

LLM-Based Playlist Curation: A Music Reasoning Approach at Scale

Zhonghao Luo^{1,*†}, Jianhong Chen^{1,†}, Nathan Apolonio¹ and Amina Shabbeer¹

¹Amazon Music, San Francisco, California, USA

Abstract

We present a production-scale playlist curation system on Amazon Music that leverages modular architecture and holistic LLM-powered reasoning to generate playlists. Our system leverages LLMs to reason over rich track metadata—including genre, mood, era, sonic descriptions, and artist context—to select cohesive track sets that satisfy both relevance and coherence criteria. We formulate the curation task under two complementary inference paradigms: track-wise (binary classification) and list-wise (subset selection). We systematically compare these paradigms and evaluate eight prompt variants across three design dimensions: playlist-level reasoning, track-level reasoning and output structure. Our prompt optimization establishes that outputting only negative tracks achieves 0.96 F1 with 84% fewer output tokens compared to naive all-track approaches. To meet production latency and cost constraints, we develop a teacher-distillation pipeline that transfers reasoning capabilities from Claude Opus 4.6 to Qwen3.5-4B via supervised fine-tuning, achieving comparable playlist quality with 227x cost reduction. Evaluation via recreating both expert-curated editorial playlists and real user personalized playlists demonstrates consistent improvements over production baselines across relevance metrics (weighted precision, weighted NDCG) and coherence metrics (consumption similarity, valence deviation, danceability deviation). Online A/B experiments further validate the framework with improvements in user engagement metric (+0.06%) and user retention metric (+0.03%), confirming that LLMs can serve as effective quality gates for algorithmic playlist curation at scale.

Keywords

Large Language Model, Recommender System, Music Information Retrieval, Chain-of-Thought Reasoning, Knowledge Distillation

1. Introduction

Music streaming platforms serve hundreds of millions of listeners with algorithmically generated playlists [1]. These playlists—often themed around genres, artists, moods, or eras—constitute a primary mode of music consumption and discovery. Curating a high-quality playlist is an intricate process that carefully balances multiple dimensions, including relevance, diversity, popularity, coherence, and serendipity [2, 3, 4, 5]. However, the algorithmic generation pipelines that populate these playlists typically optimize for a subset of these dimensions, leading to suboptimal listening experiences: a slow ballad disrupting a high-energy dance mix, or a punk track infiltrating an arena rock collection.

Traditional heuristic-based filtering approaches fail to capture the multi-dimensional nature of musical fit. A track may share genre tags with a playlist yet be sonically inappropriate due to tempo, mood, or era mismatches. Modern music recommender systems attempt to optimize multiple objectives simultaneously but struggle to find the right balance among competing dimensions such as relevance, coherence, and diversity. This motivates the application of models that can reason over rich track metadata—including genre, mood, era, sonic descriptors, and artist context—and leverage broad world knowledge to make holistic curation decisions. Modern LLMs have shown strong reasoning capabilities on real world tasks including coding and mathematics [6]. Our work demonstrates that LLM’s reasoning capability can effectively apply to playlist curation tasks.

MuRS 2026: 4th Music Recommender Systems Workshop, September 28th, 2026, Minneapolis, Minnesota, USA

*Corresponding author.

† These authors contributed equally.

✉ zhongluo@amazon.com (Z. Luo); jhchenjh@amazon.com (J. Chen); napoloni@amazon.com (N. Apolonio); ashabb@amazon.com (A. Shabbeer)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0). CEUR Workshop Proceedings (CEUR-WS.org)

In this work, we present a framework for LLM-based playlist curation deployed at production scale on Amazon Music. We use "production scale" to refer to the system's capacity to process millions of playlists under strict latency and cost constraints. Our major contributions are as follows:

- **LLM-based playlist reasoning framework:** We design a general-purpose LLM reasoning framework applicable to modern playlist curation systems, in which the model determines whether a set of songs is coherent and relevant to a given theme through chain-of-thought (CoT) reasoning.
- **Comprehensive evaluation framework:** We develop datasets and metrics to assess playlist quality across multiple dimensions of relevance and coherence, enabling systematic benchmarking of different model and prompt variants.
- **Prompt design and optimization for scalable inference:** We systematically evaluate eight prompt variants across three design dimensions—playlist-level reasoning, track-level reasoning, and output structure—establishing an optimal configuration that achieves 0.96 F1 for track filtering accuracy while reducing output tokens by 84% relative to the baseline prompt.
- **Model distillation via supervised fine-tuning (SFT):** We develop a teacher-distillation pipeline from Claude Opus 4.6 to Qwen3.5-4B [7], achieving comparable track filtering accuracy with $227\times$ cost reduction. The distilled model demonstrates consistent improvements over production baselines across relevance and coherence metrics.

2. Related Work

2.1. Music Recommendation and Playlist Curation

Most prior work on playlist curation has focused on fully automatic generation without human intervention. These techniques leverage various user facets and listening habits, such as temporal context [8] or daily activities [9], with the primary goal of capturing and encoding listener preferences to recommend songs and generate playlists [1]. Dias et al. [3] classify playlist generation algorithms into seven categories: (1) *Similarity-based algorithms* that rely on inter-song distances [10, 11]; (2) *Collaborative filtering* methods that learn user listening behavior and recommend songs based on ratings from similar users [12, 13]; (3) *Frequent pattern mining* approaches that consider both local and global patterns [14, 15]; (4) *Statistical models* that learn transition or interaction probabilities [16]; (5) *Case-based reasoning* solutions that retrieve and adapt stored representative cases [17]; (6) *Discrete optimization* approaches that satisfy defined constraints [18]; and (7) *Hybrid techniques* that combine advantages of multiple approaches.

Complementary research has sought to understand creator behaviors, needs, and interaction patterns [19, 20]. These studies demonstrate that user feedback is critical in playlist curation and should be incorporated to improve algorithms and engage users [21]. Notably, some listeners prefer handmade playlists over automatically generated ones [22, 23, 24], a preference attributed to the perceived intentionality of human curation for specific purposes or occasions.

Our work differs from fully automatic playlist generation in that we do not aim to generate playlists end-to-end. Instead, we operate as a quality gate within an existing multi-stage pipeline, filtering candidates produced by upstream retrieval and ranking systems. Unlike prior work that focuses on user preference modeling or transition optimization, our approach leverages explicit chain-of-thought reasoning over track metadata to enforce coherence and relevance criteria that are difficult to encode as heuristic rules.

2.2. LLM-Based Recommender Systems

The incorporation of LLMs into recommender systems has emerged as a key research direction. Recent surveys [25] categorize these methods into two paradigms: *LLM-as-RS* and *LLM-in-RS*. The *LLM-as-RS* paradigm formulates recommendations as text generation tasks, leveraging LLM reasoning capabilities

to directly generate items or explanations [26, 27, 28]. However, these approaches suffer from high inference latency, scaling challenges, and hallucination risks [25, 29]. The LLM-in-RS paradigm leverages semantic understanding to encode text features for downstream recommendation tasks [30, 31, 32], though using only embeddings does not exploit the full reasoning power of LLMs and introduces challenges when fusing with other signals. A third line of work employs LLMs specifically for reranking, which can be further divided into pointwise [33, 34], listwise [35, 36], and pairwise [37, 38] approaches.

Our methodology is most closely related to listwise reranking but differs in two key respects. First, we focus on subset selection (retain/remove) rather than producing a total ordering. Second, unlike prior LLM-as-RS approaches that suffer from hallucination risks when generating item identifiers, our system operates over a closed candidate set with ordinal indices, reducing the risk of hallucinations.

2.3. LLM Reasoning

LLMs demonstrate the capacity for complex deduction and inference when logical rules and principles are integrated into their learning process. Rather than relying solely on statistical patterns [39], they can make decisions grounded in explicit knowledge [40] and structured logical reasoning. Reasoning with foundation models—a rapidly advancing area of AI research—seeks to develop systems that understand and interact with complex information in a more human-like manner [41], leveraging logical reasoning and knowledge representation to handle abstract concepts and apply rule-based decision-making.

A central paradigm for endowing LLMs with reasoning capabilities involves pre-training on large-scale general-domain corpora followed by task- or domain-specific adaptation. One prominent adaptation strategy is supervised fine-tuning on reasoning datasets, where chain-of-thought (CoT) prompting is first used to elicit reasoning paths from a capable model, and the resulting traces are then used as training data for a target model [42, 43, 44, 45]. A complementary paradigm is reinforcement learning (RL) for reasoning [46], which has recently gained traction as a means of incentivizing behavior alignment and extended reasoning. Training with reinforcement learning from verifiable rewards (RLVR) [47, 6]—using signals such as mathematical answer correctness or code unit-test pass rates—enables models to develop long-form reasoning behaviors including planning, reflection, and self-correction, reframing reasoning as a capability that can be explicitly trained and scaled.

In this work, We adopt the SFT-based distillation approach rather than reinforcement learning, as our task provides clear ground-truth labels from expert-curated playlists that serve as direct supervision signals. This choice trades the potential for emergent reasoning behaviors offered by RL for training stability, speed and reproducibility. We leave RL-based reasoning exploration—particularly with composite reward functions combining coherence, diversity, and engagement signals—to future work.

3. Playlist Curation

3.1. System Overview

A typical playlist curation system consists of three stages [3]: (1) *theme selection*, (2) *candidate generation*, and (3) *ranking and post-processing*. Theme selection determines the overall playlist concept, which may center on genre, mood, era, cultural events, or other facets. For expert-curated or user-created playlists, the theme is specified by a human; for algorithmically generated personalized playlists, an algorithm selects the theme based on user preferences, listening behavior, and contextual signals. Candidate generation retrieves songs relevant to the theme—and potentially personalized to the user—typically employing multiple retrieval sources (embedding-based, tag-based, or heuristic-based) to ensure diversity. While candidate generation optimizes for recall, the ranking stage optimizes for precision by ordering more relevant candidates at the top. Then the post-processing stage would apply additional heuristics to sequence tracks and satisfy business requirements such as diversity, coherence, and licensing constraints.

We identify several limitations of current systems. In theme selection, algorithms may fail to capture emerging user interests or balance long-term versus short-term preferences. In candidate

generation, systems struggle to balance relevance and diversity, producing either homogeneous but relevant candidates or diverse but poorly targeted ones. In ranking and post-processing, fragmented heuristics may inadvertently degrade other quality dimensions—for example, promoting a less popular song to the top of a playlist without considering whether its sonic profile fits the surrounding tracks. Given their extensive music knowledge and advanced reasoning capabilities, LLMs are well-suited to address these challenges. In this work, we take an initial step by applying LLMs to the ranking and post-processing stage. We hypothesize that LLMs can follow playlist curation criteria and reason over rich track metadata to produce relevant and coherent song selections [5]. We plan to explore LLMs in theme selection and candidate generation stages in future work.

3.2. Problem Formulation

Given the theme T of a playlist and a list of candidate tracks with metadata $X = \{x_1, \dots, x_N\}$, the task is to find a subset of tracks S that maximizes playlist quality:

$$S^* = \arg \max_{S \subseteq X} Q(T, S) \quad (1)$$

where S^* is the optimal track set and Q is the playlist quality function. Since explicitly defining Q is intractable, we consider two alternative formulations.

Formulation 1: Track-Wise Inference

We formulate playlist curation as a binary classification task in which the LLM classifies each track x_i as RELEVANT (to retain) or NOT RELEVANT (to remove):

$$y_i = f(x_i, T) \quad (2)$$

where $f(\cdot)$ outputs a binary label $y_i \in \{0, 1\}$, and the optimal set predicted by the model is $\hat{S}^* = \{x_i \mid y_i = 1, 1 \leq i \leq N\}$.

This formulation simplifies the task to independent binary decisions but suffers from two limitations:

1. **Lack of global context:** Each decision is made independently, without awareness of other removal decisions that should inform the conditional context.
2. **Poor scalability:** Each playlist requires N inference calls, yielding $\mathcal{O}(N \cdot M)$ total inferences across M playlists.

Formulation 2: List-Wise Inference

Alternatively, we leverage the complex reasoning capability of LLMs to approximate Q and generate a coherent subset from the candidate pool:

$$\hat{S}^* = g(T, X) \quad (3)$$

where $g(\cdot)$ outputs the selected track set and $\hat{S}^*$ is the optimal set generated by the model. This formulation provides a holistic view in which each track decision is conditioned on all other tracks, potentially improving coherence. It also reduces inference calls to $\mathcal{O}(M)$ —one per playlist—though with longer input and output sequences. The quality function Q is implicitly learned by the model rather than explicitly defined. We compare these two formulations empirically in Section 6.

4. Prompt Design and Optimization

4.1. Track-Wise Prompt

The track-wise prompt comprises the following components:

1. **Curation Guidelines:** A set of curation principles gathered from expert curators and industry references (e.g., “Consider musical lineage, cultural context, artistic movements, and thematic resonance”).
2. **Playlist Theme:** A description of the playlist theme, including metadata such as title and description, as well as theme identifiers from upstream systems (e.g., seed songs or seed artists).
3. **Track Metadata:** A rich set of metadata collected and produced by complementary systems, including internal tags, title, album, artists, sonic properties, text descriptions, and popularity. Text descriptions are generated by a separate multi-modal language model.
4. **Output Guidelines:** The prescribed output format, instructing the LLM to provide chain-of-thought reasoning before rendering a verdict of `RELEVANT` or `NOT RELEVANT`.

An example prompt template can be found in Appendix A. We find that chain-of-thought reasoning significantly improves verdict accuracy in offline experiments. We calibrate reasoning length to satisfy latency and scalability requirements. Additionally, we find that using XML tags for the verdict enables reliable output parsing.

4.2. List-Wise Prompt

The list-wise prompt shares the same **Curation Guidelines**, **Playlist Theme**, and **Track Metadata** components, with track metadata encompassing all N candidates. For **Output Guidelines**, we systematically vary three independent design dimensions across eight prompt variants:

1. **Playlist-level reasoning:** Whether to prepend a high-level description of the ideal playlist structure before individual track decisions.
2. **Track-level reasoning:** Whether to require per-track justification before the verdict.
3. **Output structure:** Whether to emit all tracks, negatives only, or positives only.

An example prompt template that emits negative tracks only can be found in Appendix A. This factorial design addresses two research questions: (1) How much reasoning is necessary for accurate curation? (2) Which output structure best balances latency and quality?

5. Supervised Fine-Tuning via Teacher Distillation

Production systems impose strict latency and cost budgets on each component. We observe that proprietary frontier LLMs, such as Claude Opus 4.6, cannot meet these constraints. Instead, we propose distilling a frontier model’s curation judgments into a smaller open-source model via supervised fine-tuning (SFT). We hypothesize that the smaller model can achieve comparable performance after distillation while delivering substantially lower latency and cost.

5.1. Training Dataset

The objective of SFT is to teach the student model to reason through track metadata and playlist themes to produce correct verdicts. We leverage approximately 20,000 expert-curated editorial playlists as our high-quality ground-truth dataset. These playlists are created by trained music programmers following industry-standard track selection and sequencing guidelines, and they span a wide range of artists, genres, eras, activities, and cultural moments.

For **track-wise inference**, the dataset consists of (playlist, track) pairs, where positive pairs are drawn from original playlists and negative pairs from randomly sampled tracks of other playlists. We apply the track-wise prompt to the teacher model (Claude Opus 4.6) to generate reasoning traces and verdicts (`RELEVANT` or `NOT RELEVANT`), retaining only samples where the teacher’s verdict matches the ground truth. This yields approximately 343,000 training pairs.

For **list-wise inference**, we construct synthetic playlists by combining original tracks (positives) with randomly sampled tracks from other playlists (negatives). To control for playlist length and mimic

production conditions, we constrain synthetic playlists to 40–60 tracks by filtering short playlists and downsampling long ones. The positive-to-negative ratio is set to 5:1, reflecting the empirical ratio observed in personalized playlists. We acknowledge that this ratio creates a relatively easy classification setting: a naive majority-class baseline that predicts all tracks as relevant would achieve 83.3% accuracy. Therefore, we also report F1 metric to understand its capability of identifying positive tracks. We apply the list-wise prompt to the teacher model to generate reasoning traces and track selections. Unlike the track-wise setting, we retain all samples regardless of perfect alignment with ground truth, as the teacher achieves high F1 ($>90\%$) and the remaining disagreements represent genuinely ambiguous cases that may improve model robustness. This produces approximately 40,000 training playlists.

Limitations of the teacher model. We acknowledge several limitations of using Claude Opus 4.6 as the source of reasoning traces. First, the model’s training data has a knowledge cutoff that excludes recently released tracks and evolving genre trends, potentially biasing decisions toward well-established artists and canonical genre boundaries. Second, the model reasons over textual metadata rather than audio signals directly; it cannot perceive sonic qualities such as timbre, production style, or mixing characteristics that human curators assess by listening.

We partially mitigate these limitations through two mechanisms: (1) the rich metadata representation includes machine-generated sonic descriptions from a separate multi-modal model, providing the LLM with audio information it cannot derive independently; (2) we retain only teacher outputs that agree with expert-curated ground truth during training data construction, filtering out hallucinated or incorrect judgments. Nevertheless, we do not claim the teacher model is an oracle, and future work should incorporate direct human evaluation to establish an upper bound on teacher quality.

5.2. Evaluation Dataset

We develop two complementary evaluation datasets to assess both relevance and coherence.

5.2.1. Editorial Playlist Dataset

Following the same construction strategy as Section 5.1, we generate an evaluation set from editorial playlists via a train-test split that prevents playlist overlap between training and evaluation. Since ground-truth labels are available for each track, we report standard classification metrics: PRECISION, RECALL, and F1.

5.2.2. Personalized Playlist Dataset

To validate our methods on real user behavior, we collect approximately 50,000 personalized playlists from the platform. Since listening signals on these playlists are sparse, we additionally collect 8 weeks of post-creation playback sessions covering each user’s listening activity across the platform. We compute a Wilson score for each (user, track) pair based on play frequency and duration, representing observed user preference [48]. We define the following relevance metrics:

1. **Weighted Precision:** For each playlist track, we look up its Wilson score from the (user, track) pair. Tracks without observed interactions receive the population-average Wilson score as a baseline. Weighted precision is the mean score across all playlist positions.
2. **Weighted NDCG:** We apply Wilson scores with position discount ($1/\log_2(\text{rank} + 1)$) and normalize by the ideal arrangement (IDCG) at a fixed playlist length to ensure fair cross-playlist comparison.

Furthermore, we measure the coherence of resulting playlists in terms of similarity with the theme. Although there are different definitions of coherence [5], we define the following coherence metrics based on how similar are the playlist songs with the thematic seed songs.

1. **Consumption Similarity:** Using track embeddings derived from co-listening behavior, we compute the dot-product similarity between each track and the seed track, averaged across the playlist.

Table 1

Comparison of inference paradigms. Throughput measured on 8× A100-SXM4-40GB GPUs.

Paradigm	Input Tokens	Output Tokens	Inference Pattern	Throughput
Track-wise	600	400	50 calls/playlist	0.34 playlists/s
List-wise	12000	500	1 calls/playlist	7.5 playlists/s

2. **Valence Deviation:** Valence captures the positivity of a track (high for cheerful/euphoric, low for sad/angry). We compute the mean squared deviation (MSD) between each track’s valence and the seed track’s valence.
3. **Danceability Deviation:** Danceability captures suitability for dancing based on tempo, rhythm, beat strength, and regularity. We compute the MSD between each track’s danceability score and the seed track’s score.

6. Experiments

6.1. Training Setup

We use Qwen3.5-4B as student models, selected for their favorable latency-quality tradeoff in preliminary experiments. We use AdamW 8-bit optimizer with learning rate = 2×10^{-4} , linear schedule and 5 warmup steps. The model is trained on 8 A100 GPUs. We also compare LoRA ($r = 64$) with full-rank training to understand if there is any catastrophic forgetting effect. Loss converges after 1 epoch in all configurations.

6.2. Scalability: List-Wise vs. Track-Wise Inference

We compare the two inference paradigms in terms of tokens and throughput. In track-wise inference, the model evaluates one (playlist, track) pair per call, requiring approximately 50 calls per playlist. In list-wise inference, all tracks are evaluated in a single prompt, enabling holistic reasoning over inter-track relationships. Table 1 summarizes throughput. Despite longer input contexts, list-wise inference achieves a $22\times$ throughput gain—a combined effect of prompt optimization and reduced per-playlist inference calls.

6.3. Prompt Optimization for List-Wise Inference

We systematically evaluate 8 prompt variants as described in Section 4.2, using the editorial playlist dataset with known positive and negative labels. Results are shown in Table 2.

Table 2

Prompt ablation results for list-wise inference. The best results are highlighted in **bold**. * denotes recommended configuration.

Prompt	Playlist Reasoning	Track Reasoning	Output Format	F1	Accuracy	Output Tokens
P1	Yes	Yes	All tracks	0.91	0.88	2,194
P2	No	Yes	All tracks	0.92	0.89	2,019
P3*	No	Yes (Negatives)	Negatives only	0.96	0.94	344
P4	Yes	Yes (Negatives)	Negatives only	0.95	0.93	512
P5	Yes	Yes (Positives)	Positives only	0.89	0.85	1,185
P6	No	Yes (Positives)	Positives only	0.90	0.87	1,062
P7	Yes	No	Negatives only	0.95	0.93	218
P8	Yes	No	Positives only	0.93	0.90	414

Overall, prompt 3 that outputs only track-level reasoning on negative tracks without playlist-level reasoning achieves the highest F1=0.96 and accuracy=0.94, while also keeping the output tokens short. Several findings emerge from this analysis. **Output structure is the dominant factor.** Negative-only prompts achieve mean F1 = 0.953 versus 0.915 for all-track prompts and 0.907 for positive-only prompts. By constraining the model to reason only about tracks to remove, attention is focused on the decision-relevant subset—the minority of tracks that violate coherence or relevance criteria. **Playlist-level reasoning adds no measurable value** on top of track-level reasoning. Comparing P3 (no playlist reasoning) with P4 (with playlist reasoning) reveals additional output tokens without F1 or accuracy improvement, suggesting the model does not benefit from explicitly summarizing playlist context when per-track reasoning is already present. Based on these findings, we adopt P3 as the final prompt configuration for all subsequent evaluations.

6.4. Evaluation Results of Distilled Models

We evaluate SFT-distilled models on the personalized playlist dataset (Section 5.2). Table 3 reports results as percentage changes relative to the production baseline. We show models that are fine-tuned through LoRA and full-rank modes. We compare two inference paradigms for completeness, although track-wise inference doesn’t meet the scalability requirements in practice.

Table 3

Personalized playlist evaluation (Qwen3.5-4B). Metrics shown as percentage change over the production baseline. ↑: higher is better; ↓: lower is better. Best results in **bold**.

Inference Paradigm	Training Mode	Weighted Precision ↑	Weighted NDCG ↑	Consumption Similarity ↑	Valence Deviation ↓	Danceability Deviation ↓
List-wise	None	0.0%	0.002%	0.09%	-0.3%	-0.17%
Track-wise	LoRA	-0.004%	0.012%	0.33%	-3.1%	-3%
List-wise	LoRA	0.004%	0.028%	0.26%	-1.8%	-1.46%
List-wise	Full-rank	0.006%	0.092%	0.26%	-0.8%	-1.24%

Several observations merit discussion. First, all models—including the off-the-shelf (non-fine-tuned) Qwen3.5-4B—improve over the production baseline across both relevance and coherence metrics, indicating that LLMs possess inherent music reasoning capabilities. Second, SFT-distilled models demonstrate significant improvements over the pretrained model, confirming that distillation successfully transfers both appropriate reasoning traces and output formatting. Third, list-wise inference excels on relevance metrics (weighted precision, weighted NDCG), while track-wise inference achieves stronger coherence metrics (consumption similarity, valence deviation, danceability deviation). We attribute this to the track-wise prompt’s longer per-track reasoning trace, which emphasizes sonic properties of individual songs rather than holistic playlist analysis. Finally, full-rank training yields slightly better relevance metrics than LoRA, suggesting no catastrophic forgetting and confirming that full-rank training is appropriate for this task.

6.5. Online Experiments

We conduct an online A/B test to further validate our methods through user feedback. The experiment is randomized on users, where the control group sees playlists from production baseline while the treatment group sees playlists generated by the full-rank list-wise model from Table 3. Despite the inference optimization efforts, it is still challenging and cost-prohibitive to deploy the system to the full scale of Amazon Music. Therefore, we run the online experiment on a limited cohort of active users. After 4 weeks of experiment, we observe statistically significant improvements in user engagement metric (**+0.06%**) and user retention metric (**+0.03%**). This verifies our hypothesis that LLM-based framework is able to improve the relevance and coherence of playlists and lead to better user engagement on the platform.

7. Conclusion

We have presented a framework for LLM-based playlist curation that bridges the gap between frontier model reasoning capabilities and production-scale deployment constraints. Through systematic prompt optimization, we established that a negative-only output structure achieves 0.96 F1 with 84% fewer tokens than naive all-track approaches. Our teacher-distillation pipeline successfully transfers Claude Opus 4.6’s music reasoning capabilities into a deployable Qwen3.5-4B model, achieving $227\times$ cost reduction. Evaluation on both editorial and personalized playlists demonstrates consistent improvements over production baselines on relevance and coherence metrics. Online A/B experiments further validate the framework with improvements in user engagement metric (+0.06%) and user retention metric (+0.03%), confirming that LLMs can serve as effective quality gates for algorithmic playlist curation at scale.

7.1. Limitations

Evaluation sparsity: Consumption signals on personalized playlists are sparse, leading to high metric variance. We partially address this by incorporating listening behavior outside of personalized playlists, though this may not perfectly reflect playlist-specific quality. Future work should explore denser evaluation methodologies.

Negative sample quality: The current editorial dataset uses randomly injected tracks as negatives, creating relatively easy classification examples. Hard negatives—tracks that are sonically similar but thematically inappropriate (e.g., wrong era within the same genre, similar BPM but mismatched mood)—would better stress-test model capabilities and improve robustness.

Scalability constraints: Despite prompt optimization and model distillation, the framework still requires one LLM inference per playlist across millions of playlists. This poses cost and scaling challenges in production environments with limited GPU resources.

Teacher model as ground truth: Our distillation pipeline relies on Claude Opus 4.6 as the source of reasoning traces, inheriting its limitations including knowledge cutoff, inability to perceive audio directly, and potential hallucination of musical relationships. We plan to evaluate the quality of teacher model output in future work.

Narrow task scope: The LLM is applied only to the ranking and post-processing stage, filtering approximately 15–20% of candidates from pre-curated pools of 50 tracks. The more challenging and potentially higher-impact applications—theme selection, candidate generation from large catalogs, and sequential track ordering—remain unaddressed.

7.2. Future Work

Several directions are worth further investigation. First, we plan to improve training and evaluation datasets through hard negative mining—generating negatives that are sonically similar but thematically inappropriate (e.g., 1980s rock tracks for a 1990s rock playlist)—to enable finer-grained coherence reasoning. Second, we intend to explore reinforcement learning with composite reward functions that jointly optimize coherence, diversity, and engagement. Third, we aim to apply LLM reasoning to other stages of playlist curation, including theme selection and playlist description generation. Finally, we plan to evaluate whether LLM-based curation reasoning transfers to other recommendation domains such as video or product recommendations.

Declaration on Generative AI

During the preparation of this work, the author(s) used Claude Sonnet in order to: Grammar and spelling check. After using these tool(s)/service(s), the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication’s content.

References

- [1] G. Bonnin, D. Jannach, Automated generation of music playlists: Survey and experiments, *ACM Computing Surveys (CSUR)* 47 (2014) 1–35.
- [2] S. J. Cunningham, D. Bainbridge, A. Falconer, " more of an art than a science": Supporting the creation of playlists and mixes (2006).
- [3] R. Dias, D. Gonçalves, M. J. Fonseca, From manual to assisted playlist creation: a survey, *Multimedia Tools and Applications* 76 (2017) 14375–14403.
- [4] A. N. Hagen, The playlist experience: Personal playlists in music streaming services, *Popular Music and Society* 38 (2015) 625–645.
- [5] H. Schweiger, E. Parada-Cabaleiro, M. Schedl, The impact of playlist characteristics on coherence in user-curated music playlists, *EPJ Data Science* 14 (2025) 24.
- [6] D. Guo, D. Yang, H. Zhang, J. Song, P. Wang, Q. Zhu, R. Xu, R. Zhang, S. Ma, X. Bi, et al., Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, *arXiv preprint arXiv:2501.12948* (2025).
- [7] Qwen Team, Qwen3.5: Towards native multimodal agents, 2026. URL: <https://qwen.ai/blog?id=qwen3.5>.
- [8] P. Herrera, Z. Resa, M. Sordo, Rocking around the clock eight days a week: an exploration of temporal patterns of music listening, in: 1st Workshop On Music Recommendation And Discovery (WOMRAD), *ACM RecSys*, 2010, Barcelona, Spain, 2010.
- [9] R. Dias, M. J. Fonseca, R. Cunha, A user-centered music recommendation approach for daily activities., in: *CBRecSys@ RecSys*, 2014, pp. 26–33.
- [10] P. Knees, M. Schedl, T. Pohle, G. Widmer, An innovative three-dimensional user interface for exploring music collections enriched, in: *Proceedings of the 14th ACM international conference on Multimedia*, 2006, pp. 17–24.
- [11] M. Slaney, W. White, Similarity based on rating data., in: *ISMIR*, 2007, pp. 479–484.
- [12] J. Bennett, S. Lanning, et al., The netflix prize, in: *Proceedings of KDD cup and workshop*, volume 2007, New York, 2007, p. 35.
- [13] W. W. Cohen, W. Fan, Web-collaborative filtering: Recommending music by crawling the web, *Computer Networks* 33 (2000) 685–698.
- [14] S. Chen, J. L. Moore, D. Turnbull, T. Joachims, Playlist prediction via metric embedding, in: *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2012, pp. 714–722.
- [15] M. Crampes, J. Villerd, A. Emery, S. Ranwez, Automatic playlist composition in a dynamic music landscape, in: *Proceedings of the 2007 international workshop on Semantically aware document processing and indexing*, 2007, pp. 15–20.
- [16] M. Schedl, Deep learning in music recommendation systems, *Frontiers in Applied Mathematics and Statistics* 5 (2019) 44.
- [17] C. Baccigalupo, E. Plaza, Case-based sequential ordering of songs for playlist recommendation, in: *European conference on case-based reasoning*, Springer, 2006, pp. 286–300.
- [18] J.-J. Aucouturier, F. Pachet, Scaling up music playlist generation, in: *Proceedings. IEEE International Conference on Multimedia and Expo*, volume 1, IEEE, 2002, pp. 105–108.
- [19] Y.-X. Chen, A. Butz, Musicsim: integrating audio analysis and user feedback in an interactive music browsing ui, in: *Proceedings of the 14th international conference on Intelligent user interfaces*, 2009, pp. 429–434.
- [20] M. Schedl, A. Flexer, J. Urbano, The neglected user in music information retrieval research, *Journal of Intelligent Information Systems* 41 (2013) 523–539.
- [21] D. M. Weigl, C. Guastavino, User studies in the music information retrieval literature., in: *ISMIR*, 2011, pp. 335–340.
- [22] F. Bakalov, M.-J. Meurs, B. König-Ries, B. Sateli, R. Witte, G. Butler, A. Tsang, An approach to controlling user models and personalization effects in recommender systems, in: *Proceedings of the 2013 international conference on Intelligent user interfaces*, 2013, pp. 49–56.

- [23] D. Baur, B. Hering, S. Boring, A. Butz, Who needs interaction anyway: exploring mobile playlist creation from manual to automatic, in: Proceedings of the 16th international conference on Intelligent user interfaces, 2011, pp. 291–294.
- [24] A. Laplante, J. S. Downie, et al., Everyday life music information-seeking behaviour of young adults., in: ISMIR, 2006, pp. 381–382.
- [25] W. Hua, L. Li, S. Xu, L. Chen, Y. Zhang, Tutorial on large language models for recommendation, in: Proceedings of the 17th ACM Conference on Recommender Systems, 2023, pp. 1281–1283.
- [26] K. Bao, J. Zhang, Y. Zhang, W. Wang, F. Feng, X. He, Tallrec: An effective and efficient tuning framework to align large language model with recommendation, in: Proceedings of the 17th ACM conference on recommender systems, 2023, pp. 1007–1014.
- [27] J. Zhang, R. Xie, Y. Hou, W. X. Zhao, L. Lin, J.-R. Wen, Recommendation as instruction following: A large language model empowered recommendation approach, *ACM Transactions on Information Systems* 43 (2026) 1–37.
- [28] S. Doh, K. Choi, J. Nam, Talkplay: Multimodal music recommendation with large language models, *arXiv preprint arXiv:2502.13713* (2025).
- [29] Z. Yue, S. Rabhi, G. d. S. P. Moreira, D. Wang, E. Oldridge, Llamarec: Two-stage recommendation using large language models for ranking, *arXiv preprint arXiv:2311.02089* (2023).
- [30] Z. Cui, Y. Weng, X. Tang, X. Zhang, S. Li, P. Liu, B. He, D. Liu, W. Luo, X. He, et al., Semantic retrieval augmented contrastive learning for sequential recommendation, *Advances in Neural Information Processing Systems* 38 (2026) 162621–162645.
- [31] W. Wei, X. Ren, J. Tang, Q. Wang, L. Su, S. Cheng, J. Wang, D. Yin, C. Huang, Llmrec: Large language models with graph augmentation for recommendation, in: Proceedings of the 17th ACM international conference on web search and data mining, 2024, pp. 806–815.
- [32] Y. Yang, Z. Ji, Z. Li, Y. Li, Z. Mo, Y. Ding, K. Chen, Z. Zhang, J. Li, L. LIN, et al., Sparse meets dense: Unified generative recommendations with cascaded sparse-dense representations, *Advances in Neural Information Processing Systems* 38 (2026) 93746–93770.
- [33] P. Liang, R. Bommasani, T. Lee, D. Tsipras, D. Soylu, M. Yasunaga, Y. Zhang, D. Narayanan, Y. Wu, A. Kumar, et al., Holistic evaluation of language models, *arXiv preprint arXiv:2211.09110* (2022).
- [34] K. Long, S. Li, C. Xu, J. Tang, T. Wang, Precise zero-shot pointwise ranking with llms through post-aggregated global context information, in: Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2025, pp. 2384–2394.
- [35] J. Podolak, L. Perić, M. Janićijević, R. Petcu, Beyond reproducibility: Advancing zero-shot llm reranking efficiency with setwise insertion, in: Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2025, pp. 3205–3213.
- [36] W. Sun, L. Yan, X. Ma, S. Wang, P. Ren, Z. Chen, D. Yin, Z. Ren, Is chatgpt good at search? investigating large language models as re-ranking agents, in: Proceedings of the 2023 conference on empirical methods in natural language processing, 2023, pp. 14918–14937.
- [37] Q. Luo, E. Chen, Z. Shi, B. Wang, Anchor-based pairwise comparison via large language model for recommendation reranking, in: Proceedings of the 34th ACM International Conference on Information and Knowledge Management, 2025, pp. 5001–5005.
- [38] Z. Qin, R. Jagerman, K. Hui, H. Zhuang, J. Wu, L. Yan, J. Shen, T. Liu, J. Liu, D. Metzler, et al., Large language models are effective text rankers with pairwise ranking prompting, in: Findings of the Association for Computational Linguistics: NAACL 2024, 2024, pp. 1504–1518.
- [39] Z. Yang, X. Du, R. Mao, J. Ni, E. Cambria, Logical reasoning over natural language as knowledge representation: A survey, *arXiv preprint arXiv:2303.12023* (2023).
- [40] J. Mao, J. Ye, Y. Qian, M. Pavone, Y. Wang, A language agent for autonomous driving, *arXiv preprint arXiv:2311.10813* (2023).
- [41] K. Valmeekam, M. Marquez, S. Sreedharan, S. Kambhampati, On the planning abilities of large language models-a critical investigation, *Advances in neural information processing systems* 36 (2023) 75993–76005.
- [42] Y. Fu, H. Peng, L. Ou, A. Sabharwal, T. Khot, Specializing smaller language models towards multi-step reasoning, in: International Conference on Machine Learning, PMLR, 2023, pp. 10421–10430.

- [43] C.-Y. Hsieh, C.-L. Li, C.-K. Yeh, H. Nakhost, Y. Fujii, A. Ratner, R. Krishna, C.-Y. Lee, T. Pfister, Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes, in: Findings of the Association for Computational Linguistics: ACL 2023, 2023, pp. 8003–8017.
- [44] J. Huang, S. Gu, L. Hou, Y. Wu, X. Wang, H. Yu, J. Han, Large language models can self-improve, in: Proceedings of the 2023 conference on empirical methods in natural language processing, 2023, pp. 1051–1068.
- [45] S. Li, J. Chen, Y. Shen, Z. Chen, X. Zhang, Z. Li, H. Wang, J. Qian, B. Peng, Y. Mao, et al., Explanations from large language models make small reasoners better, arXiv preprint arXiv:2210.06726 (2022).
- [46] F. Xu, Q. Hao, C. Shao, Z. Zong, Y. Li, J. Wang, Y. Zhang, J. Wang, X. Lan, J. Gong, et al., Toward large reasoning models: A survey of reinforced reasoning with large language models, Patterns 6 (2025).
- [47] A. Jaech, A. Kalai, A. Lerer, A. Richardson, A. El-Kishky, A. Low, A. Helyar, A. Madry, A. Beutel, A. Carney, et al., Openai o1 system card, arXiv preprint arXiv:2412.16720 (2024).
- [48] H. Huang, Y. Wei, X. Yuan, R. Zheng, Weighted matrix factorization with wilson lower bound score, in: Proceedings of the 2023 5th Asia Pacific information technology conference, 2023, pp. 33–37.

A. Prompt Template

Track-wise Prompt Template

You are an expert music curator with deep knowledge of genres, artists, musical styles, and playlist curation best practices. Analyze whether this track fits the given playlist theme.

Playlist Theme

{playlist theme}

Track Metadata

{track metadata}

Curation Guidelines

{curation guidelines}

Output Guidelines

Provide step-by-step analysis and explore the connections you discover. Then conclude with: RELEVANT or NOT RELEVANT

List-wise Prompt Template

You are an expert music curator with deep knowledge of genres, artists, musical styles, and playlist curation best practices. Analyze whether the candidate tracks fit the given playlist theme.

Playlist Theme

{playlist theme}

Candidate Tracks Metadata

{candidate tracks metadata}

Curation Guidelines

{curation guidelines}

Output Guidelines

Only output the tracks that need to be removed. For each track to be removed, output one line in this exact format: [track number] [reason to remove]