Teaching Chorale Generation Model to Avoid Parallel Motions

Eunjin Choi[1], Hyerin Kim[2], Juhan Nam[1], and Dasaem Jeong[2]

[1] Graduate School of Culture Technology, KAIST
[2] Department of Art & Technology, Sogang University

Abstract. This paper presents a music generation model trained with Bach's chorales and classical music theory rules. Although previous work has shown promising results in generating the four-part harmony, one of the limitations is the frequent appearance of parallel 5th or 8th, which are prohibited in music theory and rarely used in Bach's chorale. To address this issue, we propose an additional loss that minimizes the probability of prohibited patterns, comparing the results with those from inference using a post-hoc probability manipulation to prevent parallel 5th and 8th. The experimental result shows that applying the proposed loss term can help to reduce parallel motion without losing other quality.

Keywords: Music generation, Bach chorale, Domain knowledge injection

1 Introduction

Music generation is a fascinating research topic that has received much attention for centuries. From W.A. Mozart's Musikalisches Würfelspiel (musical dice game), there have been several works conducted in a rule-based approach, such as David Cope's Experiments in Musical Intelligence [1]. Since the success of deep learning, however, data-driven approaches using neural networks have been dominating the music generation. Especially, several Bach chorale generation models have shown promising results [2, 3]. However, previous works [4, 5] pointed out that these models tend to generate note patterns that were avoided by Bach, such as parallel 5th and 8th, which are shown in Figure 1. Fang et al. showed that these parallel 5th and 8th patterns are the most distinctive characteristics to distinguish the model's generation from Bach's original chorales [5].

It is not surprising that the data-driven model could generate prohibited patterns, because most music generation models, including language-modeling-based, use likelihood maximization for the training objective; models only learn the pattern that exists in the training dataset. Therefore, this training strategy is not effective in teaching the model what patterns to avoid.

In this context, we raise three research questions. First, how can we inject music domain knowledge or rules into the data-driven generation model? Second, how can we

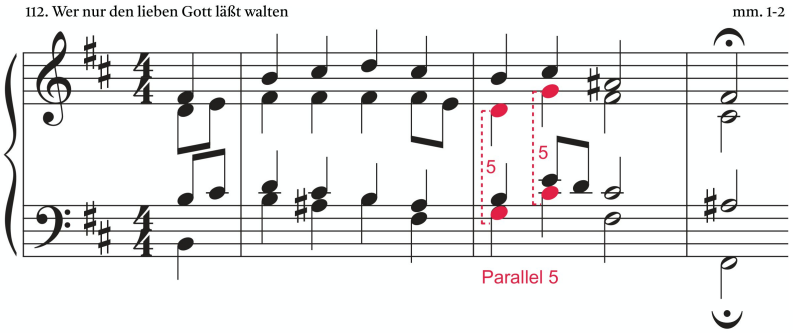

Fig. 1. Example of parallel 5th in four voice chorale. The highlighted notes are intentionally manipulated to demonstrate parallel 5th, which are rare in Bach's original chorale.

teach the generation model to recognize the absence of something rather than its presence? Third, can we improve the generation model performance using music theory?

To address this issue, we propose a novel musically-informed loss term for training a music generation model. We compared the experimental results with the JS Fake Chorale dataset [6]. Generated music samples are available on the online webpage. [3]

2 Related Works

2.1 Deep Music Generation with Domain Knowledge

While early deep learning-based music generation studies were conducted by bringing models from computer vision [7] or natural language processing (NLP) [8, 9] domains, there are an increasing number of music generation models inspired by intuition from music domain. We have summarized these cases into two main groups. One group focuses on the structure and repetition of music defined as a unit of pattern such as theme [10], loop [11], bar relations [12], skeleton [13], or hierarchical structure [14], and extracted the unit with a rule-based approach and utilized it for modeling, or focused on modeling the structure of music using hierarchical encoding methods [15–17]. The other group utilizes intuition from music domain knowledge to suggest novel music generation system paradigms: non-unidirectional music generation system suggested by Coconet [3], and BERT-based music generation system [18, 19], combinatorial music generation system [20], harmonic expectation-based music harmonization system [21].

[3] https://bit.ly/3LMajEK

To the best of our knowledge, so far there have been no (or few) studies that have used music theory directly in the training procedure of deep learning-based music generation. We conjecture that it is not necessary to follow the rules strictly unless it is classical music, and the stricter the rules we impose on the model, the less diverse the music produced. Rather, studies indirectly utilize some music knowledge by steering models to learn specific features of music.

[22] proposed a contrastive loss that steers a music transformer to have arbitrary logical music features. Studies on the disentanglement of latent representation [23, 24] proposed models to learn specific music features in certain latent dimensions, which can be useful to steer the model in a controllable way. Studies that focus on the chord-conditioned music generation [25, 26] could be an example that uses music theory in a music generation as well. However, as we noted in the introduction section, the likelihood-based models ended up rarely generating notes prohibited in music theory. In this context, our approach has novelty in that it uses music knowledge directly in the model training scheme so that the model could learn to avoid prohibited patterns according to music theory.

2.2 Bach Chorale Generation

As a representative corpus of Western classical polyphonic music, Bach's chorale has been widely adopted for music generation research. Among many, we introduce previous deep learning-based approaches to Bach chorale. BachBot [27] is one of the earliest examples of success in modeling Bach chorale with the deep neural network, or long short-term memory (LSTM) more specifically. BachBot uses 16th-grid sampling, along with additional tokens for time grid delimiter. Fermata, which plays a critical role in notating the phrase boundary in Bach's chorale, was also considered as an additional token in BachBot.

DeepBach [2] proposed to apply pseudo-Gibbs sampling instead of generating the music in sequential order. While DeepBach used LSTM as a main neural network block, CocoNet [3] applied a convolutional neural network using similar ideas of applying Gibbs sampling. The model was employed as a backbone to serve Google's first AI-powered doodle, Bach doodle, which generates Bach-like harmonization for a user's input melody.

Another recently proposed model, TonicNet[28], is closer to BachBot in the sense that it uses the 16th-grid sampling with ancestral sampling. Here, the author proposed a feature-rich encoding scheme, such as a number of sustain counts for each voice and adding a chord token at the head of each time frame. The author later proposed the JS fake Chorale [6], a dataset of machine-generated chorales, even though an explanation of the model used for the generation was not provided along.

The frequent appearance of parallel 5th and 8th is considered as a problem with deep learning-based Bach chorales generation. However, no previous research has made a direct attempt to reduce the parallel motions of the generated chorale. This paper suggests a novel loss function that directly prohibits parallel motion.

3 Methods

3.1 Problem Formulation

In ordinary language modeling, the problem can be defined as modeling the probability distribution of the next token for given previous tokens, such as $P(x_{t+1}|x_0, \dots x_t)$. However, in the symbolic music generation, one can provide more information about the current time step before predicting the token, such as a beat position or which voice the current token has to belong to. We can group this information as a condition c and formulate the music language modeling as Equation 1

$$P(x_t|x_0, \dots, x_{t-1}, c_0, \dots, c_t) \tag{1}$$

where x_t and c_t represents a predicting token and a condition token of timestep t, respectively. During the inference, c can be calculated by a rule-based approach for every timestep. Since the condition of the current time step is given explicitly, the model does not have to implicitly predict the information, such as to which beat the current time step belongs. While providing the condition also can be done synchronously with the predicting token x_t, separating the c_t from x_{t-1} has several advantages. Even though c_t is easily predictable for a given c_{t-1} in many cases, there are some exceptional cases, such as measure boundary with different time signatures. If we notate the offset of the current time step from measure starting in the sixteenth notes, the next offset for 11 is 0 for time signature 3/4 and 12 for time signature 4/4. By providing c_t instead of c_{t-1}, we can eliminate this type of ambiguity.

While any causal model, such as a transformer decoder, can be used for this task, we used a stack of uni-directional GRU as our model. We also tried a stack of transformer decoder module, but the result was not better than GRU.

3.2 Data Representation

Following the previous works on Bach chorales generation [2, 27, 28], we use 16th-grid sampling so that a single bar of 4/4 time signature is represented with sixteen-time frames. A single voice is represented as a sequence of F-dimensional tokens $v \in \mathbb{Z}^{T \times F}$, where T represents the number of total time frames and F represents the number of features. Thus, an entire four-voice chorale can be represented as $c \in \mathbb{Z}^{(T \times 4) \times F}$ and this flattened voices as (S, A, T, B)-repeated order was fed to the GRU model.

To extract metadata such as the number of sharp in the key signature and time signature, we used the music21 [29] library. Analyzing the major minor tonality was done using the Krumhansl-Schmuckler key-finding algorithm [30] in music21. For pitch representation, we adapted a sustain token for representing the same repetitive pitch without onset, following [2]. The selected features we used are described in Table 1. Besides the features in Table 1, we also considered the tonality(major or minor), num beat in 3/4 or 4/4 time, voice index of current time step, the recent previous MIDI pitch value of current voice, the number of time step current voice sustained, beat distance from last fermata, and remaining number of fermata. However, our preliminary ablation study showed that using Pitch, Fermata, Beat position, Beat strength, and Num sharp in key

Table 1. List of considered features for note encoding.

Feature	Description	Type
Pitch	MIDI pitch value of current time step	I & O
Fermata	1 when the fermata starts at current time step, 0 otherwise	I & O
Beat position	Beat position in sixteenth note grid (0 - 15 for 4/4)	I
Beat strength	Beat strength in sixteenth note grid	I
Num sharp in key	The number of [-flats/+sharps] in key signature.	I

(PFBBN) was most effective in modeling Bach-like music for our generation model. Therefore, we used PFBBN as a default encoding scheme of experiments in this paper.

It's important to highlight that the dimensions of our input and output features differ in our study. For our input, we utilized all the features previously described. However, for our output feature, x_t, which the model is tasked with predicting, we only incorporated pitch and fermata. The features that aren't predicted, c_t are initially fed into the model shifted to the left by one step so that x_{t-1} and c_t are concatenated together. This allows the model to anticipate the subsequent token x_t based on $(x_0, x_1, \ldots, x_{t-1}, c_0, c_1, \ldots, c_t)$. During the inference process, c_{t+1} was obtained using a rule-based approach. For the initial condition token c_0, we derived it from the pre-established distribution for each feature across the entire dataset.

3.3 Pitch Onset Loss

As we used note sustain as an independent token, we found that this sustain token appears 2.5 times more often than note onset, or change of pitch in the dataset. This can lead the model to predict sustain too frequently, as this single token occupies 70 % of entire pitch values. Therefore, we additionally imposed pitch onset loss, a pitch loss of a time step where onset exists, to enforce the model to focus more on the note onset and not hold the same pitch too much time during inference. The onset boolean can be represented as $o \in \{0, 1\}^T$ for a single voice $v \in \mathbb{Z}^{T \times F}$, where $o_t = 1$ if the voice has a note onset at time frame t and otherwise $o_t = 0$. The pitch onset loss L_{po} can be represented as an equation below.

$$L_{po} = \frac{1}{T} \sum_t o_t \cdot (-\log \hat{y}_t) \tag{2}$$

3.4 Loss Function Design According to Music Theory

Parallel Prohibition Loss We designed a loss function that penalizes parallel 5th and parallel 8th, which imitates one of the most marked rules for composing counterpoint. Even though we can also penalize concealed 5th and 8th along with the parallel, we only focus on the parallel error in this work. To force the model to avoid these prohibited patterns, our system calculates prohibition matrix $\mathbf{Pr}$ for a given preceding voice $v \in \mathbb{Z}^{T \times F}$ and the following voice $w \in \mathbb{Z}^{T \times F}$ using a rule-based algorithm. The result can be denoted as $\mathbf{Pr}_{v,w} \in \{0, 1\}^{T \times P}$, where T and P represent the number of time frames and total note pitch in the vocabulary, respectively.

The prohibited pitches $f(p, q, t)$ at time t for a sequence of MIDI pitch for voice, $q \in \mathbb{N}^T$, for a sequence of MIDI pitch for the preceding voice, $p \in \mathbb{N}^T$, can be represented as below:

$$f(p,q,t) = \begin{cases} q_{t-1} + (p_t - p_{t-1}) & \text{if } |p_{t-1} - q_{t-1}| \equiv 7 \text{ or } 0 (\text{mod} 12) \\ & \text{and } p_{t-1} \neq p_t \text{ and } q_t \neq p_t - p_{t-1} + q_{t-1} \\ 0 & \text{else} \end{cases} \tag{3}$$

We intentionally did not prohibit the parallel progression that actually occurred in training set for two primary reasons. Firstly, there are instances where Bach himself did not adhere to the prohibition rule. Secondly, the log-likelihood loss and the prohibit loss directly conflict with each other. While the log-likelihood loss seeks to maximize the probability of a particular note, the prohibition loss aims to minimize the probability of that very same note. Therefore, we did not apply the prohibition rule in these cases.

Using $f(p, q, t)$, the piano-roll-like prohibition matrix $\mathbf{Pr}_{v,w} \in \{0, 1\}^{T \times P}$ for the voice w and its preceding voice v, can be represented as Equation 4.

$$\mathbf{Pr}_{v,w}[n, t] = \begin{cases} 1 & \text{if } f(v, w, t) = n \\ 0 & \text{else} \end{cases} \tag{4}$$

The integrated $\mathbf{Pr}_i$, the prohibition matrix for i-th voice for every preceding voice, can be represented as Equation 5, where u_i represents a sequence of features for i-th voice. For example, if the voice order is soprano, alto, tenor, and bass, u_0 is soprano, and u_3 is bass.

$$\mathbf{Pr}_i = \sum_{j}^{i-1} \mathbf{Pr}_{u_j, u_i} \tag{5}$$

After the language model predicts the shifted events, we calculated the prohibition loss L_{phb}, the cross entropy loss between the predicted pitch token probabilities $\hat{y} \in (0, 1)^{T \times P}$ and the prohibit matrix $\mathbf{Pr}$, which can be represented as an equation below:

$$L_{phb} = -\frac{1}{T} \sum_{t} \mathbf{Pr}[t] \cdot (\log(1 - \hat{y}_t^{\alpha})) \tag{6}$$

where α is a hyperparameter, which helps to preserve loss and gradient for small $\hat{y}$. In our experiment, we used $\alpha = 0.5$. Minimizing L_{phb} forces $\hat{y}$ to be close to zero in the case of prohibited pitches. We have also tested to maximize $-\log(\hat{y})$, but this often results in unstable training since the gradient explodes around $\log(0)$.

Our final loss function is formulated as follows:

$$L_{total} = L_{LM} + \lambda_{phb} \cdot L_{phb} + \lambda_{po} \cdot L_{po} \tag{7}$$

where L_{LM} is Cross Entropy loss between predicted pitch and fermata tokens and target, L_{phb} and L_{po} are prohibited and pitch onset loss, and λ_{phb} and λ_{po} are weights for prohibit and pitch onset loss. We applied weight annealing for λphb, so that $\lambda phb = 0$ for the first 10% of iteration, and apply sigmoid annealing, so that the prohibition loss is gradually applied after the training becomes stable.

Rule-based Parallel Masking To compare the effectiveness of applying parallel prohibition loss, we also tested a rule-based parallel masking that avoids parallel progress during inference. Using a similar approach in Equation 3, we calculated the possible prohibited pitch for every step of the autoregressive inference. While we only prohibited parallel progression with exactly the same interval in the prohibited pitch during the training, we prohibited every possible pitch across the entire octave that makes the same 5th or 8th interval as a pitch class during the inference so that we could achieve zero parallel errors in the evaluation metric.

4 Experiments

In this section, we describe the experiment to investigate the effect of our suggested prohibit loss and pitch onset loss terms.

4.1 Dataset

For the train and validation dataset, we used 366 Chorales of Johann Sebastian Bach, which are provided in the format of Humdrum kern [31]. The data provides note information of each of the four voices, including the fermata symbol.

4.2 Experiment Setting

We split the dataset as 9:1 for the train and validation dataset. For the model, we used a 4-layer GRU model with a hidden size of 512 and a dropout rate of 0.2. For the hyperparameter, we used batch size 8, Adam optimizer with learning rate 1e-3, and Step LR scheduler with step size 2k and gamma 0.8. Since the model normally converges within 30k steps, we used 30k steps to train the model. For the embedding size of used features, the default feature embedding size is 512 and all features have a feature dimension ratio of 0.1 of the default feature embedding size, which corresponds to 51. As the pitch feature is the most important feature, we used dimension ratio 0.75 for the pitch, which makes 384 dimensions. The embeddings from each feature are all concatenated, forming a total of 588 dimensions.

4.3 Evaluation Metric

Since we change loss weight with different values, total validation loss values are not directly comparable to evaluate the model performance. Therefore, for evaluation, we used the metric suggested by [5], which calculates Wasserstein distance of distribution of generated note, rhythm, parallel errors, harmonic quality, intervals of each voice (S, A, T, B), repeated sequence, and overall grade values compared to the Bach's original chorale dataset. To compare with the previous works, we evaluate our suggested model with JS Fake Chorale Dataset [6].

The original implementation of the metric [5] distinguishes enharmonic like C♯ and D♭, which are encoded in the same MIDI pitch. MIDI files generated from our proposed method or from JS Fake Chorale get severe distortion when converted by music21 in

the evaluation code. For example, MIDI pitch from 64 to 63 can be encoded either E4 to D♯3 (minor second) or E4 to E♭4 (augmented first). If this interval is interpreted as E4 to E♭4 by music21, this makes a large error in Wasserstein distance because Bach's original chorale corpus uses a lot of minor seconds but not augmented first. Therefore, we modified the code to use interval and note pitch classes in MIDI pitch, not distinguishing enharmonic notes. Note that we calculated the distribution of each feature in the Bach chorale corpus based on the dataset from [31], which is slightly different from the one used in [5].

Table 2. Experiment results for suggested losses. PE: parallel error, HQ: harmonic quality, B Intervals: bass intervals, RS: repeated sequence. Lower values mean better chorales. Here, the first row means using only PFBBN feature. **Bold** values are minimum values among our model conditions.

Conditions			Metrics						
L_{phb}	Masking	L_{po}	Note	Rhythm	PE	HQ	B Intervals	RS	Grade
×	×	×	0.30 (0.16)	0.22 (0.16)	0.94 (2.39)	0.62 (0.38)	0.42 (0.21)	1.38 (0.93)	4.77 (2.76)
√	×	×	0.30 (0.17)	0.22 (0.15)	0.74 (1.77)	0.63 (0.41)	0.41 (0.23)	1.38 (0.91)	4.58 (2.28)
×	√	×	0.31 (0.20)	0.23 (0.12)	**0.0 (0.0)**	0.65 (0.38)	0.54 (0.33)	1.52 (2.11)	**4.19 (2.47)**
√	√	×	0.36 (0.23)	0.21 (0.10)	**0.0 (0.0)**	0.69 (0.39)	0.68 (0.42)	1.40 (0.77)	4.25 (1.31)
×	×	0.2	**0.29 (0.16)**	0.21 (0.09)	1.00 (2.60)	0.61 (0.36)	0.41 (0.19)	1.30 (0.81)	4.72 (2.94)
×	×	0.5	0.31 (0.18)	0.21 (0.14)	0.67 (1.56)	0.61 (0.36)	**0.40 (0.19)**	1.33 (0.86)	4.42 (1.94)
×	×	1.0	0.30 (0.17)	0.20 (0.11)	0.76 (2.24)	**0.59 (0.37)**	**0.40 (0.19)**	**1.26 (0.61)**	4.40 (2.43)
×	×	2.0	0.31 (0.17)	0.21 (0.10)	0.69 (1.86)	0.61 (0.37)	0.40 (0.20)	1.33 (0.72)	4.46 (2.27)
√	×	1.0	0.31 (0.18)	**0.20 (0.09)**	0.52 (1.3)	0.63 (0.42)	0.40 (0.20)	1.28 (0.63)	4.27 (1.61)
Bach [31]			0.27 (0.15)	0.25 (0.16)	0.30 (0.88)	0.57 (0.32)	0.40 (0.21)	1.43 (0.92)	4.14 (1.60)
JS Fake [6]			0.29 (0.14)	0.17 (0.07)	3.91 (4.01)	1.03 (0.77)	0.37 (0.16)	1.12 (0.40)	7.73 (4.43)

4.4 Effect of Losses

To investigate whether the suggested loss terms are effective, we conducted an ablation study of prohibition loss and pitch onset loss. As we mentioned earlier, we selected PFBBN as the baseline to apply the loss. Since the voice intervals except bass (S, A, T) are not significantly different among the conditions, we omit the column in the result table 2.

For prohibition loss, we experimented with $\lambda_{phb} = 1k$, which yields the lowest parallel error in our preliminary experiment. Rule-based parallel masking was also compared with the condition using λ_{phb}. Similarly, we tested the effect of the pitch onset loss with $\lambda_{po} = 0, 0.2, 0.5, 1.0, 2.0$. The results are shown in Table 2. The result shows that parallel prohibition loss helped to reduce parallel errors but could not completely avoid them. This is also partially due to the fact that the training data itself does not perfectly exclude parallel motion. For the pitch onset loss experiment, we found that using $\lambda_{po} = 1.0$ results in the best performance for most of the metrics. The combination of prohibition loss and the pitch onset loss showed the best performance, which is nearly similar to the metric of Bach's original corpus.

Table 2 reveals that our rule-based masking method can effectively eliminate parallel errors during inference. However, this approach also resulted in a degradation of the metric for harmonic quality or bass intervals, as the model must sample lower-probability pitches to avoid producing parallel fifths. The most significant impact is observed in the Wasserstein distance of the bass interval, as the bass voice is influenced by three preceding voices, resulting in a more densely constrained inference process. Therefore, rule-based hard masking has to be carefully considered.

5 Conclusion

In this paper, we suggested a music theory-based novel loss term and applied it to the Bach chorale generation. Using the previously suggested quantitative evaluation metric, we showed that the model can generate chorales in quality that follow a similar distribution of musical characteristics as Bach's corpus. We found that the suggested loss terms could improve the sample quality of generated chorale in terms of parallel errors, which was one of the main critical limitations of previous chorale generation models.

For further study, we will continue investigating the effect of voice order and pitch augmentation to improve the generated sample quality. Also, we can apply other well-known prohibitions such as concealed fifth and voice crossing into our prohibit loss term. Although our current work only studied hard prohibition (strict prohibition), soft prohibition is another topic of imposing the rule to the model.

6 Acknowledgment

This work was supported by the Year 2022 Culture Technology R&D Program by the Ministry of Culture, Sports and Tourism and Korea Creative Content Agency (Project Name: Research Talent Training Program for Emerging Technologies in Games, Project Number: R2020040211) and Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Korea Government (MSIT) (NRF-2022R1F1A1074566).

References

1. Alcedo Coenen. David Cope, Experiments in Musical Intelligence. A-R Editions, Madison, Wisconsin, USA. Vol. 12 1996. *Organised Sound*, 2(1):57–60, 1997.
2. Gaëtan Hadjeres, François Pachet, and Frank Nielsen. DeepBach: A Steerable Model for Bach Chorales Generation. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 1362–1371, 2017.
3. Cheng-Zhi Anna Huang, Tim Cooijmans, Adam Roberts, Aaron Courville, and Douglas Eck. Counterpoint by Convolution. In *Proceedings of 18th International Conference on Music Information Retrieval (ISMIR)*, pages 211–218, 2017.
4. Cheng-Zhi Anna Huang, Curtis Hawthorne, Adam Roberts, Monica Dinculescu, James Wexler, Leon Hong, and Jacob Howcroft. Approachable Music Composition with Machine Learning at Scale. In *Proceedings of the 20th International Conference on Music Information Retrieval (ISMIR)*, pages 793–800, 2019.

5. Alexander Fang, Alisa Liu, Prem Seetharaman, and Bryan Pardo. Bach or Mock? A Grading Function for Chorales in the Style of J.S Bach. In *Machine Learning for Media Discovery (ML4MD) Workshop at the International Conference on Machine Learning (ICML)*, 2020.
6. Omar Peracha. JS Fake Chorales: A Synthetic Dataset of Polyphonic Music with Human Annotation. *arXiv preprint arXiv:2107.10388*, 2021.
7. Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang, and Yi-Hsuan Yang. Musegan: Multi-track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment. In *Proceedings of the 32th AAAI Conference on Artificial Intelligence*, volume 32, 2018.
8. Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Ian Simon, Curtis Hawthorne, Andrew M Dai, Matthew D Hoffman, Monica Dinculescu, and Douglas Eck. Music Transformer. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.
9. Yu-Siang Huang and Yi-Hsuan Yang. Pop Music Transformer: Beat-based Modeling and Generation of Expressive Pop Piano Compositions. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 1180–1188, 2020.
10. Yi-Jen Shih, Shih-Lun Wu, Frank Zalkow, Meinard Muller, and Yi-Hsuan Yang. Theme Transformer: Symbolic Music Generation with Theme-conditioned Transformer. *IEEE Transactions on Multimedia*, 2022.
11. Sangjun Han, Hyeongrae Ihm, Moontae Lee, and Woohyung Lim. Symbolic Music Loop Generation with Neural Discrete Representations. In *Proceedings of 23rd International Conference on Music Information Retrieval (ISMIR)*, pages 403–410, 2022.
12. Yi Zou, Pei Zou, Yi Zhao, Kaixiang Zhang, Ran Zhang, and Xiaorui Wang. MELONS: Generating Melody with Long-term Structure using Transformers and Structure Graph. In *2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 191–195, 2022.
13. Kejun Zhang, Xinda Wu, Tieyao Zhang, Zhijie Huang, Xu Tan, Qihao Liang, Songruoyao Wu, and Lingyun Sun. Wuyun: Exploring Hierarchical Skeleton-guided Melody Generation using Knowledge-enhanced Deep Learning. *arXiv preprint arXiv:2301.04488*, 2023.
14. Shuqi Dai, Zeyu Jin, Celso Gomes, and Roger B Dannenberg. Controllable Deep Melody Generation via Hierarchical Music Structure Representation. In *Proceedings of 22nd International Conference on Music Information Retrieval (ISMIR)*, 2021.
15. Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. Jukebox: A Generative Model for Music. *arXiv preprint arXiv:2005.00341*, 2020.
16. Ziyu Wang, Yiyi Zhang, Yixiao Zhang, Junyan Jiang, Ruihan Yang, Junbo Zhao, and Gus Xia. Pianotree VAE: Structured Representation Learning for Polyphonic Music. In *Proceedings of 21st International Conference on Music Information Retrieval (ISMIR)*, 2020.
17. Junyan Jiang, Gus G Xia, Dave B Carlton, Chris N Anderson, and Ryan H Miyakawa. Transformer VAE: A Hierarchical Model for Structure-aware and Interpretable Music Representation Learning. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 516–520, 2020.
18. Ke Chen, Cheng-i Wang, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. Music SketchNet: Controllable Music Generation via Factorized Representations of Pitch and Rhythm. In *Proceedings of 21st International Conference on Music Information Retrieval (ISMIR)*, 2020.
19. Ziyu Wang and Gus Xia. MuseBERT: Pre-training Music Representation for Music Understanding and Controllable Generation. In *Proceedings of 22nd International Conference on Music Information Retrieval (ISMIR)*, pages 722–729, 2021.
20. Hyun Lee, Taehyun Kim, Hyolim Kang, Minjoo Ki, Hyeonchan Hwang, Sharang Han, Seon Joo Kim, et al. ComMU: Dataset for Combinatorial Music Generation. In *Advances in Neural Information Processing Systems*, volume 35, pages 39103–39114, 2022.

21. Yi-Wei Chen, Hung-Shin Lee, Yen-Hsing Chen, and Hsin-Min Wang. SurpriseNet: Melody Harmonization Conditioning on User-controlled Surprise Contours. In *Proceedings of 22nd International Conference on Music Information Retrieval (ISMIR)*, pages 105–112, 2021.
22. Halley Young, Vincent Dumoulin, Pablo S Castro, Jesse Engel, and Cheng-Zhi Anna Huang. Compositional Steering of Music Transformers. 2022.
23. Hao Hao Tan and Dorien Herremans. Music Fadernets: Controllable Music Generation based on High-level Features via Low-level Feature Modelling. In *Proceedings of 19th International Conference on Music Information Retrieval (ISMIR)*, 2020.
24. Ashis Pati and Alexander Lerch. Is Disentanglement Enough? On Latent Representations for Controllable Music Generation. In *Proceedings of 22nd International Society for Music Information Retrieval Conference (ISMIR)*, 2021.
25. Kyoyun Choi, Jonggwon Park, Wan Heo, Sungwook Jeon, and Jonghun Park. Chord Conditioned Melody Generation with Transformer based Decoders. *IEEE Access*, 9:42071–42080, 2021.
26. Seungyeon Rhyu, Hyeonseok Choi, Sarah Kim, and Kyogu Lee. Translating Melody to Chord: Structured and Flexible Harmonization of Melody with Transformer. *IEEE Access*, 10:28261–28273, 2022.
27. Feynman T Liang, Mark Gotham, Matthew Johnson, and Jamie Shotton. Automatic Stylistic Composition of Bach Chorales with Deep LSTM. In *Proceedings of 18th International Conference on Music Information Retrieval (ISMIR)*, pages 449–456, 2017.
28. Omar Peracha. Improving Polyphonic Music Models with Feature-rich Encoding. In *Proceedings of 21st International Conference on Music Information Retrieval (ISMIR)*, 2020.
29. Michael Scott Cuthbert and Christopher Ariza. music21: A Toolkit for Computer-aided Musicology and Symbolic Music Data. In *Proceedings of 11th International Conference on Music Information Retrieval (ISMIR)*, pages 637–642, 2010.
30. David Temperley. What's Key for Key? The Krumhansl-Schmuckler Key-Finding Algorithm Reconsidered. *Music Perception*, 17(1):65–100, 10 1999.
31. Nathaniel Condit-Schultz, Yaolong Ju, and Ichiro Fujinaga. A Flexible Approach to Automated Harmonic Analysis: Multiple Annotations of Chorales by Bach and Prætorius. In *Proceedings of 19th International Conference on Music Information Retrieval (ISMIR)*, pages 66–73, 2018.