols (e.g., Gmaj7) and chord functions (e.g., VI7). The HLSD contains 11,329 lead sheet samples, all in 4/4 time signature, covering 704 chord categories.

Wikifonia.org offered a public lead sheet repository. Unfortunately, the website ceased service in 2013. Before that, Lim et al. [72] obtained some data, including 5,533 western music lead sheets in MusicXML format and covering multiple music genres such as rock, pop, and jazz. They collected 2,252 lead sheets from the obtained dataset. Only the first chord is selected if a bar contains two or more chords. Finally, the music features are extracted and converted into CSV format.

4.3 ABC

ABC is a format for recording music in plain text. It was originally designed for folk music originated in Western Europe and was later extended to support a complete representation of classical music scores. The **Nottingham Music Dataset (NMD)** is composed of simple melodies with chords, containing ~1,200 British and American folk songs in special text format. Using the program NMD2ABC[8] and some Perl scripts, most of this dataset can be converted into ABC notation. Additionally, Henrik Norbeck has collected more than 3,000 music scores and lyrics in ABC format, mainly including traditional music from Ireland and Sweden. The accesses to these two datasets are shown in Table 7.

4.4 Aligned MIDI and Audio

Some datasets contain both the MIDI files and the aligned audio files. Except for the symbolic music generation tasks, these datasets can be capitalized in other tasks like music transcription, audio synthesis, and so on.

[8] abc.sourceforge.net/NMD/.

The MusicNet dataset [153] is a collection of 330 freely licensed classical music recordings, including 10 composers and 11 instruments, plus instrument/note annotations, resulting in more than 1 million temporal labels on 34 hours of chamber music performances. The labels are from 513 classes, indicating the precise time of each note in each recording, the instrument playing each note, and the position of the note in the rhythm structure of the music. The average recording length is 6 minutes. The shortest recording time is 55 seconds, and the longest is nearly 18 minutes. The majority of MusicNet is composed of Beethoven's compositions and piano solos. However, as Hawthorne et al. [154] discussed, the alignment between audio and score is not completely accurate.

The MAESTRO dataset [154] contains over 172 hours of virtuosic piano performances, covering about 430 compositions and 1,282 performances from nine years of International Piano-e-Competition events. Audio and MIDI files are precisely aligned (≈3ms). The MIDI data contains key strike velocities and sustain pedal positions. The audio files and MIDI files are divided into individual music pieces and annotated with composer, title, and year of performance. The repertoires are mainly classical music composed by composers from the 17th century to the early 20th century.

The Piano-Midi is a MIDI database of classical pianos containing 332 music pieces from 25 composers of three major classical periods (baroque, classical, and romantic). The dataset is rich in timing and dynamic information. The MIDI files and their corresponding mp3 and ogg audio files are available, and some music pieces have corresponding music scores in PDF format as well.

The **Groove MIDI Dataset (GMD)** is a dataset created by Gillick et al. [115] for the research of drum performance. It contains 13.6 hours, 1,150 MIDI files, and 22,000 bars of tempo-aligned expressive drumming performed by 10 drummers. The dataset was recorded on the Roland TD-112 electronic drum kit equipped with sensors to capture precise performance features in MIDI format. The MIDI notes are related to the instrument, timing, and velocity. The microtimings describe how the note timings stray from a fixed grid, and the velocities indicate how hard the notes are struck. Each performance in the dataset also contains relevant metadata (such as music style) and corresponding synthesized audio outputs.

Foscarin et al. [156] proposed a dataset named **Aligned Scores and Performances (ASAP)**, which consists of 222 digital scores and 1,068 aligned performances of western classical piano music from 15 composers. Music scores are provided in pairs of MusicXML files and quantified MIDI files, while performances are provided in the form of paired MIDI files and partial audio recordings. The 548 performances are available as MIDI only, and all the others (520) are provided as MIDI and audio recordings aligned with approximately 3ms precision. For each MIDI score and performance, ASAP offers the positions of all beats, downbeats, time signature changes, and key signature changes. Each score or performance in ASAP is labeled with metadata, such as the composer and the title of the piece.

The EMOPIA dataset proposed by Huang et al. [165] is a multi-modal database focusing on perceived emotion in pop piano music. The dataset contains 1,087 music clips from 387 solo piano performances of popular music (i.e., 2.87 clips per song on average) and clip-level emotion labels annotated by dedicated annotators. The genres of songs include Japanese anime, Korean and Western pop song covers, movie soundtracks, and personal compositions. The emotional labels follow the four-class taxonomy corresponding to the four quadrants of Russell's famous Circumplex model of affect [166].

5 EVALUATION

There are no unified evaluation criteria for the results of generative algorithms in the arts field. Indeed, since works of art such as music and painting are intrinsically subjective, it is rather hard

to measure them with a certain rigorous metric [124]. So, the quality of the generated contents inevitably requires subjective evaluation, but some objective evaluation metrics still have certain reference values for the generation quality. This section summarizes the music evaluation methods from both objective and subjective aspects. Effective evaluation can find out the deficiencies and give valuable feedback, thereby improving the quality of the generated music.

5.1 Objective

The objective evaluation includes quantitative evaluations of the generative model and the generated music. The former is often assessed by metrics such as loss, which only reflect the learning ability of the model but do not truly embody its generative quality. Therefore, this survey will not delve into the metrics used to evaluate the generative models. The latter selects appropriate music metrics for the specific music generation task. To define the music metrics, researchers need to master quite a lot of knowledge of music theory. Furthermore, several special methods for detecting specific aspects (like structure, style, etc.) of the generated music will be introduced in Section 5.1.2.

5.1.1 Music Metrics. Music metrics are derived from some music concepts, also known as musical statistical descriptors. Ideally, as the training process goes on, the distributions of the generated and the real music should be closer and closer. Therefore, many methods judge the generative quality by comparing the descriptive statistics of real music and the music generated by models. Researchers can design various music metrics according to their research tasks as long as the definitions of metrics are explained in detail, especially the corresponding relationship between the value of metrics and the quality of music.

Since the development of deep symbolic music generation, numerous research teams have defined many music metrics for various generation tasks. Here we only point out some commonly used and representative metrics, as shown in Table 8. Notably, Yang and Lerch [125] also put forward a set of simple musically informed objective metrics, including absolute and relative metrics. The absolute metrics are used to give insights into the properties and characteristics of a dataset, including pitch-based and rhythm-based features. The relative metrics compare the distributions of two datasets, including the **Overlapping Area (OA)** and the **Kullback–Leibler Divergence (KLD)**. The evaluation framework has been released as an open-source toolbox[9] that implements the demonstrated evaluation and analysis methods along with visualization tools. This evaluation system has been recognized by many researchers, e.g., Wuerkaixi et al. [126] adopted OA and KLD to assess the degree of harmony between melody and accompaniment; Mittal et al. [127] used an adjusted frame-level OA to capture local self-similar patterns of generated melody sequences; Madaghiele et al. [71] and Jiang et al. [120] used several absolute features proposed in [125]; Genchel et al. [48] directly utilized the toolbox MGEval to evaluate their music generation system.

5.1.2 Other methods. Apart from the music metrics, some methods evaluate specific aspects of music (such as structure and style) with the help of other theories or algorithms. This section briefly introduces several methods, and the concrete implementations require further reading of relevant literature.

Structure

The music structure assessment methods focus more on repetitive and self-similar structures in music. The **Variable Markov Oracle (VMO)** is an information dynamics method developed by

[9] https://github.com/RichardYang40148/mgeval.

Table 8. Music Metrics

Type	Metrics	Definition
Pitch	Empty Bar (EB) [30]	the ratio of empty bars (in %).
	Used Pitch Class (UPC) [30]	the number of used pitch classes per bar (from 0 to 12).
	Pitch Range (PR) [125]	the subtraction of the highest and lowest used pitch in semitones.
	Number of Unique Note Pitches/Durations [41]	the count of how many different pitches/durations are used in a music piece.
	Pitch Class Histogram Entropy [129]	construct the 12-dimensional pitch class histogram $\vec{\boldsymbol{h}}$, normalized by the total note count in the period such that $\sum_i h_i = 1$. Then, calculate the entropy of $\vec{\boldsymbol{h}}$: $\mathcal{H}(\vec{\boldsymbol{h}}) = -\sum_{i=0}^{11} h_i log_2(h_i)$
	Qualified Note (QN) [30]	the ratio of "qualified" notes (in %). Dong et al. [30] consider a note no shorter than three timesteps (i.e., a 32nd note) as a qualified note.
	Polyphony [95]	the average number of pitches being played at the same time, evaluated only at time steps where at least one pitch is on.
	Scale Consistency [60]	the fraction of tones that are part of a standard scale, reporting the number for the best matching such scale.
	Repetitions [60]	the repetitions of short subsequences, giving a score on how much recurrence there is in a sample.
	Consecutive Pitch Repetitions (CPR) [68]	the frequency of occurrences of ℓ consecutive pitch repetitions for a specified length ℓ.
	Durations of Pitch Repetitions (DPR) [68]	the frequency of pitch repetitions that last at least d long in total for a specified duration d.
	Tone Span [60]	the number of half-tone steps between the lowest and the highest tone in a sample.
	Tone Spans (TS) [68]	the frequency of pitch changes that span more than d half-steps for a specified tone distance d.
	Average Pitch Interval (PI) [125]	the average value of the interval between two consecutive pitches in semitones.
Rhythm	Drum Pattern (DP) [30]	the ratio of notes in 8- or 16-beat patterns, common ones for Rock songs in 4/4 time (in %).
	Qualified Rhythm Frequency (QR) [68]	the frequency of note durations within valid beat ratios of {1,1/2,1/4,1/8,1/16}, their dotted and triplet counterparts, and any tied combination of two valid ratios.
	Rhythmic Intensity Score [95]	the percentage of sub-beats with at least one note onset: $s^{rhym} = \frac{1}{B}\sum_{b=1}^{B} 1(n_{onset,\ b} \geq 1)$ where B is the number of sub-beats in a bar and $1(\cdot)$ is the indicator function.
	Grooving Pattern Similarity [129]	the grooving pattern represents the positions in a bar at which there is at least a note onset, and the similarity between a pair of grooving patterns $\vec{g}^a$, $\vec{g}^b$ as: $\mathcal{GS}(\vec{g}^a, \vec{g}^b) = 1 - \frac{1}{Q}\sum_{i=0}^{Q-1} XOR\left(g_i^a, g_i^b\right)$ where Q is the dimensionality of $\vec{g}^a$, $\vec{g}^b$, and $XOR(\cdot, \cdot)$ is the exclusive OR operation. Note that the value of $\mathcal{GS}(\cdot, \cdot)$ would always lie in between 0 and 1.
Harmony	Tonal Distance (TD) [30]	the harmonicity between a pair of tracks. Larger TD [128] implies weaker inter-track harmonic relations.
	Chord Progression Irregularity [129]	the percentage of unique chord trigrams in the chord progression of an entire piece.
	Chord histogram entropy (CHE) [73]	given a chord sequence, create a histogram of chord occurrences with $\|C\|$ bins. Then, normalize the counts to sum to 1, and calculate its entropy: $H = -\sum_{i=1}^{\|C\|} p_i log p_i$ where p_i is the relative probability of the i-th bin. The entropy is greatest when the histogram follows a uniform distribution, and lowest when the chord sequence uses only one chord throughout.
	Chord coverage (CC) [73]	the number of chord labels with non-zero counts in the chord histogram in a chord sequence.
	Chord tonal distance (CTD) [73]	the average value of the tonal distance computed between every pair of adjacent chords in a chord sequence. The CTD is highest when there are abrupt changes in the chord progression (e.g., from C chord to B chord).
	Chord tone to non-chord tone ratio (CTnCTR) [73]	define CTnCTR as $\frac{n_c+n_p}{n_c+n_n}$, n_c is the number of the chord tones, n_n is the number of the non-chord tones, n_p is the number of a subset of non-chord tones that are two semitones within the notes which are right after them, where subscript p denotes a "proper" non-chord tone. CTnCTR equals one when there are no non-chord tones at all, or when $n_p = n_n$.
	Pitch consonance score (PCS) [73]	computed by averaging these consonance scores across a 16th-note windows, excluding rest periods, where the consonance score is set to 1 for consonance intervals including unison, major/minor 3rd, perfect 5th, major/minor 6th, set to 0 for a perfect 4th, and set to -1 for other intervals, which are considered dissonant.
	Melody-chord tonal distance (MCTD) [73]	the average of the tonal distance between every melody note and the corresponding chord label calculated across a melody sequence, with each distance weighted by the duration of the corresponding melody note.

Wang and Dubnov [131], which can detect repeated music patterns in a time series by visualization. VMO uses a string-matching algorithm called **Factor Oracle (FO)** to search for repeated segments (suffixes) at every time instance in the signal [132]. A key step of VMO is to find a threshold θ to establish similarity between features. In order to select a sequence with the most abundant information patterns, the optimal threshold θ is usually selected by calculating the **Information Rate (IR)**. VMO with higher IRs captures more repeat subsequences than VMO with low IRs. In addition, the higher IR itself can reflect more self-similarity structures in the sequence [132]. Chen et al. [133] reflected the self-similarity structure in music by comparing the IRs of the generated music of different models and detected the repeated patterns in the music pattern plots generated by VMO. Lattner et al. [99] also used IR as the evaluation metrics as well. Larger IR occurs when repetition and variation are in balance, and smaller IR occurs when the sequences are either random or very repetitive.

Guo et al. [130] and Makris et al. [107] adopted COSIATEC to assess the long-term structure in music. COSIATEC utilizes a geometric approach, much like zip-file compression, to detect repeated patterns in symbolic music data. The resulting compression ratio indicates how much smaller a musical file can be made by representing it with pattern vectors and their corresponding translation vectors, reflecting the number of repeated patterns contained in the generated music.

The fitness scale plot algorithm [134, 135] and related SM toolbox [136] offer an aesthetic method for detecting and visualizing repetitive structures in music. The fitness scape is a matrix $S_{N\times N}$, where $s_{ij} \in [0,\ 1]$ is the fitness, i.e., the repeatability in the music segment derived from the self-similarity matrix of the segment specified by $(i,\ j)$. Zhang et al. [102] and Wu and Yang [129] compared the fitness scale plots of the generated and real music.

Originality

The current deep learning-based music generation algorithms rarely have musicological analysis and plagiarism analysis for the generated results [137]. To judge whether the generated music "plagiarizes" the music in the original corpus, Hadjeres et al. [46] proposed a plagiarism analysis method that finds the length of the longest subsequence from generated music which can be found identically in the training dataset and then drew the length histograms. The shorter the length of the histogram peak is, the more innovative the generated music is. Besides, Chu et al. [138] divided each generated melody into segments of two bars and then compared each segment to all segments in the rest of the 100 generated songs. The repeat time is recorded to evaluate how much the model repeats itself. Hakimi et al. [70] evaluated plagiarism by measuring the percentage of n-grams in a solo that also appear in any other solo in the source. Particularly, Yin et al. [137] use a similarity measure called the cardinality score to calculate the originality scores and found that Transformer models obtained by traditional stopping criteria produce single-note repetition patterns, resulting in lower quality and originality. While in the later training epoch, the transformer model tends to overfit and produces a copy of the input song excerpt.

Style

Researchers usually train an additional classifier to judge whether the generated music style meets expectation. For example, Jin et al. [20] used the **Minimum Distance Classifier (MDC)** algorithm to determine whether the generated music is classical. To evaluate the effect of style transfer, Brunner et al. [96] constructed a binary style classifier to output the probability distribution of distinct styles. Some feature mapping methods (like t-SNE, PCA, etc.) are also used to analyze the ability of model learning style via mapping the high-dimensional style embedding space into a low-dimensional feature space. E.g., in [92], t-SNE is used to visualize the latent vectors of 30 jazz

songs and 20 classical songs. Several evaluation metrics related to style transfer are proposed in [96, 139].

Tonality

The Keyscape [140] can illustrate the music tonality, with different colors for each key. Lattner et al. [99] utilized the humdrum keyscape to parse the tonality of a given MIDI file using the Krumhansl-Schmuckler key-finding algorithm [141]. The peak of the pyramid shows the key estimates of the whole piece, and moving down toward the base is a recursive set of decreasing window sizes that provides an estimation of the key estimates within that context window. Intuitively, the highly structured keyscape indicates that more self-references exist in the segment, with tonal centers that recur often in fractal patterns.

5.2 Subjective

Albeit many objective evaluation metrics have been proposed, the correlation between quantitative evaluation and human judgment is unclear. Thus, researchers always perform subjective tests when evaluating the generated music. The most common subjective evaluation method is the listening test, which is a complicated process involving many variables [125], such as the selection and presentation of audio samples, listening environment, selection of subjects, and expression of questions. In contrast, some subjective evaluations, like analyzing the generated musical scores, do not require listening and are conducted mainly by music experts. The number of subjects involved in the subjective evaluation may vary greatly. Human evaluation faces the following two problems: one is that it takes more time compared with machine evaluation; the other is the evaluation error caused by auditory fatigue. Nevertheless, human subjective evaluation is still foremost and indispensable now.

5.2.1 Listening Test. The listening test generally requires [142]: (1) enough subjects with sufficient diversity to offer statistically significant results; (2) the music knowledge level of subjects is evenly distributed, including both music amateurs and music experts; (3) the experiment was conducted in a controlled environment with specific acoustic characteristics and equipment; and (4) each subject received the same instructions and stimuli. To prove that the evaluation results have statistically significant differences, the hypothesis tests are usually inevitable, such as the Wilcoxon signed-rank test [28], t-test [33], and Kruskal Wallis H-Test [83]. Here are several common listening tests:

- **Turing test.** Ask the subjects to judge whether the music is machine-generated or human-created [46].
- **Binary/multiple comparison and selection.** The selection can be either preference selection, i.e., to choose the most favorite one from a group of music [43, 101], or selection based on some criteria, e.g., to choose the music closest to the original corpus style [90].
- **Rank.** According to several evaluation metrics, rank a set of music samples generated by different models [143].
- **Score.** Rate the generated music based on several evaluation metrics or by answering several subjective questions, such as how natural the music is. This listening test can be regarded as the further quantification of selection or rank. The common scoring rule is the 5-point Likert [29, 80], and other self-defined scoring rules like the 4-point scale adopted by [144].
- **Identify music type.** Ask the subjects to identify the style or emotion of the generated music [43, 44].

Another listening test only invites professional composers to assess the music. The evaluation criteria here are not subjective questions but professional music theory, which requires subjects with a strong music background. Zhou et al. [88] evaluated the notes and rhythms in the generated music according to the music theory consistent with the Beatles' music, and Wei et al. [145] invited professional drummers and composers to give descriptive feedback and detailed comments on the structural compatibility, stability, and variability of the generated drum patterns.

The listening tests can be conducted on online platforms, such as Amazon Mechanical Turk [147]. Although online testing saves manpower, it cannot ensure the authenticity and validity of the collected results, e.g., it does not rule out the situation that someone hastily completes the task to get rewards. However, it is undeniable that online evaluation can indeed invite more subjects. Liang et al. [146] conducted a public music discrimination test on bachbot.com, and a total of 2,336 participants were involved in the test, which is the largest human evaluation so far.

5.2.2 Visual Analysis. Visual analysis means that people evaluate the generated music subjectively via observing the visual music representation (like the musical score and piano roll) rather than auditory perception. Score analysis is usually carried out by experts in the music field and focuses on different musical aspects depending on the generation tasks. Dong et al. [30] analyzed the characteristics of the generated multitrack music from the piano rolls, such as rhythmic patterns and harmonic relations. Yan et al. [144] recorded the generated accompaniment score in PDF format and recruited three Ph.D. students of music theory as raters, then asked them to rate the accompaniment according to some criteria. Different judges may have different opinions on the same music score, so we summarize it under the scope of subjective evaluation.

6 CHALLENGES AND FUTURE DIRECTIONS

Automatic music generation has made great progress in the past decade, where deep learning technology has shown its powerful capabilities. However, deep symbolic music generation still faces many challenges, based on which we will point out several future research directions.

6.1 Challenges

To our best knowledge, the deep symbolic music generation may still face the following challenges.

Creativity. The magic of algorithmic composition lies in that its goal is to reproduce the real intelligence of human beings rather than complex imitation [66]. Creativity is the extrapolation of outlier patterns beyond the observed distribution, while present machine learning regimes mainly handle interpolation rather than extrapolation [70]. The boundary between musical creativity and plain clumsiness is highly uncertain and difficult to quantify. Yin et al. [137] even claimed that language-based deep learning models seem to be little more than powerful memorizers. For now, relatively few researches focus on the creativity of generated music, such as [66].

Style. The style of generated music is often consistent with that of training datasets, and different music styles require different model settings. A universal framework for all music styles may be desired [93]. For style transfer, automatically learning the disentanglement of music representations with respect to different aspects is still an effortful task.

Structure. The generated music lacks long-term structure, i.e., recurring themes, motivations, and patterns. Especially when using recurrent sequence models (like RNN and LSTM), the generated music gradually becomes boring if there is no external inducement during the generating process. Music also contains a whole structure of its own, such as the AABA or AAB form in jazz and the verse plus chorus form in pop music. It is hard to simultaneously model the local structure and the global structure as the sequence length increases. Different structure components are supposed to be modeled separately, and the change in structure makes automatic music generation more complicated.

Sentiment. There is not much research on emotional music generation, and the emotion category involved is relatively simple. There is still much room to understand the correlation between music and emotion and integrate this important relationship into the music generation system.

Interaction. It may be possible to generate innovative and fluent music completely using computers in an end-to-end way. However, humans lack participation during this process, resulting in the generated music may not meet their creative goals (such as rhythm variations and emotional trends). Nowadays, the interactive interfaces/tools available for users are limited, and some interactive generation systems are available only in studios but are not promoted in real-world scenarios. The better interaction and cooperation modes between humans and AI still need further exploration.

Pretraining. Due to the difference between natural text data and symbolic music data, directly migrating the pretraining model in the **Natural Language Processing (NLP)** field to music will face various problems, such as destroying the local structure of music [40, 62]. Symbolic music is more structural and diverse than natural language, and how to preserve music structures and the relative positions of notes (like rhythm patterns and harmony) during pretraining is worth consideration.

Evaluation. As Theis et al. [148] observed, "Good performance with respect to one criterion therefore need not imply good performance with respect to the other criteria." The metrics defined by distinct researchers are diverse, which makes it difficult to compare the music generated in different studies. There is also a lack of correlation between quantitative metrics and subjective evaluation. Music that scores high on objective metrics may perform poorly in subjective evaluation, and vice versa. Moreover, the current systems are unable to evaluate the generated music automatically, probably because the automatic analysis of music is not close to the human level yet [125].

Application. The application of the music generation system in real scenario is limited for three reasons. First, most models place restrictions on the input, such as the number and type of tracks. Second, users cannot control the generation process in a fine-grained way, which is crucial for the computation-assisted composition system [149]. Third, new music demands emerge, like automatically creating background music for short videos, while the generated music according to arbitrarily specified attributes (emotions, rhythm, etc.) is still unsatisfactory.

6.2 Future Directions

According to the above challenges, several future research directions are pointed out as follows. Some of them may be under research but have not achieved the desired results.

- Generate innovative music rather than the permutation and combination of the training data and put forward novel evaluation metrics to measure the creativity and originality of the music.
- Generate long-term music with natural ending, clear themes, and both local and global structures.
- Develop a universal generative framework suitable for music of various styles.
- Condition the music generation process on emotions, then use the generated results to adjust human emotion.
- Strengthen the interaction with humans in a more fine-grained way to meet the current needs (like creating music for short videos, etc.) and enhance the ability of real-time improvisation.
- Merge music knowledge or metrics into the training objective as rewards or penalties to improve the generation results and increase the interpretability of the generated music.

- Design pre-trained models with tailored handling of music positional information and music relations for better music representation learning.
- Explain the correlation between quantitative metrics and subjective evaluation, and finally, realize automatic music evaluation.

7 CONCLUSION

This paper provides the latest progress in applying deep learning techniques to various symbolic music generation tasks. The focus of this paper is on different generation tasks rather than concrete algorithms or models. Thus, people can have a comprehensive and profound understanding of what tasks have appeared in this field, where they have developed, and what shortcomings and challenges these tasks exist, thereby laying the foundation for breakthroughs in this field. This task-oriented taxonomy brings convenience to researchers, as they can quickly gain an overall understanding of the task they are interested in. In addition, we also introduced multiple commonly used music representations, music evaluation methods, and datasets. We hope this survey can help readers better understand the status and trend of symbolic music generation based on deep learning.

In the past decade, symbolic music generation using deep learning techniques has made rapid progress. However, the generated results are still far from expectations, with plenty of problems such as the lack of innovation and music structure, the difficulty of capturing emotion in music, the limited interaction with users, inconsistent music evaluation standards, and so on. In the future, we advocate the following practices: use new methods to improve the learning process and generate more satisfactory music; build a unified evaluation benchmark for the generated music; combine emotion with music generation to explore using the generated emotional music to adjust human emotions; further strengthen the interaction between AI music and humans, so as to achieve the ultimate goal of serving humans; and apply automatic music generation technology to real life to promote social progress and development.

REFERENCES

[1] Gerhard Nierhaus. 2009. *Algorithmic Composition: Paradigms of Automated Music Generation.* Springer Science & Business Media, (2009).

[2] Jose D. Fernández and Francisco J. Vico. 2013. AI methods in algorithmic composition: A comprehensive survey. *J. Artif. Intell. Res.* 48 (2013), 513–582.

[3] Chien-Hung Liu and Chuan-Kang Ting. 2017. Computational intelligence in music composition: A survey. *IEEE Trans. Emerg. Top. Comput. Intell.* 1, 1 (2017), 2–15.

[4] Dorien Herremans, Ching-Hua Chuan, and Elaine Chew. 2017. A functional taxonomy of music generation systems. *ACM Comput. Surv.* 50, 5 (2017), 69:1–69:30.

[5] Jean-Pierre Briot, Gaëtan Hadjeres, and François Pachet. 2017. Deep learning techniques for music generation–a survey. *arXiv preprint arXiv:1709.01620* (2017).

[6] Jean-Pierre Briot and François Pachet. 2017. Music generation by deep learning-challenges and directions. *arXiv preprint arXiv:1712.04371* (2017).

[7] Jean-Pierre Briot. 2021. From artificial neural networks to deep learning for music generation: History, concepts and trends. *Neural Comput. Appl.* 33, 1 (2021), 39–65.

[8] Gerhard Widmer and Werner Goebl. 2004. Computational models of expressive music performance: The state of the art. *J. New Music Res.* 33, 3 (2004), 203–216.

[9] Alexis Kirke and Eduardo R. Miranda. 2013. An overview of computer systems for expressive music performance. *Guide to Computing for Expressive Music Performance* (2013), 1–47.

[10] James Pritchett. 1994. The completion of John Cage's Freeman Etudes. *Perspect. New Music* (1994). 264–270.

[11] Richard C. Pinkerton. 1956. Information theory and melody. *Sci. Am.* 194, 2 (1956), 77–86.

[12] Frederick P. Brooks Jr, Albert L. Hopkins Jr, Peter G. Neumann, and William V. Wright. 1957. An experiment in musical composition. *IRE Trans. Electron. Comput.* 6, 3 (1957), 175–182.

[13] Lejaren A. Hiller Jr and Leonard M. Isaacson. 1958. Musical composition with a high-speed digital computer. *J. Audio Eng. Soc.* 6, 3 (1958), 154–160.

[14] Charles Ames. 1987. Automated composition in retrospect: 1956-1986. *Leonardo*. 20, 2 (1987), 169–185.
[15] Mary Farbood and Bernd Schöner. 2001. Analysis and synthesis of Palestrina-style counterpoint using Markov chains. In *ICMC* (2001).
[16] Peter M. Todd. 1989. A connectionist approach to algorithmic composition. *Comput. Music J.* 13, 4 (1989), 27–43.
[17] John A. Biles. 1994. GenJam: A genetic algorithm for generating jazz solos. In *Proc. of the 1994 International Computer Music Conference*. Vol. 94 (1994), 131–137.
[18] David Cope. 1987. Experiments in musical intelligence. In *Proc. of the 1987 International Computer Music Conference*. 174–181.
[19] Victor Lavrenko and Jeremy Pickens. 2003. Polyphonic music modeling with random fields. *In Proc. of the 11th ACM International Conference on Multimedia*. ACM, 120–129.
[20] Cong Jin, Yun Tie, Yong Bai, Xin Lv, and Shouxun Liu. 2020. A style-specific music composition neural network. *Neural Process. Lett.* 52, 3 (2020), 1–20.
[21] Sarthak Agarwal, Vaibhav Saxena, Vaibhav Singal, and Swati Aggarwal. 2018. LSTM based music generation with dataset preprocessing and reconstruction techniques. In *SSCI 2018*. 455–462.
[22] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.* 9, 8 (1997), 1735–1780.
[23] Douglas Eck and Jürgen Schmidhuber. 2002. Finding temporal structure in music: Blues improvisation with LSTM recurrent networks. In *Proc. of the 12th IEEE Workshop on Neural Networks for Signal Processing*. 747–757.
[24] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. 2012. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. In *ICML 2012*. 1881–1888.
[25] Elliot Waite. 2016. Project Magenta: Generating long-term structure in songs and stories. https://magenta.tensorflow.org/2016/07/15/lookback-rnn-attention-rnn/.
[26] Gaëtan Hadjeres and Frank Nielsen. 2017. Interactive music generation with positional constraints using anticipation-RNNs. *arXiv preprint arXiv:1709.06404* (2017).
[27] Daniel D. Johnson. 2017. Generating polyphonic music using tied parallel networks. In *EvoMUSART 2017*. 128–143.
[28] Adam Roberts, Jesse H. Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. 2018. A hierarchical latent vector model for learning long-term structure in music. In *ICML 2018*. 4361–4370.
[29] Li-Chia Yang, Szu-Yu Chou, and Yi-Hsuan Yang. 2017. MidiNet: A convolutional generative adversarial network for symbolic-domain music generation. In *ISMIR 2017*. 324–331.
[30] Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang, and Yi-Hsuan Yang. 2018. MuseGAN: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. In *AAAI 2018*. 34–41.
[31] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. SeqGAN: Sequence generative adversarial nets with policy gradient. In *AAAI 2017*. 2852–2858.
[32] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Ian Simon, Curtis Hawthorne, Noam Shazeer, Andrew M. Dai, Matthew D. Hoffman, Monica Dinculescu, and Douglas Eck. 2019. Music transformer: Generating music with long-term structure. In *ICLR (Poster) 2019*.
[33] Chris Donahue, Huanru Henry Mao, Yiting Ethan Li, Garrison W. Cottrell, and Julian J. McAuley. 2019. LakhNES: Improving multi-instrumental music generation with cross-domain pre-training. In *ISMIR 2019*. 685–692.
[34] Yu-Siang Huang and Yi-Hsuan Yang. 2020. Pop music transformer: Beat-based modeling and generation of expressive pop piano compositions. In *ACM Multimedia 2020*. 1180–1188.
[35] Luca Casini, Gustavo Marfia, and Marco Roccetti. 2018. Some reflections on the potential and limitations of deep learning for automated music generation. In *PIMRC 2018*. 27–31.
[36] MIDI Manufacturers Association (MMA). MIDI Specifications. https://www.midi.org/specifications.
[37] Yu-Hua Chen, Yu-Hsiang Huang, Wen-Yi Hsiao, and Yi-Hsuan Yang. 2020. Automatic composition of guitar tabs by transformers and grooves modeling. In *ISMIR 2020*.
[38] Dasaem Jeong, Taegyun Kwon, Yoojin Kim, and Juhan Nam. 2019. Graph neural network for music score data and modeling expressive piano performance. In *ICML 2019*. 3060–3070.
[39] Ziyu Wang, Yiyi Zhang, Yixiao Zhang, Junyan Jiang, Ruihan Yang, Junbo Zhao, and Gus Xia. 2020. PianoTree VAE: Structured representation learning for polyphonic music. In *ISMIR 2020*. 368–375.
[40] Mingliang Zeng, Xu Tan, Rui Wang, Zeqian Ju, Tao Qin, and Tie-Yan Liu. 2021. MusicBERT: Symbolic music understanding with large-scale pre-training. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP* (2021), 791–800.
[41] Ning Zhang. 2020. Learning adversarial transformer for symbolic music generation. *IEEE Transactions on Neural Networks and Learning Systems* (2020).
[42] Wen-Yi Hsiao, Jen-Yu Liu, Yin-Cheng Yeh, and Yi-Hsuan Yang. 2021. Compound word transformer: Learning to compose full-song music over dynamic directed hypergraphs. In *AAAI 2021*. 178–186.

[43] Huanru Henry Mao, Taylor Shin, and Garrison W. Cottrell. 2018. DeepJ: Style-specific music generation. In *ICSC 2018*. 377–382.
[44] Kun Zhao, Siqi Li, Juanjuan Cai, Hui Wang, and Jingling Wang. 2019. An emotional symbolic music generation system based on LSTM networks. In *Proc. of the IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*. 2039–2043.
[45] Douglas Eck and Juergen Schmidhuber. 2002. A first look at music composition using LSTM recurrent neural networks. *Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale* 103 (2002), 48.
[46] Gaëtan Hadjeres, François Pachet, and Frank Nielsen. 2017. DeepBach: A steerable model for Bach chorales generation. In *ICML 2017*. 1362–1371.
[47] Luca Angioloni, Tijn Borghuis, Lorenzo Brusci, and Paolo Frasconi. 2020. CONLON: A pseudo-song generator based on a new pianoroll, Wasserstein autoencoders, and optimal interpolations. In *ISMIR 2020*. 876–883.
[48] Benjamin Genchel, Ashis Pati, and Alexander Lerch. 2019. Explicitly conditioned melody generation: A case study with interdependent RNNs. *arXiv preprint arXiv:1907.05208* (2019).
[49] Ashis Pati, Alexander Lerch, and Gaëtan Hadjeres. 2019. Learning to traverse latent spaces for musical score inpainting. In *ISMIR 2019*. 343–351.
[50] Chris Walshaw. 2016. abc notation home page (Accessed on 21/12/2016). https://abcnotation.com
[51] Bob L. Sturm, João Felipe, Santos, Oded Ben-Tal, and Iryna Korshunova. 2016. Music transcription modelling and composition using deep learning. *arXiv preprint arXiv:1604.08723* (2016).
[52] Ching-Hua Chuan and Dorien Herremans. 2018. Modeling temporal tonal relations in polyphonic music through deep networks with a novel image-based representation. In *AAAI 2018*. 2159–2166.
[53] Shiqi Wei and Gus Xia. 2021. Learning long-term music representations via hierarchical contextual constraints. In *ISMIR 2022*. 738–745.
[54] Gaëtan Hadjeres, Frank Nielsen, and François Pachet. 2017. GLSR-VAE: Geodesic latent space regularization for variational autoencoder architectures. In *SSCI 2017*. 1–7.
[55] Duhyeon Bang and Hyunjung Shim. 2021. MGGAN: Solving mode collapse using manifold-guided training. In *Proc. of the 2021 IEEE/CVF International Conference on Computer Vision Workshops*. 2347–2356.
[56] Natasha Jaques, Shixiang Gu, Richard E. Turner, and Douglas Eck. 2017. Tuning recurrent neural networks with reinforcement learning. In *ICLR (Workshop) 2017*.
[57] Natasha Jaques, Shixiang Gu, Dzmitry Bahdanau, José Miguel Hernández-Lobato, Richard E. Turner, and Douglas Eck. 2017. Sequence tutor: Conservative fine-tuning of sequence generation models with KL-control. In *ICML 2017*. 1645–1654.
[58] Otto Fabius, Joost R. van Amersfoort, and Diederik P. Kingma. 2015. Variational recurrent auto-encoders. In *ICLR (Workshop) 2015*.
[59] Zachary M. Ziegler and Alexander M. Rush. 2019. Latent normalizing flows for discrete sequences. In *ICML 2019*. 7673–7682.
[60] Olof Mogren. 2016. C-RNN-GAN: Continuous recurrent neural networks with adversarial training. In *NIPS 2016 (Workshop)*. 1–6.
[61] Christine Payne. 2019. MuseNet. http://openai.com/blog/musenet.
[62] Ziyu Wang and Gus Xia. 2021. MuseBERT: Pre-training music representation for music understanding and controllable generation. In *ISMIR 2021*. 722–729.
[63] Hao-Wen Dong and Yi-Hsuan Yang. 2018. Convolutional generative adversarial networks with binary neurons for polyphonic music generation. In *ISMIR 2018*. 190–196.
[64] Faqian Guan, Chunyan Yu, and Suqiong Yang. 2019. A GAN model with self-attention mechanism to generate multi-instruments symbolic music. In *IJCNN 2019*. 1–6.
[65] Ian Simon, Adam Roberts, Colin Raffel, Jesse H. Engel, Curtis Hawthorne, and Douglas Eck. 2018. Learning a latent space of multitrack measures. *arXiv preprint arXiv:1806.00195* (2018).
[66] Gong Chen, Yan Liu, Sheng-hua Zhong, and Xiang Zhang. 2018. Musicality-novelty generative adversarial nets for algorithmic composition. In *ACM Multimedia 2018*. 1607–1615.
[67] Hongyuan Zhu, Qi Liu, Nicholas Jing Yuan, Chuan Qin, Jiawei Li, Kun Zhang, Guang Zhou, Furu Wei, Yuanchun Xu, and Enhong Chen. 2018. XiaoIce Band: A melody and arrangement generation framework for pop music. In *KDD 2018*. 2837–2846.
[68] Nicholas Trieu and Robert M. Keller. 2018. JazzGAN: Improvising with generative adversarial networks. In *Proc. of the 6th International Workshop on Musical Metacreation (MUME)*.
[69] Google. Improv RNN. Magenta. https://github.com/tensorflow/magenta/tree/master/magenta/models/improv_rnn.
[70] Shunit Haviv Hakimi, Nadav Bhonker, and Ran El-Yaniv. 2020. BebopNet: Deep neural models for personalized jazz improvisations. In *ISMIR 2020*. 828–836.

[71] Vincenzo Madaghiele, Pasquale Lisena, and Raphaël Troncy. 2021. MINGUS: Melodic improvisation neural generator using Seq2Seq. In *ISMRI 2021*. 412–419.

[72] Hyungui Lim, Seungyeon Rhyu, and Kyogu Lee. 2021. Chord generation from symbolic melody using BLSTM networks. In *ISMIR 2017*. 621–627.

[73] Yin-Cheng Yeh, Wen-Yi Hsiao, Satoru Fukayama, Tetsuro Kitahara, Benjamin Genchel, Hao-Min Liu, Hao-Wen Dong, Yian Chen, Terence Leong, and Yi-Hsuan Yang. 2021. Automatic melody harmonization with triad chords: A comparative study. *J. New Music Res.* 50, 1 (2021), 37–51.

[74] Chung-En Sun, Yi-Wei Chen, Hung-Shin Lee, Yen-Hsing Chen, and Hsin-Min Wang. 2021. Melody harmonization using orderless NADE, chord balancing, and blocked Gibbs sampling. In *ICASSP 2021*. 4145–4149.

[75] Cheng-Zhi Anna Huang, Tim Cooijmans, Adam Roberts, Aaron C. Courville, and Douglas Eck. 2017. Counterpoint by convolution. In *ISMIR 2017*. 211–218.

[76] Yi-Wei Chen, Hung-Shin Lee, Yen-Hsing Chen, and Hsin-Min Wang. 2021. SurpriseNet: Melody harmonization conditioning on user-controlled surprise contours. In *ISMIR 2021*. 105–112.

[77] Shangda Wu, Yue Yang, Zhaowen Wang, Xiaobing Li, and Maosong Sun. 2021. Melody harmonization with controllable harmonic rhythm. 2021. *arXiv preprint arXiv:2112.11122* (2021).

[78] Cheng-Zhi Anna Huang, Curtis Hawthorne, Adam Roberts, Monica Dinculescu, James Wexler, Leon Hong, and Jacob Howcroft. 2019. The Bach Doodle: Approachable music composition with machine learning at scale. In *ISMIR 2019*. 793–800.

[79] Ian Simon, Dan Morris, and Sumit Basu. 2008. MySong: Automatic accompaniment generation for vocal melodies. In *Proc. of the 2008 Conference on Human Factors in Computing Systems*. ACM, 725–734.

[80] Hao-Min Liu and Yi-Hsuan Yang. 2018. Lead sheet generation and arrangement by conditional generative adversarial network. In *ICMLA 2018*, 722–727.

[81] Yi Ren, Jinzheng He, Xu Tan, Tao Qin, Zhou Zhao, and Tie-Yan Liu. 2020. PopMAG: Pop music accompaniment generation. In *Proc. of the 28th ACM International Conference on Multimedia*. 1198–1206.

[82] Daphne Ippolito, Anna Huang, Curtis Hawthorne, and Douglas Eck. 2018. Infilling piano performances. In *NIPS Workshop on Machine Learning for Creativity and Design*. (2018).

[83] Wayne Chi, Prachi Kumar, Suri Yaddanapudi, Rahul Suresh, and Umut Isik. 2020. Generating music with a self-correcting non-chronological autoregressive model. In *ISMIR 2020*. 893–900.

[84] Ke Chen, Cheng-i Wang, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. 2020. Music SketchNet: Controllable music generation via factorized representations of pitch and rhythm. In *ISMIR 2020*. 77–84.

[85] Taketo Akama. 2021. A contextual latent space model: Subsequence modulation in melodic sequence. In *ISMIR 2021*. 27–34.

[86] Chin-Jui Chang, Chun-Yi Lee, and Yi-Hsuan Yang. 2021. Variable-length music score infilling via XLNet and musically specialized positional encoding. In *ISMIR 2021*. 97–104.

[87] Rui Guo, Ivor Simpson, Chris Kiefer, Thor Magnusson, and Dorien Herremans. 2022. MusIAC: An extensible generative framework for Music Infilling Applications with multi-level Control. In *EvoMUSART 2022*. 341–356.

[88] Yichao Zhou, Wei Chu, Sam Young, and Xin Chen. 2019. BandNet: A neural network-based, multi-instrument Beatles-style MIDI music composition machine. In *ISMIR 2019*. 655–662.

[89] Hongyuan Zhu, Qi Liu, Nicholas Jing Yuan, Kun Zhang, Guang Zhou, and Enhong Chen. 2020. Pop music generation: From melody to multi-style arrangement. *ACM Trans. Knowl. Discov. Data.* 14, 5 (2020), 54:1-54:31.

[90] Kristy Choi, Curtis Hawthorne, Ian Simon, Monica Dinculescu, and Jesse H. Engel. 2020. Encoding musical style with transformer autoencoders. In *ICML 2020*. 1899–1908.

[91] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. 2016. Image style transfer using convolutional neural networks. In *CVPR 2016*. 2414–2423.

[92] Gino Brunner, Andres Konrad, Yuyi Wang, and Roger Wattenhofer. 2018. MIDI-VAE: Modeling dynamics and instrumentation of music with applications to style transfer. In *ISMIR 2018*. 747–754.

[93] Wei Tsung Lu and Li Su. 2018. Transferring the style of homophonic music using recurrent neural networks and autoregressive model. In *ISMIR 2018*. 740–746.

[94] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu. 2016. WaveNet: A generative model for raw audio. In *Proc. of the 9th ISCA Speech Synthesis Workshop*. 125.

[95] Shih-Lun Wu and Yi-Hsuan Yang. 2021. MuseMorphose: Full-song and fine-grained music style transfer with one transformer VAE. *arXiv preprint arXiv:2105.04090* (2021).

[96] Gino Brunner, Yuyi Wang, Roger Wattenhofer, and Sumu Zhao. 2018. Symbolic music genre transfer with CycleGAN. In *ICTAI 2018*. 786–793.

[97] Ruihan Yang, Dingsu Wang, Ziyu Wang, Tianyao Chen, Junyan Jiang, and Gus Xia. 2019. Deep music analogy via latent representation disentanglement. In *ISMIR 2019*. 596–603.

[98] Gabriele Medeot, Srikanth Cherla, Katerina Kosta, Matt McVicar, Samer Abdallah, Marco Selvi, Ed Newton-Rex, and Kevin Webster. 2018. StructureNet: Inducing structure in generated melodies. In *ISMIR 2018*. 725–731.
[99] Stefan Lattner, Maarten Grachten, and Gerhard Widmer. 2018. Imposing higher-level structure in polyphonic music generation using convolutional restricted Boltzmann machines and constraints. *J. Creative Music Syst.* 2 (2018), 1–31.
[100] Harsh Jhamtani and Taylor Berg-Kirkpatrick. 2019. Modeling self-repetition in music generation using generative adversarial networks. In *ICML (Workshop) 2019*.
[101] Shuqi Dai, Zeyu Jin, Celso Gomes, and Roger B. Dannenberg. 2021. Controllable deep melody generation via hierarchical music structure representation. In *ISMIR 2021*. 143–150.
[102] Xueyao Zhang, Jinchao Zhang, Yao Qiu, Li Wang, and Jie Zhou. 2021. Structure-enhanced pop music generation via harmony-aware learning. *arXiv preprint arXiv:2109.06441* (2021).
[103] Jian Wu, Xiaoguang Liu, Xiaolin Hu, and Jun Zhu. 2020. PopMNet: Generating structured pop music melodies using neural networks. *Artif. Intell.* 286 (2020), 103303.
[104] Yi Zou, Pei Zou, Yi Zhao, Kaixiang Zhang, Ran Zhang, Xiaorui Wang. 2022. MELONS: Generating melody with long-term structure using transformers and structure graph. In *ICASSP 2022*. 191–195.
[105] Lucas Ferreira and Jim Whitehead. 2019. Learning to generate music with sentiment. In *ISMIR 2019*. 384–390.
[106] Lucas N. Ferreira, Levi H. S. Lelis, and Jim Whitehead. 2020. Computer-generated music for tabletop role-playing games. In *Proc. of the 16th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* 16, 1 (2020), 59–65.
[107] Dimos Makris, Kat R. Agres, and Dorien Herremans. 2021. Generating lead sheets with affect: A novel conditional seq2seq framework. In *IJCNN 2021*. 1–8.
[108] Hao Hao Tan and Dorien Herremans. 2020. Music FaderNets: Controllable music generation based on high-level features via low-level feature modelling. In *ISMIR 2020*. 109–116.
[109] Gerhard Widmer. 2001. Using AI and machine learning to study expressive music performance: Project survey and first report. *AI Commun.* 14, 3 (2001), 149–162.
[110] Akira Maezawa, Kazuhiko Yamamoto, and Takuya Fujishima. 2019. Rendering music performance with interpretation variations using conditional variational RNN. In *ISMIR 2019*. 855–861.
[111] Dasaem Jeong, Taegyun Kwon, Yoojin Kim, Kyogu Lee, and Juhan Nam. 2019. VirtuosoNet: A hierarchical RNN-based system for modeling expressive piano performance. In *ISMIR 2019*. 908–915.
[112] Ian Simon and Sageev Oore. 2017. Performance RNN: Generating music with expressive timing and dynamics. *Magenta Blog* (2017).
[113] Sageev Oore, Ian Simon, Sander Dieleman, Douglas Eck, and Karen Simonyan. 2020. This time with feeling: Learning expressive musical performance. *Neural Comput. Appl.* 32, 4 (2020), 955–967.
[114] Aashiq Muhamed, Liang Li, Xingjian Shi, Suri Yaddanapudi, Wayne Chi, Dylan Jackson, Rahul Suresh, Zachary C. Lipton, and Alexander J. Smola. 2021. Symbolic music generation with transformer-GANs. In *AAAI 2021*. 35, 1 (2021), 408–417.
[115] Jon Gillick, Adam Roberts, Jesse H. Engel, Douglas Eck, and David Bamman. 2019. Learning to groove with inverse sequence transformations. In *ICML 2019*. 2269–2279.
[116] Kıvanç Tatar and Philippe Pasquier. 2019. Musical agents: A typology and state of the art towards musical metacreation. *J. New Music Res.* 48, 1 (2019), 56–105.
[117] Mason Bretan, Sageev Oore, Jesse H. Engel, Douglas Eck, and Larry P. Heck. 2017. Deep music: Towards musical dialogue. In *AAAI 2017*. 5081–5082.
[118] Yotam Mann. 2017. AI duet. https://experiments.withgoogle.com/ai/ai-duet.
[119] Pablo Samuel Castro. 2019. Performing structured improvisations with pre-trained deep learning models. In *ICCC 2019*. 306–310.
[120] Nan Jiang, Sheng Jin, Zhiyao Duan, and Changshui Zhang. 2020. RL-Duet: Online music accompaniment generation using deep reinforcement learning. In *AAAI 2020*. 710–718.
[121] Chris Donahue, Ian Simon, and Sander Dieleman. 2019. Piano genie. In *Proc. of the 24th International Conference on Intelligent User Interfaces*. 160–164.
[122] Ryan Louie, Andy Coenen, Cheng Zhi Huang, Michael Terry, and Carrie J. Cai. 2020. Novice-AI music co-creation via AI-steering tools for deep generative models. In *CHI 2020*. 1–13.
[123] Zhaolin Qiu, Yufan Ren, Canchen Li, Hongfu Liu, Yifan Huang, Yiheng Yang, Songruoyao Wu, Hanjia Zheng, Juntao Ji, Jianjia Yu, and Kejun Zhang. 2019. Mind Band: A crossmedia AI music composing platform. In *ACM Multimedia 2019*. 2231–2233.
[124] Ivan P. Yamshchikov and Alexey Tikhonov. 2020. Music generation with variational recurrent autoencoder supported by history. *SN Applied Sciences* 2, 12 (2020), 1–7.
[125] Li-Chia Yang and Alexander Lerch. 2020. On the evaluation of generative models in music. *Neural Comput. Appl.* 32, 9 (2020), 4773–4784.

[126] Abudukelimu Wuerkaixi, Christodoulos Benetatos, Zhiyao Duan, and Changshui Zhang. 2021. CollageNet: Fusing arbitrary melody and accompaniment into a coherent song. In *ISMIR 2021*. 786–793.
[127] Gautam Mittal, Jesse H. Engel, Curtis Hawthorne, and Ian Simon. 2021. Symbolic music generation with diffusion models. In *ISMIR 2021*. 468–475.
[128] Christopher Harte, Mark Sandler, and Martin Gasser. 2006. Detecting harmonic change in musical audio. In *Proc. of the 1st ACM Workshop on Audio and Music Computing Multimedia*. 21–26.
[129] Shih-Lun Wu and Yi-Hsuan Yang. 2020. The jazz transformer on the front line: Exploring the shortcomings of AI-composed music through quantitative measures. In *ISMIR 2020*. 142–149.
[130] Zixun Guo, Dimos Makris, and Dorien Herremans. 2021. Hierarchical recurrent neural networks for conditional melody generation with long-term structure. In *IJCNN 2021*. 1–8.
[131] Cheng-i Wang and Shlomo Dubnov. 2014. Guided music synthesis with variable Markov oracle. In *AIIDE Workshop 2014*.
[132] Eunjeong Stella Koh, Shlomo Dubnov, and Dustin Wright. 2018. Rethinking recurrent latent variable model for music composition. In *MMSP (Workshop) 2018*. 1–6.
[133] Ke Chen, Weilin Zhang, Shlomo Dubnov, and Gus Xia. 2018. The effect of explicit structure encoding of deep neural networks for symbolic music generation. In *Proc. of the International Workshop on Multilayer Music Representation and Processing*. 77-84.
[134] Meinard Müller, Peter Grosche, and Nanzhu Jiang. 2011. A segment-based fitness measure for capturing repetitive structures of music recordings. In *ISMIR 2011*. 615–620.
[135] Meinard Müller and Nanzhu Jiang. 2012. A scape plot representation for visualizing repetitive structures of music recordings. In *ISMIR 2012*. 97–102.
[136] Meinard Müller, Nanzhu Jiang, and Harald G. Grohganz. 2014. SM Toolbox: MATLAB implementations for computing and enhancing similarity matrices. In *AES International Conference on Semantic Audio*.
[137] Zongyu Yin, Federico Reuben, Susan Stepney, and Tom Collins. 2021. "A good algorithm does not steal–it imitates": The originality report as a means of measuring when a music generation algorithm copies too much. In *EvoMUSART 2021*. 360–375.
[138] Hang Chu, Raquel Urtasun, and Sanja Fidler. 2017. Song from PI: A musically plausible network for pop music generation. In *ICLR (Workshop) 2017*.
[139] Ondrej Cífka, Umut Simsekli, and Gaël Richard. 2019. Supervised symbolic music style translation using synthetic data. In *ISMIR 2019*. 588–595.
[140] David Huron. 2002. Music information processing using the Humdrum toolkit: Concepts, examples, and lessons. *Comput. Music J.* 26, 2 (2002), 11–26.
[141] Carol L. Krumhansl. 2001. *Cognitive Foundations of Musical Pitch*. Vol. 17. Oxford University Press, 2001.
[142] Flavio P. Ribeiro, Dinei A. F. Florêncio, Cha Zhang, and Michael L. Seltzer. 2011. CrowdMOS: An approach for crowdsourcing mean opinion score studies. In *ICASSP 2011*. 2416–2419.
[143] Behzad Haki and Sergi Jordà. 2019. A bassline generation system based on sequence-to-sequence learning. In *NIME 2019*. 204–209.
[144] Yujia Yan, Ethan Lustig, Joseph VanderStel, and Zhiyao Duan. 2018. Part-invariant model for music generation and harmonization. In *ISMIR 2018*. 204–210.
[145] I-Chieh Wei, Chih-Wei Wu, and Li Su. 2019. Generating structured drum pattern using variational autoencoder and self-similarity matrix. In *ISMIR 2019*. 847–854.
[146] Feynman T. Liang, Mark Gotham, Matthew Johnson, and Jamie Shotton. 2017. Automatic stylistic composition of Bach chorales with deep LSTM. In *ISMIR 2017*. 449–456.
[147] Michael Buhrmester, Tracy Kwang, and Samuel D. Gosling. 2011. Amazon's Mechanical Turk: A new source of inexpensive, yet high-quality data? *Perspect. Psychol. Sci.* 6 1 (2011), 3–5.
[148] Lucas Theis, Aäron van den Oord, and Matthias Bethge. 2016. A note on the evaluation of generative models. *arXiv preprint arXiv:1511.01844* (2015).
[149] Jeffrey Ens and Philippe Pasquier. 2020. MMM: Exploring conditional multi-track music generation with the transformer. *arXiv preprint arXiv:2008.06048* (2020).
[150] Michael Scott Cuthbert and Christopher Ariza. 2010. Music21: A toolkit for computer-aided musicology and symbolic music data. In *ISMIR 2010*. 637–642.
[151] Thierry Bertin-Mahieux, Daniel P. W. Ellis, Brian Whitman, and Paul Lamere. 2011. The million song dataset. In *ISMIR 2011*. 591–596.
[152] Qiuqiang Kong, Bochen Li, Jitong Chen, and Yuxuan Wang. 2020. GiantMIDI-piano: A large-scale midi dataset for classical piano music. *arXiv preprint arXiv:2010.07061* (2020).
[153] John Thickstun, Zaïd Harchaoui, and Sham M. Kakade. 2017. Learning features of music from scratch. In *ICLR (Poster) 2017*.

[154] Curtis Hawthorne, Andriy Stasyuk, Adam Roberts, Ian Simon, Cheng-Zhi Anna Huang, Sander Dieleman, Erich Elsen, Jesse H. Engel, and Douglas Eck. 2019. Enabling factorized piano music modeling and generation with the MAESTRO dataset. In *ICLR 2019*.
[155] Ziyu Wang, Ke Chen, Junyan Jiang, Yiyi Zhang, Maoran Xu, Shuqi Dai, Xianbin Gu, and Gus Xia. 2020. POP909: A pop-song dataset for music arrangement generation. In *ISMIR 2020*. 38–45.
[156] Francesco Foscarin, Andrew McLeod, Philippe Rigaux, Florent Jacquemard, and Masahiko Sakai. 2020. ASAP: A dataset of aligned scores and performances for piano transcription. In *ISMIR 2020*. 534–541.
[157] Nathan Fradet, Jean-Pierre Briot, Fabien Chhel, Amal El Fallah-Seghrouchni, and Nicolas Gutowski. 2021. MidiTok: A Python package for MIDI file tokenization. In *Extended Abstracts for the Late-Breaking Demo Session of the 22nd Int. Society for Music Information Retrieval Conf., Online, 2021*.
[158] Ashis Pati and Alexander Lerch. 2021. Attribute-based regularization of latent spaces for variational auto-encoders. *Neural Comput. Appl.* 33 (2021), 4429–4444.
[159] Stuart Geman and Donald Geman. 1984. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Trans. Pattern Anal. Mach. Intell.* 6 (1984), 721–741.
[160] Jun S. Liu. 1994. The collapsed Gibbs sampler in Bayesian computations with applications to a gene regulation problem. *J. Am. Stat. Assoc.* 89, 427 (1994), 958–966.
[161] Shlomo Dubnov, Ke Chen, and Kevin Huang. 2022. Deep music information dynamics. *J. Creative Music Syst.* 1, 1 (2022).
[162] Hao-Wen Dong, Ke Chen, Shlomo Dubnov, Julian McAuley, and Taylor Berg-Kirkpatrick. 2023. Multitrack music transformer. In *ICASSP 2023*.
[163] Conan Lu and Shlomo Dubnov. 2021. ChordGAN: Symbolic music style transfer with chroma feature extraction. In *Proceedings of the 2nd Conference on AI Music Creativity, AIMC. 2021*.
[164] Botao Yu, Peiling Lu, Rui Wang, Wei Hu, Xu Tan, Wei Ye, Shikun Zhang, Tao Qin, and Tie-Yan Liu. 2022. Museformer: Transformer with fine- and coarse-grained attention for music generation. In *NIPS 2022*.
[165] Hsiao-Tzu Huang, Joann Ching, Seungheon Doh, Nabin Kim, Juhan Nam, and Yi-Hsuan Yang. 2021. EMOPIA: A multi-modal pop piano dataset for emotion recognition and emotion-based music generation. In *ISMIR 2021*.
[166] James A. Russell. 1980. A circumplex model of affect. *J. Pers. Soc. Psychol.* 39, 6 (1980), 1161.
[167] Curtis Hawthorne, Anna Huang, Daphne Ippolito, and Douglas Eck. 2018. Transformer-NADE for piano performances. In *NIPS 2nd Workshop on Machine Learning for Creativity and Design. 2018*.

Received 22 June 2022; revised 17 April 2023; accepted 2 May 2023