

# Um Estudo sobre o uso de Modelos de Linguagem Abertos na Tarefa de Recomendação de Próximo Item

Marcos Avner Pimenta de Lima<sup>1</sup>, Eduardo Alves da Silva<sup>1,2</sup>, Altigran Soares da Silva<sup>1</sup>

<sup>1</sup>Instituto de Computação – Universidade Federal do Amazonas (UFAM)  
Manaus – AM – Brasil

<sup>2</sup>Escola Politécnica – Universidade do Vale do Itajaí (Univali)  
Itajaí – SC – Brasil

{marcos.lima,eduardo.silva,alti}@icomput.ufam.edu.br

**Abstract.** *Large Language Models (LLMs) have been employed in recommendation systems to enhance the user experience and reduce information overload. With the rise of generative AI, this approach has gained attention and demonstrated promising results. Open LLMs are of great interest due to their accessibility and potential for fine-tuning. We investigate the effectiveness of open LLMs in recommendation by utilizing a method from the literature to recommend new items, both with and without fine-tuning. We conclude that open LLMs can outperform closed ones, even with fewer parameters, and fine-tuning enhances the performance of models, depending on hyperparameter exploration and data quality.*

**Resumo.** *Modelos de Linguagem de Larga Escala (LLMs) vêm sendo utilizados em sistemas de recomendação para melhorar a experiência dos usuários e reduzir a sobrecarga de informação. Com a popularidade da IA generativa, essa abordagem cresce e mostra resultados promissores. LLMs abertos são de grande interesse devido à sua acessibilidade e potencial para ajuste fino. Investigamos a eficácia de LLMs abertos em recomendação sequencial utilizando um método da literatura para recomendar novos itens, com e sem ajuste fino. Concluímos que LLMs de código abertos podem superar os proprietários, mesmo com menos parâmetros, e que o ajuste fino melhora o desempenho dos modelos, dependendo da exploração de hiperparâmetros e da qualidade dos dados.*

## 1. Introdução

Sistemas de recomendação desempenham um papel fundamental em plataformas on-line de *e-commerce*, redes sociais, mídias audiovisuais (músicas, vídeos e filmes), notícias, entre outras. Eles estão presentes no dia a dia de todos e têm como objetivo amenizar o problema da sobrecarga de informação [Shao et al. 2021], ou seja, reduzir o volume de conteúdo acessado até que o usuário encontre o que procura. Os sistemas de recomendação utilizam informações do comportamento e preferências do usuário para sugerir os itens relevantes e reduzir o seu esforço na busca de conteúdo.

Da mesma forma, os Modelos de Linguagem de Larga Escala (LLMs – Large Language Models) estão em destaque e se tornando cada vez mais presentes no cotidiano das pessoas. O lançamento do ChatGPT em novembro de 2022<sup>1</sup> acelerou a adoção massiva de LLMs. Embora esses modelos sejam primariamente treinados para a geração de texto,

<sup>1</sup>ChatGPT Release Notes – <https://help.openai.com/en/articles/6825453-chatgpt-release-notes>

eles têm apresentado expressiva capacidade de generalização e aplicabilidade em outras tarefas [Wang and Lim 2023, Bao et al. 2023, Hou et al. 2024b].

A capacidade de generalização dos LLMs vem despertando o interesse dos pesquisadores de sistemas de recomendação [Fan et al. 2023]. No entanto, a natureza recente dessa área enfrenta desafios quanto a viabilidade da utilização de LLMs em recomendação. Por exemplo, há trabalhos que adotam LLMs diretamente como recomendador [Dai et al. 2023, Liu et al. 2023, Wang and Lim 2023, Hou et al. 2024b], outros realizam o ajuste fino (*fine-tuning*) [Bao et al. 2023, Liu et al. 2024, Xu et al. 2023] dos modelos para adaptá-los à tarefa de recomendação. Além disso, o número de novos LLMs pré-treinados lançados em 2023 mais que dobrou em relação ao ano anterior, totalizando 149 novos modelos no fim daquele ano, sendo 65.7% deles modelos abertos<sup>2</sup>.

Apesar do crescente interesse na aplicação de LLMs em recomendação, os experimentos têm sido conduzidos de maneira particular em cada estudo. Alguns comparam recomendadores com e sem LLMs, outros exploram LLMs com e sem ajuste fino. Também, há pesquisas que testam arquiteturas de LLMs na recomendação, como *encoder-decoder* ou *decoder-only* [Liu et al. 2023, Xu et al. 2023]. Há trabalhos que comparam resultados de uma única tarefa [Wang and Lim 2023, Sanner et al. 2023] ou de múltiplas tarefas [Dai et al. 2023, Liu et al. 2023, Hou et al. 2024b] de recomendação entre LLMs e recomendadores tradicionais, apenas com engenharia de prompt. Há estudos que investigam o desempenho de modelos proprietários [Wang and Lim 2023, Dai et al. 2023], ou de um modelo aberto e outro proprietário [Zhang et al. 2023, Hou et al. 2024b]. Por fim, há estudos que verificam o desempenho de LLMs pré-treinados para a tarefa de recomendação versus LLMs com ajuste fino [Bao et al. 2023, Xu et al. 2023, Liu et al. 2024]. Essa diversidade de abordagens reflete a complexidade e o aumento do interesse na aplicação de LLMs na recomendação de itens.

Dessa forma, a escolha de um método de recomendação com LLMs depende de lacunas existentes nos estudos propostos. Por exemplo, ainda não está claro se algum LLM aberto se destaca como um recomendador eficaz. Em busca de preencher algumas dessas lacunas e responder a pergunta anterior, neste estudo exploramos: (i) o método de engenharia de prompt proposto por Wang & Lim [Wang and Lim 2023] para a tarefa de recomendação sequencial nos LLMs abertos Llama [Touvron et al. 2023], Mistral [Jiang et al. 2023] e Falcon [Almazrouei et al. 2023]; e (ii) o impacto do ajuste fino na qualidade das recomendações. Portanto, esta pesquisa visa examinar a capacidade dos modelos, originalmente treinados para predição de *tokens* sequenciais, em antecipar com precisão o próximo item em uma sequência de itens previamente consumidos.

A escolha do método de Wang & Lim objetiva: (i) estabelecer um *baseline* de um modelo GPT que superou os modelos tradicionais na recomendação sequencial; (ii) avaliar a capacidade de inferência dos LLMs abertos num cenário de recomendação *zero-shot*; (iii) comparar modelos abertos no mesmo cenário; e (iv) verificar o desempenho do prompt de três etapas proposto naquele trabalho, que são inferir preferências, itens mais representativos e recomendações.

Além de preencher essas lacunas e focar na avaliação de um único método com LLMs na tarefa de recomendação sequencial, não foram identificados trabalhos que comparem LLMs abertos nesse contexto. Também não foram encontrados estudos que utilizem modelos recentes e competitivos em tarefas de processamento de linguagem natural,

---

<sup>2</sup>Stanford AI Index Report 2024 – <https://aiindex.stanford.edu/report/>

como o Llama, Mistral e Falcon, para recomendação. Assim, esperamos que os resultados contribuam para a compreensão das vantagens e desvantagens de cada modelo e auxiliem futuras pesquisas de recomendação com LLMs. Além disso, esperamos que a melhoria contínua dos modelos abertos leve à sua utilização efetiva em sistemas on-line.

Entre os principais resultados, verificamos que um modelo aberto com 8 bilhões de parâmetros superou um proprietário de 175 bilhões na acurácia das recomendações em *zero-shot*. Também, que um único método de ajuste fino gerou variabilidade no comportamento dos modelos, melhorando a acurácia em até 9% em alguns casos, mas piorando em outros. Portanto, fatores determinantes para o sucesso no ajuste fino são a investigação de hiperparâmetros em cada modelo, e a qualidade e o volume do conjunto de dados de refinamento. Dessa forma, as principais contribuições do estudo são: (1) um comparativo de LLMs abertos na tarefa de recomendação sequencial em *zero-shot*; (2) um pipeline de ajuste-fino para múltiplos modelos de linguagem na tarefa de recomendação<sup>3</sup>; e (3) uma análise de cenários com e sem ajuste fino em LLMs abertos na recomendação sequencial.

O artigo é organizado como segue. A Seção 2 discute os estudos da literatura recente relacionados ao uso de LLM em recomendadores e as estratégias adotadas nessa tarefa. Na Seção 3, o método no qual baseamos nosso estudo é detalhado. Na Seção 4 descrevemos os experimentos realizados e apresentamos os resultados na Seção 5. Por fim, a Seção 6 destaca as contribuições, principais resultados e elenca trabalhos futuros.

## 2. Trabalhos Relacionados

Os Modelos de Linguagem de Larga Escala (LLMs) têm apresentado resultados promissores em diversos domínios devido à sua expressiva capacidade de generalização [Bao et al. 2023]. Isso despertou a atenção de pesquisadores da área de sistemas de recomendação para explorar o uso de LLMs nessa tarefa, principalmente após o lançamento do ChatGPT (modelo gpt-3 [Brown et al. 2020]) em novembro de 2022.

Os modelos GPT têm sido regularmente utilizados como referência (*baselines*) na experimentação em virtude de resultados preliminares positivos [Dai et al. 2023, Liu et al. 2023, Lyu et al. 2023]. A abordagem mais utilizada para recomendação é a preparação de *prompts*, comumente chamada de *engenharia de prompt* [Dai et al. 2023, Liu et al. 2023, Wang and Lim 2023, Sanner et al. 2023, Hou et al. 2024b]. Essa abordagem exige apenas a inferência de modelos pré-treinados, o que poupa tempo e recursos computacionais, embora possa envolver custos como o consumo de APIs.

O *prompt* é o texto ou a pergunta a ser enviada como entrada especificando a tarefa a ser realizada pelo LLM. Nas tarefas de recomendação, os trabalhos recentes da literatura incorporaram o histórico de itens do usuário e a instrução em forma de texto de acordo com a tarefa desejada. A recomendação sequencial vem sendo a tarefa mais explorada [Liu et al. 2023, Wang and Lim 2023, Rajput et al. 2023, Bao et al. 2023, Xu et al. 2023, Hou et al. 2024b] e se baseia na predição do próximo item a ser consumido. Um exemplo de *prompt* usado para este tipo de tarefa é ilustrado na Figura 1. As técnicas *zero-shot*, onde não são fornecidos exemplos da saída desejada [Wang and Lim 2023, Hou et al. 2024b], e *few-shot*, onde alguns exemplos são fornecidos [Dai et al. 2023, Liu et al. 2023], são as mais utilizadas com o objetivo de aumentar a qualidade das recomendações. A primeira explora ao máximo a capacidade de inferência do LLM para obter a recomendação sem nenhuma informação prévia e a segunda fornece alguns exemplos para direcionar e facilitar o processo de inferência.

<sup>3</sup>Mais informações disponíveis no repositório: <https://github.com/bdri-ufam/sbbd2024-nir-llms>.

```
### Persona
Você é um recomendador de filmes que responde solicitações baseadas no contexto fornecido. Se você não souber a resposta, não compartilhe informação falsa.

### Contexto
O histórico de filmes assistidos é: {histórico ordenado do mais antigo para o mais recente}.

### Tarefa
Baseado nos últimos filmes assistidos do histórico, recomende o próximo filme a ser assistido.

### Formato
Forneça uma lista numerada com 10 títulos de filmes.
```

Figura 1. Exemplo de um prompt para recomendação sequencial zero-shot.

Também com o objetivo de melhorar a qualidade na recomendação, trabalhos recentes usam LLMs para o aumento de dados [Lyu et al. 2023] ou incorporaram identificadores de itens no modelo para a recomendação direta [Rajput et al. 2023], e outros são voltados ao *ajuste fino* – *fine-tuning* [Bao et al. 2023, Xu et al. 2023, Liu et al. 2024]. Essa última abordagem vem ganhando destaque por aproveitar a informação existente no (pré-)treinamento e aprimorar os modelos na tarefa de recomendação.

O ajuste fino tem sido realizado em nível de instrução com combinações de prompts e respostas esperadas na recomendação [Zhang et al. 2023]. Essa técnica também é usada para refinar um modelo menor com respostas de um maior, para que o comportamento do primeiro seja tão bom quanto do segundo [Liu et al. 2024], ou ainda, para que modelos abertos sejam competitivos com os proprietários. Isso é importante porque viabiliza o uso de recursos computacionais limitados, permite a reprodutibilidade, e melhora o tempo de inferência, pois modelos menores geram respostas mais rápidas.

Dos modelos abertos, o Llama tem se destacado [Bao et al. 2023, Xu et al. 2023, Liu et al. 2024]. Embora os 7 bilhões de parâmetros da sua menor versão [Touvron et al. 2023] represente 4% dos 175 bilhões de parâmetros do GPT-3 [Brown et al. 2020], o ajuste fino com recursos acessíveis só é viável com PEFT (Parameter Efficient Fine-tuning – Refinamento Eficiente de Parâmetros) [Houlsby et al. 2019], como a LoRA (Low Rank Adaptation – Baixa Adaptação de Ranking) [Hu et al. 2022]. Essa tem sido a prática comum tanto na academia quanto no mercado<sup>4</sup>.

Dada a recenticidade do tema, para posicionar os trabalhos que usam LLMs com a literatura de sistemas de recomendação, os conjuntos de dados mais utilizados são os de filmes<sup>5</sup>, produtos<sup>6</sup> e notícias<sup>7</sup>, todos bem estabelecidos em pesquisas de recomendação sem o uso de LLMs. O MovieLens é o conjunto de dados mais utilizado nos estudos [Dai et al. 2023, Wang and Lim 2023, Sanner et al. 2023, Lyu et al. 2023, Bao et al. 2023, Hou et al. 2024b, Xu et al. 2023]. Nestes experimentos, as métricas mais adotadas são HR (Hit Ratio – Taxa de Acerto) e NDCG (Normalized Discounted Cumulative Gain – Ganho Cumulativo Descontado Normalizado) [Liu et al. 2023, Zhang et al. 2023, Dai et al. 2023, Wang and Lim 2023, Lyu et al. 2023, Sanner et al. 2023, Rajput et al. 2023, Xu et al. 2023, Hou et al. 2024b, Liu et al. 2024].

Dessa forma, o nosso estudo explora o uso de LLMs abertos com engenharia de prompt, com e sem refinamento de parâmetros, para comparar as recomendações geradas pelos diferentes modelos. Além disso, avaliamos o impacto do ajuste fino com LoRA e como os resultados obtidos se comparam com os de um modelo fechado previamente

<sup>4</sup>NVIDIA – PEFT: <https://docs.nvidia.com/nemo-framework/user-guide/latest/llms/gemma/peft.html>

<sup>5</sup>MovieLens [Harper and Konstan 2015]: <https://grouplens.org/datasets/movielens/>

<sup>6</sup>Amazon Product Reviews [Hou et al. 2024a]: [https://cseweb.ucsd.edu/~jmcauley/datasets.html#amazon\\_reviews](https://cseweb.ucsd.edu/~jmcauley/datasets.html#amazon_reviews)

<sup>7</sup>MIND – Microsoft News Dataset [Wu et al. 2020]: <https://msnews.github.io/index.html>

utilizado na literatura. O método que utilizamos é uma adaptação do trabalho de Wang & Lim [Wang and Lim 2023] e consiste em usar prompts no estilo *zero-shot* para a tarefa de recomendação sequencial. Detalharemos esse método na Seção 3.

Além disso, buscamos responder se é possível simplificar a adoção de LLMs em recomendação ao utilizar um único formato de prompt com *fine-tuning* específico em múltiplos modelos, diferentemente do que tem sido notado na literatura em que múltiplos prompts e estratégias de refinamento são combinados, por exemplo, um modelo aberto com treino completo e outro com refinamento [Xu et al. 2023]. O conjunto de dados utilizado para isso foi o MovieLens e as métricas adotadas são HR@10 e NDCG@10, os quais detalharemos na Seção 4.

Embora trabalhos da literatura explorem diferentes tarefas e algoritmos de recomendação com conjuntos de dados distintos, nosso estudo foca em explorar o método de Wang & Lim aplicado a LLMs abertos para ampliar os resultados obtidos. Desta forma, mantivemos o método proposto pelos autores, incluindo aí o cálculo de similaridade e os prompts, de forma a permitir uma comparação mais direta com este trabalho.

### 3. Recomendação Zero-Shot de Próximo Item

Nesta seção, detalhamos a abordagem de recomendação que utilizamos em nosso estudo. Ela combina a eficiência da recomendação de próximo item com a flexibilidade das técnicas de prompt *zero-shot* para modelos de linguagem de larga escala (LLMs), resultando em uma abordagem efetiva para diversos cenários de recomendação. O método que usamos baseado nessa abordagem foi proposto por Wang & Lim [Wang and Lim 2023] e sua escolha se justifica pelo seu desempenho competitivo, demonstrado através de experimentos realizados pelos autores com *baselines* bem conhecidos de recomendação de próximo item. Apesar de o artigo original apresentar várias configurações, utilizamos em nosso trabalho somente aquela que obteve os melhores resultados nos experimentos nele reportados. O Algoritmo 1 apresenta uma descrição em alto nível deste método.

No Algoritmo 1, a função ZeroShotNIR é iniciada com dois parâmetros: *hist\_usuario*, que é o histórico do usuário alvo, ou seja, o conjunto de itens com os quais o usuário alvo da recomendação interagiu, e *todos\_itens*, que é o conjunto de todos os itens disponíveis para recomendação. Um primeiro prompt é usado (Linha 2) para solicitar ao LLM um sumário das características que são mais importantes para o usuário ao selecionar os itens, um resumo de suas preferências. O prompt é passado para o LLM (Linha 3) usando o histórico do usuário como contexto. Em seguida, um segundo prompt é usado (Linha 4) solicitando ao LLM que selecione deste histórico os itens mais condizentes com as preferências do usuário, apresentados em ordem decrescente de preferência. Isso é feito concatenando (“+”) a este prompt as preferências obtidas. O segundo prompt é passado para o modelo de linguagem (Linha 5) também usando o histórico do usuário como contexto. Um terceiro prompt é usado (Linha 7) solicitando ao modelo de linguagem que recomende  $N$  itens que sejam semelhantes aos itens representativos obtidos anteriormente. Este prompt é passado para o modelo de linguagem (Linha 8) para obter as recomendações, usando como contexto um conjunto chamado *Conjunto de Itens Candidatos*, que é gerado na Linha 6 pela função “ConjCandidatos” que explicaremos a seguir. As recomendações são formatadas (Linha 9) para apresentação final e retornadas formatadas (Linha 10).

Na Figura 2, ilustramos exemplos dos 3 prompts descritos acima considerando o domínio de recomendação de filmes, mais especificamente a coleção MovieLens, univer-

**Algorithm 1** Procedimento ZeroShotNIR

---

**Require:**  $hist\_usuario$   $\triangleright$  Conj. de itens com os quais o usuário interagiu  
**Require:**  $todos\_itens$   $\triangleright$  Conj. de todos os itens para recomendação  
**Ensure:**  $recs$   $\triangleright$  Lista de itens recomendados

- 1: **function** ZEROHOTNIR( $hist\_usuario, todos\_itens$ )
- 2:  $prompt \leftarrow$  "Quais características são mais importantes para mim ao selecionar itens (resuma minhas preferências brevemente)?"
- 3:  $preferencias \leftarrow$  LLM( $prompt, contexto = hist\_usuario$ )
- 4:  $prompt \leftarrow$  "Você irá selecionar os itens que mais me atraem apresentados em ordem decrescente de preferência" +  $preferencias$
- 5:  $itens\_representativos \leftarrow$  LLM( $prompt, contexto = hist\_usuario$ )
- 6:  $conj\_candidatos \leftarrow$  ConjCandidatos( $hist\_usuario, todos\_itens$ )
- 7:  $prompt \leftarrow$  "Recomende  $N$  itens do Conjunto de Candidatos semelhantes a" +  $itens\_representativos$
- 8:  $recs \leftarrow$  LLM( $prompt, contexto = conj\_candidatos$ )
- 9:  $recs\_extraidas \leftarrow$  formatar\_recomendacoes( $recs$ )
- 10: **return**  $recs\_extraidas$
- 11: **end function**

---

**Algorithm 2** Função ConjCandidatos (por Filtragem de Usuários)

---

**Require:**  $hist\_usuario$   $\triangleright$  Conj. de itens com os quais o usuário interagiu  
**Require:**  $itens$   $\triangleright$  Conj. de todos os itens para recomendação  
**Ensure:**  $conj\_candidatos$   $\triangleright$  Conjunto de itens candidatos

- 1: **function** CONJCANDIDATOS( $hist\_usuario, itens$ )
- 2: Seja  $usuarios$  o conjunto de todos os usuários
- 3: Seja  $M$  o número de usuários mais similares a serem selecionados
- 4: Seja  $S$  o número de itens mais populares a serem selecionados
- 5:  $conj\_candidatos \leftarrow \emptyset$
- 6:  $vetor\_usuario \leftarrow$  MULTIHOTVEC( $hist\_usuario$ )
- 7:  $similaridades \leftarrow \{\}$
- 8: **for** cada usuario em  $usuarios$  **do**
- 9:  $vetor\_outro \leftarrow$  MULTIHOTVEC( $usuario$ )
- 10:  $similaridade \leftarrow$  COS( $vetor\_usuario, vetor\_outro$ )
- 11:  $similaridades \leftarrow similaridades \cup \{usuario : similaridade\}$
- 12: **end for**
- 13:  $usuarios\_similares \leftarrow$  MAISSIMILARES( $similaridades, M$ )
- 14:  $itens\_populares\_usuarios \leftarrow$  ITENSPOPULARES( $usuarios\_similares, S$ )
- 15:  $conj\_candidatos \leftarrow conj\_candidatos \cup itens\_populares\_usuarios$
- 16: **return**  $conj\_candidatos$
- 17: **end function**

---

salmente utilizada em experimentos com sistemas de recomendação e adotada por Wang & Lim [Wang and Lim 2023]. Neste exemplo,  $N = 10$  itens são recomendados.

### Conjunto de Itens Candidatos

A ideia do conjunto de itens candidatos utilizada pelo método é fornecer ao LLM um subconjunto de itens que sejam mais relevantes para o usuário-alvo, dentre todos os itens possíveis, e ao mesmo tempo evitar usar uma lista de itens muito grande que ultrapasse o limite de tokens do prompt de cada LLM. Existem algumas opções para gerar o conjunto que satisfaça essa condições, mas em nosso trabalho utilizamos a *Filtragem de Usuários*, que nos experimentos reportados por Wang & Lim [Wang and Lim 2023] levou aos melhores resultados na recomendação.

A função ConjCandidatos é descrita em alto nível no Algoritmo 2. Ela inicializa o conjunto de candidatos como vazio (Linha 5). Em seguida, representa o histórico do usuário como um vetor *multi-hot* (Linha 6) e inicializa um conjunto de similaridades vazio (Linha 7). Um vetor *multi-hot* é uma representação binária de um conjunto de itens onde cada posição do vetor corresponde a um item específico, e o valor é 1 se o usuário interagiu com o item e 0 caso contrário. Esta representação é útil para calcular similaridades entre usuários, permitindo comparar vetores para identificar usuários similares com base em interações comuns. Para cada usuário no conjunto de usuários (Linha 9), o vetor do

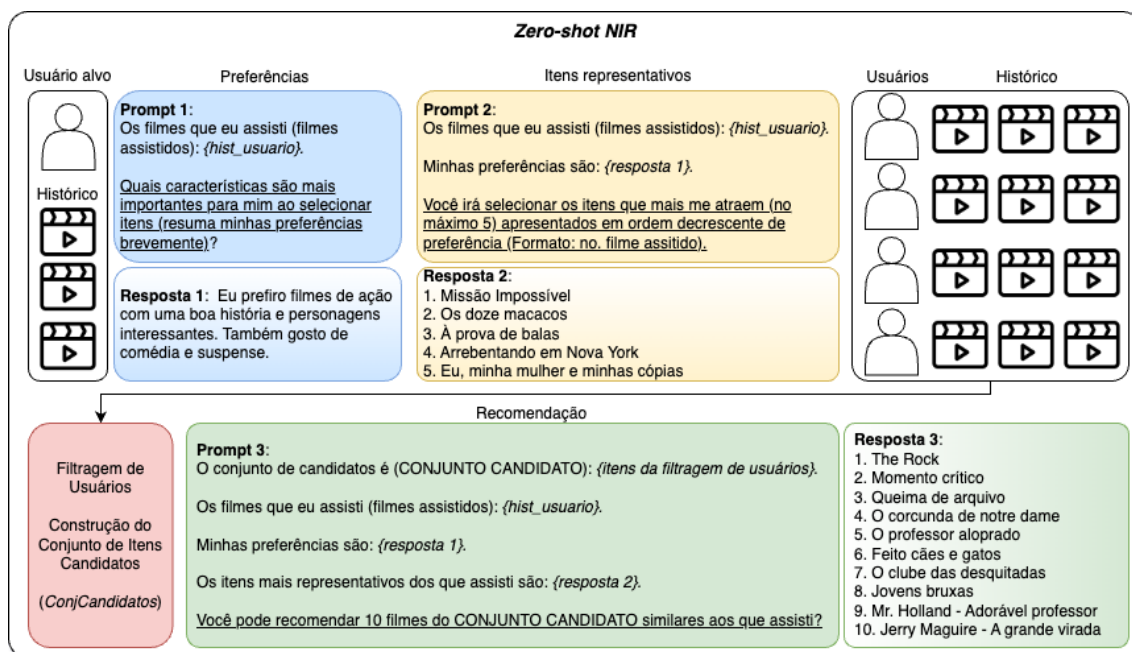


Figura 2. Exemplo de prompt utilizado para recomendação

usuário é representado como um vetor *multi-hot*, e a similaridade de cosseno é calculada entre o vetor do usuário-alvo e o vetor do outro usuário (Linha 10). As similaridades são armazenadas em um conjunto (Linha 11). Os  $M$  usuários mais similares ao usuário-alvo são então selecionados (Linha 13), e os  $S$  itens mais populares, ou seja, os que têm a maior frequência de interações entre esses usuários são identificados (Linha 14). Esses itens populares são adicionados ao conjunto de candidatos (Linha 15). Finalmente, a função retorna o conjunto de candidatos (Linha 16).

#### 4. Experimentos com Modelos de Linguagem Abertos

Nesta seção, detalhamos os experimentos realizados utilizando diferentes modelos de linguagem abertos para a tarefa de recomendação de próximo item.

##### 4.1. Coleção MovieLens

Assim como Wang & Lim [Wang and Lim 2023], usamos a coleção MovieLens 100k [Harper and Konstan 2015] nos experimentos, pois é bastante utilizada em trabalhos de recomendação da literatura. A coleção é composta por 100 mil avaliações com notas de 1 a 5 de 943 usuários sobre um acervo de 1682 filmes, onde cada usuário tem pelo menos 20 filmes avaliados<sup>8</sup>.

No trabalho de Wang & Lim [Wang and Lim 2023], o conjunto de itens candidatos (Algoritmo 2) foi gerado com os  $M=12$  usuários mais similares e os  $S=19$  itens mais populares desses usuários. Esse tamanho equilibra diversidade e simplicidade, otimizando o desempenho do GPT-3. Assim, apenas 170 dos 943 usuários da coleção foram mantidos, pois somente para esses usuários o item correto estava presente no conjunto de itens candidatos. O demais usuários foram descartados, pois o LLM não seria capaz de recomendar o item correto que não estivesse presente no conjunto de itens candidatos.

Em nosso trabalho, como experimentamos com outros LLMs, além dessa configuração, que chamaremos de *Original*, usamos uma outra configuração, que chamaremos

<sup>8</sup>Coleção disponível em: <https://github.com/AGI-Edgerunners/LLM-Next-Item-Rec>

de *Ampliada*, com  $M=14$  usuários mais similares e  $S=38$  itens mais populares desses usuários. Isso nos permitiu utilizar um conjunto maior de 264 usuários da coleção. Escolhemos  $M=14$  porque observamos que aumentos adicionais em  $M$  não resultavam em ganhos significativos. Além disso,  $S=38$  foi usado pois além deste valor o tamanho da amostra da coleção não aumentava significativamente, permitindo assim um equilíbrio entre diversidade e eficiência nas recomendações.

## 4.2. Avaliação e Métricas

A tarefa de recomendação sequencial, ou de próximo item, recebe o histórico de avaliações do usuário e visa prever qual será o próximo item a ser consumido por ele. Assim, nossa avaliação é baseada na estratégia *leave-one-out*, em que um histórico de  $n-1$  itens já consumidos de um usuário é utilizado para recomendar o próximo item a ser consumido, ou seja, o  $n$ -ésimo item. Assim, o gabarito (*ground-truth*) usado para nossa avaliação consiste no  $n$ -ésimo item consumido por cada um dos usuários.

Nessa avaliação são tipicamente utilizadas métricas que capturam a acurácia do modelo na sugestão de itens que um usuário provavelmente consumirá e elas devem considerar a relevância e a ordem das recomendações. Sendo assim, em nossos experimentos foram adotadas as métricas  $HR@K$  e  $NDCG@K$ , com  $K=10$ , ou seja, consideramos a recomendação de 10 itens.

A métrica  $HR@K$  (*Hit Ratio at K*) representa a proporção de vezes em que o item verdadeiro (*ground-truth*) está entre os top  $K$  itens recomendados. Essa métrica reflete a habilidade do recomendador em posicionar corretamente os itens mais relevantes para o usuário dentre uma lista de recomendações.

De maneira complementar, a métrica  $NDCG@K$  (*Normalized Discounted Cumulative Gain at K*) vai além, pois considera a presença do item relevante e a sua posição na lista de recomendação. Quanto mais próximo do topo da lista o item relevante estiver, melhor. Essa abordagem proporciona uma avaliação mais refinada da acurácia das recomendações, pois a ordem das recomendações impacta na experiência dos usuários.

## 4.3. Modelos Abertos Utilizados

Utilizamos em nossos experimentos os LLMs *Falcon*, *Llama* e *Mistral*, que se destacaram em momentos distintos no ano de 2023 quando foram inicialmente lançados. Assim, é razoável supor que os conjuntos de dados nos seus treinamentos incluem informações sobre os filmes presentes na coleção MovieLens 100k, que foi construída entre 1997 e 1998. Como vários modelos de linguagem deste tipo, estes são baseados em uma arquitetura de transformadores auto-regressivos pré-treinados do tipo *decoder-only*, em que apenas o componente decodificador é utilizado. Este também é o caso do modelo GPT-3<sup>9</sup> que foi usado por Wang & Lim e que, portanto, serviu de *baseline* para o nosso estudo.

A Tabela 1 apresenta uma comparação resumida dos LLMs utilizados em nossos experimentos. Nela, destacamos os desenvolvedores, tamanhos disponíveis, contexto de tokens e suas principais características. Para poupar espaço, os detalhes das características foram omitidos, e encorajamos o leitor a consultar os links fornecidos para uma descrição mais completa e aprofundada de cada modelo, Falcon<sup>10</sup>, Llama<sup>11</sup> e Mistral<sup>12</sup>.

<sup>9</sup>Improving Language Understanding by Generative Pre-Training – [https://cdn.openai.com/research-covers/language-unsupervised/language\\_understanding\\_paper.pdf](https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf)

<sup>10</sup>Disponível em: <https://huggingface.co/tiiuae/falcon-7b-instruct>

<sup>11</sup>Disponível em: <https://huggingface.co/meta-llama>

<sup>12</sup>Disponível em: <https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.2>



**Tabela 1. Comparação dos Modelos Utilizados**

Modelo	Desenvolvedor	Tamanho	Contexto	Versão	Características
Falcon	Technology Innovation Institute	7B, 40B, 180B	2.048	Falcon 7B Instruct	Embeddings Posicionais Rotativos (RoPE), Otimizações Hiperparamétricas, Adaptado do PaLM
Llama	Meta	7B, 8B, 13B, 70B	4.096 (v2), 8.000+ (v3)	Llama 2-7B Chat HF, Llama 2-13B Chat HF, Llama 3-8B Instruct	Atenção Agrupada por Consulta (GQA), Tokenizador mais eficiente (Llama 3), Janela de contexto ampliada (Llama 3)
Mistral	Mistral AI	7B, 8x7B, 8x22B	4.096	Mistral 7B Instruct	Atenção Agrupada por Consulta, Atenção em Janela Deslizante (SWA), Modelos de Mistura de Especialistas Esparsos (SMoE)

#### 4.4. Ajuste Fino

Neste estudo, além dos LLMs pré-treinados descritos acima, usamos versões obtidas com ajustes finos desses modelos. Ajuste fino é o processo de adaptar um modelo pré-treinado a um conjunto de dados específico para melhorar seu desempenho em uma tarefa particular. Em nosso caso, investigamos se esse processo traz melhorias para a recomendação. Para o primeiro experimento de ajuste fino, o conjunto de dados foi dividido em 80% dos dados para treinamento e 20% para teste tomados aleatoriamente (80/20). Para o segundo experimento de ajuste fino, optou-se por usar validação cruzada (VC), obtendo 4 subconjuntos de mesmo tamanho para permitir uma avaliação mais robusta da estabilidade do desempenho dos modelos ajustados. Além disso, em ambos os experimentos os LLMs foram refinados utilizando a técnica de *Low Rank Adaptation (LoRA)* [Hu et al. 2022] (ver Seção 2).

A Tabela 2 apresenta os números das instâncias utilizadas no ajuste fino, considerando as configurações Original e Ampliada.

**Tabela 2. Instâncias utilizadas no ajuste fino.**

Tipo de Ajuste Fino	Configuração			
	Original		Ampliada	
	Treino	Teste	Treino	Teste
80/20	136	34	211	53
VC (por Fold)	128	42	198	66

Em todos os casos, o mesmo conjunto de hiperparâmetros foi utilizado para garantir a consistência e replicabilidade dos experimentos. Os hiperparâmetros adotados foram *temperature* = 0.1 para produzir respostas mais conservadoras, *penalty* = 1.15 para evitar repetições frequentes e *epochs* = 3, número de épocas de treinamento no ajuste fino, devido ao tamanho reduzido das amostras. Os demais hiperparâmetros adotados no ajuste fino podem ser consultados no repositório disponibilizado<sup>3</sup>.

### 5. Resultados

Nesta seção, apresentamos os resultados experimentais dos LLMs, tanto pré-treinados quanto gerados por ajuste fino, na tarefa de recomendação Zero-Shot de Próximo Item.

A Tabela 3 apresenta os resultados dos modelos pré-treinados. Na esquerda, estão os resultados para todo o *dataset*, e na direita, apenas para as amostras em que o *ground-truth* está na lista de itens candidatos. O melhor resultado está em **negrito** e o segundo melhor sublinhado. Note que Wang & Lim não testaram a configuração ampliada. O Llama 3-8B obteve os melhores scores de HR@10, superando o *baseline* GPT-3 no conjunto de dados completo e na amostra original usada por Wang & Lim. Na configuração ampliada, o Llama 3-8B também foi superior. Analisando o NDCG@10, observa-se que o Llama 2-13B e o Llama 3-8B superaram o *baseline*, ou seja, ainda que a taxa de acerto do GPT-3 tenha sido superior ao Llama 2-13B em todo dataset, o Llama 2-13B posicionou o item relevante (*ground-truth*) mais próximo do topo da lista de recomendações.

**Tabela 3. Resultados dos Experimentos de Recomendação Zero-Shot.**

Modelo	Todo o Dataset				Amostras com <i>Ground-truth</i>			
	Original		Ampliada		Original		Ampliada	
	HR	NDCG	HR	NDCG	HR	NDCG	HR	NDCG
GPT3 (baseline)	0.1187	0.0546	-	-	0.6706	-	-	-
Llama 3 8B	<b>0.1283</b>	0.0566	<b>0.1018</b>	0.0359	<b>0.7059</b>	<b>0.3118</b>	<b>0.3471</b>	0.1136
Llama 2 13B Chat HF	0.1166	<b>0.0577</b>	0.0923	<b>0.0414</b>	0.6235	0.3075	0.3118	<b>0.1446</b>
Llama 2 7B Chat HF	0.1018	0.0466	0.0600	0.0283	0.5588	0.2566	0.1882	0.0948
Mistral 7B Instruct v0.2	0.0954	0.0513	0.0774	0.0320	0.5059	0.2727	0.3059	0.1233
Falcon 7B Instruct	0.0424	0.0191	0.0435	0.0199	0.1914	0.0907	0.1588	0.0756

Ao comparar os resultados das duas Configurações, observa-se uma queda significativa na Configuração Ampliada, onde o conjunto de itens candidatos inseridos no prompt é maior. Esta redução de desempenho deve-se ao aumento de opções disponíveis para recomendação, o que, embora aumente a diversidade, reduz a probabilidade de acerto dos modelos.

Nos experimentos, notou-se que alguns modelos geravam menos de 10 recomendações, o que pode ser atribuído a um *trade-off* relacionado à serendipidade dos modelos. Isso está relacionado com os hiperparâmetros que usamos e que conferem aos modelos uma certa liberdade de inferência, permitindo que eles gerem respostas variadas para um mesmo *prompt*. Essa abordagem, contudo, pode aumentar a ocorrência de respostas fora do esperado.

A Tabela 4 apresenta os resultados dos experimentos com ajuste fino 80/20. Por concisão, apresentamos apenas os resultados de HR@10. Houve melhoria no desempenho geral em comparação com os modelos pré-treinados. No entanto, os modelos tiveram desempenho inferior na configuração ampliada, onde o conjunto de filmes candidatos é maior. Os resultados dos modelos Llama 3-8B e Llama 2-13B foram inferiores aos obtidos sem o ajuste fino, devido à distribuição de itens-alvos (*ground-truth*), sem repetições nas respostas, dada a esparsidade do dataset. Cada caso de teste apresenta um item-alvo diferente, dificultando a generalização e a aprendizagem de padrões.

**Tabela 4. Resultados dos Experimentos de Ajuste Fino (80-20) - HR@10.**

Modelo	Config. Original	Config. Ampliada
Llama 3 8B	0.5999	0.2841
Llama 2 13B Chat HF	0.5594	<b>0.3371</b>
Llama 2 7B Chat HF	0.5936	0.2500
Mistral 7B Instruct v0.2	<b>0.6056</b>	0.2879
Falcon 7B Instruct	0.1999	0.1477

O *boxplot* apresentado na Figura 3 resume os resultados dos experimentos com os modelos gerados por ajuste fino usando validação cruzada. Para a Configuração Original, o Mistral 7B apresentou o melhor desempenho, com HR@10 médio de 0.6056, seguido pelos modelos Llama 3-8B e Llama 2-7B com 0.5999 e 0.5936, respectivamente. Além disso, o Mistral apresentou menor variação nos resultados, indicada pelo desvio padrão e distância inter-quartis, quando comparado com o Llama 2-7B. Isso sugere que apesar dos resultados semelhantes, o Mistral 7B foi mais consistente em suas recomendações, perdendo neste quesito apenas para o Llama 3-8B. Para a Configuração Ampliada, o melhor desempenho foi o do Llama 2-13B com HR@10 médio de 0.3371. Neste segundo cenário, o Mistral 7B apresentou maior variação nos resultados, enquanto que os modelos Llama 2-13B e Llama 3-8B foram mais consistentes. O Falcon 7B teve pouca variação em ambas as configurações, embora seu desempenho absoluto tenha sido o pior. Os

modelos Llama apresentaram variações moderadas, com Llama 3-8B mostrando a maior consistência entre os três, especialmente na Configuração Original.

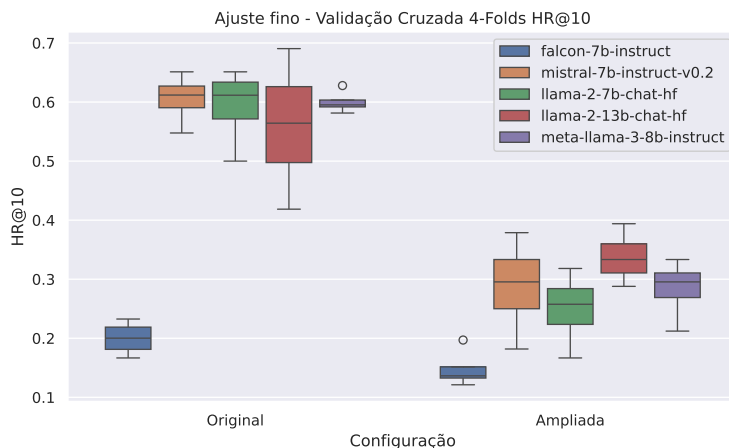


Figura 3. Resultados com ajuste fino por validação cruzada.

## 6. Conclusão e Trabalhos Futuros

Neste artigo, apresentamos um estudo experimental sobre a eficácia da aplicação de LLMs abertos na tarefa de recomendação sequencial em cenário *zero-shot*, usando modelos pré-treinados e posteriormente adaptados com ajuste fino. Dentre os LLMs utilizados, o Llama 3-8B superou a acurácia do GPT-3, modelo proprietário usado aqui como *baseline*, na geração de respostas com o item relevante nas recomendações, mostrando-se eficaz e eficiente na tarefa, e foi estável na geração das respostas nos experimentos de ajuste fino com validação cruzada.

A análise ampliada que realizamos, com duas configurações experimentais, permitiu verificar como a seleção e composição do conjunto de itens candidatos à recomendação influenciam o desempenho dos modelos. Observamos que ampliar o conjunto de candidatos aumentou a diversidade, mas reduziu a eficácia das recomendações, pois aumenta as possibilidades de escolha e combinações no processo de inferência.

Os experimentos de ajuste fino apresentaram melhoras em alguns modelos, mas não houve impacto significativo devido às características da coleção experimental. Isso reforça que a qualidade e o volume dos dados são determinantes para que a melhoria seja expressiva. Além disso, a utilização da validação cruzada permitiu avaliar a capacidade de generalização dos modelos em diferentes amostras, contribuindo para uma análise mais ampla da estabilidade e robustez dos modelos na tarefa de recomendação.

Futuros trabalhos focarão na avaliação da generalização dos resultados do método com outras coleções de diferentes domínios de aplicação, a fim de verificar a aplicabilidade dos LLMs abertos em variados contextos. A combinação de abordagens, como a filtragem colaborativa e outros métodos de recomendação, também será explorada para reduzir o número de itens candidatos visando aumentar a precisão e balancear a diversidade das recomendações. Além disso, pretendemos estudar e aplicar técnicas para reduzir o volume de respostas não-plausíveis geradas pelos modelos e aumentar a qualidade das recomendações. Por fim, pretendemos realizar avaliações contínuas de novos LLMs abertos para identificar avanços e novos cenários de aplicação, bem como, a evolução do desempenho desses modelos na tarefa de recomendação.

## Agradecimentos

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES-PROEX) - Código de Financiamento 001. Este trabalho foi parcialmente financiado pela Fundação de Amparo à Pesquisa do Estado do Amazonas – FAPEAM – por meio do projeto POSGRAD 2024/2025 e do Projeto NeuralBond (UNIVERSAL 2023 Proc. 01.02.016301.04300/2023-04); pelo CNPq no âmbito do Projeto IAIA (406417/2022-9) e uma bolsa individual Altigran da Silva (307248/2019-4); e pela FAPESP/MCTIC/CGI através do Centro de Inovação em Inteligência Artificial para a Saúde (CIIA-Saúde) (Proc. 2020/09866-4).

## Referências

- Almazrouei, E. et al. (2023). The falcon series of open language models. arXiv, cs.CL, 2311.16867.
- Bao, K. et al. (2023). TALLRec: An Effective and Efficient Tuning Framework to Align Large Language Model with Recommendation. In *Proc. of the 17th ACM Conf. on Recommender Systems (RecSys)*, p. 1007–1014.
- Brown, T. B. et al. (2020). Language models are few-shot learners. In *Proc. of the 34th Intl. Conf. on Neural Information Processing Systems (NeurIPS)*, p. 1877–1901.
- Dai, S. et al. (2023). Uncovering ChatGPT’s Capabilities in Recommender Systems. In *Proc. of the 17th ACM Conf. on Recommender Systems (RecSys)*, p. 1126–1132.
- Fan, W. et al. (2023). Recommender Systems in the Era of Large Language Models (LLMs). arXiv, cs.IR, 2307.02046.
- Harper, F. M. and Konstan, J. A. (2015). The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.*, vol. 5, n. 4, p. 1–19.
- Hou, Y. et al. (2024a). Bridging Language and Items for Retrieval and Recommendation. arXiv, cs.IR, 2403.03952.
- Hou, Y. et al. (2024b). Large Language Models are Zero-Shot Rankers for Recommender Systems. In *Proc. of the 46th European Conf. on Information Retrieval (ECIR)*, p. 364–381.
- Houlsby, N. et al. (2019). Parameter-Efficient Transfer Learning for NLP. In *Proc. of the 36th Intl. Conf. on Machine Learning (ICML)*, p. 2790–2799.
- Hu, E. J. et al. (2022). LoRA: Low-Rank Adaptation of Large Language Models. In *Proc. of the 10th Intl. Conf. on Learning Representations (ICLR)*, p. 1–13.
- Jiang, A. Q. et al. (2023). Mistral 7B. arXiv, cs.CL, 2310.06825.
- Liu, J. et al. (2023). Is ChatGPT a Good Recommender? A Preliminary Study. arXiv, cs.IR, 2304.10149.
- Liu, Q. et al. (2024). ONCE: Boosting Content-based Recommendation with Both Open- and Closed-source Large Language Models. In *Proc. of the 17th ACM Intl. Conf. on Web Search and Data Mining (WSDM)*, p. 452–461.
- Lyu, H. et al. (2023). LLM-Rec: Personalized Recommendation via Prompting Large Language Models. arXiv, cs.CL, 2307.15780.
- Rajput, S. et al. (2023). Recommender Systems with Generative Retrieval. In *Proc. of 37th Conf. on Neural Information Processing Systems (NeurIPS)*, p. 1–17.

- Sanner, S. et al. (2023). Large Language Models are Competitive Near Cold-start Recommenders for Language- and Item-based Preferences. In *Proc. of the 17th ACM Conf. on Recommender Systems (RecSys)*, p. 890–896.
- Shao, B., Li, X., and Bian, G. (2021). A survey of research hotspots and frontier trends of recommendation systems from the perspective of knowledge graph. *Expert Systems with Applications*, 165, p. 113764.
- Touvron, H. et al. (2023). Llama 2: Open Foundation and Fine-Tuned Chat Models. arXiv, cs.CL, 2307.09288.
- Wang, L. and Lim, E.-P. (2023). Zero-Shot Next-Item Recommendation using Large Pretrained Language Models. arXiv, cs.IR, 2304.03153.
- Wu, F. et al. (2020). MIND: A large-scale dataset for news recommendation. In *Proc. of the 58th Annual Meeting of the Association for Computational Linguistics*, p. 3597–3606.
- Xu, S. et al. (2023). OpenP5: An Open-Source Platform for Developing, Training, and Evaluating LLM-based Recommender Systems. arXiv, cs.IR, 2306.11134.
- Zhang, J. et al. (2023). Recommendation as Instruction Following: A Large Language Model Empowered Recommendation Approach. arXiv, cs.IR, 2305.07001.