Learning When to Personalize: LLM Based Playlist Generation via Query Taxonomy and Classification

Fedor Buzaev[1,2], **Ivan Sukharev**[1] , **Rinat Mullahmetov**[1,3] , **Roman Bogachev**[1] , **Ilya Sedunov**[1] , **Oleg Pavlovich**[1] , **Daria Pugacheva**[2,4]
[1]Zvuk , [2]HSE University , [3]Innopolis University , [4]Domain-specific NLP Group
Correspondence: {fa.buzaev,dpugacheva}@hse.ru, ivan@sukharev.me

Abstract

Playlist generation based on textual queries using large language models (LLMs) is becoming an important interaction paradigm for music streaming platforms. User queries span a wide spectrum from highly personalized intent to essentially catalog-style requests. Existing systems typically rely on non-personalized retrieval/ranking or apply a fixed level of preference conditioning to every query, which can overfit catalog queries to a single user or underpersonalize explicitly listener-dependent requests. We present an industrial-scale LLM-based playlist generation system with dynamic personalization that adapts the personalization strength to the query type. We define a query taxonomy, train a query-type classifier on 5,000 manually labeled queries, and use its predicted probability to modulate the mixture of LLM-based semantic scoring and personalized evaluation. In a blind user study with pairwise comparisons and ELO aggregation, this approach consistently outperforms both non-personalized and fixed-personalization baselines.

1 Introduction

Large language models (LLMs) are progressively being employed to power natural language interfaces for recommender systems (Wu et al., 2024). In the context of music streaming, users are able to formulate playlist requests using free-form text, such as 'upbeat indie rock for a Friday night with friends' or 'calm instrumental music for deep work'. LLMs effectively parse such rich queries and retrieve items that are semantically aligned with the articulated user intent (Palumbo et al., 2025). At the same time, music recommendation is inherently personal. Different users may submit identical queries yet expect vastly different playlists, influenced by their long-term preferences, listening histories, and current context. Consequently, a practical LLM-based playlist generator must integrate two key signals: semantic relevance to the query and personalization to the user. Modern platforms typically utilize collaborative or sequence-based recommender systems to capture these individual preferences (Zeng et al., 2024; Deldjoo et al., 2024). However, not all queries should be personalized to the same degree. Some queries are explicitly dependent on the preferences of an individual user (e.g., 'my favorite pop hits of the 2010s'), others are primarily constrained by external limitations and specific descriptions (e.g., 'best disco hits of the 1980s'). Applying a strong collaborative approach to the latter may degrade the perceived quality by replacing canonical tracks with niche songs that appeal only to a particular user. Conversely, weak personalization for taste-dependent queries may result in the creation of generic playlists that feel detached from the listening history of the user.

In this work, we argue that the degree of personalization should be adapted to the query type. We analyze real user queries from a large-scale music streaming platform and construct a taxonomy of 19 categories, grouped into queries that warrant strong personalization and queries that should primarily adhere to the textual description. Building on this taxonomy, we train a query-type classifier on 5,000 manually labeled queries sourced from the *Zvuk.com* music streaming platform, and use its predicted probability as a continuous personalization strength.

Our system starts from an LLM-based embedder that retrieves candidate tracks by semantic similarity to the text query. A collaborative recommender scores the same candidates based on the listener's historical behavior. For each query, the classifier predicts the probability that this query benefits from strong personalization. Then the semantic and collaborative scores are fused.

We evaluate several model variants in a blind

Proceedings of the 4th Workshop on NLP for Music and Audio (NLP4MusA 2026), pages 51–57

user study with pairwise comparisons between systems. Participants see two anonymized playlists for the same query and indicate which one better matches their expectations, or that they are equivalent. We aggregate outcomes with an ELO-style rating system and complement this with non-parametric significance tests. The dynamic personalization variant achieves the highest ELO rating and win rate.

Our contributions are:

- A taxonomy of text-to-playlist queries from a commercial music streaming platform, highlighting when strong personalization is desirable or harmful.
- A method for dynamically fusing LLM-based and collaborative scores based on the type of a query.
- A blind user study with pairwise comparisons and ELO aggregation, showing that query-type-aware dynamic personalization improves perceived playlist quality over non-personalized and fixed-personalization baselines.

2 Related Work

Large language models (LLMs) enable text-to-playlist generation by mapping natural language queries to semantically relevant tracks. Prototypes like LLMusic integrate LLMs with Spotify APIs for prompt-based playlist creation without fine-tuning (Preda, 2023). Industrial systems like Text2Tracks extend this with generative retrieval, which generates track IDs directly from a textual query (Palumbo et al., 2025).

Traditional music recommenders fuse semantic relevance with collaborative filtering (Zeng and Umrawal, 2025), but fixed-weight hybrids often over-personalize catalog queries or underfit taste-dependent ones. Sequential models incorporate personalized popularity awareness to balance long-term preferences and novelty in next-track prediction (Abbattista et al., 2024). Dynamic user interest models adapt recommendations over time by modeling evolving preferences, yet lack query-specific modulation (Mao et al., 2024).

3 Methodology

In this section, we formalize the text-to-playlist task, outline the semantic retrieval and query expansion approach based on LLMs, present the collaborative re-ranking component, and introduce the dynamic personalization system guided by the query-type classifier.

3.1 Problem Setting and Notation

We consider a platform with a set of users $\mathcal{U}$ and a catalog of tracks $\mathcal{T}$. A user $u \in \mathcal{U}$ issues a free-form natural language query x. Given (u, x), the system must produce an ordered list of tracks

$$\pi(u, x) = (t_1, t_2, \ldots, t_K),$$

where $t_k \in \mathcal{T}$ and K is the playlist length.

The pipeline operates in two main stages:

1. **Semantic candidate generation** with an LLM-based embedder.
2. **Personalized re-ranking** with a collaborative recommender, optionally modulated by a query-type classifier.

We then denote the semantic (Embedder) score by $s_{\text{llm}}(x, t)$ and the collaborative (RecSys) score by $s_{\text{cf}}(u, t)$. Both scores are normalized to $(0, 1]$ via monotonic transformations so that multiplicative fusion is well-defined.

3.2 LLM-Based Semantic Retrieval

The first stage uses an LLM-based text encoder to map the query x to a dense vector representation $\mathbf{v}_{\text{query}}(x) \in \mathbb{R}^d$. Each track $t \in \mathcal{T}$ is associated with one or more textual descriptions (e.g., title, artist names, editorial tags), which are encoded into vectors $\mathbf{v}_t \in \mathbb{R}^d$ using the same or a compatible encoder. The LLM-based text encoder is fine-tuned using the InfoNCE contrastive objective on triplets of anchor, positive, and negative examples to map queries and tracks into a shared semantic embedding space that enables meaning-based retrieval.

We perform nearest-neighbor search in the track embedding index to find the set $\mathcal{C}(x)$ of top-N candidates. We define the semantic relevance score as

$$s_{\text{llm}}(x, t) = \text{sim}(\mathbf{v}_{\text{query}}(x), \mathbf{v}_t), \qquad (1)$$

where $\text{sim}(\cdot, \cdot)$ is the cosine similarity (Manning et al., 2008).

This stage ensures that all candidates are thematically connected to the textual description, even for users with sparse histories or for niche queries that do not align well with collaborative signals.

3.3 Collaborative Re-Ranking

The second stage uses a production collaborative recommender to personalize the candidate set $\mathcal{C}(x)$ for user u. The model takes as input the user identifier and track features (including historical interactions such as listens, skips, and likes) and outputs a score $s_{\text{cf}}(u,t) \in (0,1]$, which reflects how well track t matches long-term preferences of the user.

For this purpose, the expanded vector $\mathbf{v}_{\text{exp}}(u,x)$ (see the exact definition in the Appendix A) replaces $\mathbf{v}_{\text{query}}(x)$ in the similarity computation (1), yielding a score that reflects both the textual description and the explicit feedback of the user.

A simple fixed-personalization baseline fuses the two scores using a global exponent $\alpha_{\text{fixed}} \geq 0$:

$$s^{\text{fixed}}_{\text{final}}(u,x,t) = s_{\text{llm}}(x,t) \cdot \big(s_{\text{cf}}(u,t)\big)^{\alpha_{\text{fixed}}}. \quad (2)$$

When $\alpha_{\text{fixed}} = 0$, the ranking is purely semantic. As α_{fixed} increases, the model becomes more personalized and elevates preferred tracks, even if their connection to the query is weak.

However, a single global α_{fixed} cannot capture the diversity of query types. A fixed value may be both insufficient for queries that are highly dependent on personal preferences or excessive for descriptive or catalog-oriented queries. To address this issue, we introduce a query-type classifier and define α as a query-dependent function.

3.4 Dynamic Personalization

Given that queries fall into two types, one strictly personalized and the other corresponding to non-personalized catalog queries, let us introduce a binary variable $y \in \{0,1\}$. Here, $y = 1$ if a query falls into the first type and $y = 0$ if it belongs to the second. Thus, a binary classifier trained on the labeled data returns the probability that a query x is strictly personalized, and the coefficient α is calculated as

$$\alpha(x) = p_\theta(y = 1 \mid x) \in [0,1]. \quad (3)$$

Given $\alpha(x)$, we define the dynamically personalized fusion score as

$$s^{\text{dyn}}_{\text{final}}(u,x,t) = s_{\text{llm}}(x,t) \cdot \big(s_{\text{cf}}(u,t)\big)^{\alpha(x)}. \quad (4)$$

When $\alpha(x) \to 0$, the collaborative factor tends to 1 and the final score is dominated by the LLM component:

$$\lim_{\alpha(x)\to 0} s^{\text{dyn}}_{\text{final}}(u,x,t) = s_{\text{llm}}(x,t). \quad (5)$$

When $\alpha(x)$ is close to 1, the collaborative score retains its full strength and strongly re-orders the semantic candidates. In intermediate cases, the exponent provides a smooth interpolation between semantic and collaborative ranking, driven by the query-type classifier.

4 Taxonomy and Labels

To determine when strong personalization is appropriate, we manually examine user queries and classify them into 19 categories (see the Appendix B). We further differentiate between categories for which the playlist should depend on preferences of the user and those for which it should primarily reflect external constraints. Data were sourced from the *Zvuk.com* music streaming platform. We sampled no more than two queries per user and processed all logs in anonymized form.

Nine categories (e.g., mood, activity and etc.) are treated as *personalized*, while the remaining ten categories (e.g., historical or cultural events, explicit artists and etc) are treated as *non-personalized* (see Table 3). Then, we annotate 5,000 real user queries with this taxonomy. After annotation, the query distribution comprised 42% personalized requests, while 58% did not require personalization.

For dynamic personalization we collapse the taxonomy into a binary label $y \in \{0,1\}$, where $y = 1$ denotes that the query should be strongly personalized (belongs to one of the personalized categories) and $y = 0$ denotes that the query should mostly follow the textual description.

5 Experimental Setup and Results

We evaluate the proposed approach in an internal blind user study (please refer to the Appendix C for details) and compares different model variants.

5.1 Model Variants

We instantiate the general architecture in several concrete model variants, all of which are built on a `Qwen2.5-7B`–based (Qwen, 2024) semantic retriever.

In the first variant, final ranking are performed exclusively by the LLM retriever, such that the ordering is determined solely by $s_{\text{llm}}(x,t)$. The second variant retains the same retriever but uses it to generate a candidate set of size $N = 500$, which is subsequently re-ranked in a collaborative manner using $s_{cf}(u,t)$; the two signals are combined via

Model	#, M	Acc	Macro-F1	ROC-AUC
RuBERT	178	83.1	81.2	88.31
RuRoBERTa	355	84.13	82.2	89.5
SBERT	400	86.2	85.02	91.33
RuModernBERT	150	**87.55**	**86.7**	**92.67**

Table 1: Performance of query-type classifiers on the test set.

Model	Wins	ELO
Retriever	37	1382.10
Retriever + CR	73	1525.54
Retriever + CR + DP	125	1592.37

Table 2: Number of wins and ELO ratings for the non-personalized baseline (Retriever), the fully personalized setting applied to all queries (Retriever + CR), and the dynamic-personalization approach (Retriever + CR + DP).

the fixed-exponent fusion defined in Eq.(2). The third variant follows the same two-stage candidate generation with $N = 500$ and collaborative scoring, but additionally incorporates a query-type classifier that predicts $\alpha(x)$, and the final ranking is obtained using the query-dependent fusion in Eq.(4).

These variants enable an ablation-style analysis of the contributions of collaborative re-ranking, and query-type-aware dynamic personalization within a unified retrieval framework.

5.2 Query-type classification.

We fine-tune four Russiaan-language transformer encoders as query-type classifiers, namely `RuBERT-base`, `RuRoBERTa-large` (Zmitrovich et al., 2024), `sbert_large_nlu_ru` (Abramov et al., 2024) and `RuModernBERT-base` (Spirin et al., 2025), since Russian is the primary language of user queries on the streaming platform.

Each model takes the tokenized query x as input, encodes it into a contextual representation, and applies a linear classification head with a softmax over the two classes. We split the 5,000 labeled queries into nto training, validation, and test sets in a 70%, 20%, and 10% ratio, respectively, and train with cross-entropy loss. Training details (batch size 200, learning rate $5e$-4, number of epochs $2e3$) follow standard fine-tuning practice.

Table 1 summarizes the performance of the four models on the held-out test set. Due to architectural modifications and the pretraining design, `RuModernBERT-base` is expected to outperform the other models (Spirin et al., 2025). Consequently, we select the `RuModernBERT-base` to compute $\alpha(x)$ for the internal blind user study stage.

5.3 User study results

In total, 20 unique users participated in the study and submitted 254 pairwise judgments.

Each comparison corresponds to a single query and a single model pair. We aggregate the pairwise outcomes using an ELO-style rating system (Olesker-Taylor and Zanetti, 2024) and complement this with non-parametric significance tests (Mann–Whitney) (Groggel, 2000) where appropriate. Table 2 reports the number of wins and the resulting ELO ratings. Among the results, there were 19 cases in which users judged the two playlists as equivalent.

The dynamically personalized variant wins 125 comparisons (53.2%) versus 37 wins (15.7%) for the retriever-only solution and 73 wins (31.1%) for the the fully personalized setting. The difference in ELO ratings of more than 143.44 points indicates a substantial advantage of personalization over the non-personalized baseline. Query-type-aware dynamic fusion delivers additional benefits, outperforming both the non-personalized baseline and the always-personalized configuration. Non-parametric tests confirm that the difference in pairwise preferences is statistically significant ($p < 0.05$).

Conclusion

We presented a query-type-aware dynamic personalization mechanism for LLM-based playlist generation. By training a classifier on a manually constructed taxonomy of user queries and using its output probability as an exponent for the collaborative score, the system can smoothly interpolate between purely semantic and strongly personalized rankings. Combined with query expansion from likes and dislikes, this approach improves user preferences in a blind pairwise study compared to non-personalized and fixed-personalization baselines. Future work includes richer multi-label query understanding, user-controllable personalization sliders, and automatic learning of personalization schedules from interaction data.

Discussion and Limitations

Any personalized recommender may reinforce existing preferences and reduce diversity, thereby lim-

iting exposure to novel content. Dynamic personalization partially mitigates this effect by attenuating the contribution of the collaborative component for many non-personalized queries. In designing and training the pipeline, we focused on a single language that is predominant on the streaming platform. However, extending the approach to multilingual settings may improve robustness and reliability (Pletenev et al., 2025).

Ethics

Our study relies on user testing and crowdsourced data annotation conducted by paid contributors through the streaming platform's internal service. Participation in the user testing was strictly voluntary and anonymized. Contributors were compensated at rates above the average level to ensure fair remuneration for their time and effort. This approach reflects our commitment to labor ethics and our respect for the value of human contributions to AI research.

References

Davide Abbattista, Vito Walter Anelli, Tommaso Di Noia, Craig MacDonald, and Aleksandr Vladimirovich Petrov. 2024. Enhancing sequential music recommendation with personalized popularity awareness. In *Proceedings of the 18th ACM Conference on Recommender Systems, RecSys 2024, Bari, Italy, October 14-18, 2024*, pages 1168–1173. ACM.

Aleksandr Abramov, Denis Antykhov, and Ibragim Badertdinov. 2024. Bert large model (uncased) for sentence embeddings in russian language.

Yashar Deldjoo, Markus Schedl, and Peter Knees. 2024. Content-driven music recommendation: Evolution, state of the art, and challenges. *Comput. Sci. Rev.*, 51:100618.

David J. Groggel. 2000. Practical nonparametric statistics. *Technometrics*, 42(3):317–318.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to information retrieval*. Cambridge University Press.

Chunyan Mao, Shuaishuai Huang, Mingxiu Sui, Haowei Yang, and Xueshe Wang. 2024. Analysis and design of a personalized recommendation system based on a dynamic user interest model. *CoRR*, abs/2410.09923.

Sam Olesker-Taylor and Luca Zanetti. 2024. An analysis of elo rating systems via markov chains. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

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

Sergey Pletenev, Maria Marina, Nikolay Ivanov, Daria Galimzianova, Nikita Krayko, Mikhail Salnikov, Vasily Konovalov, Alexander Panchenko, and Viktor Moskvoretskii. 2025. Will it still be true tomorrow? multilingual evergreen question classification to improve trustworthy QA. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 8614–8631, Suzhou, China. Association for Computational Linguistics.

David Preda. 2023. Llmusic: An llm-based playlist generator connected to spotify.

Team Qwen. 2024. Qwen2.5: A party of foundation models.

Egor Spirin, Boris Malashenko, and Sokolov Andrey. 2025. Rumodernbert: Modernized bert for russian.

Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, Hui Xiong, and Enhong Chen. 2024. A survey on large language models for recommendation. *World Wide Web (WWW)*, 27(5):60.

Terence Zeng and Abhishek K. Umrawal. 2025. Content filtering methods for music recommendation: A review. *CoRR*, abs/2507.02282.

Zhaodian Zeng, Yining Wang, Yanbin Zhao, and Wenxuan Shi. 2024. A survey of music recommendation systems. In *Proceedings of the 5th International Conference on Computer Information and Big Data Applications*, CIBDA '24, page 507–519, New York, NY, USA. Association for Computing Machinery.

Dmitry Zmitrovich, Aleksandr Abramov, Andrey Kalmykov, Vitaly Kadulin, Maria Tikhonova, Ekaterina Taktasheva, Danil Astafurov, Mark Baushenko, Artem Snegirev, Tatiana Shavrina, Sergei S. Markov, Vladislav Mikhailov, and Alena Fenogenova. 2024. A family of pretrained transformer language models for russian. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation, LREC/COLING 2024, 20-25 May, 2024, Torino, Italy*, pages 507–524. ELRA and ICCL.

A Query Expansion from Likes and Dislikes

To better align the semantic search with long-term preferences of the user, we expand the query representation using embeddings of tracks that the user has explicitly liked or disliked.

Let $\mathbf{v}_{\text{pos}}(u)$ be the average embedding of tracks that user u has liked, and $\mathbf{v}_{\text{neg}}(u)$ the average embedding of tracks that the user has disliked or hidden. We introduce three non-negative scalar weights:

- q — weight of the original query embedding;
- p — weight of positive history (likes);
- n — weight of negative history (dislikes).

These weights are controlled in the range of $[0, 1]$.

We form an expanded query vector

$$\mathbf{v}_{\text{exp}}(u, x) = \frac{q\,\mathbf{v}_{\text{query}}(x) + p\,\mathbf{v}_{\text{pos}}(u) - n\,\mathbf{v}_{\text{neg}}(u)}{q + p - n}, \tag{6}$$

assuming $q + p - n > 0$. Here the negative sign in front of $n \cdot \mathbf{v}_{\text{neg}}(u)$ explicitly pushes the expanded query away from regions of the space corresponding to disliked content.

B Queries Taxonomy

Table 3 presents the query taxonomy used in our analysis, separating user requests into personalized and non-personalized types. Personalized queries capture situations where the appropriate response is expected to depend on the individual listener (e.g., mood, activities, life events, travel or romantic context, and explicit taste-dependent requests). Non-personalized queries cover requests that can be satisfied largely independently of user identity, including cultural references, known artists/genres, trends and charts, "in the style of" prompts, children's content, media soundtracks, concrete musical attributes, references to third parties or public venues, and noisy or malformed inputs. Each category is illustrated with a representative example query.

C User Study Interface

The study uses a web-based interface integrated into the music streaming platform. Each participant provides their platform identifier (u) and enters a free-form text query x in their native language. For each evaluation trial, the system samples a pair of models from a predefined comparison set and generates two anonymized playlists:

- Playlist A: output of model M_a for (u, x).
- Playlist B: output of model M_b for (u, x).

The interface does not reveal which underlying model produced which playlist. Participants see only the track lists and can interact with them (e.g., by inspecting track titles and artists). For each pair they are asked to choose one of three options:

1. Playlist A is better.
2. Playlist B is better.
3. Both playlists are equivalent.

Each response is recorded as a pairwise outcome for the corresponding models (M_a, M_b). A single user can submit multiple queries; queries and model pairs are randomized across trials.

Type	Category	Example
Personalized	Emotional state	*for a melancholic mood*
	Sports and physical activity	*for running*
	Home leisure	*for a cozy evening at home*
	Work and study	*for concentration at work*
	Life situation	*for exam preparation*
	Travel	*for a night drive through the city*
	Fantasy situation	*to imagine yourself as Marilyn Monroe*
	Romantic context	*for a romantic mood*
	Explicit taste-dependent requests	*punk rock for an evening mosh that I will like*
Non-personalized	Cultural or historical context	*Victory Day*
	Existing genres, styles, artists, or tracks	*the most popular song of Louis Armstrong*
	Existing trends and charts	*TikTok trends 2025*
	"In the style of X" requests	*songs in the style of the Beatles*
	Children's content	*for a children's morning party*
	Soundtracks and media	*songs from the cartoon N*
	Concrete musical characteristics	*guitar in djent style*
	Queries referencing other people or groups	*for a party with my father-in-law*
	Public places	*for a coffee shop*
	Noise or random strings	*fjdhk.!!668!*

Table 3: Categories of personalized and non-personalized queries with examples.