Recomendação de Próximo Item usando Modelos de Linguagem Abertos

Marcos A. P. de Lima[1]

[1]Instituto de Computação – Universidade Federal do Amazonas (UFAM)
Manaus – Amazonas – Brazil

marcos.lima@icomp.ufam.edu.br

Resumo. *Este artigo aborda a Recomendação de Próximo Item por meio da utilização de Modelos de Linguagem Abertos, com foco nos modelos* Llama2 7b *e* Llama2 13b GPTQ*, aplicados ao conjunto de dados* MovieLens 100k *para recomendação de filmes. Os experimentos compreendem três etapas distintas de engenharia de prompt. Inicialmente, aos modelos foi fornecido um conjunto de filmes candidatos, solicitando então um resumo das preferências do usuário-alvo. Na segunda etapa, as preferências geradas foram empregadas para selecionar, dentre o conjunto de filmes candidatos, aqueles mais representativos para o perfil do usuário-alvo. Por fim, as informações das etapas anteriores foram combinadas em um único prompt, requisitando do modelo a geração de uma lista de 10 filmes recomendados para avaliação. Os modelos alcançaram um valor de* Hit@10 *de 0.4457 e 0.6353, respectivamente, em uma amostra de 170 usuários dos 943 presentes no conjunto de dados. Esses resultados indicam a eficácia dos modelos abertos na tarefa de recomendação e a utilidade da abordagem de engenharia de prompt para aprimorar o processo de geração de recomendações.*

1. Introdução

Sistemas de Recomendação são ferramentas e técnicas de software que oferecem sugestões de itens que podem ser úteis para um usuário [Resnick and Varian 1997, Mahmood and Ricci 2009, Ricci et al. 2011]. Um dos primeiros trabalhos encontrados foi o *Tapestry* [Goldberg et al. 1992], sistema desenvolvido em 1992 por pesquisadores da Xerox PARC para filtragem de documentos digitais de forma colaborativa por meio de anotações dos usuários. Em 1994 foi apresentado o *GroupLens* [Resnick et al. 1994], um sistema de recomendação de notícias baseado nas avaliações dos usuários numa escala de cinco níveis. Mais tarde o projeto *GroupLens* daria origem ao conjunto de dados *MovieLens 100k* adotado neste estudo. Outro trabalho que também ajudou a disseminar a área de Sistemas de Recomendação foi o *RINGO* [Shardanand and Maes 1995], que recomendava músicas por meio das preferências explicitamente fornecidas pelo usuários. Mais tarde essa tecnologia foi adotada por empresas como Yahoo!, ZDNet e Barnes and Noble.

Atualmente são amplamente utilizados no comércio eletrônico, serviços de streaming, redes sociais e muito mais, por entregar conteúdo mais relevante e melhorar a experiência do usuário. Existem muitas abordagens para implementação desses sistemas, podendo ser divididos em três categorias principais: métodos baseados em conteúdo, filtragem colaborativa e abordagens híbridas. Os sistemas de recomendação por filtragem colaborativa recomendam itens para um usuário com base nas preferências

de usuários semelhantes [Hu et al. 2008, Sarwar et al. 2001]. De modo análogo, os sistemas de recomendação baseados em conteúdo também utilizam preferências, neste caso do próprio usuário, junto com as características dos itens para gerar suas recomendações [Lops et al. 2011, Pazzani and Billsus 2007]. Por fim, os sistemas de recomendação híbridos combinam múltiplos métodos de recomendação para melhorar a precisão e a diversidade das recomendações.

O desempenho dos sistemas de recomendação está diretamente ligado à quantidade de dados de treinamento disponíveis, criando assim, segundo [Ding et al. 2021], um dilema para produtos em estágios iniciais. Como forma de tentar resolver esse problema, algoritmos chamados *Zero-Shot* estão sendo explorados por permitirem a generalização de um conjunto de dados antigo para um novo conjunto de dados.

Recentemente, o uso de *Large Language Models* (LLMs) em sistemas de recomendação têm proporcionado um avanço significativo. Modelos como *GPT-3* [Radford et al. 2018] demonstram habilidade em padrões contextuais e nuances semânticas em extensos dados textuais. Além disso, estudos como o de [Sun et al. 2019] que exploram a aplicação de modelos como *BERT* [Devlin et al. 2019] em sistemas de recomendação sequencial, demonstram a capacidade de melhora na precisão das recomendações. Por fim, pesquisas recentes, como o trabalho de [Wang and Lim 2023] trazem uma estratégia para recomendação de itens em cenários *zero-shot* por meio de *engenharia de prompt*.

Neste trabalho, serão replicados os experimentos de [Wang and Lim 2023] usando *LLMs abertos*. O modelo *Llama2* [Touvron et al. 2023] foi escolhido por também apresentar três variações de parâmetros (7 bilhões, 13 bilhões e 70 bilhões) o que possibilita uma comparação inicial em termos de performance e complexidade. Além disso, segundo os autores, o z*Llama2* superou os demais concorrentes na maioria dos *benchmarks* executados.

2. Objetivos

Este trabalho visa investigar a viabilidade da Recomendação de próximo item por meio de modelos abertos. Além do objetivo central, propomos também dois objetivos secundários. Primeiramente, realizaremos uma análise comparativa entre duas versões do modelo selecionado, *Llama2 7b* e *Llama2 13b*, avaliando tanto à assertividade das recomendações quanto o custo computacional associado. Por fim, faremos um confronto dos resultados alcançados com os resultados previamente apresentados por [Wang and Lim 2023].

3. Metodologia

3.1. Dataset

O dataset utilizado foi o *MovieLens 100K* [Harper and Konstan 2015], cujos dados foram coletados pelo Projeto de Pesquisa *GroupLens* e incluem 100.000 avaliações de 943 usuários sobre 1.682 filmes. Cada usuário avaliou pelo menos 20 filmes, e informações demográficas simples, como idade, gênero, ocupação e código postal, estão incluídas. A coleta de dados ocorreu por meio de um site durante um período de sete meses, de 19 de setembro de 1997 a 22 de abril de 1998. O conjunto de dados passou por uma limpeza, removendo usuários com menos de 20 avaliações ou informações demográficas incompletas.

3.2. Amostragem

Dada a complexidade computacional associada ao processamento integral do conjunto de dados, decidimos conduzir as análises em uma amostra aleatória, selecionando 170 casos de teste dos 943 disponíveis no conjunto de dados *MovieLens 100k*. Essa estratégia busca permitir uma análise inicial e também resultados primários em tempo hábil.

Além disso, o conjunto de dados *MovieLens 100k* apresenta 1.683 filmes, quantidade essa impossível de ser colocada num único prompt, pois os modelos de linguagem trabalham com janelas de contexto, sendo incapazes de trabalhar com todo o universo de itens. Portanto, optou-se por construir um *conjunto de filmes candidatos* para cada usuário-alvo que atende-se a dois critérios:

- ser mais relevantes para o usuário em vez de filmes selecionados aleatoriamente.
- o tamanho do conjunto deve ser pequeno o suficiente para se adequar a um prompt.

Para atender a esses critérios, empregamos o método de *filtragem por usuário*. Essa abordagem fundamenta-se na premissa de que os *filmes candidatos* devem ser apreciados por usuários de preferências similares com o usuário-alvo. Para realizar isso, representamos cada usuário por meio de um *vetor multi-hot* que reflete seus filmes vistos, identificamos usuários semelhantes por meio da medida de similaridade de cosseno e, posteriormente, selecionamos os 12 usuários mais próximos. O *conjunto de filmes candidatos*, composto por 19 itens, é constituído pela escolha dos filmes mais populares entre aqueles interagidos por esses 12 usuários semelhantes. Essa estratégia de *filtragem por usuário* contribui para refinar o escopo da recomendação, considerando as preferências de usuários cujos padrões de visualização se assemelham ao usuário-alvo.

3.3. Engenharia de prompt

Adotando a metodologia apresentada por [Wang and Lim 2023], nosso pipeline de recomendação foi estruturado em três etapas. Inicialmente, realiza-se a coleta das preferências do usuário-alvo, proporcionando uma compreensão aprofundada de suas inclinações cinematográficas. Em seguida, efetua-se a classificação do *conjunto de filmes previamente vistos*, utilizando como critério as preferências identificadas na etapa anterior. Por último, requisitamos do modelo uma recomendação de 10 filmes similares, cuidadosamente selecionados a partir do *conjunto de filme candidatos*.

Na identificação das preferências do usuário-alvo, empregamos um template de prompt preenchido com o *conjunto de filmes candidatos* e o *conjunto de filmes previamente vistos* pelo próprio usuário-alvo. Dessa maneira, cada rodada de solicitação foi capaz de capturar nuances nas preferências, estabelecendo assim as bases para as etapas subsequentes do processo de recomendação.

> Candidate Set (candidate movies): ...
> The movies I have watched (watched movies): ...
> Step 1: What features are most important to me when selecting movies (Summarize my preferences briefly)?

Na etapa subsequente, desenvolvemos um novo prompt baseado no *conjunto de filmes candidados*, no *conjunto de filmes previamente vistos*, nas *preferências do usuário-alvo* para então gerar um segundo prompt instruindo o modelo a gerar uma lista composta por até cinco filmes que melhor refletem as preferências do usuário-alvo:

Candidate Set (candidate movies): ...
The movies I have watched (watched movies): ...
Step 1: What features are most important to me when selecting movies (Summarize my preferences briefly)?
Answer: ...
Step 2: Create an enumerated list selecting the most featured movies from the watched movies according to my preferences.

Por último, na fase conclusiva, elaboramos um terceiro prompt, mantendo quase toda base do template empregado na etapa anterior, agora ampliado pela inclusão do *conjunto de filmes mais representativos*. Nesse estágio, instruímos explicitamente o modelo a gerar uma lista composta por 10 filmes que estão no *conjunto de filmes candidatos* com base no *conjunto de filmes mais representativos*.

Candidate Set (candidate movies): ...
The movies I have watched (watched movies): ...
Step 1: What features are most important to me when selecting movies (Summarize my preferences briefly)?
Answer: ...
Step 2: Selecting the most featured movies (at most 5 movies) from the watched movies according to my preferences in descending order.
Answer: ...
Step 3: Can you recommend 10 movies from the "Candidate Set" similar to the previous selected movies list I've watched? For each recommended movie use format ("Recommended movie" # "Similar movie").

4. Experimentos

Os experimentos foram conduzidos no ambiente *Google Colaboratory* num backend de 50GB de RAM e 16GB de GPU RAM. Ambos os modelos avaliados foram carregados da plataforma *HuggingFace*[1].

Ambos os modelos utilizaram o mesmo conjunto de hiperparâmetros:

- **do_sample=True**: para que a seleção do próximo token ocorra por meio de uma distribuição de probabilidades.
- **temperature=0.1**: para que a escolha ocorra mais focada em tokens com maior probabilidade.
- **repetition_penalty=1.15**: para penalizar a repetição de tokens.
- **max_new_tokens=1024**: para garantir que a resposta gerada pelo modelo seja completa (alguns nomes de filmes são muito longos).
- **top_k=50**: para considerar somente os top 50 tokens na amostra.

4.1. Llama2 7b

O modelo Llama2 7b (7 bilhões de parâmetros) utilizado é disponibilizado dentro do repositório da *Meta*[2].

[1]https://huggingface.co/
[2]https://huggingface.co/meta-llama/Llama-2-70b

4.2. Llama2 13b

O modelo Llama 2 13b (13 bilhões de parâmetros) precisou ser *quantizado* para reduzir o custo computacional e possibilitar os experimentos. Sendo assim, foi utilizado o modelo já quantizado e disponibilizado dentro do repositório *TheBloke*[3]. Esse modelo foi quantificado usando o algoritmo *GPTQ* [Frantar et al. 2023].

5. Resultados

Para a avaliação dos modelos propostos, optamos pela métrica *Hit@10* por ser uma escolha frequente na análise de sistemas de recomendação, por ser capaz de representar o desempenho dos modelos de recomendação num único valor e também por sua adoção no trabalho baseline.

A Tabela 1 apresenta os resultados obtidos pelos modelos avaliados. Analisando os valores de *Hit@10*, ambos os modelos implementados e avaliados neste estudo foram capazes de superar os resultados obtidos por [Wang and Lim 2023]. Além disso, comparando o tempo necessário para executar os testes na amostra, temos que o modelo *Llama2 13b GPTQ* embora mais custoso computacionalmente ainda assim foi quase duas vezes mais rápido na execução de nosso pipeline.

Model	**Hit@10**	**Tempo (min)**	Prompt/Min
NIR-Multi-UF	0.1187	-	-
Llama-2-7b	0.4457	223	0.7623
Llama-2-13b GPTQ	0.6353	142	1.1972

Table 1. Resultados dos experimentos

Adicionalmente, a Figura 5 mostra o valor acumulado de *Hit@10* no decorrer dos experimentos. Podemos perceber uma estabilização no valor quando passamos da metade da amostra, o que poderia sugerir que os testes realizados em todo o conjunto de dados resultariam num valor de *Hit@10* próximo de 0.6353.

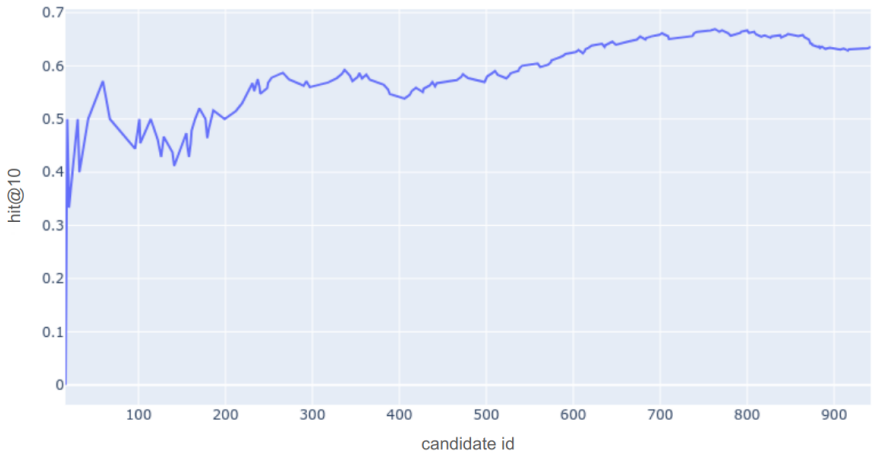

Figure 1. Llama2 13B GPTQ - Hit@10 acumulado

[3]https://huggingface.co/TheBloke/Llama-2-13B-chat-GPTQ

6. Premissas e limitações

Primeiramente, é importante enfatizar que a métrica *Hit@10* possui limitações. Por exemplo, não leva em consideração a posição exata do item relevante na lista (se está em 1º ou 10º lugar), e não pondera a relevância dos itens, tratando todos os acertos da mesma forma. Além disso, não existe ponderação entre erros e acertos. Portanto, outras métricas, como *NDCG* (normalized discounted cumulative gain), *F1*, *Precisão*, *Revocação* e *MAP* (mean average precision), podem ser usadas em conjunto para fornecer uma visão mais abrangente do desempenho do sistema de recomendação.

A seleção aleatória de uma amostra de 170 usuários de um total de 943 introduz considerações importantes sobre a representatividade e generalização dos resultados obtidos. Reconhecendo as limitações inerentes a essa abordagem, é sugerido que análises estatísticas, como intervalos de confiança e testes de significância, sejam aplicadas para validar os resultados em relação à totalidade da população. Portanto, as conclusões derivadas desse estudo podem estar sujeitas a limitações inerentes à representatividade da amostra.

Por fim, este estudo avaliou os uso de *LLMs* para sistemas de recomendação sem realizar *fine-tuning* nos hiperparâmetros. Isso pode levar a desempenho subótimo, generalização limitada e sensibilidade aos dados de treinamento. Recomenda-se a exploração de diferentes combinações de hiperparâmetros para mitigar essas limitações.

7. Conclusão

Este estudo demonstrou a eficácia e viabilidade da recomendação de próximo item utilizando modelos de linguagem abertos, especificamente os modelos *Llama2 7b* e *Llama2 13b GPTQ*, aplicados ao conjunto de dados *MovieLens 100k*. Através de experimentos conduzidos com uma amostra de 170 casos de teste do conjunto de dados, aplicamos *Engenharia de Prompt* em três etapas distintas para aprimorar o processo de recomendação.

Os resultados alcançados, com valores de *Hit@10* de 0.4457 e 0.6353, respectivamente, evidenciam a promissora capacidade desses modelos na realização da tarefa de recomendação de próximo item, mesmo sem a implementação de uma etapa de ajuste fino (*fine-tuning*). Esses resultados superam os reportados por [Wang and Lim 2023] e são particularmente relevantes em cenários *zero-shot*. A aplicação da abordagem de *Engenharia de Prompt* proposta pelo autor também revelou-se crucial, possibilitando a geração de recomendações mais alinhadas com as preferências individuais dos usuários.

Podemos ainda destacar não apenas a efetividade dos *modelos de abertos* quando comparados com *modelos proprietários* na recomendação de itens, mas também a importância de estratégias específicas, como a *Engenharia de Prompt*, para otimizar o desempenho desses modelos. Como desdobramento futuro, explorar outras técnicas de *fine-tuning* e considerar conjuntos de dados mais amplos pode contribuir para uma compreensão mais abrangente e aprimorada da aplicabilidade desses modelos em sistemas de recomendação.

References

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,*

Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Ding, H., Ma, Y., Deoras, A., Wang, Y., and Wang, H. (2021). Zero-shot recommender systems.

Frantar, E., Ashkboos, S., Hoefler, T., and Alistarh, D. (2023). Gptq: Accurate post-training quantization for generative pre-trained transformers.

Goldberg, D., Nichols, D., Oki, B. M., and Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70.

Harper, F. M. and Konstan, J. A. (2015). The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4).

Hu, Y., Koren, Y., and Volinsky, C. (2008). Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE international conference on data mining*, pages 263–272. Ieee.

Lops, P., De Gemmis, M., and Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. *Recommender systems handbook*, pages 73–105.

Mahmood, T. and Ricci, F. (2009). Improving recommender systems with adaptive conversational strategies. pages 73–82.

Pazzani, M. J. and Billsus, D. (2007). Content-based recommendation systems. In *The adaptive web: methods and strategies of web personalization*, pages 325–341. Springer.

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. (2018). Improving language understanding by generative pre-training.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. (1994). Grouplens: an open architecture for collaborative filtering of netnews. In *Conference on Computer Supported Cooperative Work*.

Resnick, P. and Varian, H. R. (1997). Recommender systems. *Commun. ACM*, 40:56–58.

Ricci, F., Rokach, L., Shapira, B., and Kantor, P. B. (2011). *Recommender systems handbook*. Springer, New York; London.

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*, pages 285–295.

Shardanand, U. and Maes, P. (1995). Social information filtering: algorithms for automating "word of mouth". In *International Conference on Human Factors in Computing Systems*.

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*, pages 1441–1450.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C. C., Chen, M.,

Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Koura, P. S., Lachaux, M.-A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E. M., Subramanian, R., Tan, X. E., Tang, B., Taylor, R., Williams, A., Kuan, J. X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., and Scialom, T. (2023). Llama 2: Open foundation and fine-tuned chat models.

Wang, L. and Lim, E.-P. (2023). Zero-shot next-item recommendation using large pre-trained language models.