LLM-Based Intelligent Agents for Music Recommendation: A Comparison with Classical Content-Based Filtering

Ronald Carvalho Boadana[1], **Ademir Guimarães da Costa Junior**[1]
Ricardo Rios[1], **Fábio Santos da Silva**[1]

[1]Escola Superior de Tecnologia - Universidade do Estado do Amazonas (UEA)
Manaus - AM - Brasil

{rcb.cid25, agdcj.cid25, rrios, fssilva}@uea.edu.br

Abstract. *The growing availability of music on streaming platforms has led to information overload for users. To address this issue and enhance the user experience, increasingly sophisticated recommendation systems have been proposed. This work investigates the use of Large Language Models (LLMs) from the Gemini and LLaMA families, combined with intelligent agents, in a multi-agent personalized music recommendation system. The results are compared with a traditional content-based recommendation model, considering user satisfaction, novelty, and computational efficiency. LLMs achieved satisfaction rates of up to 89,32%, indicating their promising potential in music recommendation systems.*

Resumo. *A crescente oferta de músicas nas plataformas de streaming tem causado sobrecarga de informação aos usuários. Para mitigar esse problema e aprimorar a experiência, sistemas de recomendação mais sofisticados têm sido propostos. Este trabalho investiga o uso de Modelos de Linguagem de Grande Escala (LLMs) das famílias Gemini e LLaMA, aliados a agentes inteligentes, em um sistema de recomendação musical personalizado multiagente. Os resultados são comparados a um modelo tradicional baseado em conteúdo, considerando satisfação do usuário, novidade e eficiência computacional. Os LLMs atingiram até 89,32% de satisfação, indicando seu potencial promissor nos sistemas de recomendação de músicas.*

1. Introdução

A música é uma das mais antigas formas de manifestação artística do ser humano. Desde os primórdios, ela desempenha um papel fundamental na expressão e na comunicação de sentimentos, emoções, opiniões, ideias etc. Devido à sua capacidade de gerar reações como, por exemplo, felicidade, tristeza ou nostalgia, torna a música um meio poderoso de conexão humana. Além disso, ela funciona como um veículo de transmissão de conhecimento, registro e preservação de histórias, permitindo que tradições e memórias sejam compartilhadas entre gerações e culturas ao longo do tempo.

Com o avanço da tecnologia e com a popularização da *Internet*, a forma como as pessoas consomem música mudou radicalmente. Serviços de *streaming* como, por exemplo, *Spotify*®, *Apple Music*®, *Deezer*®e *Amazon Music*®, tornaram-se as principais plataformas de distribuição musical, oferecendo aos usuários acesso rápido a catálogos com milhões de faixas. Essa grande variedade de músicas fez com que a escolha de uma música se tornasse um desafio devido à sobrecarga de informação [Fouad et al. 2025].

Para minorar essa sobrecarga, sistemas de recomendação têm se tornado ferramentas essenciais para personalizar a experiência do usuário, facilitando assim a descoberta de músicas alinhadas aos seus gostos e preferências.

Em trabalhos recentes, diversas técnicas e mecanismos de descoberta e filtragem de músicas têm sido explorados com o intuito de melhorar a experiência do usuário e minorar a sobrecarga de informação [Fouad et al. 2025]. No entanto, tais abordagens enfrentam limitações importantes como, por exemplo, a dificuldade de fazer recomendações para usuários que não possuem um histórico de músicas ouvidas (*cold start*) [Maheshwari 2023].

Nesse cenário, este trabalho propõe uma abordagem baseada em agentes inteligentes construídos sobre Modelos de Linguagem de Grande Escala (*LLMs — Large Language Models*), como o *Gemini* [Gemini Team and Google DeepMind 2023] e o *LLaMA* [Grattafiori et al. 2024], visando explorar sua capacidade de compreender linguagem natural e capturar relações contextuais e semânticas profundas. A proposta se estrutura em uma arquitetura modular composta por agentes especializados, o que favorece a flexibilidade, escalabilidade e reprodutibilidade do sistema de recomendação. Além disso, a utilização de dados reais e a avaliação com usuários finais conferem maior validade prática aos resultados obtidos. Com isso, este trabalho busca investigar não apenas as potencialidades e limitações dessa abordagem, mas também seu impacto direto na experiência do usuário em sistemas de recomendação musical.

Esse artigo está organizado como segue. A Seção 2 apresenta os trabalhos relacionados. A Seção 3 detalha a metodologia utilizada no trabalho. A Seção 4 apresenta os resultados e uma discussão dos experimentos realizados. A conclusão do trabalho é apresentada na Seção 5.

2. Trabalhos Relacionados

A recomendação musical tem sido tradicionalmente realizada por meio de abordagens em filtragem colaborativa e filtragem baseada em conteúdo. A filtragem colaborativa infere preferências a partir de padrões de comportamento semelhantes de usuários [Sarwar et al. 2001]. Embora eficiente, essa técnica não consegue tratar o conhecido problema do *cold start*, o que dificulta a recomendação de itens para novos usuários. Para lidar com isso, alguns sistemas baseados em conteúdo utilizam metadados musicais como, por exemplo, gênero, ritmo e instrumentação, para recomendar itens de interesse do usuário [Schedl et al. 2014]. Embora essa abordagem seja menos suscetível ao *cold start*, ela tende a gerar recomendações mais homogêneas, o que limita a diversidade de itens recomendados.

Além dessas técnicas, métodos mais recentes incorporam aprendizado profundo para aprimorar a precisão das recomendações. Em [Nguyen et al. 2024], é proposto um sistema de recomendação baseado na análise de emoções faciais e da emoção que a música possibilita. Esse estudo avaliou o uso de Redes Neurais Convolucionais (CNNs — *Convolutional Neural Networks*) para a classificação das emoções do usuário a partir de imagens. A classificação a emoção que a música possibilita é feita utilizando Máquinas de Vetores de Suporte (SVM — *Support Vector Machine*). Apesar de apresentar resultados promissores, o modelo possui limitações como, por exemplo, a dificuldade de aplicação em tempo real e a incapacidade de considerar fatores individuais, como idade, gênero e

histórico de consumo do usuário.

Em [Yang 2022], é apresentado um modelo de recomendação baseado em características vocais das canções. O sistema analisa o áudio das músicas para extrair informações relevantes sobre vozes e sons. As características extraídas são utilizadas como entrada de modelos convolucionais para treinamento e classificação em gêneros musicais. O resultado da classificação é combinado com o histórico de consumo do usuário para fazer as recomendações. Entretanto, o classificador proposto é ineficaz em classificar músicas em seus respectivos gêneros.

Nos últimos anos, a incorporação de LLMs aos sistemas de recomendação abriu novas possibilidades, principalmente pela capacidade dessas arquiteturas em modelar dependências contextuais e compreender nuances semânticas. Estudos como, por exemplo, [Sun et al. 2019] e [Kang and McAuley 2018] demonstraram a eficácia de arquiteturas baseadas em *Transformer*, como o BERT4Rec, no sequenciamento de interações de usuários para prever preferências futuras. Embora esses métodos tenham sido inicialmente desenvolvidos para sistemas de recomendação gerais, eles inspiraram adaptações específicas no contexto musical, como o uso de *embeddings* derivados de textos descritivos de faixas e artistas [Vagliano et al. 2018].

O estudo [Vagliano et al. 2018] propõe o uso de *autoencoders* adversariais para a continuação automática de *playlists*. Para isso, combina múltiplas fontes de dados para enriquecer os *embeddings* musicais e melhorar a qualidade das recomendações. Os resultados evidenciam que o uso de *embeddings* multimodais melhora a capacidade de prever músicas adequadas para serem adicionadas a *playlists*, especialmente quando as sequências são curtas ou incompletas.

Este trabalho se difere dos anteriores, especialmente, pelo uso de agentes inteligentes baseados em recomendação de músicas, comparando com o método de filtragem baseada em conteúdo amplamente utilizada em sistemas de recomendação de música.

3. Material e Métodos

Para realizar os experimentos foi feita uma coleta de dados de usuários em tempo real. Para isso, optou-se por utilizar a API pública do *Spotify*®. A coleta ocorreu ao longo de 13 meses, com a participação de usuários voluntários que consentiram com o uso de seus dados para fins de pesquisa. O consentimento foi acordado considerando os princípios éticos de privacidade e confidencialidade. Após a coleta, os dados foram organizados em uma base estruturada, que contém um total de 19 usuários e 22.178 faixas únicas.

3.1. Configuração do Ambiente de Experimentos

Para realizar os experimentos optou-se por utilizar recursos computacionais em nuvem. A API de recomendação foi desenvolvida em Python e Django a um custo de US$2 e foi hospedada na Amazon AWS [AWS 2025] em uma instância *t3.small* (2 vCPUs e 2GB RAM) e era executada em um container Docker [Docker 2025]. Essa API recebe dados do MongoDB [MongoDB 2025], envia-os para agentes de IA (Inteligência Artificial) e retorna as recomendações como resultado. Os agentes utilizam os modelos LLaMA 3.3, via API da Groq, e Gemini 2.0 Flash, via API do Google. Os modelos não foram personalizados utilizando qualquer treinamento específico. As recomendações são geradas dinamicamente a cada requisição.

O *front-end*, desenvolvimento em React e TypeScript e hospedado gratuitamente no Firebase, exibe as recomendações e coleta as avaliações feitas pelos usuários. O *back-end*, desenvolvido em Node.js com TypeScript, atua como intermediário entre o *front-end* e a API de recomendação. O *back-end*, hospedado no Render a um custo de US$ 2, gerencia as requisições recebidas do *front-end* e envia para o banco de dados (MongoDB).

3.2. Dataset

O conjunto de dados coletados inclui dados como: nome da música, artista(s), álbum, duração da faixa, presença de conteúdo explícito, data de lançamento e data de reprodução pelo usuário. Nesse trabalho foram considerados apenas três dados: nome da música, nome do artista principal e o gênero musical. A identificação do gênero foi realizada via requisições adicionais à API do *Spotify*. A base foi dividida em dois subconjuntos principais:

1. **Catálogo musical:** composto por todas as faixas únicas identificadas na amostra;
2. **Histórico musical dos usuários:** registros das músicas efetivamente ouvidas por cada participante.

Considerando as limitações impostas pela quantidade de *tokens* suportados pelos modelos de LLMs, foi realizada uma amostragem controlada. Foram selecionados os 20 gêneros musicais mais recorrentes na base original. Em seguida, utilizou-se a função `sample()` da biblioteca `Pandas` para gerar um catálogo musical com 300 faixas. Para o histórico de cada usuário, foram selecionadas as 30 músicas mais reproduzidas durante o período analisado. A Tabela 1 apresenta uma amostragem dos dados de catálogo musical.

Tabela 1. Amostragem do catálogo musical

Nome da Música	Artista (s)	Gêneros
Decode	Sabrina Carpenter	Pop
Permission to Dance	BTS	K-Pop, K-Pop Boy Group, Pop
Bicycle Song - 2006 Remaster	Red Hot Chili Peppers	Alternative Rock, Funk Metal, Funk Rock, Permanent Wave, Rock

3.3. Método Tradicional de Filtragem Baseada em Conteúdo

O método tradicional, utilizado como *benchmark*, consiste em um sistema de recomendação de músicas que se baseia em filtragem por conteúdo. Similarmente às implementações de [Fiarni and Maharani 2019], ele calcula a similaridade de cosseno entre vetores de gêneros musicais (representados por TF-IDF), uma técnica também utilizada em [Singh et al. 2020, Falah and Suryawan 2022].

Inicialmente, o sistema faz requisições a uma *API REST* para coletar ambos os conjuntos de dados: o histórico de músicas consumidas por um usuário; e o catálogo geral de músicas disponíveis. Em seguida, os gêneros de todas as músicas do catálogo e os gêneros preferidos do usuário são transformados em vetores numéricos por meio da técnica TF-IDF.

Com os vetores gerados, é aplicada a métrica de similaridade de cosseno (eq. 1) para calcular o grau de semelhança entre os gêneros preferidos do usuário e os gêneros

presentes no catálogo. O sistema seleciona as 20 músicas mais próximas aos cinco gêneros mais consumidos pelo usuário.

$$\text{similaridade}(x, y) = cos(\theta) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2}\sqrt{\sum_i y_i^2}} \quad (1)$$

3.4. Método de Filtragem de Conteúdo baseado em Agentes Inteligentes

O processo de recomendação foi conduzido por meio de agentes inteligentes baseados em LLMs, implementados com a ajuda do framework CrewAI [CrewAI 2025], que permite a criação de arquiteturas multiagente colaborativas. Nesse contexto, agentes inteligentes podem ser compreendidos como programas que utilizam técnicas de inteligência artificial para executar tarefas que normalmente exigiriam intervenção humana [Russell and Norvig 2022], sendo capazes de operar de forma autônoma e adaptativa.

Ao invés de um único *prompt* genérico, a recomendação foi dividida em subtarefas executadas por agentes especializados, cada um com objetivos bem definidos como, por exemplo:

- **ReadingAgt**: Leitura e compreensão do catálogo musical;
- **AnalistAgt**: Análise do histórico de reprodução do usuário;
- **ExtractAgt**: Extração de padrões de preferência com base em gêneros e artistas;
- **RecommendAgt**: Geração da lista final de recomendações.

Esses agentes foram configurados para atuar de forma independente, mas colaborativa, compartilhando informações intermediárias. Toda comunicação entre os agentes e a base de dados foi realizada via APIs REST, garantindo reprodutibilidade e modularidade no processo, conforme a Figura 1.

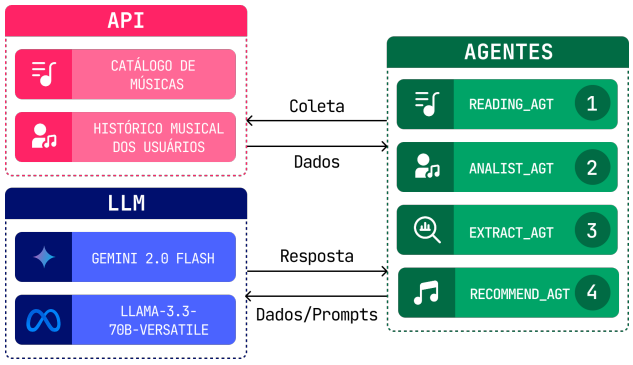

Figura 1. Fluxo do sistema de recomendação proposto

3.4.1. Especificação dos Agentes e *Prompts*

Os *prompts* empregados nos agentes seguem a abordagem *zero-shot*, na qual os modelos de linguagem realizam inferências diretamente, sem a necessidade de exemplos explícitos no próprio *prompt*. Esse tipo de configuração busca avaliar a capacidade dos LLMs em gerar recomendações com base na descrição textual do histórico musical do usuário e do catálogo de faixas disponíveis [Sun et al. 2019, Vagliano et al. 2018].

Nas tabelas de 2 a 4, são apresentados os *prompts* utilizados na construção dos agentes, bem como as ferramentas e tarefas atribuídas a cada um. A Tabela 2 apresenta as ferramentas utilizadas pelos agentes **ReadingAgt** e **AnalistAgt**, respectivamente. Essas ferramentas foram desenvolvidas utilizando o *CrewAI Tools* [CrewAI 2025] e consistem em funções que realizam requisições à API, permitindo o retorno dos dados necessários para os agentes.

Tabela 2. Ferramentas utilizadas e suas respectivas descrições

Ferramenta	Descrição
`GetUserHistory DataTool`	*Fetches the music data from a given URL and returns it as a list of dictionaries.*
`GetMusicCatalogue Tool`	*Fetches the user listening history from a given URL and returns the first 30 items as a list.*

A Tabela 3 apresenta os agentes previamente introduzidos, acompanhados de suas respectivas descrições de objetivo e contexto. A descrição de objetivo serve como diretriz principal, orientando as decisões e ações de cada agente ao longo do processo de recomendação. O contexto fornece informações adicionais que moldam a personalidade, o comportamento e a forma de atuação dos agentes [CrewAI 2025].

Tabela 3. Agentes utilizados, objetivos e ferramentas associadas

Agente	Objetivo (Goal)	Contexto (Backstory)	Ferramenta (Tool)
`ReadingAgt`	*Read all the songs from a catalogue.*	*Specializes in handling and returning a song catalogue.*	`GetMusic CatalogueTool`
`AnalistAgt`	*Read all a music history from an user.*	*Specializes in handling and returning a music history.*	`GetUserHistory DataTool`
`ExtractAgt`	*Inferring the user's favorite music genre.*	*Specializes in analysing the user music history to infer their 5 favorite music genres.*	-
`RecommendAgt`	*Recommend songs to a user using their listening histories.*	*You are a personalized music recommender. You analyze song genres and recommend tracks using content-based filtering techniques.*	-

A Tabela 4 apresenta a estruturação dos *prompts* utilizados nas tarefas, organizados de forma a guiar a execução passo a passo do processo de recomendação. Essa organização visa garantir clareza, coesão e eficiência na atuação de cada agente dentro do sistema. As tarefas, por sua vez, correspondem a atribuições específicas designadas aos agentes, contendo todas as informações necessárias para sua execução, como uma descrição detalhada, o agente responsável, as ferramentas requeridas, entre outros elementos essenciais que viabilizam a correta execução das ações [CrewAI 2025].

3.4.2. Seleção dos Modelos

Para a implementação do sistema de recomendação com agentes baseados em LLMs, a seleção dos modelos subjacentes foi guiada por critérios de capacidade técnica e viabilidade operacional, especialmente no contexto de acesso e custo. Foram avaliados o Gemini 2.0 Flash [Google 2025] e o LLaMA-3.3-70B-VERSATILE [Groq 2025] como os LLMs principais para as inferências dos agentes.

Tabela 4. Tarefas utilizadas no sistema

Descrição	Agente	Saída Esperada
Read and return all the song catalogue at this URL: `/getAllDataEniac?limit=300`.	`Song Catalogue Reader`	*Song Catalogue*
Read and return the user music history at this URL: `/getUserData/{user_id}`	`User Music History Reader`	*User listening history*
Infer the user's favorite music genres. Use the user's listening history to identify their 5 most preferred music genres.	`User Music Genres`	*User Music Genres*
Generate a list of 20 recommended songs from the catalogue. Use the user's listening history and inferred genres to build a personalized profile. Select songs that match the user's musical preferences. Return a JSON list with `genre`*,* `song_name` *and* `artist_name`*, along with* `liked` *and* `known` *flags.*	`Content-Based Music Recommender`	*Recommended Songs for the user*

O Gemini 2.0 Flash foi escolhido principalmente devido à sua capacidade de processar um elevado número de *tokens* em sua versão gratuita, oferecendo uma janela de contexto de até 1 milhão de *tokens*. Essa característica se mostrou decisiva para o projeto, pois permitiu o processamento de catálogos musicais extensos e históricos de reprodução detalhados dos usuários sem incorrer em custos adicionais significativos durante a fase de desenvolvimento e experimentação.

O modelo LLaMA-3.3-70B-VERSATILE foi escolhido devido à sua facilidade de acesso e custo-benefício através do serviço sob demanda da plataforma *Groq*, na qual permite o uso de LLMs aceleradas por meio de acesso de API [Groq 2025]. Apesar de não apresentar uma janela de *tokens* tão elevada quanto a versão gratuita do Gemini Flash, o custo acessível do serviço viabilizou a utilização de um modelo de grande porte (70 bilhões de parâmetros) para as operações de inferência mais complexas como, por exemplo, a análise de histórico e a inferência de padrões de preferência. A escolha de um modelo com 70B de parâmetros subentende a busca por uma maior capacidade de compreensão e geração de linguagem natural, permitindo aos agentes capturar nuances contextuais e semânticas mais profundas necessárias para uma recomendação musical sofisticada, mesmo que a velocidade e o custo tenham sido fatores mais diretos na decisão de acesso via *Groq*.

Ambos os modelos foram acessados por meio de chaves de API, o Gemini foi acessado via API proprietária, enquanto o LLaMA foi acessado via serviço sob demanda oferecido pela plataforma *Groq*.

3.5. Método de Avaliação

Foi desenvolvida uma interface específica para que os usuários interagissem com as *playlists* geradas pelos diferentes métodos de recomendação. Cada participante teve acesso às três *playlists*, compostas com 10 faixas recomendadas, correspondentes aos métodos avaliados (tradicional, LLaMA e Gemini), de forma cega, e realizou a avaliação de cada uma individualmente. A análise foi conduzida com base nos seguintes critérios:

1. Se gostou da música recomendada (resposta binária: Gostei / Não gostei);
2. Se já conhecia a faixa apresentada (resposta binária: Sim / Não);

3. Avaliação geral da *playlist*, em uma escala ordinal de 0 a 10, sendo 0 equivalente a ”péssimo”e 10 a ”ótimo”.

Com as respostas coletadas, são calculadas as seguintes métricas de avaliação, as quais serão usadas para comparação de desempenho dos diferentes modelos na tarefa de recomendação de músicas.

Taxa de Apreciação (*Like Rate*). Define-se como a proporção de músicas curtidas pelo usuário:

$$\text{LR} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(\text{like}_i = 1) \tag{2}$$

onde N é o número total de músicas avaliadas pelo usuário na *playlist*, e $\mathbb{I}$ é a função indicadora.

Taxa de Descoberta (*Novelty Rate*). Corresponde à fração de músicas não previamente conhecidas pelo usuário:

$$\text{NR} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(\text{known}_i = 0) \tag{3}$$

Taxa de Descobertas Bem-Sucedidas (*Successful Novelty Rate*). Proporção de músicas *novas* que foram curtidas, refletindo a eficácia da recomendação de faixas inéditas:

$$\text{SNR} = \frac{\sum_{i=1}^{N} \mathbb{I}(\text{known}_i = 0 \wedge \text{like}_i = 1)}{\sum_{i=1}^{N} \mathbb{I}(\text{known}_i = 0)} \tag{4}$$

Essa métrica é condicional à existência de pelo menos uma música nova na *playlist*.

Nota da Playlist (*Playlist Rating*). Refere-se à nota atribuída pelo usuário à *playlist* como um todo, após ouvir as dez faixas. Essa é uma medida subjetiva global da qualidade percebida da recomendação.

As métricas descritas permitem avaliar não apenas a eficácia da recomendação em termos de apreciação geral, mas também sua capacidade de promover descobertas relevantes aos usuários.

4. Resultados e Discussão

Os experimentos seguiram o protocolo descrito na Seção 3. A Tabela 5 e a Figura 2 apresentam os resultados médios das avaliações feitas por dez usuários, de forma cega, expressos em termos de média e desvio padrão. Considerando exclusivamente a métrica

de *Rating*, que reflete a avaliação subjetiva global da qualidade da playlist, observa-se uma diferença clara entre os modelos.

Tabela 5. Métricas de avaliação dos modelos em termos de média e desvio padrão.

Modelo	LR (%)	NR (%)	SNR (%)	Rating (1–10)	Tempo de Inferência (s)
Tradicional	$61{,}00 \pm 22{,}00$	$58{,}50 \pm 22{,}48$	$21{,}00 \pm 18{,}55$	$6{,}70 \pm 3{,}37$	$1{,}37 \pm 0{,}28$
LLaMA	$89{,}32 \pm 7{,}34$	$11{,}85 \pm 8{,}76$	$3{,}17 \pm 3{,}61$	$8{,}70 \pm 2{,}45$	$84{,}07 \pm 13{,}84$
Gemini	$65{,}00 \pm 21{,}10$	$52{,}00 \pm 23{,}79$	$18{,}50 \pm 18,31$	$7{,}25 \pm 2{,}18$	$70{,}76 \pm 32{,}80$

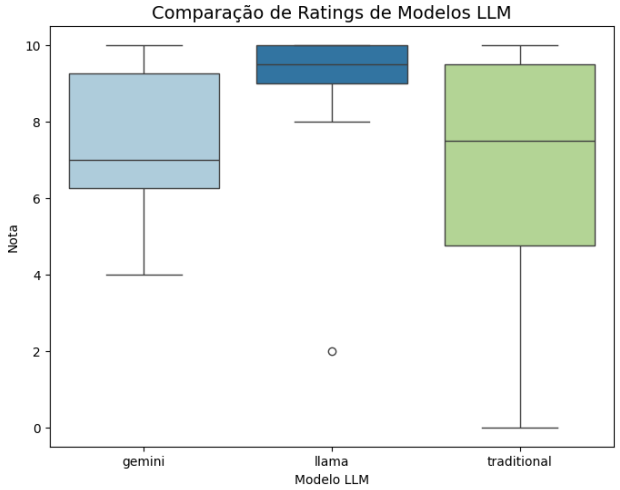

Figura 2. Boxplot da Comparação de Ratings dos Modelos LLM

A análise das métricas de desempenho revelou padrões distintos entre os modelos LLaMA, Gemini e o Tradicional, especialmente no que tange à satisfação do usuário e à capacidade de descoberta. O modelo **LLaMA** obteve a maior média de *Rating*, com $8{,}70 \pm 2{,}45$, indicando uma forte aceitação por parte dos participantes. Esse desempenho sugere que suas recomendações foram percebidas como mais coerentes, agradáveis ou relevantes em comparação às demais. Em segundo lugar, o modelo **Gemini** alcançou média de $7{,}25 \pm 2{,}18$, o que também representa uma avaliação positiva, embora com leve inferioridade em relação ao LLaMA.

Por outro lado, o modelo **Tradicional** apresentou a menor média de nota, com $6{,}70 \pm 3{,}37$. Além da menor média, o desvio padrão mais elevado sugere uma maior variabilidade na percepção dos usuários sobre suas *playlists*, indicando que, embora algumas tenham sido bem recebidas, outras foram avaliadas de forma mais crítica.

O modelo **LLaMA** também obteve a maior taxa de apreciação ($89{,}32\% \pm 7{,}34$), sugerindo que, mesmo com baixa novidade, suas recomendações estavam altamente alinhadas às preferências dos usuários. Isso pode indicar uma forte capacidade de personalização, porém possivelmente com tendência a recomendações conservadoras, já conhecidas ou semelhantes a preferências passadas. O **Gemini**, por sua vez, obteve uma taxa de apreciação intermediária ($65{,}00\%$), alinhando-se mais ao desempenho do modelo **Tradicional** ($61{,}00\%$), mas ainda com melhor aceitação geral da *playlist*.

Peculiarmente, os dados mostram uma inversão entre novidade e apreciação. O modelo **LLaMA** apresentou a menor taxa de descoberta ($11{,}85\%$), o que pode indicar

que a maioria das faixas já era conhecida pelos usuários. Em contrapartida, o modelo **Tradicional** atingiu a maior taxa de descoberta ($58{,}50\%$), seguido pelo Gemini ($52{,}00\%$). Isso sugere que, apesar de mais inovadores, esses modelos tiveram menor alinhamento com as preferências dos usuários, o que pode estar relacionado às suas avaliações mais baixas.

A métrica de descoberta bem-sucedida (SNR) revelou diferenças significativas entre os modelos. O **Tradicional** se destacou com a maior SNR ($21{,}00\%$), demonstrando sua eficácia em apresentar músicas inovadoras e apreciadas, quando comparado aos demais modelos. Esse desempenho contrasta fortemente com a do **LLaMA**, que registrou a menor SNR ($3{,}17\%$), confirmando a hipótese de que esse modelo tende a focar em faixas já conhecidas, limitando sua capacidade de exploração. O **Gemini**, por sua vez, atingiu um equilíbrio com uma SNR de $18{,}50\%$. Isso indica que ele é mais apto a introduzir novas músicas agradáveis do que o **LLaMA**, mas ainda não alcança o desempenho do **Tradicional** nesse quesito.

Além das métricas de satisfação e descoberta, a eficiência computacional dos modelos, medida pelo tempo de inferência, revelou diferenças significativas que impactam diretamente sua aplicabilidade. O modelo **Tradicional** demonstrou ser o mais rápido e eficiente, com um tempo de inferência de apenas $1{,}37 \pm 0{,}28$ seg. Essa velocidade o torna mais adequado para cenários que exigem respostas em tempo real ou processamento eficiente de grandes volumes de dados. Em contrapartida, os modelos baseados em LLMs apresentaram tempos de inferência substancialmente mais longos. O **Gemini** registrou um tempo de $70{,}76 \pm 32{,}80$ seg., enquanto o **LLaMA** foi o mais lento, com $84{,}07 \pm 13{,}84$ seg. Essa diferença de magnitude em relação ao **Tradicional** é um fator crítico. Embora o **LLaMA** tenha se destacado em termos de satisfação do usuário e apreciação, o tempo de resposta pode ser um gargalo em sistemas que necessitam de baixa latência.

5. Conclusão e Trabalhos Futuros

Este trabalho investigou e comparou o desempenho de modelos de recomendação de *playlists*, Tradicional, LLaMA e Gemini, sob as perspectivas de satisfação do usuário, capacidade de descoberta de novas músicas e eficiência computacional. Os resultados obtidos revelam um cenário complexo, onde cada abordagem apresenta vantagens e limitações distintas.

Embora os LLMs selecionados (LLaMA e Gemini) tenham a capacidade de gerar recomendações altamente personalizadas e apreciadas pelos usuários (altas notas de *Rating* do LLaMA e sua elevada taxa de apreciação), eles tendem a ser conservadores em termos de taxa de novidade. Por outro lado, o modelo Tradicional, embora com menores índices de satisfação geral, provou ser mais eficaz na introdução de descobertas bem-sucedidas (SNR). Isso sugere que, para aplicações onde a curadoria e a introdução de material novo são prioridades, as abordagens tradicionais ainda podem ser mais vantajosas.

Um ponto crítico revelado na avaliação dos modelos foi a eficiência computacional. Apesar de apresentar melhores resultados quanto à satisfação do usuário, os LLMs apresentam uma baixa eficiência computacional, contrastando com o modelo Tradicional, que opera em baixa latência. Em ambientes que exigem respostas em tempo real, a superioridade dos LLMs em termos de personalização pode ser ofuscada por suas limitações

de desempenho, tornando-os menos viáveis nessa tarefa. Para sistemas que priorizam a satisfação imediata do usuário e a personalização de preferências já estabelecidas e que toleram maior tempo de resposta, modelos como o LLaMA mostram-se promissores. Para cenários que valorizam a descoberta genuína de novos conteúdos e a eficiência computacional, as abordagens tradicionais ainda mantêm sua relevância. O Gemini, por sua vez, posiciona-se como um intermediário, pois equilibra personalização e alguma capacidade de descoberta, embora ainda com um custo computacional elevado.

Como trabalhos futuros, planejamos aprimorar a personalização das recomendações musicais por meio do uso de um maior volume de dados de catálogo e histórico musical, além da ampliação dos atributos utilizados como entrada dos modelos. Pretendemos ir além de informações básicas como nome da música, artista e gênero, incorporando também letras, sentimentos extraídos, descrições contextuais e outros elementos que possam explorar melhor as capacidades dos LLMs. Para isso, investigaremos abordagens híbridas de filtragem, combinando técnicas baseadas em conteúdo e filtragem colaborativa. Para avaliar o uso de catálogos e históricos de consumo maiores, pretendemos utilizar bancos de dados vetoriais como suporte à arquitetura do sistema e avaliar seu impacto no desempenho dos LLMs nesse cenário. Pretende-se também utilizar outros modelos como, por exemplo, Gemma, Mistral e Qwen. Por fim, será conduzida uma análise do efeito de diferentes valores do hiperparâmetro de temperatura nas respostas dos modelos, para avaliar se esse fator influencia a qualidade, diversidade e relevância das recomendações geradas.

Referências

AWS (2025). Amazon web services. `https://www.aws.amazon.com/`. Acesso em: 13 mai. 2025.

CrewAI (2025). The leading multi-agent platform. `https://www.crewai.com/`. Acesso em: 30 jun. 2025.

Docker (2025). Docker: Accelerated container application development. `https://www.docker.com/`. Acesso em: 30 jun. 2025.

Falah, Z. F. and Suryawan, F. (2022). Recommendation system to propose final project supervisors using cosine similarity matrix. *Khazanah Informatika Jurnal Ilmu Komputer dan Informatika*, 8(2).

Fiarni, C. and Maharani, H. (2019). Product recommendation system design using cosine similarity and content-based filtering methods. *IJITEE (International Journal of Information Technology and Electrical Engineering)*, 3(2):42–48.

Fouad, O., Fouad, R., Hussen, N., and Abuhadrous, I. (2025). A comprehensive review of music recommendation systems. *Adv. Sciences and Technology Journal*, 2(1):1–18.

Gemini Team and Google DeepMind (2023). Gemini: A Family of Highly Capable Multimodal Models. *arXiv preprint arXiv:2312.11805*. Acesso em: 5 jun. 2025.

Google (2025). Gemini 2.0: Flash, Flash-Lite and Pro - Google Developers Blog. `https://developers.googleblog.com/en/gemini-2-family-expands/`. Acesso em: 28 mai.2025.

Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., and et al. (2024). The llama 3 herd of models.

Groq (2025). Groq is fast ai inference. `https://www.groq.com/`. Acesso em: 30 jun. 2025.

Kang, W.-C. and McAuley, J. (2018). Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 197–206.

Maheshwari, C. (2023). Music recommendation on spotify using deep learning.

MongoDB (2025). Mongodb: The world's leading modern database — mongodb. `https://www.mongodb.com/`. Acesso em: 30 jun. 2025.

Nguyen, H., Tran, N., Ly, D., Tran, A., Nguyen, A., Vo, H., et al. (2024). A model for song recommendation based on facial emotion analysis and musical emotion. *International Journal of Intelligent Engineering & Systems*, 17(4).

Russell, J. and Norvig, P. (2022). *Artificial Intelligence - A Modern Approach*. GEN LTC, 4th edition.

Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, page 285–295, New York, NY, USA. Association for Computing Machinery.

Schedl, M., Gómez, E., Urbano, J., et al. (2014). Music information retrieval: Recent developments and applications. *Foundations and Trends® in Information Retrieval*, 8(2-3):127–261.

Singh, R. H., Inderprastha Engineering College, AKTU, Maurya, S., Tripathi, T., Narula, T., Srivastav, G., Inderprastha Engineering College, AKTU, Inderprastha Engineering College, AKTU, Inderprastha Engineering College, AKTU, and Inderprastha Engineering College, AKTU (2020). Movie recommendation system using cosine similarity and KNN. *Int. J. Eng. Adv. Technol.*, 9(5):556–559.

Sun, F., Liu, J., Wu, J., Pei, C., Lin, X., Ou, W., and Jiang, P. (2019). Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, page 1441–1450, New York, NY, USA. Association for Computing Machinery.

Vagliano, I., Galke, L., Mai, F., and Scherp, A. (2018). Using adversarial autoencoders for multi-modal automatic playlist continuation. In *Proceedings of the ACM Recommender Systems Challenge 2018*, RecSys Challenge '18, New York, NY, USA. Association for Computing Machinery.

Yang, J. (2022). Personalized song recommendation system based on vocal characteristics. *Mathematical Problems in Engineering*, 2022(1):3605728.