CRSLLM: A Hybrid Conversational Recommender System Integrating Collaborative Filtering and Large Language Models

Marcos André de Barros Prudêncio[1], **Byron Leite Dantas Bezerra**[1]

[1]Escola Politécnica de Pernambuco – Universidade de Pernambuco – Recife – PE – Brazil

prudenciomarcos10@gmail.com, byronleite@ecomp.poli.br

Abstract. *Conversational Recommender Systems (CRSs) extend traditional recommender systems by incorporating information obtained through user interactions. This work proposes CRSLLM, a hybrid architecture that integrates collaborative filtering and LLMs for automatic user profile extraction, zero-shot recommendation, and contextual recommendation refinement. The system employs the EASE algorithm as its recommendation engine and ChatGPT (gpt-4o-mini) as the language model, prioritizing a modular and computationally efficient architecture. Experiments conducted on the ReDial dataset demonstrated high effectiveness in extracting user preferences and improvements in Recall@K and NDCG@K compared with the baseline collaborative filtering method, while achieving competitive performance against recent LLM-based approaches.*

Resumo. *Sistemas de Recomendação Conversacionais ampliam os sistemas de recomendação tradicionais ao incorporar informações obtidas durante a interação com o usuário. Este trabalho propõe o CRSLLM, uma arquitetura híbrida que integra filtragem colaborativa e LLMs para extração automática do perfil do usuário, geração zero-shot e refinamento contextual das recomendações. O sistema utiliza o algoritmo EASE como núcleo de recomendação e instâncias do ChatGPT (gpt-4o-mini), priorizando uma arquitetura modular e de baixa complexidade computacional. Os experimentos na base ReDial demonstraram alta efetividade na extração de preferências e melhorias nas métricas Recall@K e NDCG@K em relação ao método base, além de desempenho competitivo frente a abordagens recentes baseadas em LLMs.*

1. Introduction

Sistemas de recomendação (RS, do inglês Recommender System) se tornaram indispensáveis em plataformas digitais, como serviços de música, e-commerce e streaming de filmes, personalizando a experiência do usuário [Roy and Dutta 2022]. RSs tradicionais são eficazes em usar dados históricos de usuários e itens para fazer recomendações. Já sistemas de recomendação conversacionais (CRS, do inglês Conversational RS) incluem a interação entre usuário e máquina, tornando o processo de recomendação mais interativo [Sun and Zhang 2018, Christakopoulou et al. 2016]. Um CRS pode, por exemplo, capturar o contexto atual do usuário em um diálogo, a fim de refinar suas recomendações.

A interação entre usuário e máquina via linguagem natural (LN) gera oportunidades para a integração de CRSs com serviços externos como, por exemplo, os modelos de linguagem de larga escala (LLMs, do inglês Large Language Models)

[He et al. 2023, Zhu et al. 2025a, Zhu et al. 2025b], buscando explorar a capacidade desses modelos de compreender e gerar texto, mantendo diálogos coesos. Estudos sobre essa integração são relativamente recentes e crescentes em relevância, cobrindo aplicações práticas em assistentes virtuais, plataformas de streaming e sistemas de e-commerce, nos quais os usuários podem expressar preferências em linguagem natural [Wang et al. 2023].

Apesar dos avanços, a integração de LLMs a técnicas de recomendação ainda representa um desafio, uma vez que dados de interação entre usuários, itens e sistema, fundamentais para algoritmos de filtragem colaborativa, não fazem parte dos corpora de treinamento dos LLMs e nem sempre podem ser representados de forma eficaz em linguagem natural [Zhao et al. 2023]. Nesse contexto, o objetivo geral deste trabalho foi investigar essa integração, desenvolvendo um CRS híbrido que combina filtragem colaborativa com LLMs para auxílio na recomendação (CRSLLM). A arquitetura proposta faz uso de LLMs em fases distintas do processo de recomendação: (1) para aquisição do perfil do usuário, através da extração de itens e avaliações citadas em diálogos; (2) para geração zero-shot de itens recomendados; e (3) para refinamento das recomendações a partir de uma avaliação contextual. A implementação utiliza versões do ChatGPT (como *gpt-4o-mini* e *gpt-4.1-mini*) e o algoritmo EASE [Steck 2019], um modelo linear simples de filtragem colaborativa, assim, gerando um sistema modular e de baixa complexidade.

A solução implementada foi avaliada com a base ReDial [Li et al. 2018], que consiste em diálogos em LN entre pares de usuários, simulando um ambiente de recomendação de filmes. Cada etapa da arquitetura foi avaliada separadamente. A etapa de extração obteve alto grau de efetividade, com valores de precisão e cobertura em torno de 90%. Os experimentos também mostraram que as etapas de geração zero-shot e revisão contextual trouxeram ganhos em comparação com o método base de filtragem colaborativa, seguindo as métricas Recall@K e NDCG@K comumente usadas na literatura.

Ainda foram realizados experimentos adicionais comparando o CRSLLM com vários baselines da literatura. Nos resultados obtidos, o CRSLLM mostrou-se superior a modelos de filtragem colaborativa complexos tanto em Recall@K como em NDCG@K. Contudo, o CRSLLM não foi superior a outras abordagens baseadas em LLMs. No entanto, esses trabalhos contemplam arquiteturas mais complexas, envolvendo, por exemplo, o uso de RAGs e Redes Neurais de Grafos. Nossa arquitetura, por outro lado, apresentou resultados competitivos, sendo modular e computacionalmente mais leve.

A seguir, a Seção 2 apresenta conceitos fundamentais referentes a CRSs, além dos trabalhos relacionados. A Seção 3 traz a solução proposta e sua implementação. A Seção 4 mostra os resultados obtidos, e a Seção 5 apresenta conclusões e trabalhos futuros.

2. Sistemas de Recomendação Conversacionais (CRSs)

RSs tradicionais operam com avaliações passadas de usuários para estimar preferências. Por sua vez, CRSs adotam interações mais completas, pois fazem uso de diálogos com o usuário, dando liberdade de expressar interesses em LN. Entre as vantagens dos CRSs estão: (a) mitigação da partida fria, pois é possível coletar preferências dos usuários nos diálogos; (b) indução do contexto, a partir da análise do diálogo; e (c) análise contínua de mensagens entre usuário e sistema [Jannach et al. 2021].

2.1. Componentes Básicos de um CRS

A grande maioria dos CRSs atuais são baseados em técnicas de processamento de LN implementadas em chatbots [Sun and Zhang 2018]. Apesar de não existir uma arquitetura consensual, existem componentes comumente usados nesses sistemas.

Interpretação da Entrada do Usuário: Analisa a entrada do usuário utilizando técnicas de compreensão de LN [Sun and Zhang 2018] ou LLMs [Jannach et al. 2021], podendo operar tanto com formulários estruturados quanto com LN. Seus objetivos incluem: detecção de intenções e de preferências explícitas e implícitas, extração de entidades e atributos, e tratamento de ambiguidades, sinônimos e correferências.

Módulo de Recomendação: Gera recomendações a partir das preferências e do contexto extraído. Esse módulo é acionado após o processamento da entrada do usuário para geração de itens candidatos, reordenação contextual ou refinamento após feedback do usuário [Sun and Zhang 2018, Tran et al. 2020], atualizando suas previsões a cada turno de diálogo para tornar as recomendações mais precisas [Jannach et al. 2021].

Existem outros componentes de um CRS, como o sistema de gerenciamento de diálogo (que decide a próxima ação do sistema após recebimento de entrada) [Jannach et al. 2021] e a geração de output (que transforma as recomendações em uma resposta em linguagem) [Jannach et al. 2021, Ding et al. 2020], porém o foco da nossa pesquisa é na interpretação da entrada e no módulo de recomendação de CRSLLMs.

2.2. Desafios de CRSs

Segundo [Gao et al. 2021], mesmo solucionando alguns dos problemas existentes em RSs tradicionais, CRSs continuam tendo desafios diversos, tais como:

Elicitação de Preferências através de Perguntas: Refere-se à necessidade de um CRS ser capaz de formular questões que revelem as preferências do usuário de maneira objetiva, tendo que saber o que perguntar e como adaptar recomendações dependendo do feedback, visando não sobrecarregar ou causar confusão no diálogo.

Estratégias de Recomendação Multiturno: Quando trabalhando com diálogos multiturno, com perguntas, respostas e feedback do usuário, CRSs necessitam saber quando fazer a recomendação ou continuar o diálogo, de maneira a evitar diálogos longos.

Compreensão e Geração de Linguagem Natural: Uma das tarefas mais complexas de um SC é conseguir interpretar o texto do usuário considerando contexto, intenções e sentimento, e gerar respostas coerentes que se adéquem ao usuário específico.

Equilíbrio entre Exploração e Explotação: Um CRS deve equilibrar a exploração de novas opções para descobrir preferências desconhecidas com a explotação de preferências já consolidadas, evitando falta de diversidade.

2.3. Integração LLMs com CRSs

Como já mencionado, a combinação de CRSs e LLMs tem atraído crescente atenção na literatura, uma vez que os LLMs trazem capacidades de compreensão de LN e raciocínio contextual que RSs tradicionais não possuem. Uma abordagem relevante nesse contexto é a de [He et al. 2023], que investiga o uso de LLMs como recomendadores conversacionais em cenário *zero-shot*, ou seja, sem qualquer ajuste fino específico para a tarefa. Os autores demonstram que LLMs são capazes de realizar recomendações razoáveis

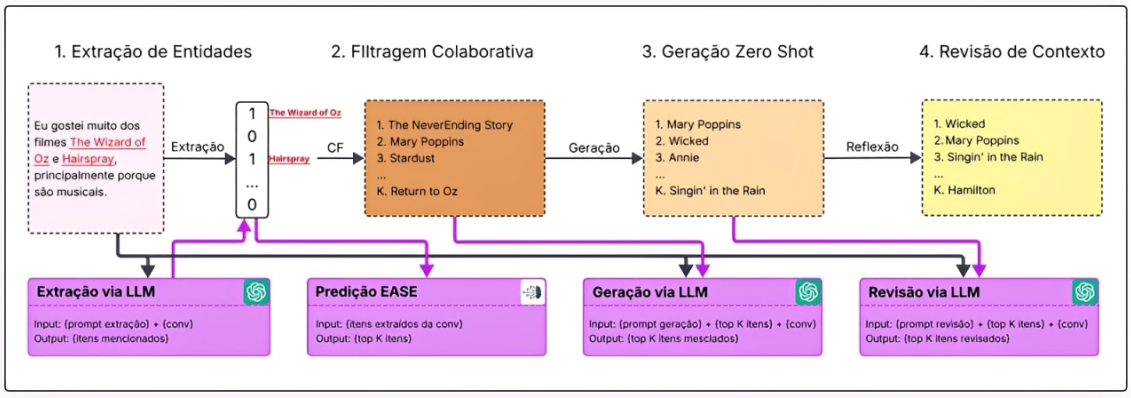


Figura 1. Arquitetura geral do sistema completo e seus componentes

baseando-se apenas no contexto do diálogo, sem acesso a dados históricos de interação dos usuários. Contudo, o trabalho também evidencia limitações importantes: LLMs em modo *zero-shot* tendem a recomendar itens populares e precisam de personalização baseada em padrões de comportamento colaborativo, o que limita sua qualidade.

Para mitigar essa limitação, [Zhu et al. 2025b] utilizam LLMs para extração de preferências em diálogos, compreensão do contexto, re-ranqueamento de recomendações e geração de respostas. Já o CRAVE [Zhu et al. 2025a] incorpora informações de filtragem colaborativa ao LLM por meio de descrições em LN, preservando suas capacidades generativas enquanto fundamenta as recomendações em padrões históricos de usuários semelhantes. E [Liang et al. 2024] exploram LLMs tanto para geração de diálogos sintéticos quanto como recomendadores ajustados para sugerir itens diretamente.

3. Solução Proposta para CRSLLMs

Esta seção apresenta a arquitetura do CRSLLM (ver Figura 1), que investiga o uso de LLMs em três etapas da recomendação. Inicialmente, no módulo de Extração de Entidades o LLM é empregado para construir o perfil do usuário, identificando nos diálogos os itens mencionados e suas respectivas avaliações. Essas informações alimentam o módulo de Filtragem Colaborativa, que provê recomendações iniciais para os usuários. No módulo de Geração *Zero-Shot*, o LLM produz uma segunda lista de itens diretamente a partir do diálogo, que é combinada com a recomendação da filtragem colaborativa. Por fim, no módulo de Revisão do Contexto, o LLM refina a lista resultante com base na análise contextual do diálogo. Os componentes da arquitetura são detalhados a seguir.

3.1. Extração de Entidades

Esse módulo faz uso de um LLM para extrair itens e opiniões mencionados por um usuário no diálogo de entrada. Seja $\mathcal{U} = \{u_1, \ldots, u_M\}$, o conjunto de usuários, e seja $\mathcal{I} = \{i_1, \ldots, i_N\}$, o conjunto de itens. Considere que o usuário u_j expresse no diálogo sua opinião sobre um dado item i_k. O LLM será instruído a classificar a opinião dentre os valores: $\mathcal{A} = \{a_1, \ldots, a_L\}$. Assim, o modelo irá extrair uma avaliação $r_{jk} \in \mathcal{A}$ a partir da interpretação do diálogo de entrada. As avaliações extraídas serão usadas para preencher a matriz de avaliações $X \in \{0, 1\}^{M \times N}$, usada pelo método de filtragem colaborativa.

Neste módulo, o LLM recebe um diálogo d e um prompt de extração pr^e (ver Figura 2). Assim, será extraído um conjunto de triplas relacionando um usuário u_j, um

Prompt pr^e

You are a data extraction system for a movie recommender. You will be given a full conversation between a user and a system. Each line begins with User or System. Your task is to extract movies that represent the USER'S PREFERENCES.
CRITICAL LOGIC:
2. VERY POSITIVE: If the user really likes, enthusiastically accepts a suggestion, or wants to watch very much a movie → (Attitude = 2).
1. POSITIVE: If the user likes, accepts a suggestion, or wants to watch a movie → (Attitude = 1).
0. NEUTRAL: If the user does not care, does not explicitly like or dislike or does not express opinion about a movie -¿ (Attitude = 0).
-1. NEGATIVE: If the user explicitly dislikes, rejects, or does not likes a movie → (Attitude = -1).
-2. VERY NEGATIVE: If the user hates, strongly dislikes, or despises a movie → (Attitude = -2).
OUTPUT FORMAT: Output ONLY the Exact Movie Title and the value of the attitude. Format: Movie Title #### X
IMPORTANT RULES: 1. Focus ONLY on the User's reaction. Do NOT infer preference just because the System suggested it. 2. The final attitude must reflect the user's LAST explicit opinion. 3. Each movie must appear ONLY ONCE. 4. Only give the neutral rating if the user does not mention the item in the dialog. 5. The user's opinion must only be about movies suggested by the system. Do not include movies the user mentioned themselves.
Conversation: {conversation text }
Example output format: Police Academy 2: Their First Assignment (1985)####2 Lethal Weapon (1987)####1 The Mummy (1999)####0

Figura 2. Prompt utilizado na etapa de extração de entidades.

item i_k e a opinião r_{jk} inferida pelo LLM a partir do diálogo. O LLM foi instruído a gerar opiniões dentre os valores $\mathcal{A} = \{-2, -1, 0, 1, 2\}$, representativas da escala de preferência: -2 (Muito Negativo), -1 (Negativo), 0 (Neutro), 1 (Positivo) e 2 (Muito Positivo). Esses valores foram escolhidos para diminuir a ambiguidade na interpretação das respostas.

O CRSLLM é um sistema de recomendação que opera em cima de feedback com avaliações binárias, e assim foi feito um processamento simples pra preencher a matriz de avaliações. Para os valores negativos retornados pelo LLM ($r_{jk} = -2$ ou $r_{jk} = -1$), foi definido então $X(j, k) = 0$. Para os valores não negativos ($r_{jk} = 0$, $r_{jk} = 1$ ou $r_{jk} = 2$), foi definido que $X(j, k) = 1$. Essa abordagem foi adotada para evitar o risco de extrapolar itens mencionados negativamente através de filtragem colaborativa, visto que a razão para a avaliação negativa de um item tende a ser mais subjetiva [Zhu et al. 2025b].

3.2. Filtragem Colaborativa

O modelo de filtragem colaborativa empregado foi o EASE [Steck 2019], um método linear projetado para dados esparsos. Dada a matriz de avaliações $X \in \{0, 1\}^{M \times N}$, o EASE estima uma matriz de pesos item-item B minimizando a função objetivo:

$$\min_B ||X - XB||_F^2 + \lambda ||B||_F^2 \quad s.a. \quad diag(B) = 0 \tag{1}$$

em que o primeiro termo minimiza o erro de reconstrução da matriz de avaliações e o segundo termo controla a complexidade do modelo (através do parâmetro de regularização λ). A restrição $diag(B) = 0$ impede que um item contribua para sua própria predição. A solução possui forma fechada obtida pelo método dos multiplicadores de Lagrange:

Prompt pr^{zs}

You are an expert movie recommender. Your task is to generate 25 highly personalized movie recommendations for a user based ONLY on their ongoing conversation.
CONVERSATION CONTEXT: {conversation history}
TASK: Based on the user's explicit and implicit preferences in the conversation, generate exactly 25 movie recommendations.
IMPORTANT RULES: 1. Respect History: Do NOT recommend movies the user has already explicitly mentioned (whether they liked them or not). 2. Focus on Preferences: Match the user's requested genres, mood, actors, or directors perfectly based ONLY on what was said. 3. Exact Count: You MUST generate exactly 25 recommendations. No more, no less. 4. Strict Formatting: Do NOT include any conversational filler, greetings, explanations, bullet points, or numbering.
OUTPUT FORMAT: Output exactly one standard movie title per line. Include the release year in parentheses if possible.
Example output format: The Matrix (1999) Inception (2010) Interstellar (2014)
You MUST output exactly 25 lines. Nothing else.

Figura 3. Prompt utilizado na etapa de geração das recomendações.

$$\hat{B} = (X^\top X + \lambda I)^{-1} \cdot (X^\top X - diagMat(\gamma)) \tag{2}$$

em que γ representa o vetor de multiplicadores de Lagrange. Para um usuário u_j, com o vetor de avaliações $\vec{x}_j \in \{0,1\}^{1\times N}$, os scores de recomendação são obtidos pela multiplicação desse vetor pela matriz $\hat{B}$ e os K itens mais relevantes são selecionados:

$$I_K^{fc} = Top_K\left((\vec{x}_j \cdot B) \odot m(i_k)\right) \tag{3}$$

O vetor $\vec{x}_j \cdot B$ contém os escores preditos para todos os itens do catálogo, enquanto $m(i_k)$ corresponde a uma máscara que remove os itens já presentes no histórico do usuário. A função Top_K retorna os índices dos K maiores escores. O conjunto I_K^{fc} constitui a lista ordenada de recomendações produzidas pelo EASE.

3.3. Geração Zero-Shot

A etapa anterior retorna uma lista ordenada I_K^{fc}. Nessa etapa, essa lista é revisada, combinando com uma lista I_K^{zs} retornada pelo LLM, operando diretamente em um cenário *zero-shot*. O objetivo é explorar a capacidade dos LLMs de compreender preferências expressas em linguagem natural. Inicialmente, um prompt pr^{zs} (ver Figura 3) instrui o LLM a gerar a lista ordenada de recomendações I_K^{zs} baseada diretamente no diálogo d:

$$I_K^{zs} = LLM(pr^{zs}, d) \tag{4}$$

Após essa geração, cada item i_k recebe um score posicional $\hat{S}^{zs}(i_k)$, definido pela equação (5), de modo que o primeiro item recebe score 1.0 e o último recebe um valor próximo a 0.0:

$$\hat{S}^{zs}(i_k) = 1 - \frac{\text{rank}(i_k)}{\max(1,,|I_K^{zs}|)} \tag{5}$$

Prompt pr^{ac}

You are a movie recommendation expert. Analyze each candidate movies relevance to the user's conversation.
CONVERSATION CONTEXT: conversation history
CANDIDATE MOVIES (from collaborative filtering): recommended movies
TASK: For each movie, assign a relevance score based on how well it matches the users' expressed preferences.
SCORING CRITERIA: 3 (Highly Relevant): Perfect match for the user's stated preferences. Same genre/theme as movies the user loved. Features actors/directors user mentioned positively. Directly addresses the user's request.
2 (Relevant): Good fit for user's preferences. Related genre or theme. Similar tone/mood to liked movies. Reasonable next recommendation in this context.
1 (Somewhat Relevant): Loose connection. Somewhat related. Could work, but not a strong match.
0 (Not Relevant): No connection to the user's preferences. Wrong genre. Contradicts stated preferences. Does not fit the conversation context.
IMPORTANT RULES: - Do NOT assume popularity = relevance. - Do NOT favor recent movies unless the user mentioned recency. - Focus on USER's explicit and implicit preferences. - If user likes horror, romantic comedies should get 0 or 1 at most.
OUTPUT FORMAT: For EACH movie, output exactly one line: Movie Title####X
You MUST score ALL movies. Output one line per movie, nothing else.

Figura 4. Prompt utilizado na etapa de revisão do contexto

Um procedimento similar foi feito para gerar o score posicional $\hat{S}^{fc}(i_k)$ a partir da saída do EASE para cada item $i_k \in I_K^{fc}$. Os dois *scores* são combinados pela equação:

$$\hat{S}^{c}(i_k) = \alpha \hat{S}^{fc}(i_k) + (1-\alpha)\hat{S}^{zs}(i_k) + \mathcal{B}^{int} \tag{6}$$

O termo $\mathcal{B}^{int}$ corresponde a um bônus aplicado aos itens pertencentes à interseção $I_K^{zs} \cap I_K^{fc}$, favorecendo recomendações presentes nas duas formas de recomendação. Por fim, o hiperparâmetro α controla a contribuição relativa do EASE e do LLM na composição do score combinado $\hat{S}^{c}(i_k)$. Finalmente, a lista final I_K^{c} é definida como:

$$I_K^{c} = TopK\left(\hat{S}^{c}(i_k)\right), i_k \in I_K^{zs} \cup I_K^{fc} \tag{7}$$

3.4. Revisão do Contexto

A seguir, o CRSLLM, realiza uma análise contextual para reorganizar os itens recomendados em I_K^{c}, penalizando itens sem coerência com as preferências do usuário na conversa d e promovendo os mais alinhados ao diálogo. Esse processo é formalizado por:

$$\hat{S}^{ac}(i_k) = \text{LLM}(pr^{ac},\, d,\, I_K^{c}) \tag{8}$$

$$\hat{S}^{acr}(i_k) = \sigma \cdot \hat{S}^{c}(i_k) + (1-\sigma) \cdot \hat{S}^{ac}(i_k) \tag{9}$$

$$I_K^{acr} = \text{TopK}_{i_k \in I_K^{c}}\left(S^{acr}(i_k)\right) \tag{10}$$

em que pr^{ac} é um prompt (ver Figura 4) que instrui o LLM a atuar como um avaliador, atribuindo a cada item $i_k \in I_K^c$ um score de relevância $\hat{S}^{ac}(i_k)$ com base na análise contextual inferida no diálogo d. Esses scores são inteiros no conjunto $\{0, 1, 2, 3\}$, categorizados como: 0 (não relevante), 1 (pouco relevante), 2 (relevante) e 3 (extremamente relevante). Esse score é normalizado para o intervalo $[0, 1]$ por $\hat{S}^{ac}(i_k) \leftarrow \hat{S}^{ac}(i_k)/3$.

Paralelamente, $\hat{S}^{c}(i_k)$ é o score dos itens de I_K^c, já no intervalo $[0, 1]$. O hiperparâmetro σ rege o peso relativo entre as duas fontes na composição do score da análise contextual revisada $\hat{S}^{acr}(i_k)$, a partir do qual é gerada a lista final I_K^{acr}.

4. Experimentos e Resultados

Esta seção apresenta os experimentos realizados, incluindo a descrição da base de dados, as métricas de avaliação e os resultados comparativos com os vários baselines.

4.1. Base de Dados

Os dados utilizados nos experimentos são provenientes da base de dados ReDial [Li et al. 2018], que consiste em diálogos em linguagem natural entre pares de usuários. Os diálogos foram coletados através da plataforma de crowdsourcing Amazon Mechanical Turk. Os participantes da conversa assumem papéis distintos: um atua como buscador de recomendação, que expressa suas preferências, e o outro é o recomendador.

Além do texto dos diálogos, a base fornece metadados relevantes para tarefas de recomendação, como registro de menções a filmes e marcações de preferências do usuário (com informações sobre itens sugeridos, já assistidos e avaliados positivamente ou negativamente). O ReDial contém 11.348 entradas, tendo a separação de treino e teste já feita, sendo 10.006 (88.27%) de treino e 1.342 (11.83%) de teste. No contexto do CRSLLM, o ReDial é explorado como fonte tanto para a extração de preferências a partir do texto, realizada por uma instância de LLM, quanto para a construção da matriz usuário–item.

4.2. Avaliação da Extração de Entidades

Inicialmente, apresentamos os resultados obtidos com a extração feita pelo LLM. A avaliação tem objetivo de medir o quão similar é a lista de itens extraídos, comparada com a lista fornecida pelo conjunto Redial. Para essa avaliação, foram usadas três métricas tradicionais para extração de informação: (a) Jaccard, que mede similaridade entre as listas de itens extraídos e itens conhecidos; (b) Precision, que avalia a qualidade das extrações, medindo a proporção de entidades extraídas que estão corretas; e (c) Recall, que avalia a proporção de entidades corretas que foram efetivamente identificadas pelo LLM.

Foram testados diferentes modelos de LLM do GPT para extração. Os resultados foram compilados na Tabela 1. A extração usada pelo sistema foi a realizada pelo modelo GPT-5, que obteve melhor resultado em todas as métricas. No geral foram obtidas taxas altas para todas as métricas com destaque para valores de Precision próximos a 90%.

4.3. Avaliação do CRSLLM

Inicialmente, avaliamos os componentes individuais do CRSLLM, com as seguintes versões: (a) EASE, modelo de filtragem colaborativa puro; (b) CRSLLM-RG, modelo CRSLLM parcial com recomendação generativa zero-shot; e (c) CRSLLM, modelo completo com recomendação generativa e análise contextual. Adicionalmente,

Tabela 1. Desempenho dos modelos GPT na extração de entidades

Modelo	Jaccard	Precision	Recall
GPT-4o-mini	0.7153	0.8482	0.7581
GPT-4.1-mini	0.7012	0.8455	0.7603
GPT-4o	0.6961	0.8318	0.7342
GPT-4.1	0.7173	0.8469	0.7643
GPT-5	**0.7509**	**0.8990**	**0.8066**

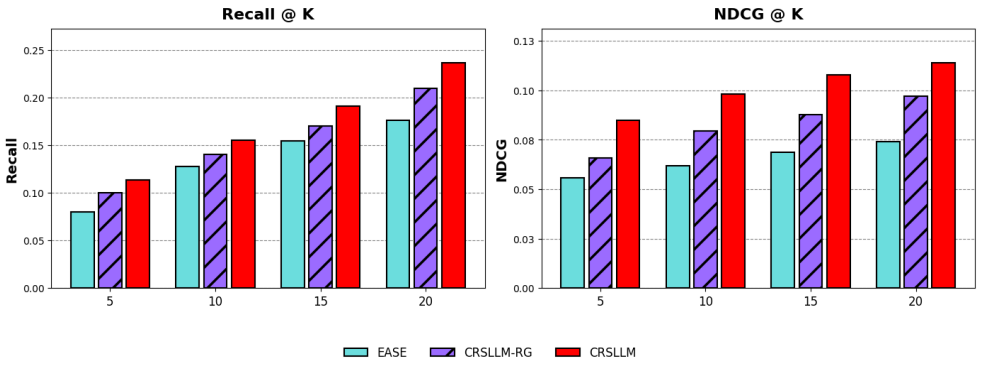


Figura 5. Comparação entre as diferentes versões do CRSLLM

foi realizada uma análise de sensibilidade sobre os valores de K utilizados, que são: $K \in \{5, 15, 20, 25\}$.

Para avaliar o desempenho do CRSLLM, foram utilizadas métricas quantitativas amplamente adotadas na literatura de RS. O conjunto de itens relevantes utilizado como referência para avaliação é construído a partir dos metadados do ReDial. Em cada turno do diálogo, um item i_k é inserido na lista ground truth se o usuário não havia assistido ao item anteriormente e se avaliou o item positivamente. Adicionalmente, itens já mencionados em turnos são excluídos, evitando recomendações redundantes.

A partir da lista ground truth, a lista gerada pelo CRSLLM é avaliada pelas métricas: (a) o Recall@K avalia a proporção de itens relevantes que o sistema conseguiu recuperar nas K primeiras recomendações, em relação ao total de itens relevantes; (b) o Normalized Discounted Cumulative Gain (NDCG@K) avalia a qualidade do ranking, penalizando itens relevantes que aparecem em posições mais baixas da lista. Diferente do Recall@K, que apenas verifica se os itens relevantes foram recuperados, o NDCG@K considera a posição em que eles aparecem, avaliando a qualidade da ordenação. A avaliação ocorre na forma turno a turno. O modelo analisa o diálogo sequencialmente, utilizando como entrada apenas o histórico acumulado até o momento da sugestão.

Como pode ser visto na Figura 5, houve um ganho em Recall@K e NDCG@K para todos os valores de K, com um ganho percentual médio de 30,13% em Recall@K e 55,45% em NDCG@K ao compararmos o pipeline completo do CRSLLM diretamente com o modelo base (EASE). Ao analisarmos os ganhos marginais de cada componente, observamos que a etapa de Geração Zero-Shot (CRSLLM-RG) foi responsável por um aumento médio de 15,92% no Recall@K e 26,56% no NDCG@K sobre o modelo EASE. Posteriormente, a aplicação da Análise Contextual gerou um ganho adicional médio de

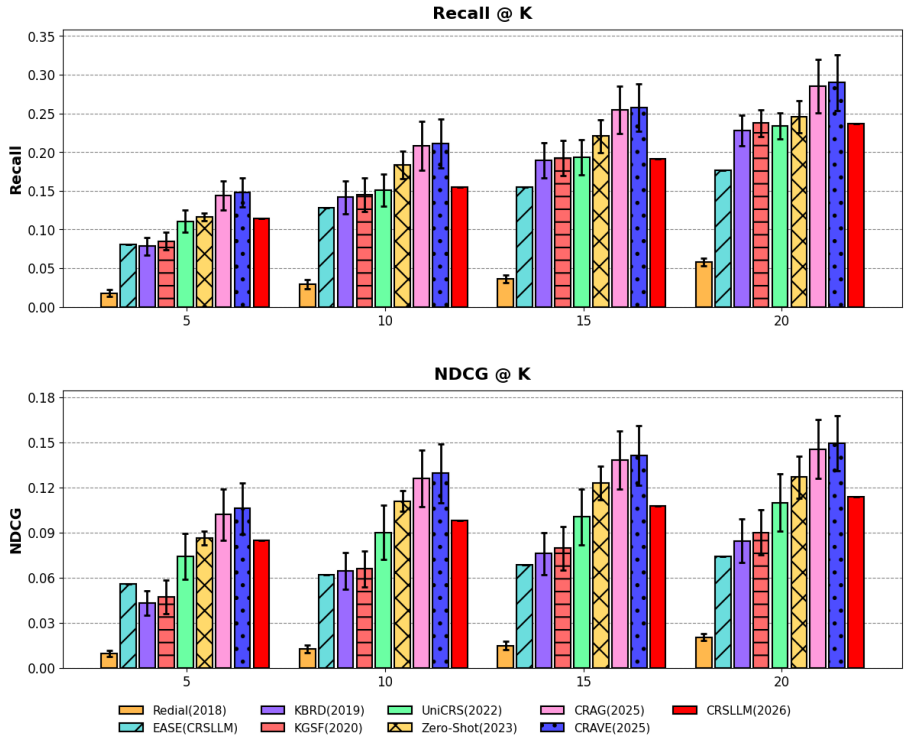


Figura 6. Comparação entre CRSLLM e outras baselines que utilizam Redial como dataset

12,20% no Recall@K e 22,97% no NDC@KG em relação ao passo anterior. Desta forma, cada etapa contribuiu positivamente com a qualidade final das recomendações.

Além da análise de cada componente, avaliamos o CRSLLM com baselines da literatura: (a) **ReDial (2018)** utiliza um autoencoder para modelar itens e uma RNN para modelar os diálogos [Li et al. 2018]; (b) **KBRD (2019)** emprega uma Rede Neural de Grafos Relacional (RGNN) sobre o grafo de conhecimento DBpedia para representar entidades e realizar fusão semântica [Chen et al. 2019]; (c) **KGSF (2020)** utiliza o ConceptNet para modelar diálogos e integra informações de entidades [Zhou et al. 2020]; (d) **UniCRS (2022)** combina um modelo *transformer* com uma RGNN e mecanismo de *cross-attention* para incorporar informações do grafo de conhecimento [Wang et al. 2022]; (e) **Zero-Shot LLM (2023)** utiliza um LLM para gerar recomendações diretamente a partir do diálogo, sem recorrer a bases externas [He et al. 2023]; (f) **CRAG (2025)** integra LLMs e filtragem colaborativa por meio de recuperação aumentada e recuperação colaborativa contextual [Zhu et al. 2025b]; e (g) **CRAVE (2025)** combina LLMs com experiências em linguagem natural recuperadas de interações históricas [Zhu et al. 2025a].

A Figura 6 apresenta os resultados do CRSLLM em relação aos baselines. O CRSLLM supera consistentemente os modelos que não utilizam LLMs (ReDial, KBRD e KGSF) e apresenta desempenho competitivo em relação ao UniCRS. Embora fique abaixo de abordagens recentes baseadas em LLMs, como Zero-Shot, CRAG e CRAVE, essa diferença é esperada, pois esses métodos empregam arquiteturas significativamente mais complexas, envolvendo recuperação colaborativa densa, múltiplas chamadas a LLMs e modelos de maior capacidade.O CRAG requer 5 chamadas ao LLM por conversa de

teste, enquanto o CRAVE exige 6 chamadas por conversa de treino na construção *offline* dos bancos de experiência, além de 3 chamadas por conversa na inferência com o sistema debatedor-crítico, ambos utilizando o GPT-4o como modelo base. Em contraste, o CRSLLM foi projetado com foco em simplicidade e eficiência, utilizando o algoritmo linear EASE e apenas 1 chamada por diálogo na etapa de extração e 2 chamadas por turno na inferência, com custo independente do comprimento do diálogo e empregando um modelo de menor capacidade (*gpt-4o-mini*). Ainda assim, alcançou resultados competitivos, evidenciando a eficácia da integração com LLMs. Além disso, sua arquitetura modular facilita a manutenção, a substituição de componentes e a incorporação de futuras melhorias.

5. Conclusão

Este trabalho apresentou o CRSLLM, uma arquitetura híbrida para CRS que integra filtragem colaborativa e LLMs. O sistema foi implementado e avaliado sobre a base ReDial, obtendo resultados satisfatórios e demonstrando que a combinação entre modelos colaborativos leves e LLMs é capaz de produzir recomendações relevantes e contextualizadas. Entre as principais contribuições destacam-se a proposta de uma arquitetura modular e extensível, o desenvolvimento de um processo de pré-processamento automático da base ReDial, a integração de LLMs para interpretação dos diálogos, a utilização do algoritmo EASE como núcleo de filtragem colaborativa e a introdução de etapas de pós-processamento baseadas em LLMs para geração e refinamento das recomendações. Em conjunto, essas estratégias aumentaram a cobertura e a qualidade das listas recomendadas.

Como trabalhos futuros, destacam-se a incorporação de técnicas de RAG para lidar com diálogos longos e preservar o contexto conversacional, a substituição do EASE por algoritmos mais sofisticados e a realização de avaliações *online* com usuários reais, permitindo analisar aspectos como qualidade da interação e satisfação dos usuários. Esses avanços podem ampliar a eficácia e a aplicabilidade do CRSLLM em cenários reais. Além disso, a avaliação foi realizada apenas em ambiente *offline*, utilizando a base ReDial no domínio de filmes, o que limita a generalização dos resultados para outros domínios.

Referências

Chen, Q., Lin, J., Zhang, Y., Ding, M., Cen, Y., Yang, H., and Tang, J. (2019). Towards knowledge-based recommender dialog system. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Christakopoulou, K., Radlinski, F., and Hofmann, K. (2016). Towards conversational recommender systems. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 815–824. ACM.

Ding, X., Yu, W., Xie, Y., and Liu, S. (2020). Efficient model-based collaborative filtering with fast adaptive pca. *arXiv preprint arXiv:2009.02251*.

Gao, C. et al. (2021). Advances and challenges in conversational recommender systems. *Journal of Information Processing*.

He, Z., Xie, Z., Jha, R., Steck, H., Liang, D., Feng, Y., Majumder, B. P., Kallus, N., and McAuley, J. (2023). Large language models as zero-shot conversational recommenders. In *Proceedings of the 32nd ACM International CIKM*.

Jannach, D., Manzoor, A., Cai, W., and Chen, L. (2021). A survey on conversational recommender systems. *ACM Computing Surveys*, 54(4):105:1–105:35.

Li, R., Kahou, S. E., Schulz, H., Michalski, V., Charlin, L., and Pal, C. (2018). Towards deep conversational recommendations. In *Advances in Neural Information Processing Systems 31 (NIPS 2018)*.

Liang, T., Jin, C., Wang, L., Fan, W., Xia, C., Chen, K., and Yin, Y. (2024). LLM-REDIAL: A large-scale dataset for conversational recommender systems created from user behaviors with LLMs. pages 8926–8939, Bangkok, Thailand. Association for Computational Linguistics.

Roy, D. and Dutta, M. (2022). A systematic review and research perspective on recommender systems. *Journal of Big Data*, 9(1):59.

Steck, H. (2019). Embarrassingly shallow autoencoders for sparse data. In *Proceedings of The Web Conference (WWW)*, pages 3251–3257. ACM.

Sun, Y. and Zhang, Y. (2018). Conversational recommender system. In *Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 235–244, Ann Arbor, MI, USA. ACM.

Tran, D. H., Sheng, Q. Z., Zhang, W. E., Hamad, S. A., Zaib, M., Tran, N. H., Yao, L., and Khoa, N. L. D. (2020). Deep conversational recommender system: A new frontier for goal-oriented dialogue systems. *arXiv preprint arXiv:2004.13245*.

Wang, X., Tang, X., Zhao, W. X., Wang, J., and Wen, J.-R. (2023). Rethinking the evaluation for conversational recommendation in the era of large language models. *arXiv preprint arXiv:2305.13112*.

Wang, X., Zhou, K., Wen, J.-R., and Zhao, W. X. (2022). Towards unified conversational recommender systems via knowledge-enhanced prompt learning. In *Proc. of the 28th ACM SIGKDD Conf. on Knowledge Discovery and Data Mining*, pages 1929–1937.

Zhao, Z., Fan, W., Li, J., Liu, Y., Mei, X., Wang, Y., Wen, Z., Wang, F., Zhao, X., Tang, J., and Li, Q. (2023). Recommender systems in the era of large language models (llms). *IEEE Transactions on Knowledge and Data Engineering*. arXiv:2307.02046.

Zhou, K., Zhao, W. X., Bian, S., Zhou, Y., Wen, J.-R., and Yu, J. (2020). Improving conversational recommender systems via knowledge graph based semantic fusion. In *Proc. of the 26th ACM SIGKDD Conf. on KDD*, pages 1006–1014.

Zhu, Y., Steck, H., Liang, D., He, Y., Kallus, N., and Li, J. (2025a). Llm-based conversational recommendation agents with collaborative verbalized experience. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 2207–2220.

Zhu, Y., Wan, C., Steck, H., Liang, D., Feng, Y., Kallus, N., and Li, J. (2025b). Collaborative retrieval for large language model-based conversational recommender systems. In *Proceedings of the ACM Web Conference 2025*, WWW '25. ACM.