Multimodal Music Tokenization with Residual Quantization for Generative Retrieval

Wo Jae Lee
Amazon Music

Rifat Joyee
Amazon Music

Emanuele Coviello
Amazon Music

Sudev Mukherjee
Amazon Music

Abstract

Recent advances in generative retrieval allow large language models (LLMs) to recommend items by generating their identifiers token by token, rather than using nearest-neighbor search over embeddings. This approach requires each item, such as a music track, to be represented by a compact and semantically meaningful token sequence that LLMs can generate. We propose a multimodal music tokenizer (3MToken) that transforms rich metadata from a music database, including audio, credits, semantic tags, song and artist descriptions, musical characteristics, release dates, and consumption patterns into discrete tokens using a Residual-Quantized Variational Autoencoder. Our method learns hierarchical representations, capturing coarse features in early quantization levels and refining them at later levels, preserving fine-grained information. We train and evaluate our model on a large-scale dataset of 1.6 million tracks, and it achieves +40.0%, +43.4%, and +15.8% improvements in Precision@k, Recall@k, and Hit@k, respectively, over the baselines.

1 Introduction

With the advancement of generative AI, music streaming services are beginning to offer new user experiences, such as prompt-based music discovery and recommendation [1, 9]. One of the primary challenges in this setting is enabling LLMs to interact effectively with the data in the music database to generate high-quality recommendations [3, 10]. Traditional recommender systems have relied on continuous embeddings and similarity-based retrieval, where a dense vector represents each item (e.g., a music track), and retrieval is performed using nearest-neighbor search in the embedding space [7, 22]. In contrast, recent research has proposed a new paradigm known as generative retrieval, which reformulates the recommendation task as a sequence generation problem [2, 13, 15, 23].

In generative retrieval, an autoregressive model directly generates the identifier of the next item token by token, similar to how a language model generates words in a sentence. This paradigm enables a recommender to be built on top of an LLM, which can seamlessly combine dialogue, reasoning, and recommendation in a single framework [6, 11]. One approach to generating music recommendations is to have an LLM directly produce human-readable track titles and artist names from natural-language prompts [1]. However, this free-text generation faces challenges such as entity resolution, title ambiguity, multilingual variations, and increased decoding latency in real-world applications. Due to these limitations, the preferred approach has shifted toward generative retrieval using learned discrete IDs, where the model directly predicts items [18, 24]. Rajput et al. [13] showed that representing items with Semantic IDs (sequences of discrete codes learned from item embeddings) allows a transformer-based model to generate item IDs as outputs, thereby unifying recommendation with natural language generation. Similarly, Doh et al. [2] extended this concept to music recommendation by expanding the LLM vocabulary with music tokens derived from

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: AI for Music.

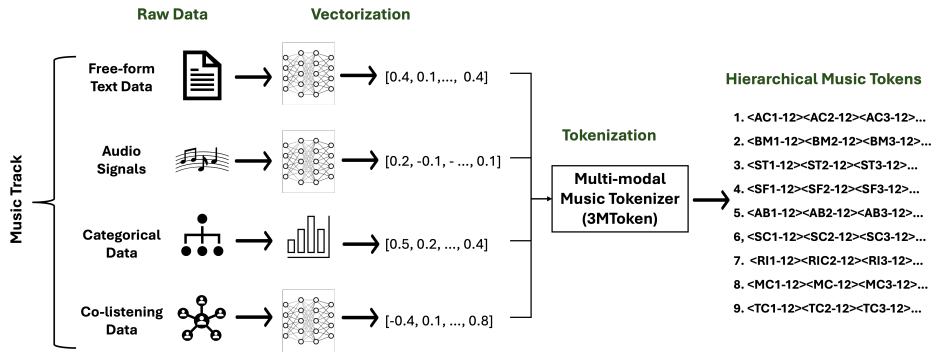

Figure 1: An overview of multimodal music tokenizer (3MToken). The proposed system ingests diverse modalities in heterogeneous formats and transforms them into embeddings and subsequently tokenizes them through a multi-stage quantization process.

multimodal data such as audio, lyrics, playlist, metadata, and tags. The proposed model outperforms unimodal approaches based solely on text or listening history in the music recommendation task.

In this context, the emergence of LLMs has catalyzed interest in developing discrete tokenization schemes for multimodal domains [25]. In the case of the music domain, each item (i.e., music track) is inherently multimodal including audio, lyrics, tags, and co-occurrence consumption. While recent studies have proposed using discrete tokenization schemes for music recommendation, there is limited research on systematically evaluating the quality of tokenizers across the diverse modalities that constitute music data, and prior works have primarily focused on end-task performance.

In this paper, we propose a multimodal music tokenizer (3MToken) based on a Residual-Quantized Variational Autoencoder (RQ-VAE) [20]. We refer to our method as 3MToken. Recent advances in neural discrete representation learning, particularly Vector Quantization Variational Autoencoder (VQ-VAE) and their variants, have shown promise in various domains including image synthesis and audio generation [16, 14]. RQ-VAE extends traditional vector quantization by employing multiple codebooks in a residual manner. Our approach is motivated by [13, 2] which applies residual quantization for creating hierarchical semantic ID and enriches LLM vocabularies with music tokens.

2 Method

Our goal is to (1) process raw music catalog data into multimodal embeddings (i.e., vectorization) and (2) transform each track's multimodal embedding into a compact sequence of discrete tokens (i.e., tokenization). We present an overview in Fig. 1 from raw inputs to vectorization and tokenization, as used in our model. Each modality is processed through a modality-specific encoder and tokenized through a multi-stage quantization process to generate discrete multimodal music tokens.

Music Data A wide range of track- and artist-level data is available in various formats. In this paper, we leverage rich metadata aggregated from multiple sources to build a multimodal music tokenizer. Typically, this metadata is heterogeneous in format - categorical (e.g., genre, mood, music key, playlist ID, singing language), scalar (e.g., tempo), and textual (e.g., song titles, artist names, credits, and biographies). To better structure these metadata, we categorize them into nine categories: (1) *Artist Roles & Collaborations* (AC) – band members, featured artists, vocalists, and instrumentalists, (2) *Basic Metadata* (BM) – track title and artist name, (3) *Semantic Tags* (ST) – genre, mood, and activity, (4) *Sonic Characteristics* (SC) – audio embeddings derived from a pre-trained audio encoder that capture timbral, rhythmic, and other acoustic properties, (5) *Musical Characteristics* (MC) – tempo and musical key, (6) *Release Information* (RI) – release date of a track, (7) *Song Facts* (SF) – recording details such as song history, artist backstory, and production details, (8) *Artist Biography* (AB) – artist attributes such as gender, birth year, and origin, (9) *Track Consumption* (TC) – data capturing co-listening patterns. We divide the music metadata into nine categories based on common user demands in music streaming services.

Vectorization After organizing the music data into the nine categories, the next step is to map each track-level data source into a multimodal embedding space. Since the music data has heterogeneous formats (ranging from raw audio signals to textual metadata and numerical fields), we apply modality-specific vectorization techniques that ensure all information is represented as dense vectors. To do this, we extract modality-specific embeddings with (1) a pre-trained text encoder [17] for textual data, (2) a CLAP-like audio encoder for audio signal snippets [19], (3) one-hot embedding for categorical data and binning for scalar data, and (4) a session-based collaborative filtering embedding model for consumption data. Full details of these vectorization steps are presented in Appendix A.

Music Tokenizer with RQ-VAE For each modality, we train a RQ-VAE model that consists of an encoder, a multi-level vector quantizer, and a decoder. Given an input embedding $\mathbf{x} \in \mathbb{R}^d$, the encoder network $f_\theta(\cdot)$ maps it to a latent representation $\mathbf{z}_e \in \mathbb{R}^{d_z}$ as follows $\mathbf{z}_e = f_\theta(\mathbf{x})$. Then, this latent space is quantized by multiple codebooks in series, which is the key to residual quantization. Instead of a single quantization step, RQ-VAE applies L sequential codebooks to iteratively refine the approximation. At the l-th level, for a given residual $\mathbf{r}_{l-1}$, the quantizer selects the closest codeword $\mathbf{e}_{k_l}$ from a learned codebook $\mathcal{E}_l = \{\mathbf{e}_1, \ldots, \mathbf{e}_{K_l}\}$ in Euclidean distance as follows: $k_l = \arg\min_k \|\mathbf{r}_{l-1} - \mathbf{e}_k\|_2^2$ where $k_l \in \{1, 2, \ldots, K_l\}$ denotes the index of the selected codeword at the l-th quantization level, K_l is the size of the codebook, $\mathbf{r}_0 = \mathbf{z}_e$ is the initial residual, and $\mathbf{r}_l = \mathbf{r}_{l-1} - \mathbf{e}_{k_l}$ is the updated residual after quantization. After multi-stage residual quantization, the final quantized representation is represented by the sum of all selected codewords: $\hat{\mathbf{z}}_q = \sum_{l=1}^{L} \mathbf{e}_{k_l}$. Then, the decoder $g_\phi(\cdot)$ reconstructs the original embedding from the quantized representation as follows: $\hat{\mathbf{x}} = g_\phi(\hat{\mathbf{z}}_q)$.

The total RQ-VAE loss can be defined by combining a reconstruction loss, a codebook loss, and a commitment loss to encourage the encoder outputs to stay close to the selected codewords:

$$\mathcal{L} = \underbrace{\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2}_{\text{reconstruction loss}} + \sum_{l=1}^{L} \Big(\underbrace{\|\text{sg}[\mathbf{r}_{l-1}] - \mathbf{e}_{k_l}\|_2^2}_{\text{codebook loss}} + \underbrace{\beta \|\mathbf{r}_{l-1} - \text{sg}[\mathbf{e}_{k_l}]\|_2^2}_{\text{commitment loss}} \Big)$$

where $\text{sg}[\cdot]$ denotes the stop-gradient operator, β is the commitment weight [16]. During training, the codebook loss updates codewords by moving them toward the encoder outputs, while the commitment loss encourages the encoder outputs to remain close to selected codewords. $\text{sg}[\cdot]$ is used to control gradient flow. The discrete token sequence for each modality is obtained by concatenating the indices from all codebooks such as $(k_{mod,1}, k_{mod,2}, ..., k_{mod,L})$.

Token Formation Once we have trained a RQ-VAE model for each modality, we can encode multimodal features of a track into a set of tokens as follows: we (1) pass each modality-specific embedding through the corresponding encoder and quantize it through L codebooks to get indices and (2) map each index to a token string of the form "<modality<level>-<index>". For example, for *Sonic Characteristics* (SC) modality, if the three quantization levels produce indices 5, 2, and 17, the resulting tokens would be <SC1-5><SC2-2><SC3-17>.

3 Experiments

Dataset and Setup We conduct experiments on a proprietary large-scale music catalog containing track- and artist-level metadata, covering 1.6M tracks spanning diverse genres, eras, and popularity levels. Music data are sourced through a combination of public databases (e.g., Wikipedia), automated labeling system (e.g., music tagging model), and expert review (e.g., human annotation). After preparing raw data, we categorize them into the nine categories as explained in Section 2. Then, we extract modality-specific embeddings with pre-trained text encoder [17], CLAP-like audio encoder [19], one-hot embedding for categorical data, binning for scalar data, and session-based collaborative filtering embedding model. We split the dataset into 80% for training and 20% for testing.

RQ-VAE Configuration We set the latent dimension to d_z = 32 and use L = 3 codebooks as done in Rajput et al. [13]. For each modality, we use $(K_1, K_2, K_3) = (32, 64, 128)$ resulting in 224 tokens per modality where the first level captures broad categories and the subsequent levels capture increasingly subtle differences in a larger codebook. Each modality-specific RQ-VAE was trained for 150 epochs using AdamW optimizer with a learning rate of 1e-4 and a batch size of 512. Further details of the model architecture are provided in Appendix B.

Table 1: Performance comparison on content-based retrieval. The last three rows present % improvement with 3MToken relative to the best baseline (underlines denote the second-best metric).

		Curated Playlist				Co-occurrence			
Method	**Metric**	**k=5**	**k=10**	**k=20**	**k=50**	**k=5**	**k=10**	**k=20**	**k=50**
K-means	Precision@k	<u>0.0678</u>	<u>0.0563</u>	<u>0.0431</u>	<u>0.0266</u>	<u>0.0596</u>	<u>0.0490</u>	<u>0.0366</u>	<u>0.0241</u>
	Recall@k	<u>0.0531</u>	<u>0.0848</u>	<u>0.1269</u>	<u>0.1903</u>	<u>0.0480</u>	<u>0.0751</u>	<u>0.1076</u>	<u>0.1604</u>
	Hit@k	<u>0.2246</u>	<u>0.2931</u>	<u>0.3856</u>	<u>0.4949</u>	<u>0.2282</u>	<u>0.3092</u>	<u>0.3868</u>	<u>0.4954</u>
VQ-VAE	Precision@k	0.0530	0.0442	0.0336	0.0213	0.0447	0.0366	0.0274	0.0180
	Recall@k	0.0417	0.0664	0.0989	0.1523	0.0368	0.0573	0.0808	0.1218
	Hit@k	0.1838	0.2577	0.3319	0.4428	0.1782	0.2474	0.3220	0.4298
3MToken (ours)	Precision@k	**0.1059**	**0.0871**	**0.0611**	**0.0340**	**0.0863**	**0.0695**	**0.0497**	**0.0282**
	Recall@k	**0.0834**	**0.1299**	**0.1770**	**0.2412**	**0.0747**	**0.1137**	**0.1523**	**0.1956**
	Hit@k	**0.2842**	**0.3523**	**0.4176**	**0.5133**	**0.3004**	**0.3746**	**0.4330**	**0.5096**
% Improvement	Precision@k	+56.19%	+54.69%	+41.76%	+27.82%	+44.80%	+41.84%	+35.79%	+17.01%
	Recall@k	+57.06%	+53.24%	+39.48%	+26.75%	+55.63%	+51.40%	+41.54%	+21.95%
	Hit@k	+26.53%	+20.20%	+8.30%	+3.72%	+31.65%	+21.15%	+11.94%	+2.87%

Baselines and Evaluation Metrics K-means clustering is used to quantize embeddings into music tokens by directly partitioning the embedding space without learned representations [2]. This method serves as a fundamental comparison point to evaluate whether the added complexity of neural quantization provides meaningful improvements over the traditional clustering technique. VQ-VAE serves as an ablation of our model, using a single codebook for quantization. We train one VQ-VAE per modality using the same encoder-decoder architecture as our RQ-VAE model, but replace the multi-level residual quantization with a single vector quantizer. We set the codebook size of VQ-VAE and the number of K-means clusters to 1024 per modality, following [2] which is 4.6 times larger than that of RQ-VAE. For evaluation, we measure the performance on reconstruction and content-based retrieval. Reconstruction is assessed using Mean Squared Error (MSE) and cosine similarity, while retrieval performance is measured on curated playlist and co-occurrence data, reporting Precision@k, Recall@k, and Hit@k. Further details of the evaluation are provided in Appendix C.

3.1 Results

Model Training and Qualitative Analysis We analyze the training losses and qualitative properties of the learned token space in Appendix D. From the analysis, the training losses vary by modality, reflecting differing complexity in modality signals. Also, we observe that early-level codes capture coarse semantics as evidenced by ground-truth labels. Lastly, token usage analysis shows that the encoder consistently maps test set tracks to codebook regions activated during training.

Reconstruction and Content Retrieval Tasks Our analysis shows that 3MToken achieves the lowest average reconstruction error and the highest cosine similarity (0.01182/0.9298) compared to K-means (0.01644/0.8861) and VQ-VAE (0.01911/0.8699), representing a 28.10%/4.9% and 38.15%/6.89% improvement, respectively. These results suggest that the residual quantization approach effectively captures fine-grained details that are lost in the vector quantization method. For the retrieval task, we present the results in Table 1 where 3MToken consistently outperforms the baselines across all metrics at all k values followed by K-means. K-means outperforms VQ-VAE, which indicates that the direct clustering better preserves the original embedding structure. In Appendix E, we provide additional analysis including the impact of d_z, unimodal vs. multimodal tokens, and ablation study. From the results, 3MToken consistently outperforms unimodal models on the retrieval tasks, demonstrating the effectiveness of the multimodal approach. In the ablation study, the full tokens achieves the strongest results with TC showing as the most critical modality.

4 Conclusion

We develop a framework for tokenizing multimodal music data into a set of discrete tokens. Our approach (3MToken) effectively compresses multimodal information into a sequence of hierarchical music tokens and demonstrates superior performance on the evaluation tasks while using a lower number of tokens. In future work, we will evaluate the proposed multimodal music tokens to enable an LLM to perform generative music retrieval.

References

[1] Mathieu Delcluze, Antoine Khoury, Clémence Vast, Valerio Arnaudo, Léa Briand, Walid Bendada, and Thomas Bouabça. Text2playlist: Generating personalized playlists from text on deezer. In *European Conference on Information Retrieval*, pages 164–170. Springer, 2025.

[2] Seungheon Doh, Keunwoo Choi, and Juhan Nam. Talkplay: Multimodal music recommendation with large language models. *arXiv preprint arXiv:2502.13713*, 2025.

[3] Daeyong Kwon, SeungHeon Doh, and Juhan Nam. Must-rag: Musical text question answering with retrieval augmented generation, 2025.

[4] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11523–11532, 2022.

[5] Greg Linden, Brent Smith, and Jeremy York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, 7(1):76–80, 2003.

[6] Enze Liu, Bowen Zheng, Cheng Ling, Lantao Hu, Han Li, and Wayne Xin Zhao. Generative recommender with end-to-end learnable item tokenization, 2025.

[7] Rishabh Mehrotra, James McInerney, Hugues Bouchard, Mounia Lalmas, and Yizhou Yue. Contextual and sequential user embeddings for large-scale music recommendation. In *Proceedings of the 14th ACM Conference on Recommender Systems (RecSys)*, 2020.

[8] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

[9] Enrico Palumbo, Gustavo Penha, Andreas Damianou, José Luis Redondo García, Timothy Christopher Heath, Alice Wang, Hugues Bouchard, and Mounia Lalmas. Text2tracks: Prompt-based music recommendation via generative retrieval. *arXiv preprint arXiv:2503.24193*, 2025.

[10] Gustavo Penha, Ali Vardasbi, Enrico Palumbo, Marco De Nadai, and Hugues Bouchard. Bridging search and recommendation in generative retrieval: Does one task help the other? In *Proceedings of the 18th ACM Conference on Recommender Systems*, pages 340–349, 2024.

[11] Haohao Qu, Wenqi Fan, Zihuai Zhao, and Qing Li. Tokenrec: Learning to tokenize id for llm-based generative recommendation, 2024.

[12] Milos Radovanovic, Alexandros Nanopoulos, and Mirjana Ivanovic. Hubs in space: Popular nearest neighbors in high-dimensional data. *Journal of Machine Learning Research*, 11(sept):2487–2531, 2010.

[13] Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Tran, Jonah Samost, et al. Recommender systems with generative retrieval. *Advances in Neural Information Processing Systems*, 36:10299–10315, 2023.

[14] Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32, 2019.

[15] Weiwei Sun, Lingyong Yan, Zheng Chen, Shuaiqiang Wang, Haichao Zhu, Pengjie Ren, Zhumin Chen, Dawei Yin, Maarten Rijke, and Zhaochun Ren. Learning to tokenize for generative retrieval. *Advances in Neural Information Processing Systems*, 36:46345–46361, 2023.

[16] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

[17] Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. Improving text embeddings with large language models. *arXiv preprint arXiv:2401.00368*, 2023.

[18] Wenjie Wang, Honghui Bao, Xinyu Lin, Jizhi Zhang, Yongqi Li, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. Learnable item tokenization for generative recommendation. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 2400–2409, 2024.

[19] 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 *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[20] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. Soundstream: An end-to-end neural audio codec, 2021.

[21] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. Soundstream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:495–507, 2021.

[22] Xiangyu Zhao, Maolin Wang, Xinjian Zhao, Jiansheng Li, Shucheng Zhou, Dawei Yin, Qing Li, Jiliang Tang, and Ruocheng Guo. Embedding in recommender systems: A survey. *arXiv preprint arXiv:2310.18608*, 2023.

[23] Bowen Zheng, Yupeng Hou, Hongyu Lu, Yu Chen, Wayne Xin Zhao, Ming Chen, and Ji-Rong Wen. Adapting large language models by integrating collaborative semantics for recommendation. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pages 1435–1448. IEEE, 2024.

[24] Bowen Zheng, Hongyu Lu, Yu Chen, Wayne Xin Zhao, and Ji-Rong Wen. Universal item tokenization for transferable generative recommendation. *arXiv preprint arXiv:2504.04405*, 2025.

[25] Jing Zhu, Mingxuan Ju, Yozen Liu, Danai Koutra, Neil Shah, and Tong Zhao. Beyond unimodal boundaries: Generative recommendation with multimodal semantics, 2025.

A Embedding Generation for Multimodal Music Data

A.1 Textual Description

For textual data such as *Artist Roles & Collaborations*, *Basic Metadata*, *Semantic Tags*, *Song Facts*, *Artist Biography*, we apply a state-of-the-art pre-trained text encoder [17] to generate a text embedding. This encoder maps free-form text into a semantic embedding space, allowing similar textual descriptions to yield similar embeddings. The model we used generates a vector with a length of 4096 for a given textual data.

A.2 Audio Signals

For audio-based signals such as *Sonic Characteristics*, which capture the acoustic properties of a track, we leverage a CLAP (Contrastive Language–Audio Pretraining) like audio encoder model [19]. The CLAP model aligns raw audio signals and textual descriptions into a shared embedding space using contrastive learning. In our case, we trained a CLAP-like model with a proprietary dataset consisting of 1M audio-text pairs where the audio encoder takes the raw audio waveform of a track as input, while the text encoder takes the track's description including metadata-derived textual context. This joint training encourages audio embeddings to remain semantically aligned with text-based descriptions, improving cross-modal consistency. After training, we only use the audio-encoder tower to generate a vector with a length of 128 for a given sound recording.

A.3 Categorical Metadata

We have two types of categorical metadata: *Release Information* and *Musical Characteristics*. For categorical data, we apply a simple data binning approach.

First, to capture both fine-grained and coarse-grained time information from the release information, we transform the release date into a compact temporal embedding. To do this, first, we encode normalized year where the release year is normalized to a $[0, 1]$ range as $year_{norm} = \frac{\text{year}-1910}{2025-1910}$, where 1910 is the earliest possible release year and 2025 is the upper bound. And then, we encode cyclical month using sine/cosine functions to capture the periodic nature of months by encoding the month m as $\sin_m = \sin\big(2\pi\frac{m-1}{12}\big)$ and $\cos_m = \cos\big(2\pi\frac{m-1}{12}\big)$, while the day d is encoded as $\sin_d = \sin\big(2\pi\frac{d-1}{31}\big)$ and $\cos_d = \cos\big(2\pi\frac{d-1}{31}\big)$. In addition, we assign each release year to one of 11 decade bins (pre-1930, 1930s, …, 2020s), represented by a one-hot vector. Also, we add two binary flags indicating data quality - an imputation flag (1 if the date is missing or invalid) and an availability flag (1 if a valid release date is present). By concatenating all components, the final release-date embedding is represented by an 18-dimensional vector that preserves both exact timing and coarse historical context.

For *Musical Characteristics*, we adopt a one-hot binning approach to discretize continuous tempo values into 220 bins, covering the range from 30 BPM to 249 BPM. Valid tempo values within this range are mapped to their corresponding bins, while missing or out-of-range values are assigned to a dedicated imputation bin (i.e., last index). For the musical key, which is categorical (e.g., C major, B major, A minor), we create a class-to-index mapping for all observed keys and encode each key as a 25-dimensional one-hot vector, reserving the last index for missing or unknown keys. In this work, the embeddings from tempo and musical key are concatenated to form the representation for *Musical Characteristics*.

A.4 Consumption Data

For *Track Consumption*, the consumption patterns which are derived from co-occurrence in a listening session are used. We built the consumption song embeddings as follows. First, from historical playback data, we generate track pairs that frequently appear in the same session, recording both pairwise co-occurrence counts and marginal frequencies. To improve data quality, we filter out pairs that do not meet a minimum threshold of sessions and distinct listeners. We then compute similarity scores for each remaining pair using a statistical significance test [5], retain the top-k most similar tracks for each seed track, and apply hubness [12] and popularity bias reduction. To learn track-level embeddings v_i, we define the affinity between two tracks as the dot product of their embeddings. We optimize a weighted cross-entropy loss over a softmax distribution of top-k similar tracks, using

negative sampling [8] as follows $\sum_{i,j} s_{i,j} \log \text{softmax}_j \exp v_i' v_j$ where $s_{i,j}$ is an increasing function of the similarity strength of the pair (i, j). Hyperparameters are tuned based on a track-ranking task on a held-out validation set of track pairs. This approach can be viewed as a session-based skip-gram embedding model (Track2Vec) where co-listening relationships are learned directly from session co-occurrence data.

After applying modality-specific encoders, the resulting dimensions of each modality are 4096, 4096, 4096, 128, 247, 18, 4096, 4096, 128 for *Artist Roles & Collaborations*, *Basic Metadata*, *Semantic Tags*, *Sonic Characteristics*, *Musical Characteristics*, *Release Information*, *Song Facts*, *Artist Biography*, and *Track Consumption*, respectively.

B Model Architecture

We use $(K_1, K_2, K_3) = (32, 64, 128)$ for our default setting. These default sizes are chosen so that the finest level often has a larger codebook (e.g., 128) to capture subtle differences, whereas the first level has a small codebook to capture broad categories (e.g., 32). For encoder ($f_\theta(\cdot)$) and decoder ($g_\phi(\cdot)$), we adopt the following architecture: the encoder is a 4-layer feedforward neural network that compresses the input embedding through layers of size $512 \to 256 \to 128 \to d_z$, using ReLU activations and batch normalization at each layer and the decoder mirrors this structure symmetrically with layers of size $d_z \to 128 \to 256 \to 512$, reconstructing the original embedding from the latent representation.

C Evaluation Metrics

We report performance on reconstructions and content retrieval tasks to compare 3MToken with its variants and the baselines described in Section 3.

Reconstruction measures how well the compressed tokens preserve the original feature information. We compute Mean Squared Error (MSE) and cosine similarity between an original modality vector $\mathbf{x}$ and a RQ-VAE reconstructed vector $\hat{\mathbf{x}} = g_\phi(\hat{\mathbf{z}}_q)$ on the test set. Lower MSE and higher cosine similarity indicate better preservation of the information. For K-means baseline, we use a corresponding cluster centroid as a reconstructed vector.

For retrieval tasks, we simulate a content-based retrieval scenario in which, given a query track, the goal is to retrieve top-k most similar tracks based on their item representations (i.e., music tokens) and evaluate whether the retrieved tracks are relevant to the query. To do this, we construct two proxy tasks using the curated playlist (including 15,000 playlists) and the co-occurrence dataset (including 30,000 track co-occurrence pairs derived from listening sessions). For the curated playlist, we select a track from a playlist as the query and retrieve the top-k tracks, assuming that tracks within the same playlist tend to share semantic similarity. Similarly, for the co-occurrence, we treat tracks that co-occur within the same listening session as semantically related and evaluate whether a model retrieves top-k tracks from the same session. Note that we only use tracks from the test set in both curated playlist and co-occurrence data.

To compare token sequences between tracks, we assign equal weights (i.e., 1) to each modality. For RQ-VAE, we apply hierarchical matching, where level-2 matching is conditioned on level-1 matches, and level-3 matching is further conditioned on successful matches at levels 1 and 2. We then find top-k tracks based on a token matching score (greater token overlap yields a higher score). For the retrieval tasks, we report standard recommendation metrics including Precision@k, Recall@k, and Hit@k.

D Model Training and Qualitative Analysis

When computing the loss, β=0.25 is used. Also, to prevent the situation where most of the input gets mapped to only a few codebook vectors (i.e., codebook collapse problem), as proposed in [21], we run K-means on the first training batch and initialize codebooks with the learned centroids instead of a random initialization for the codebook vectors. To train our model, we used two NVIDIA Tesla V100 GPUs.

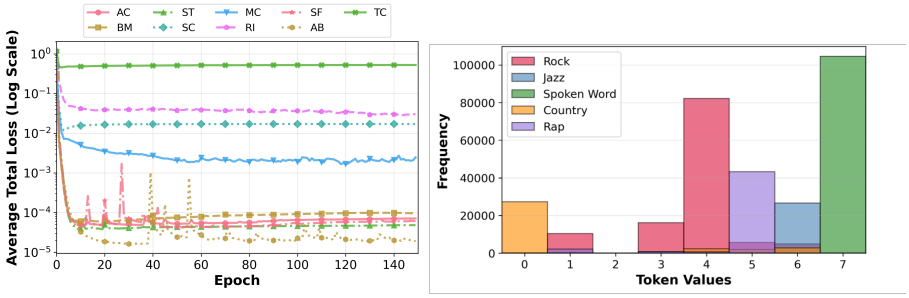

Figure 2: Average total loss for RQ-VAE training (left) and qualitative study of music tokens generated by RQ-VAE (right).

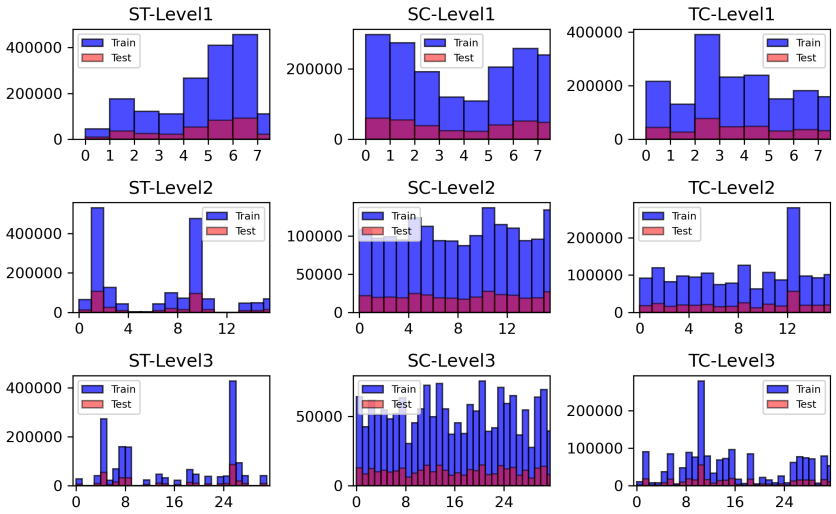

Figure 3: Token distribution across quantization levels for three selected modalities. The histograms show token usage frequency for train and test sets.

Figure 2 (left) shows the training loss of RQ-VAE across all nine modalities over 150 epochs. The losses converge after nearly 100 epochs, and distinct patterns are observed which may reflect the inherent complexity and characteristics of different musical modalities. Several modalities including AC, BM, SF, ST, and AB present relatively low loss values (between 10^{-4} and 10^{-5} range), suggesting these modalities contain well-structured, learnable patterns that are effectively captured by the quantization approach. TC and RI modalities maintain higher loss values throughout training, suggesting quantization is more challenging to preserve information.

A primary property of the residual quantization is its ability to represent tracks hierarchically. The first-level code $c_{mod,1}$ captures coarse semantic aspects of a modality such as a main musical genre (e.g., Rock) in Semantic Tag (ST) modality while the second ($c_{mod,2}$) and third levels ($c_{mod,3}$) refine the representation to capture sub-genres (e.g., Hard Rock, Punk Rock, and Alternative Rock). As a result, tracks sharing the same first-level code are likely to be similar in a broad sense for that modality, even if their full token sequences differ. For visualization purposes, we set the codebook sizes to $(K_1, K_2, K_3) = (8, 16, 32)$ and show the distribution of ground-truth music categories over the first-level codebook in ST modality in Fig. 2 (right). It clearly demonstrates that the first-level codes capture high-level categorical information.

Figure 3 presents token distributions across the three quantization levels for three selected modalities—ST (text-driven), SC (audio-driven), and TC (co-listening-driven)—for both training and test sets. As shown in the figure, some token indices are unused during training (e.g., in ST-Level3). Importantly, none of the test set tracks are mapped to these unused tokens, indicating that the encoder consistently generates embeddings within the codebook regions activated during training. This suggests that

Table 2: Reconstruction error (↓) and cosine similarity (↑)

Modality	K-means	VQ-VAE	3MToken (ours)
AC	2.700×10^{-5} / 0.9420	3.400×10^{-5} / 0.9272	$\mathbf{2.500 \times 10^{-5}}$ / **0.9478**
BM	4.900×10^{-5} / 0.8940	5.200×10^{-5} / 0.8865	$\mathbf{4.500 \times 10^{-5}}$ / **0.9028**
ST	1.500×10^{-5} / 0.9697	3.100×10^{-5} / 0.9351	$\mathbf{1.300 \times 10^{-5}}$ / **0.9730**
SC	3.229×10^{-3} / 0.760	3.293×10^{-3} / 0.7530	$\mathbf{1.984 \times 10^{-3}}$ / **0.8614**
MC	2.328×10^{-3} / 0.8335	2.588×10^{-3} / 0.8153	$\mathbf{3.900 \times 10^{-5}}$ / **0.9979**
RI	1.639×10^{-2} / 0.9701	3.406×10^{-2} / 0.9375	$\mathbf{4.109 \times 10^{-3}}$ / **0.9982**
SF	3.100×10^{-5} / 0.9319	3.000×10^{-5} / **0.9383**	$\mathbf{2.800 \times 10^{-5}}$ / 0.9349
AB	$\mathbf{4.000 \times 10^{-6}}$ / **0.9927**	1.000×10^{-5} / 0.9793	4.000×10^{-6} / 0.9913
TC	1.259×10^{-1} / 0.6810	1.319×10^{-1} / 0.6610	$\mathbf{1.001 \times 10^{-1}}$ / **0.7578**
Avg.	1.644×10^{-2} / 0.8861	1.911×10^{-2} / 0.8699	$\mathbf{1.182 \times 10^{-2}}$ / **0.9298**

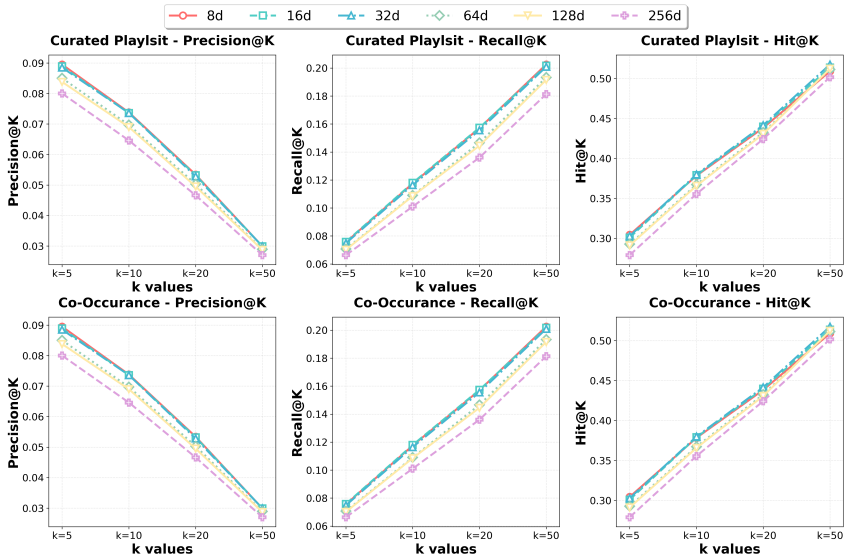

Figure 4: Performance measured in Precision@k, Recall@k, and Hit@k at different latent dimensions (d_z).

although the token space has high capacity, the encoder effectively learns to utilize a stable and meaningful subset of tokens, enhancing both the reliability and interpretability of the representations for downstream retrieval tasks.

E Further Analysis

E.1 Reconstruction Accuracy

In Table 2, we present a comparison of our proposed model against two baseline approaches across nine different modalities. Overall, our proposed model demonstrates superior performance with the lowest average reconstruction error (0.01182) and highest average cosine similarity (0.9298). Notably, 3MToken achieves exceptional results on MC and RI modalities with near-perfect cosine similarity scores of 0.9979 and 0.9982 respectively, representing significant improvements over both baseline methods. These results demonstrate that our hierarchical token extraction approach effectively captures the underlying structure across all modalities.

E.2 Impact of Latent Dimension in RQ-VAE

In RQ-VAE, the choice of latent dimension (d_z) plays a role in balancing reconstruction quality, computational efficiency, and latent space expressiveness [4]. To investigate the impact of d_z on the content-based retrieval tasks, we report Precision@k, Recall@k, and Hit@k at d_z = 8, 16, 32, 64, 128, and 256 in Fig. 4. Based on the evaluation, RQ-VAE with d_z = 32 is shown as the best performing

across most of the metrics on both Curated Playlist and Co-occurrence followed by d_z = 16. Also, as shown in the figure, the performance tends to decline as d_z increases. One possible explanation is that larger latent vectors allow the model to capture excessive fine-grained, reconstruction-specific details (e.g., noise) that are not semantically relevant, while quantization noise accumulates over more dimensions. This reduces the discriminative power of the learned tokens, meaning that even if reconstruction error improves with larger d_z, semantic retrieval performance can deteriorate.

E.3 Unimodal vs. Multimodal Approaches

In Table 3, we present Hit@k performance of unimodal and multimodal tokens on the content retrieval tasks discussed in Section 3.1, selecting the 7 best-performing modalities for brevity. As shown in the table, the multimodal token approach (i.e., 3MToken) consistently outperforms unimodal methods across all k values, demonstrating the effectiveness of the multimodal approach. Among unimodal baselines, TC achieves the best performance. The performance gap between multimodal and unimodal approaches is more significant at lower k values, indicating that multimodal approach is particularly effective at identifying highly relevant music tracks early in the ranking process. Overall, the results validate that integrating information across modalities yields significant performance gains over unimodal approaches.

Table 3: Comparison of multimodal and unimodal approaches on content-based retrieval tasks. The last row presents % improvement with 3MToken relative to the best unimodal baseline (underlines denote the second-best metric).

	Curated Playlist				**Co-occurrence**			
Method	**Hit@5**	**Hit@10**	**Hit@20**	**Hit@50**	**Hit@5**	**Hit@10**	**Hit@20**	**Hit@50**
	Multimodal Approach							
3MToken (ours)	**0.2842**	**0.3523**	**0.4176**	**0.5133**	**0.3004**	**0.3746**	**0.4330**	**0.5096**
	Unimodal Approaches							
TC	<u>0.0994</u>	<u>0.1511</u>	<u>0.2161</u>	<u>0.3070</u>	<u>0.1648</u>	<u>0.2394</u>	<u>0.3218</u>	<u>0.4262</u>
ST	0.0725	0.1120	0.1576	0.2325	0.0908	0.1324	0.1832	0.2690
SC	0.0548	0.1004	0.1538	0.2301	0.0780	0.1228	0.1788	0.2598
AC	0.0844	0.1076	0.1304	0.1715	0.1092	0.1376	0.1644	0.2030
SF	0.0848	0.1096	0.1321	0.1593	0.1394	0.1802	0.2056	0.2294
AB	0.0279	0.0459	0.0681	0.1099	0.0262	0.0456	0.0708	0.1142
BM	0.0170	0.0320	0.0555	0.1021	0.0236	0.0374	0.0640	0.1096
Improvement (%)	**+186.0%**	**+133.2%**	**+93.2%**	**+67.2%**	**+82.3%**	**+56.5%**	**+34.6%**	**+19.7%**

E.4 Ablation Study

For the ablation study, we remove one modality at a time and report Hit@k performance on the content-based retrieval tasks in Table 4, using the modalities selected in Section E.3. Overall, the full multimodal model achieves the strongest results, however, certain ablations outperform it (e.g., removing AB at Hit@50). As expected, TC is shown as the most critical modality — its removal causes the largest and most consistent performance drops across all k values in Curated Playlist and in four out of five k values in Co-occurrence, consistent with the findings in Section E.3. AC, ST, and SC are also important, as excluding them consistently degrades performance. In Curated Playlist, most modalities contribute positively at lower k values (5 and 10), whereas some (e.g., AB and SF) may introduce noise at higher k values. In Co-occurrence, the full multimodal model consistently outperforms nearly all ablation variants, with only minor exceptions.

Table 4: Ablation study results by removing one modality at a time. Percentage changes are reported relative to the full multimodal model.

	Curated Playlist				Co-occurrence			
Configuration	**Hit@5**	**Hit@10**	**Hit@20**	**Hit@50**	**Hit@5**	**Hit@10**	**Hit@20**	**Hit@50**
3MToken (ours)	0.2842	**0.3523**	0.4176	0.5133	**0.3004**	0.3746	**0.4330**	**0.5096**
remove AC	0.2777 (-2.3%)	0.3455 (-1.9%)	0.4125 (-1.2%)	0.5085 (-0.9%)	0.2862 (-4.7%)	0.3600 (-3.9%)	0.4268 (-1.4%)	0.4978 (-2.3%)
remove ST	0.2720 (-4.3%)	0.3404 (-3.4%)	0.4054 (-2.9%)	0.4959 (-3.4%)	0.2872 (-4.4%)	0.3640 (-2.8%)	0.4184 (-3.4%)	0.4884 (-4.2%)
remove BM	**0.2845** (+0.1%)	0.3523 (0.0%)	0.4176 (0.0%)	0.5092 (-0.8%)	0.2950 (-1.8%)	**0.3748** (+0.1%)	0.4318 (-0.3%)	0.5050 (-0.9%)
remove SF	0.2798 (-1.5%)	0.3499 (-0.7%)	0.4170 (-0.1%)	0.5157 (+0.5%)	0.2720 (-9.5%)	0.3476 (-7.2%)	0.4148 (-4.2%)	0.4982 (-2.2%)
remove AB	0.2740 (-3.6%)	0.3431 (-2.6%)	**0.4224** (+1.1%)	**0.5276** (+2.8%)	0.2728 (-9.2%)	0.3510 (-6.3%)	0.4174 (-3.6%)	0.5032 (-1.3%)
remove SC	0.2747 (-3.3%)	0.3438 (-2.4%)	0.4091 (-2.0%)	0.4980 (-3.0%)	0.2906 (-3.3%)	0.3594 (-4.1%)	0.4182 (-3.4%)	0.4902 (-3.8%)
remove TC	0.2658 (-6.5%)	0.3308 (-6.1%)	0.3955 (-5.3%)	0.4850 (-5.5%)	0.2752 (-8.4%)	0.3424 (-8.6%)	0.3942 (-9.0%)	0.4598 (-9.8%)