MUSETOK: SYMBOLIC MUSIC TOKENIZATION FOR GENERATION AND SEMANTIC UNDERSTANDING

Jingyue Huang *Zachary Novack* *Phillip Long* *Yupeng Hou*
Ke Chen *Taylor Berg-Kirkpatrick* *Julian McAuley*

University of California San Diego, USA

ABSTRACT

Discrete representation learning has shown promising results across various domains, including generation and understanding in image, speech and language. Inspired by these advances, we propose MuseTok, a tokenization method for symbolic music, and investigate its effectiveness in both music generation and understanding tasks. MuseTok employs the residual vector quantized-variational autoencoder (RQ-VAE) on bar-wise music segments within a Transformer-based encoder-decoder framework, producing music codes that achieve high-fidelity music reconstruction and accurate understanding of music theory. For comprehensive evaluation, we apply MuseTok to music generation and semantic understanding tasks, including melody extraction, chord recognition, and emotion recognition. Models incorporating MuseTok outperform previous representation learning baselines in semantic understanding while maintaining comparable performance in content generation. Furthermore, qualitative analyses on MuseTok codes, using ground-truth categories and synthetic datasets, reveal that MuseTok effectively captures underlying musical concepts from large music collections.

***Index Terms*—** Representation Learning, Music Tokenization, Symbolic Music Generation, Music Understanding

1. INTRODUCTION

Discrete representation learning aims to train models to represent data within a finite set of discrete codes [1]. It has proven effective across diverse generative tasks, including image generation [1–3], neural speech codec [4], generative retrieval for recommender systems [5] and signal-level music generation [6, 7]. In music information retrieval (MIR), discrete representations have also been applied to genre classification [8] and melody transcription [9]. Such methods span a wide range of compression bottlenecks, from lightly compressed discrete codes for improved modeling [7], to highly compressed codes capturing deep semantic information [10]. However, work in discrete representation learning currently lags in the *symbolic* music domain. While some limited previous research has explored the use of discrete representations for classification tasks [11, 12] or controllable generation [13–15], such work only focused on the specific application rather than on general representations for diverse tasks, with limited attention to *how* one should learn semantic embeddings, nor to the *quality* of such representations.

Thus, we introduce **MuseTok**, the first tokenization method for general symbolic music representations that can support multiple applications, including symbolic music generation and semantic music understanding in multiple perspectives. We leverage an encoder-decoder architecture with residual quantization [3] to learn bar-wise music residual codes through reconstruction, on top of music sequences derived by REMI+ [15]. Analysis of code usage and similarities demonstrates its effectiveness in capturing music theoretical concepts, such as textures and musical intervals.

Regarding the applications, for music generation, we employ a Transformer decoder to predict MuseTok codes, then pass codes to another Transformer decoder to generate REMI+ events. For semantic music understanding, three classification tasks are considered to assess the note-level, bar-level and song-level music semantics embedded in the codes. We adopt public-domain symbolic music data for model training, including a large-scale dataset PDMX [16, 17] and several small datasets [18–23] spanning diverse genres, with a main focus on piano pieces to explore discrete representation learning in a single-instrument setting. Our contributions are three-fold:

- We propose MuseTok, the first discrete representation learning framework of symbolic music for general purpose, applicable to both generation and understanding tasks.
- MuseTok achieves comparable performance on symbolic music generation and superior performance on two of three classification tasks to previous baselines, demonstrating its effectiveness on content generation and semantic understanding.
- We provide analysis on how MuseTok learns underlying musical concepts, such as key, interval, time signature, and texture.

We present generation samples and more details about datasets and experiments on the website[1]. Code implementation and checkpoints are open sourced[2].

2. RELATED WORK

Music Representation Learning. The success of representation learning methods [1–7, 24–26] has inspired exploration in the symbolic music domain. For music understanding, BERT-like models have been applied for classification tasks [11, 12], while contrastive methods integrate music with language [27]. For generation, VAE-based models disentangle latent variables to encode attributes like chord and texture, enabling controllable generation [14, 28] and style transfer [13, 15]. Unlike prior work targeting specific tasks, this paper focuses on general symbolic music representations, exploring their broad potential in music downstream tasks and quality.

Symbolic Music Encoding. Various encoding formats have been proposed for symbolic music generation. MIDI-Message [29, 30] and REMI [31] encode MIDI data as sequence of events like note, beat and time shift. Later work introduced representations for compound attributes [19], multiple instrument tracks [15, 32, 33], expressive performance [34, 35] and emotion control [20, 36]. This paper investigates discrete music representation learning on the bar-level segments, introducing a new tokenization for music generation and understanding.

[1] https://musetok.github.io/

[2] https://github.com/Yuer867/MuseTok

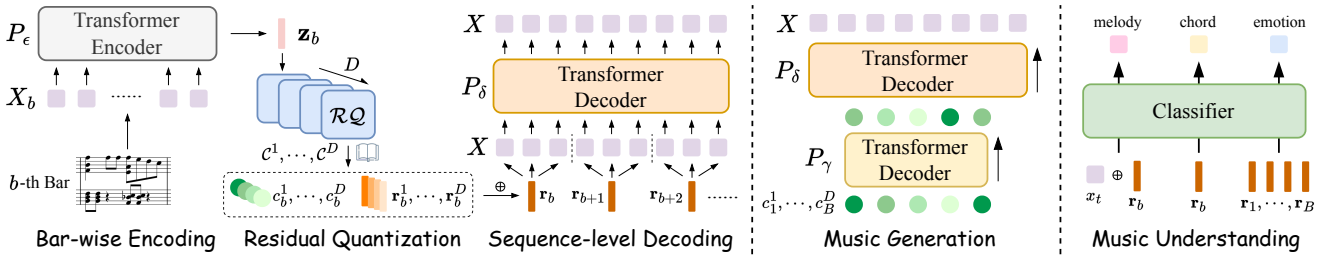

Fig. 1. Overview of MuseTok (left) and its downstream generation (middle) and understanding (right) tasks.

3. METHODS

3.1. Music Tokenization

As illustrated in Fig. 1 (left), we adopt the idea of RQ-VAE [3] to construct an encoder-decoder architecture with residual quantization (RQ) blocks to learn discrete representations of symbolic music.

Encoder. Symbolic music input is converted into a REMI+ [15] sequence $X = \{X_1, \cdots, X_B\}$ over B bars, where X_b contains all REMI+ events within the b-th bar. A Transformer encoder P_ϵ processes each bar to produce latent embeddings $\mathbf{z}_1, \cdots, \mathbf{z}_B$.

Residual Quantization. Residual quantization blocks $\mathcal{RQ}$ discretize each $\mathbf{z}_b$ into embeddings $\mathbf{r}$ and corresponding codes c (indices) from codebooks $\mathcal{C}^1, \cdots, \mathcal{C}^D$:

$$(c_b^1, \mathbf{r}_b^1), \cdots, (c_b^D, \mathbf{r}_b^D) = \mathcal{RQ}(\mathbf{z}_b; \mathcal{C}^1, \cdots, \mathcal{C}^D) \quad (1)$$

where D is the number of codebooks or quantization depth, $(c_b^d, \mathbf{r}_b^d)$ is the retrieved code (index) and corresponding embedding in the d-th codebook $\mathcal{C}^d$ for $\mathbf{z}_b$. Each codebook $\mathcal{C}^d$ contains K index-embedding pairs $\{(k^d, \mathbf{e}_k^d)\}_{k=1}^K$, where k^d is the index and $\mathbf{e}_k^d$ is its corresponding embedding in $\mathcal{C}^d$, so $\mathbf{r}_b^d \in \{\mathbf{e}_k^d\}_{k=1}^K$, $c_b^d \in \{k^d\}_{k=1}^K$.

The first code c_b^1 of $\mathbf{z}_b$ is the index in the codebook $\mathcal{C}^1$ whose embedding $\mathbf{r}_b^1$ is nearest to $\mathbf{z}_b$. Then $\mathcal{RQ}$ recursively computes codes $c_b^2 \cdots c_b^D$ so that their embeddings are nearest to the residuals:

$$c_b^1 = \text{argmin}_k \, ||\mathbf{z}_b - \mathbf{e}_k^1||, \quad (2)$$

$$c_b^d = \text{argmin}_k \, ||\mathbf{z}_b - \mathbf{e}_k^d - \Sigma_{i=1}^{d-1} \mathbf{r}_b^i||, 2 \le d \le D. \quad (3)$$

To capture different granularities of music contents, the codes and embeddings in D codebooks are not shared.

Decoder. After obtaining all aggregated embeddings $\{\mathbf{r}_b = \Sigma_{d=1}^D \mathbf{r}_b^d | b = 1, ..., B\}$ from the $\mathcal{RQ}$ module, a Transformer decoder P_δ decodes these embeddings to predict the music sequence X in an auto-regressive mode. The reconstruction objective is:

$$\mathcal{L}_{recon} = -\sum_{t=1}^{T} \log P_\delta(x_{t+1} | x_{\le t}; \mathbf{r}_{\le b}), b = \text{bar}(t) \quad (4)$$

where x_t denotes the t-th event in X, as $X = \{X_1, \cdots, X_B\} = \{x_1, \cdots, x_T\}$, and $\text{bar}(t)$ is the bar index where the t-th event lies.

Codebook Utility. To better utilize the codebook during training, we adopt SimVQ [37] and rotation trick [38] to improve codebook utility and reconstruction quality, with the commitment objective as

$$\mathcal{L}_{commit} = \sum_{d=1}^{D} \sum_{b=1}^{B} \left|\left| \mathbf{z}_b - \text{sg}\left[\sum_{d'=1}^{d} \mathbf{r}_b^{d'} W^d \right] \right|\right|_2^2 \quad (5)$$

where sg is the stop-gradient operator, W^d is the linear transformation per codebook from SimVQ. The final training objective is the combination of two objectives: $\mathcal{L} = \mathcal{L}_{recon} + \mathcal{L}_{commit}$. Codebooks are updated by the exponential moving average [1].

3.2. Music Generation

As shown in Fig. 1 (middle), with the trained tokenization model, another Transformer decoder P_γ is connected before the MuseTok decoder P_δ for two-stage music generation. Using code sequence $c_1^1, \cdots, c_B^D$ converted from REMI+ sequence by the frozen P_ϵ and $\mathcal{RQ}$, the generator P_γ is trained to perform next-token prediction on codes with cross-entropy loss. During inference, P_γ first generates MuseTok codes for high-level musical structure, then decoded by the frozen P_δ into REMI+ events for fine-grained music details.

3.3. Music Understanding

We employ three classification tasks to assess semantic understanding of MuseTok at the note, bar and song levels in Fig. 1 (right).

Melody Extraction. The symbolic-domain melody extraction aims to identify melody notes from single-track polyphonic music [11]. A classifier is trained to assign each pitch event x_t of X to one of three classes, vocal melody, instrumental melody or accompaniment, using the code embedding $\mathbf{r}_b$ ($b = \text{bar}(t)$) as a condition to provide note-level semantic context, as in the MuseTok decoding process.

Chord Recognition. We introduce a symbolic-domain chord recognition task that extracts chord progressions from single-track polyphonic music to evaluate the harmony information captured by MuseTok. Given the code embedding $\mathbf{r}_b$ for a bar, a classifier predicts a chord label for each beat from a set of predefined categories.

Emotion Recognition. Emotion recognition classifies a song into one of four categories defined by high/low positiveness and high-/low activation [39], evaluating the song-level semantic capability of MuseTok. For a given song, a classifier is trained to take the code embeddings $\mathbf{r}_1, \cdots, \mathbf{r}_B$ as input and predicts its emotion label.

4. EXPERIMENTS

4.1. Datasets and Pre-processing

We collect piano pieces from the large-scale PDMX [16] and six small datasets covering diverse genres: POP909 [18], EMOPIA [20], Pop1k7 [19], Hymnal [22], Multipianomide [21] and Ragtime [23]. Pieces with time signature changes are segmented into clips with a consistent time signature and at least 8 bars. Since most samples maintain constant tempo and velocity, these performance-related attributes are removed to focus on structural and harmonic aspects. To improve structural consistency across datasets, we align note onsets and durations to valid sheet music positions. After pre-processing and REMI+ [15] encoding, the resulting 195,187 sequences (83.7% monophonic, 13.1% chorale and 3.2% polyphonic) are randomly split into training, validation and test sets at an 8:1:1 ratio, yielding a vocabulary of 140 music events.

	PPL ↓			Acc (%) ↑			Util ↑
	mono.	chora.	poly.	mono.	chora.	poly.	
VAE	1.123±0.007	1.086±0.007	**1.159**±0.021	96.70	83.59	82.21	-
MuseTok-Small	1.093±0.005	1.169±0.021	1.498±0.053	97.76	80.85	62.92	**99.58**%
– w/o. SimVQ+Rota.	1.151±0.008	1.268±0.027	1.667±0.059	80.55	64.04	54.59	87.77%
– w/o. SimVQ	1.129±0.006	1.242±0.025	1.620±0.057	85.87	66.19	56.15	90.50%
– w/. PDMX only	1.106±0.005	1.174±0.021	-	97.91	78.11	-	99.29%
MuseTok-Large	**1.091**±0.005	**1.081**±0.010	1.217±0.030	**99.58**	**93.71**	**82.68**	98.96%

Table 1. Reconstruction performance across different texture groups and codebook utility for ablated versions of MuseTok.

4.2. Music Tokenization

Model Settings and Ablations. The tokenization model uses 12-layer, 8-head Transformers with 512 hidden dim. for both encoder and decoder. Training is performed on 16-bar sequences, augmented by random transposition within ± 6 semitones, with chorale and polyphonic pieces unsampled to balance texture groups. Using a 128-dim VAE as the upper bound, we evaluate several residual quantization variants, including ablations on quantization techniques (Rotation trick, SimVQ), dataset usage (PDMX only) and model size: MuseTok-Small (quantization depth D=8, codebook size K=1024, 128-dim) and MuseTok-Large (D=16, K=2048). All models are trained with Adam optimizer (learning rate $1e{-}4$) with 200-step warm-up, converging in 45k steps on a single RTX A6000.

Evaluation Metrics. We evaluate reconstruction quality and codebook utility using three metrics. Perplexity (PPL) measures how well a model predicts a sequence of music events, defined as inversely proportional to the log-probability of the test split. Accuracy (Acc) is computed as $1 - \mathcal{D}(X, X')/|X|$, where $\mathcal{D}(X, X')$ refers to the edit distance between the reconstructed sequence X' and the original X. Both metrics are evaluated on three texture groups: monophonic, chorale, and polyphonic, reflecting increasing musical complexity. Codebook utility (Util) measures the fraction of codes used at least once when encoding the test set, averaged across quantization layers.

Results. From Table 1, most models exhibit highest perplexity and lowest reconstruction accuracy on polyphonic pieces, due to their complex textures and chord progressions. Incorporating diverse datasets during training improves reconstruction on PDMX (mostly monophonic and chorale pieces) compared to training on PDMX alone, validating the effectiveness of balanced sampling and preprocessing in aligning dataset distributions. Among MuseTok-Small variants, combining SimVQ with rotation tricks yields the highest codebook utility and best reconstruction quality. MuseTok-Large further approaches or surpasses the VAE upper bound, particularly improving on polyphonic pieces over MuseTok-Small. Based on these results, we select MuseTok-Large as the tokenization model for generation and understanding tasks.

4.3. Music Generation

Model Settings. The first-stage generator is a 12-layer, 16-head Transformer with 1024 hidden dim., totaling 152M parameters, trained with sequence length 256 using the same hyperparameters as tokenization, converging in $\sim$200k steps over 4 days. The second stage adopts the trained tokenization decoder. Datasets are augmented by offline key transposition (± 6). Unbalanced pieces are resampled during training. During inference, nucleus sampling [40] is applied (τ=1.1, p=0.9), followed by top-k downsampling (k=30).

Baselines. We compare two baselines of standard symbolic music encoding methods, a Transformer decoder trained on REMI+ sequence with the same datasets and model size as ours (REMI) [31], and Anticipatory Music Transformer (AMT) [41] using provided `music-small-100k` checkpoint, on a music continuation task, which allows reliable assessments through relative comparison. VAE-based models like MuseMorphose [13] and FIGARO [15] are not compared as they require reference music for generation.

Evaluation Metrics. We evaluate continuation results using two objective metrics and a subjective online listening test. To quantitatively evaluate the similarity to the primers, we adopt bar-wise chroma similarity $\mathrm{sim}_{\mathrm{chr}}$ and grooving similarity $\mathrm{sim}_{\mathrm{grv}}$ [13], measuring tonal closeness via cosine similarity of chroma vectors [42] and rhythmic resemblance via grooving vectors [43]. Sequence-level similarity is computed by averaging the highest similarity scores between each generated bar and the primer bars.

In the listening test, based on 4-bar primers, participants rate generated continuations on a 5-point Likert scale for four aspects: Pitch (Pit.), Structure (Str.), Harmony (Har.), and Development (Dev.). We collected 24 responses from participants spanning a wide spectrum of musical expertise, each evaluating 8 groups of random samples, yielding 192 ratings per model per aspect.

Results. As shown in Table 2, MuseTok outperforms both baselines on objective metrics, demonstrating effective harmonic and rhythmic continuation. On subjective metrics, it lags behind REMI and AMT on Pitch, producing more out-of-key notes that also affect Harmony perception. However, it performs comparably on Structure and Development, matching REMI and surpassing AMT, highlighting its potential for developing musical ideas through code generation.

Beyond these metrics, our two-stage generation model is more robust with long-context primers. Encoding 16-bar music with REMI+ produces $\sim$ 800 events, whereas our code sequence remains fixed at 256 codes when D=16, supporting long-term generation. However, this fixed depth can introduce noise for simpler primers. From Table 1, monophonic pieces reconstruct well with only 8 codes, with extra codes may act as a bias during generation. These results highlight the need for adaptive quantization and generation strategies for varying musical complexity.

4.4. Music Understanding

Model Settings and Baselines. The melody extraction task is evaluated on POP909 [18] using a 3-layer, 4-head Transformer with 128 hidden dim. as classifier, compared with Bi-LSTM (RNN) [44] and MIDI-BERT [11] trained on REMI [31] sequences. The chord recognition task is evaluated on POP909 with 133 labels (11 qualities $\times$ 12 roots + "no chord"), using a 2-layer MLP with 256 hidden dim. as classifier. An RNN on REMI+ with bar-level prediction is compared. The emotion recognition task is conducted on

	Objective		Subjective			
	sim_{chr}	sim_{grv}	Pit.	Str.	Har.	Dev.
REMI [31]	94.61	87.41	**4.099**	**3.927**	**3.927**	**3.646**
AMT [41]	94.72	84.08	3.839	3.328	3.516	3.156
MuseTok	**95.19**	**88.77**	3.698	3.839	3.604	3.635

Table 2. Objective and subjective evaluations on music continuation.

	Melody	Chord	Emotion
RNN [44]	89.98	38.03	53.46
MIDI-BERT [11]	**90.97**	-	67.74
MusicBERT [12]	-	-	77.78
MuseTok	81.92	**49.87**	**78.95**

Table 3. Classification accuracies on three understanding tasks.

EMOPIA [20] with a 2-layer MLP with 256 hidden dim., compared against RNN, MIDI-BERT [11] and MusicBERT [12]. For this task, MuseTok is retrained with velocity included in the REMI+ encoding.

Results. Classification accuracies are reported in Table 3. Our model outperforms all baselines on emotion recognition, demonstrating the ability of MuseTok to capture song-level semantic information. The learned codes also excel at modeling harmony in the challenging 133-class chord recognition task. Lower performance on melody extraction, along with out-of-key pitch generation above, highlights the need for improved melody modeling of tokenization.

4.5. How MuseTok Learns Music?

To explore the musical concepts learned by MuseTok, we conduct two case studies on MuseTok-Small: code usage frequencies across music groups and code embedding similarities on synthetic datasets.

Code Usage Frequency. Fig. 2 (a) and (b) present the top-50 most frequent codes used from 1000 random samples of monophonic, chorale and polyphonic textures at the first (d=1) and last (d=8) codebooks. The frequent code sets are largely distinct across textures, indicating that MuseTok employs different codes and embeddings to represent different textures, treating them as distinct motif and structure development. This differentiation persists across all codebooks, as further illustrated on the demo website.

A similar analysis across six time signatures is shown in Fig. 2 (c) and (d). Unlike textures, MuseTok almost omits the time signature difference in the first codebook, but gradually diverges in deeper ones, suggesting that different codebooks are allocated to capture different aspects of musical knowledge, with the first one focusing on shared information beyond time signature.

Embedding Similarity. This study examines how code embeddings change across codebooks when applying pitch shifting (key transposition) on music samples. Fig. 3 illustrates the cosine similarity of code embeddings (y-axis) between original and transposed samples across all 8 codebooks, where transposed samples are generated by shifting all notes from -6 to 6 semitones (x-axis). Embeddings in the first codebook (d=1) maintain over 70% similarity across transpositions, while deeper codebooks gradually diverge. This indicates that (1) transposition-invariant attributes, such as rhythmic information (onset, duration) and relative melodic contour, are mainly processed in earlier codebooks, and (2) absolute pitch information is further processed in deeper codebooks. Although attributes are not fully disentangled, MuseTok demonstrates an unsupervised ability to separate musical concepts across codebooks via data-driven learning.

Another observation relates to musical intervals. Across all

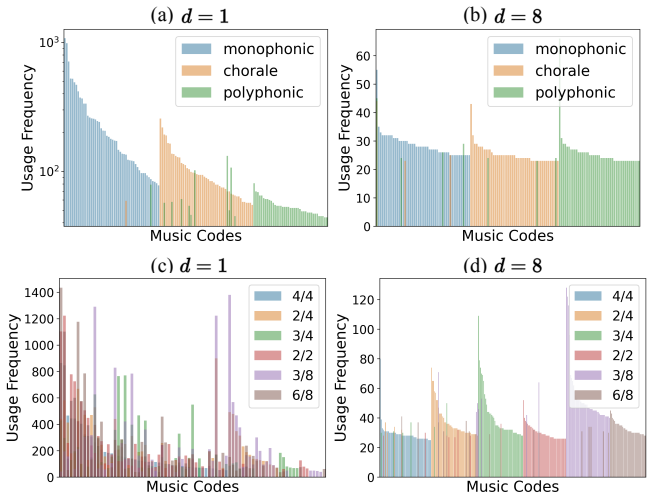

Fig. 2. Top-50 used codes across three texture groups, or across six time signatures in the first (d=1) and last (d=8) codebook.

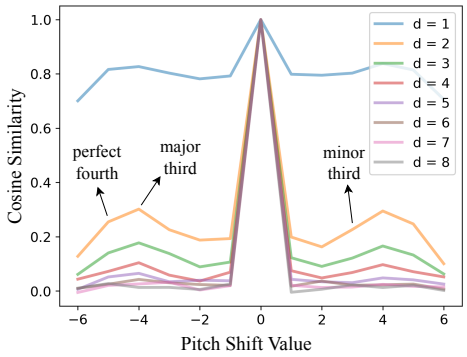

Fig. 3. The cosine similarity of code embeddings (y-axis) between original and transposed samples by pitch shifts (x-axis) across codebooks. Different colored lines denote different codebooks.

codebooks, the highest similarity scores (excluding zero-shift) occur at ± 4 (major third), followed by ± 5 (perfect fourth) and ± 3 (minor third), while ± 6 (augmented fourth) yield the lowest. This suggests that MuseTok captures interval concepts and their relative prevalence in music (major third, perfect forth, and minor third are commonly-used development), as well as interval symmetry, shown by consistent rankings for ascending and descending shifts.

These observations suggest that MuseTok effectively captures fundamental musical concepts, including rhythm, texture, and interval, even without explicit supervision from musical annotations.

5. CONCLUSION

In this paper, we introduce MuseTok, a discrete representation learning framework for symbolic music. RQ-VAE is applied on bar-wise music segments to learn music codes with high-fidelity reconstruction capability. We investigate the quality of learned codes through symbolic music generation and classification tasks in multiple perspectives, showing its effectiveness on both content generation and semantic understanding. Further qualitative analyses reveal the underlying musical concepts learned by MuseTok. In the future, we wish to focus on adaptive tokenization methods, leading to better music generation performance across all music groups.

6. REFERENCES

[1] Aäron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu, "Neural discrete representation learning," in *Proc. NeurIPS*, 2017.

[2] Ali Razavi, Aäron van den Oord, and Oriol Vinyals, "Generating diverse high-fidelity images with VQ-VAE-2," in *Proc. NeurIPS*, 2019.

[3] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han, "Autoregressive image generation using residual quantization," in *Proc. CVPR*, 2022.

[4] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi, "Soundstream: An end-to-end neural audio codec," *IEEE Trans. Audio, Speech, Lang. Process.*, 2022.

[5] Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Q. Tran, Jonah Samost, Maciej Kula, Ed H. Chi, and Mahesh Sathiamoorthy, "Recommender systems with generative retrieval," in *Proc. NeurIPS*, 2023.

[6] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever, "Jukebox: A generative model for music," *CoRR*, vol. abs/2005.00341, 2020.

[7] Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez, "Simple and controllable music generation," in *Proc. NeurIPS*, 2023.

[8] Rodrigo Castellon, Chris Donahue, and Percy Liang, "Codified audio language modeling learns useful representations for music information retrieval," in *Proc. ISMIR*, 2021.

[9] Chris Donahue, John Thickstun, and Percy Liang, "Melody transcription via generative pre-training," in *Proc. ISMIR*, 2022.

[10] Andrea Agostinelli, Timo I Denk, Zalán Borsos, Jesse Engel, Mauro Verzetti, Antoine Caillon, Qingqing Huang, Aren Jansen, Adam Roberts, Marco Tagliasacchi, et al., "Musiclm: Generating music from text," *CoRR*, vol. abs/2301.11325, 2023.

[11] Yi-Hui Chou, I-Chun Chen, Chin-Jui Chang, Joann Ching, and Yi-Hsuan Yang, "Midibert-piano: Large-scale pre-training for symbolic music understanding," *CoRR*, vol. abs/2107.05223, 2021.

[12] Mingliang Zeng, Xu Tan, Rui Wang, Zeqian Ju, Tao Qin, and Tie-Yan Liu, "Musicbert: Symbolic music understanding with large-scale pre-training," in *Findings of ACL*, 2021.

[13] Shih-Lun Wu and Yi-Hsuan Yang, "MuseMorphose: Full-song and fine-grained piano music style transfer with one Transformer VAE," *IEEE Trans. Audio, Speech, Lang. Process.*, 2023.

[14] Ziyu Wang, Dingsu Wang, Yixiao Zhang, and Gus Xia, "Learning interpretable representation for controllable polyphonic music generation," in *Proc. ISMIR*, 2020.

[15] Dimitri von Rütte, Luca Biggio, Yannic Kilcher, and Thomas Hofmann, "FIGARO: Generating symbolic music with fine-grained artistic control," in *Proc. ICLR*, 2023.

[16] Phillip Long, Zachary Novack, Taylor Berg-Kirkpatrick, and Julian J. McAuley, "PDMX: A large-scale public domain musicxml dataset for symbolic music processing," in *Proc. ICASSP*, 2025.

[17] Weihan Xu, Julian McAuley, Taylor Berg-Kirkpatrick, Shlomo Dubnov, and Hao-Wen Dong, "Generating symbolic music from natural language prompts using an llm-enhanced dataset," *CoRR*, vol. abs/2410.02084, 2024.

[18] Ziyu Wang, Ke Chen, Junyan Jiang, Yiyi Zhang, Maoran Xu, Shuqi Dai, and Gus Xia, "POP909: A pop-song dataset for music arrangement generation," in *Proc. ISMIR*, 2020.

[19] Wen-Yi Hsiao, Jen-Yu Liu, Yin-Cheng Yeh, and Yi-Hsuan Yang, "Compound Word Transformer: Learning to compose full-song music over dynamic directed hypergraphs," in *Proc. AAAI*, 2021.

[20] Hsiao-Tzu Hung, Joann Ching, Seungheon Doh, Nabin Kim, Juhan Nam, and Yi-Hsuan Yang, "EMOPIA: A multi-modal pop piano dataset for emotion recognition and emotion-based music generation," in *Proc. ISMIR*, 2021.

[21] "Classical piano midi page," `http://piano-midi.de/`.

[22] "Humnal.net," `https://www.hymnal.net/en/home`.

[23] "Rag's rag," `https://www.ragsrag.com/pr/pr.html`.

[24] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," in *Proc. ACL*, 2019.

[25] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever, "Learning transferable visual models from natural language supervision," in *Proc. ICML*, 2021.

[26] Yusong Wu, Ke Chen, Tianyu Zhang, Yuchen Hui, Taylor Berg-Kirkpatrick, and Shlomo Dubnov, "Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation," in *Proc. ICASSP*, 2023.

[27] Shangda Wu, Dingyao Yu, Xu Tan, and Maosong Sun, "Clamp: Contrastive language-music pre-training for cross-modal symbolic music information retrieval," in *Proc. ISMIR*, 2023.

[28] Ziyu Wang, Yiyi Zhang, Yixiao Zhang, Junyan Jiang, Ruihan Yang, Gus Xia, and Junbo Zhao, "PIANOTREE VAE: structured representation learning for polyphonic music," in *Proc. ISMIR*, 2020.

[29] 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, "Music Transformer: Generating music with long-term structure," in *Proc. ICLR*, 2019.

[30] Ian Simon and Sageev Oore, "Performance rnn: Generating music with expressive timing and dynamics," 2017.

[31] Yu-Siang Huang and Yi-Hsuan Yang, "Pop Music Transformer: Beat-based modeling and generation of expressive pop piano compositions," in *Proc. ACM Multimed.*, 2020.

[32] Hao-Wen Dong, Ke Chen, Shlomo Dubnov, Julian McAuley, and Taylor Berg-Kirkpatrick, "Multitrack Music Transformer," in *Proc. ICASSP*, 2023.

[33] Jeffrey Ens and Philippe Pasquier, "MMM : Exploring conditional multi-track music generation with the transformer," *CoRR*, vol. abs/2008.06048, 2020.

[34] Gaëtan Hadjeres and Léopold Crestel, "The piano inpainting application," *CoRR*, vol. abs/2107.05944, 2021.

[35] Julian Lenz and Anirudh Mani, "Pertok: Expressive encoding and modeling of symbolic musical ideas and variations," in *Proc. ISMIR*, 2024.

[36] Jingyue Huang, Ke Chen, and Yi-Hsuan Yang, "Emotion-driven piano music generation via two-stage disentanglement and functional representation," in *Proc. ISMIR*, 2024.

[37] Yongxin Zhu, Bocheng Li, Yifei Xin, and Linli Xu, "Addressing representation collapse in vector quantized models with one linear layer," *CoRR*, vol. abs/2411.02038, 2024.

[38] Christopher Fifty, Ronald G. Junkins, Dennis Duan, Aniketh Iger, Jerry W. Liu, Ehsan Amid, Sebastian Thrun, and Christopher Ré, "Restructuring vector quantization with the rotation trick," *CoRR*, vol. abs/2410.06424, 2024.

[39] James A. Russell, "A circumplex model of affect," *Journal of Personality and Social Psychology*, 1980.

[40] Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi, "The curious case of neural text degeneration," in *Proc. ICLR*, 2019.

[41] John Thickstun, David Leo Wright Hall, Chris Donahue, and Percy Liang, "Anticipatory music transformer," *IEEE Trans. Mach. Learn. Res.*, 2024.

[42] Takuya Fujishima, "Realtime chord recognition of musical sound: a system using common lisp music," in *Proc. ICMC*, 1999.

[43] Simon Dixon, Fabien Gouyon, and Gerhard Widmer, "Towards characterisation of music via rhythmic patterns," in *Proc. ISMIR*, 2004.

[44] Zhouhan Lin, Minwei Feng, Cícero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio, "A structured self-attentive sentence embedding," in *Proc. ICLR*, 2017.