LIMITAÇÕES DA ARQUITETURA DE DUAS TORRES EM SISTEMAS DE RECOMENDAÇÃO: UMA AVALIAÇÃO EMPÍRICA

Mateus Rico, Leila Weitzel e Rafael Medeiros
Universidade Federal Fluminense, Brasil

RESUMO

Sistemas de recomendação tornaram-se parte integrante de nossas interações digitais, selecionando conteúdo e ofertas personalizadas com base em nosso comportamento, preferências e interações passadas. Ao longo dos anos, diversas metodologias foram desenvolvidas para melhorar o desempenho e a precisão desses sistemas. Um método que ganhou destaque nos últimos anos é o Modelo de Duas Torres. Este algoritmo utiliza duas redes neurais para classificação de texto. O objetivo principal desta pesquisa é avaliar o desempenho do modelo de duas torres em sistemas de recomendação. A importância deste trabalho reside na crescente necessidade de sistemas de recomendação mais eficientes e personalizados, capazes de melhorar a experiência do usuário e contribuir para o avanço da IA.

PALAVRAS-CHAVE

Sistemas de Recomendação, *Deep Learning*, Duas Torres

1. INTRODUÇÃO

Sistemas de recomendação são algoritmos que oferecem sugestões personalizadas de itens, selecionando aqueles mais relevantes para cada usuário. Eles se baseiam em dois tipos principais de informação: (1) informações de características, que descrevem os itens (palavras-chave, categorias, etc.) e os usuários (preferências, perfis demográficos, etc.); e (2) informações de interação usuário-item, que registram as ações dos usuários em relação aos itens, como avaliações, compras, curtidas, etc. Em essência, esses sistemas capturam as preferências e o comportamento passado de cada usuário para filtrar a grande quantidade de dados disponível e apresentar apenas as sugestões mais pertinentes, reduzindo assim a sobrecarga cognitiva e melhorando a experiência do usuário. Esses sistemas são amplamente utilizados em diversas setores, por exemplo, indústrias, incluindo e-commerce, streaming de mídia, redes sociais e outros (Kar et al., 2024, Li et al., 2024).

Os sistemas de recomendação têm como objetivo (Li et al., 2024):

- Ser relevante - ser capaz de fazer previsões precisas sobre as preferências pessoais dos clientes e sugerir os produtos relevantes de seu interesse a eles. A lógica principal por trás disso é que uma pessoa tem uma maior probabilidade de consumir um item que seja de sua escolha ou preferência.
- Novidade – esses sistemas são úteis se forem bem-sucedidos em sugerir produtos de interesse ao usuário quando o usuário não viu o produto no passado.
- Serendipity - Como às vezes alguns dos produtos sugeridos por um sistema são inesperados para um cliente, portanto, há um elemento de uma descoberta fortuita, em vez de apenas ser algo que o usuário não conhece.
- Aumentando a diversidade de recomendações - geralmente, os sistemas de recomendação sugerem a lista dos k itens principais em uma determinada lista. Mas se todos os sistemas retornarem as mesmas listas, então é mais provável que o usuário não goste de nenhuma delas. Por outro lado, se houver sugestões diversas, então há uma chance maior de um usuário gostar de pelo menos um desses itens.

Esse sistemas têm evoluído significativamente ao longo dos anos, passando de técnicas tradicionais para abordagens modernas e sofisticadas. As três técnicas tradicionais principais são:

1. **Filtragem Colaborativa**: Baseia-se no comportamento passado dos usuários e nas preferências de usuários semelhantes. Utiliza técnicas de aprendizado de máquina para aprender padrões de comportamento dos usuários e fazer previsões com base em usuários semelhantes (Bobadilla et al., 2010);

2. **Filtragem Baseada em Conteúdo**: Baseia-se nas características dos itens que o usuário gostou no passado. Utiliza técnicas de processamento de linguagem natural (PLN) e aprendizado de máquina para entender o conteúdo dos itens e fazer recomendações com base em itens semelhantes que o usuário gostou no passado (Almazro et al., 2010);

3. **Filtragem Híbrida**: Combina a filtragem colaborativa e baseada em conteúdo para aproveitar os pontos fortes de ambos (Sundaresan, 2011). Entretanto, essas técnicas não são muito robustas e apresentam limitações.

Os métodos tradicionais de recomendação, como filtragem colaborativa e filtragem baseada em conteúdo, apresentam limitações significativas. A filtragem colaborativa sofre do problema do "novo usuário" (cold start), dificuldades de escalabilidade e esparsidade de dados. A filtragem baseada em conteúdo, por sua vez, pode levar à superespecialização, necessitando de profundo conhecimento do domínio e tendo dificuldades em capturar preferências subjetivas. Embora a filtragem híbrida busque mitigar essas desvantagens combinando diferentes abordagens, ela introduz complexidade adicional na implementação, integração de dados e ainda pode enfrentar problemas de escalabilidade e esparsidade As restrições gerais e de igual importância estão relacionadas aos desafios de interpretabilidade. Pode ser difícil explicar por que um sistema tradicional fez uma determinada recomendação, especialmente em modelos híbridos complexos. Exemplo: Um usuário pode querer saber por que um determinado livro foi recomendado para ele, mas o sistema pode não fornecer uma explicação clara. Os sistemas tradicionais podem não capturar nuances individuais nas preferências dos usuários, resultando em recomendações menos personalizadas. Exemplo: Dois usuários podem ter gostos ligeiramente diferentes, mas o sistema pode recomendar os mesmos itens para ambos, sem considerar essas diferenças sutis (Zhang et al., 2019, Li et al., 2024).

Desta forma, este trabalho visa avaliar a capacidade de generalização da arquitetura Two Towers em sistemas de recomendação, investigando se sua estrutura simples garante robustez e desempenho eficaz.

2. *DEEP LEARNING* EM SISTEMAS DE RECOMENDAÇÃO

Os sistemas de recomendação evoluíram significativamente desde seus primórdios, passando de métodos simples baseados em regras a algoritmos sofisticados alimentados por inteligência artificial. As técnicas tradicionais, embora eficazes em cenários simples, enfrentam desafios em lidar com a complexidade crescente dos dados e das preferências dos usuários. As técnicas modernas, por outro lado, exploram novas abordagens para superar essas limitações e oferecer recomendações mais personalizadas e precisas. As técnicas modernas, como Deep Learning (DL) e sistemas híbridos, ajudam a mitigar as restrições dos sistemas tradicionais (Kulkarni et al., 2023). Vários pesquisadores têm empregado arquiteturas de DL para melhorar o desempenho dos sistemas de recomendação, por exemplo, (Wu et al., 2016) propuseram um método de recomendação chamado Collaborative Denoising Auto-Encoder (CDAE). A estrutura do Denoising Auto-Encoder é usada para modelar representações distribuídas dos usuários e itens através da formulação dos dados de feedback usuário-item. (Hassan and Hamada, 2017) exploraram a eficácia do uso de redes neurais artificiais na modelagem de problemas de recomendação multicritério. (Hao and Zhang, 2021) usam um modelo de Markov oculto para analisar sequências de preferências dos usuários. (Bag et al., 2019) propuseram um método que corrige ruídos casuais usando o coeficiente de Bhattacharya e o conceito de autocontradição. (Yu et al., 2019) identificaram usuários mais informativos modelando todos os dados de treinamento como uma rede de informações heterogênea, para obter a representação de incorporação. Dentro das pesquisas de DL para sistemas de recomendação, chama-se a atenção para os modelos Duas Torres (Two Towers Embedding - TTE). As pesquisas em TTE avançaram, pois esses modelos possuem fatoração de matrizes, ao mesmo tempo em que fornecem a capacidade de incorporar recursos de consulta e item; além disso, podem refinar os embeddings de consulta em tempo real com base no contexto e são mais flexíveis. Os autores (Lee and Cho, 2023, Li et al., 2022) tratam especificamente o problema de “cold Start”. (Raza et al., 2022) apresenta um sistema de recomendação de notícias onde utiliza a função de perda categórica que alinha a representação de itens de categorias de notícias desiguais. (Yang et al., 2020) utiliza uma arquitetura modificada do TTE para tratar a Amostragem Negativa Mista (MNS - Mixed Negative Sampling) para lidar com o viés presente no feedback implícito do usuário.

2.1 *Two Towers Embeddings*

Arquiteturas de duas torres mapeiam a semântica de consultas para um espaço vetorial, agrupando entidades semanticamente semelhantes. Assim, a busca por candidatos similares se torna eficiente por meio da comparação de embeddings vetoriais. A denominação "Duas Torres" deriva da sua estrutura, composta por duas redes neurais que processam independentemente informações de usuário e item. A "torre do usuário" aprende padrões de comportamento, preferências e interações, enquanto a "torre do item" processa características como especificações e histórico de interações. A recomendação resulta da combinação das saídas dessas torres, gerando um processo colaborativo e multidimensional. A metodologia envolve: (1) entrada de dados de usuário e item em suas respectivas torres; (2) extração de características relevantes por meio de algoritmos de aprendizado profundo; (3) geração de embeddings que representam usuários e itens; (4) comparação dos embeddings para prever a probabilidade de interação; e (5) ajuste iterativo dos embeddings via gradiente descendente, otimizando as previsões. Este processo de treinamento permite que o sistema aprenda e refine suas recomendações ao longo do tempo (Ling, 2023, Wortz and Totten, 2023).

3. EXPERIMENTOS E RESULTADOS

Nesta pesquisa implementa-se um sistema de recomendação de filmes. A Torre do Usuário analisa o comportamento do usuário, como os gêneros de filmes assistidos, as avaliações dadas e a frequência de visualização. Neste caso, o sistema pode descobrir que o Usuário A frequentemente assiste e avalia filmes de ação, enquanto o Usuário B prefere comédias românticas. A Torre do Item analisa as características dos filmes, como gênero, diretor, atores e avaliações dos espectadores. O sistema pode descobrir que o Filme X é um filme de ação que foi altamente avaliado por espectadores que gostam de filmes de ação, enquanto o Filme Y é uma comédia romântica popular entre espectadores que preferem esse gênero. Ao conectar as duas torres, o sistema pode recomendar o Filme X ao Usuário A e o Filme Y ao Usuário B, com base em suas respectivas preferências. De um modo geral, o funcionamento do algoritmo se dá da seguinte maneira: (a) **Entrada de Dados**: Torre do Usuário: Processa as características do usuário, como histórico de navegação, preferências e interações anteriores. Torre do Item: Processa as características dos itens, como descrições de produtos, categorias e avaliações. (b) **Geração de Embeddings**: Cada torre gera um vetor de embeddings a partir das suas respectivas entradas. Esses vetores são representações numéricas das características dos usuários e itens. (c) **Cálculo da Similaridade**: Os vetores de embeddings gerados pelas duas torres são comparados usando uma métrica de similaridade, como o produto escalar. A similaridade indica o quão bem um item corresponde às preferências de um usuário.

3.1 *Dataset*

Conduzimos vários os experimentos com a base de dados MovieLens (Harper and Konstan, 2015) que é amplamente utilizada em pesquisas em sistemas de recomendação. O dataset contém avaliações de filmes feitas por usuários do site MovieLens, mantido pelo grupo de pesquisa GroupLens. Existem várias versões da base de dados, incluindo MovieLens 100K, 1M, 10M, 20M, e 25M, que variam em tamanho e detalhes. Os campos (colunas) do dataset MovieLens são: movieId: Identificador único do filme; movie_title: Título do filme, incluindo o ano de lançamento entre parênteses; genres: Gêneros aos quais o filme pertence, separados por barras; user_ids: Identificador único do usuário que fez a avaliação; user_rating: Avaliação dada pelo usuário (user_ratings) ao filme, geralmente em uma escala de 0 a 5 estrelas; timestamp: Data e hora em que a avaliação foi feita, representada em segundos desde 1º de janeiro de 1970 (Epoch time); Ocupação do usuário (user_ocupation_text). Nesta pesquisa utilizou-se o dataset MovieLens 100k nativo do framework TensorFlow.

3.2 Arquitetura da Rede

Uma classe contendo as duas torres do modelo TTE foi implementada. O esquema de treinamento, ilustrado na Figura 1, mostra as torres de usuário e item (filmes). Após a entrada dos dados, embeddings são gerados para cada torre, posteriormente concatenados via produto escalar para calcular a função de perda. Para avaliar

o desempenho, utilizaram-se as métricas MSE (Mean Squared Error) e MAE (Mean Absolute Error). A arquitetura do modelo TTE compreende: camadas de embedding (user_embedding e movie_embedding) gerando representações densas para usuários e filmes, respectivamente; uma camada Flatten; uma camada de concatenação combinando os vetores usuário-filme; e camadas densas com inicialização HeNormal (Kaiming Initialization) (He et al., 2015) que mantém a variância dos sinais, promovendo convergência mais rápida e estável. Camadas de normalização batch e dropout foram adicionadas. Este modelo combina embeddings e camadas densas com técnicas de regularização para aprimorar a precisão das recomendações.

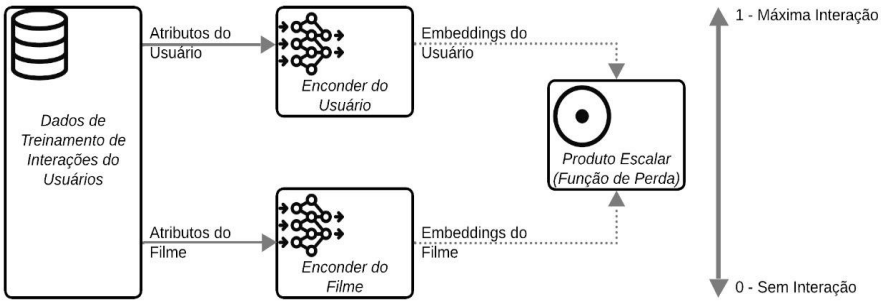

Figura 1. Esquemático do treinamento da rede TTE

3.3 Principais Resultados

Diversos experimentos foram conduzidos, modificando a arquitetura do modelo TTE. Inicialmente, utilizou-se movie_title, user_ratings e user_id como entradas. Embora o overfitting não tenha sido significativo, os valores da função de perda MAE (acima de 0,7) e MSE (em torno de 0,9) na validação foram considerados elevados, considerando que a saída varia de 1 a 5. A Figura 2 (à direita) apresenta o gráfico do MSE de treinamento e validação, mostrando convergência para valores relativamente baixos. Entretanto, a discrepância entre o MSE de treinamento e validação indica overfitting, sugerindo que o modelo aprendeu padrões específicos do conjunto de treinamento que não generalizam bem para o conjunto de validação. Para mitigar o overfitting, foram empregadas técnicas de regularização L1/L2 e dropout, além de *early stopping*. Apesar dessas intervenções, os resultados não foram satisfatórios.

Os 20 melhores resultados da primeira etapa são apresentadas na Tabela 2 na ordem que os modelos são descrito na Tabela 1. Descrição dos modelos:

(i) USER-0-MOVIE-0-TOWER onde a arquitetura da concatenação que esta sendo alterada.
(ii) USER-0-MOVIE-0 apenas as torres de usuário e movie estão sendo modificadas
(iii) USER apenas essa torre está sendo modificada.

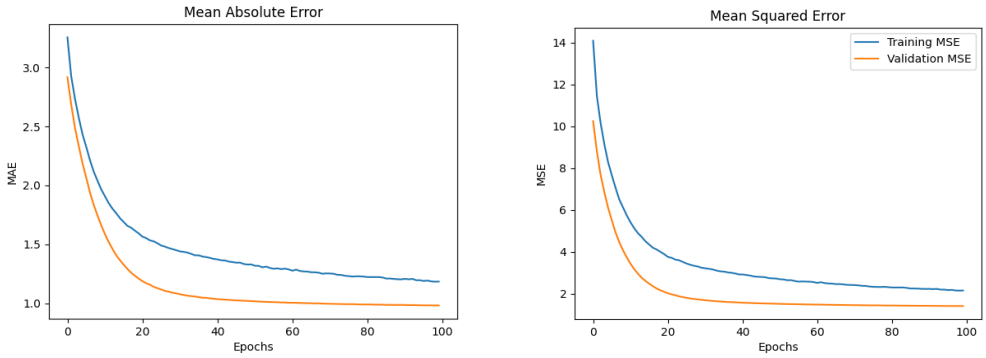

Figura 2. Gráfico do Erro no treinamento e validação, lado esquerdo MAE, lado direito MSE

Tabela 1. Arquitetura testadas, mostrando as 20 top

Arquiteturas testadas
USER-0-MOVIE-0-TOWER-DENSE32-DENSE16-DENSE1
USER-0-MOVIE-0-TOWER-DENSE32-BatchNormalization-DENSE16-DENSE1
USER-0-MOVIE-0-TOWER-DENSE32-Dropout(0,9)-BatchNormalization-DENSE16-DENSE1
USER-0-MOVIE-0-TOWER-DENSE32-Dropout(0,9)-BatchNormalization-DENSE16-Dropout(0,9)-DENSE1
USER-0-MOVIE-0-TOWER-DENSE32-Dropout(0,9)-DENSE16-Dropout(0,9)-DENSE8-DENSE1
USER-0-MOVIE-0-TOWER-DENSE32-Dropout(0,9)-DENSE16-Dropout(0,9)-DENSE8-DENSE1
USER-0-MOVIE-0-TOWER-DENSE8-DENSE4-DENSE2-DENSE1
USER-0-MOVIE-0-TOWER-DENSE8-DENSE4-Dropout(0,6)-DENSE4-DENSE1
USER-0-MOVIE-0-TOWER-DENSE4-Dropout(0,8)-BatchNormalization-DENSE4-Dropout(0,8)-BatchNormalization-DENSE4-DENSE1
USER-0-MOVIE-0-TOWER-DENSE8-Dropout(0,9)-DENSE1
USER-0-MOVIE-0-TOWER-DENSE8-DENSE1
USER-0-MOVIE-0-TOWER-DENSE8-DENSE32-DENSE8-DENSE1
USER-0-MOVIE-0-TOWER-Flatten()-DENSE64-DENSE32-DENSE1
USER-0-MOVIE-0-TOWER-Flatten()-DENSE8-Dropout(0,2)-DENSE8-DENSE1
USER-0-MOVIE-0-DENSE256-Dropout(0,9)-BatchNormalization()-DENSE64-Dropout(0,8)-DENSE32-DENSE1
USER-0-MOVIE-0-DENSE256-Dropout(0,9)-BatchNormalization()-DENSE128-Dropout(0,8)-DENSE32-DENSE64(L2)-DENSE1
USER-0-MOVIE-0-DENSE256-Dropout(0,9)-DENSE256(L2)-DENSE128-Dropout(0,2)-DENSE32-BatchNormalization()-DENSE1
USER-0-MOVIE-0-DENSE1024-Dropout(0,9)-DENSE1024(L2)-BatchNormalization()-DENSE1
USER-0-MOVIE-0-DENSE2048-Dropout(0,8)-DENSE1024(L2)-Dropout(0,4)-DENSE2048-DENSE1
USER-0-MOVIE-0-DENSE2048-Dropout(0,8)-DENSE1024(L2)-Dropout(0,4)-DENSE2048-DENSE1
USER-DENSE128-Dropout(0,8)-DENSE64-MOVIE-DENSE128-Dropout(0,8)-DENSE64-TOWER-DENSE256-Dropout(0,5)-BatchNormalization()-DENSE32-DENSE1

Tabela 2. Resultados dos primeiros experimentos apresentados na ordem apresentada na Tabela 1

LR	emb_dim	MAE	MAE_VAL	MSE	MSE_VAL
0,001	64	0,718	0,7626	0,829	0,93
0,001	64	0,721	0,7598	0,833	0,92
0,001	64	0,716	0,7576	0,818	0,91
0,001	64	0,728	0,7613	0,854	0,93
0,001	32	0,730	0,7659	0,855	0,94
0,0001	32	3,461	3,4533	13,249	13,19
0,001	8	3,388	3,3772	12,742	12,67
0,001	8	0,929	0,9138	1,363	13,21
0,001	8	0,879	0,8407	11,387	10,78
0,001	16	10,978	10,6896	18,836	18,03
0,001	16	12,029	11,5730	22,232	20,91
0,001	16	0,754	0,7742	0,903	0,95
0,001	16	0,748	0,7746	0,900	0,96
0,0001	8	3,469	3,4618	13,301	13,25
0,001	256	0,165	0,8116	0,051	10,40
0,001	256	0,138	0,8541	0,033	11,31
0,001	512	0,157	0,8417	0,031	11,12
0,001	512	0,202	0,9441	0,059	13,26
0,001	512	1,674	15,5520	36,734	3,19
0,001	1024	0,249	10,9607	0,123	1,75
0,001	256	0,712	0,7535	0,814	0,91

Na segunda parte dos experiementos, optou-se por incluir na entrada dos dados os seguintes campos: user_occupation_text, movie_title e user_ratings. A Tabela 3 mostra as arquiteturas que foram avaliadas, a arquitetura é mostrada na Tabela 3. É possível observar na Tabela 4, que com os três novos campos a rede não conseguiu bons resultados. O problema do overfitting ainda se manteve apesar de se utilizar diversos métodos de regularização.

Tabela 3. Arquiteturas testadas na segunda fase do experimento

Arquiteturas Testesdas na fase 2
USER-0-MOVIE-0-TOWER-DENSE8-Flatten()-DENSE8-DENSE1
USER-0-MOVIE-0-TOWER-DENSE64-Dropout(0.9)-DENSE32(L2)-DENSE1
USER-0-MOVIE-0-TOWER-DENSE8-DENSE32-DENSE8-DENSE1
USER-0-MOVIE-0-TOWER-DENSE32-Dropout(0.8)-DENSE16-Dropout(0.2)-DENSE1
USER-0-MOVIE-0-TOWER-DENSE32-Dropout(0.8)-DENSE8-BatchNormalization()-DENSE1
USER-0-MOVIE-0-TOWER-DENSE32-Dropout(0.2)-DENSE16(L2)-DENSE1
USER-0-MOVIE-0-TOWER-DENSE32-Dropout(0.2)-DENSE32(L2)-DENSE1
USER-0-MOVIE-0-TOWER-Flatten()-DENSE4-DENSE1
USER-0-MOVIE-0-TOWER-DENSE32-Dropout(0.8)-DENSE8-DENSE1
USER-0-MOVIE-0-TOWER-DENSE8-Dropout(0.2)-DENSE1
USER-0-MOVIE-0-TOWER-DENSE32-Dropout(0.9)-Flatten()-DENSE16-Dropout(0.2)-DENSE16-DENSE1
USER-0-MOVIE-0-TOWER-DENSE32-Dropout(0.9)-DENSE16-Dropout(0.2)-DENSE8-DENSE1
USER-0-MOVIE-0-TOWER-DENSE16-Dropout(0.9)-DENSE8-Dropout(0.2)-DENSE8-DENSE1
USER-0-MOVIE-0-TOWER-DENSE16-Dropout(0.9)-DENSE8-Dropout(0.9)-DENSE8-DENSE1
USER-0-MOVIE-0-TOWER-DENSE16-Dropout(0.9)-DENSE8-DENSE1
USER-0-MOVIE-0-TOWER-DENSE32-Dropout(0.9)-BatchNormalization()-DENSE16-DENSE1
USER-0-MOVIE-0-TOWER-DENSE32-BatchNormalization()-DENSE16-DENSE1
USER-0-MOVIE-0-TOWER-DENSE32-DENSE64-DENSE32-DENSE1
USER-0-MOVIE-0-TOWER-DENSE32-DENSE64-Dropout(0.9)-DENSE32-DENSE1
USER-0-MOVIE-0-TOWER-DENSE32-Dropout(0.9)-Flatten()-DENSE16-BatchNormalization()-DENSE16-DENSE1
USER-0-MOVIE-0-TOWER-DENSE32-DENSE16-DENSE1
USER-0-MOVIE-0-TOWER-DENSE128-Dropout(0.9)-DENSE128(L2)-DENSE64-Dropout(0.8)-DENSE1
USER-0-MOVIE-0-TOWER-BatchNormalization()-DENSE128-Dropout(0.9)-DENSE128(L2)-DENSE64-Dropout(0.8)-DENSE1
USER-DENSE64-Dropout(0.8)-DENSE32-MOVIE-DENSE64-Dropout(0.8)-DENSE32-TOWER-DENSE128-Dropout(0.9)-DENSE128(L2)-DENSE64-Dropout(0.8)-DENSE1
USER-0-MOVIE-0-TOWER-DENSE32-Dropout(0.4)-DENSE1

Tabela 4. Resultados do segundo experimento na ordem apresentada na Tabela 3

learning_rate	emb_dim	MAE	MAE_VAL	MSE	MSE_VAL
0,001	8	0,8	0,82	1	1,04
0,001	8	0,79	0,81	1	1,04
0,01	16	0,73	0,86	0,9	1,18
0,001	16	0,8	0,81	1	1,04
0,001	16	0,79	0,82	1	1,04
0,01	32	0,7	0,87	0,8	1,21
0,001	32	0,78	0,82	1	1,05
0,01	32	0,73	0,87	0,9	1,2
0,001	32	0,78	0,82	1	1,05
0,01	32	0,72	0,86	0,9	1,19
0,001	64	0,79	0,82	1	1,05
0,0001	64	0,91	0,9	1,3	1,28
0,0001	64	1,18	1,15	2,2	2,04
0,0001	64	1,11	1,08	1,9	1,82
0,001	64	0,77	0,82	0,9	1,06
0,001	64	0,76	0,83	0,9	1,08
0,001	64	0,77	0,82	0,9	1,07
0,01	64	0,72	0,89	0,9	1,23
0,01	64	0,72	0,87	0,9	1,21
0,001	64	0,75	0,83	0,9	1,09
0,001	64	0,76	0,83	0,9	1,08
0,001	128	0,72	0,87	0,8	1,18
0,001	128	0,7	0,87	0,8	1,2
0,001	128	0,77	0,83	0,9	1,09
0,001	128	0,77	0,82	0,9	1,07

4. CONCLUSÃO E TRABALHOS FUTUROS

Este trabalho avaliou o desempenho da arquitetura Two Towers (TTE) padrão em um sistema de recomendação de filmes utilizando o dataset MovieLens 100k. Os resultados demonstram que a arquitetura TTE, em sua configuração original, não apresentou desempenho satisfatório devido à presença de ruído considerável no dataset, caracterizado por desequlíbrio na distribuição de avaliações e notas. A arquitetura não conseguiu lidar adequadamente com essa heterogeneidade nos dados, resultando em métricas de perda insatisfatórias.

Além disso, a análise destaca os desafios inerentes a modelos de aprendizado auto-supervisionado, suscetíveis a colapso de embeddings. Como trabalhos futuros, sugerimos investigar a mitigação desses problemas através da implementação de funções de perda por contraste ou Gumbel-Softmax para uma seleção estocástica de codificadores, buscando melhorar a robustez do modelo e a generalização em datasets ruidosos. A exploração dessas estratégias pode levar ao desenvolvimento de sistemas de recomendação mais robustos e precisos.

REFERÊNCIAS

Almazro, D., Shahatah, G., Albdulkarim, L., Kherees, M., Martinez, R. & Nzoukou, W. (2010). A Survey Paper on Recommender Systems.

Bag, S., Kumar, S., Awasthi, A. & Tiwari, M. K. (2019). A noise correction-based approach to support a recommender system in a highly sparse rating environment. *Decision Support Systems,* 118**,** 46-57.

Bobadilla, J., Serradilla, F. & Bernal, J. J. K.-B. S. (2010). A new collaborative filtering metric that improves the behavior of recommender systems. 23**,** 520-528.

Hao, Y. & Zhang, F. (2021). An unsupervised detection method for shilling attacks based on deep learning and community detection. 25**,** 477–494.

Harper, F. M. & Konstan, J. A. (2015). The MovieLens Datasets: History and Context. 5**,** Article 19.

Hassan, M. & Hamada, M. Performance analysis of neural networks-based multi-criteria recommender systems. 2017 2nd International conferences on Information Technology, Information Systems and Electrical Engineering (ICITISEE), 1-2 Nov. 2017 2017. 490-494.

He, K., Zhang, X., Ren, S. & Sun, J. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. 2015 IEEE International Conference on Computer Vision (ICCV), 7-13 Dec. 2015 2015. 1026-1034.

Kar, P., Roy, M. & Datta, S. 2024. *Recommender Systems: Algorithms and their Applications*, Singapore, Springer Nature Singapore.

Kulkarni, A., Shivananda, A., Kulkarni, A. & Krishnan, V. A. (2023). *Applied Recommender Systems with Python: Build Recommender Systems with Deep Learning, NLP and Graph-Based Techniques*, Berkeley, CA, Apress.

Lee, W. M. & Cho, Y. S. (2023). A Flexible Two-Tower Model for Item Cold-Start Recommendation. *IEEE Access,* 11**,** 146194-146207.

Li, D., Lian, J., Zhang, L., Ren, K., Lu, T., Wu, T. & Xie, X. (2024). *Recommender systems frontiers and practices,* Singapore, Springer.

Li, P., Chen, R., Liu, Q., Xu, J. & Zheng, B. (2022). Transform Cold-Start Users into Warm via Fused Behaviors in Large-Scale Recommendation. *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval.* Madrid, Spain: Association for Computing Machinery.

Ling, B. (2023). Innovative Recommendation Applications Using Two Tower Embeddings at Uber. Available at: https://www.uber.com/en-BR/blog/innovative-recommendation-applications-using-two-tower-embeddings/. Retrieved 2024/07/26.

Raza, S., Bashir, S. R., Naseem, U., Liu, D. D. & Reji, D. J. Incorporating Accuracy and Diversity in a News Recommender System. 2022 IEEE 9th International Conference on Data Science and Advanced Analytics (DSAA), 13-16 Oct. 2022 2022. 1-10.

Sundaresan, N. Recommender systems at the long tail. 2011. ACM Press, 1-1.

Wortz, J. & Totten, J. (2023). Scaling deep retrieval with TensorFlow Recommenders and Vertex AI Matching Engine. Available at: https://cloud.google.com/blog/products/ai-machine-learning/scaling-deep-retrieval-tensorflow-two-towers-architecture. Retrieved 2024/03/24.

Wu, Y., DuBois, C., Zheng, A. X. & Ester, M. (2016). Collaborative Denoising Auto-Encoders for Top-N Recommender Systems. *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining.* San Francisco, California, USA: Association for Computing Machinery.

Yang, J., Yi, X., Cheng, D. Z., Hong, L., Li, Y., Wang, S. X., Xu, T. & Chi, E. H. (2020). Mixed Negative Sampling for Learning Two-tower Neural Networks in Recommendations. *Companion Proceedings of the Web Conference 2020.* Taipei, Taiwan: Association for Computing Machinery.

Yu, J., Gao, M., Yin, H., Li, J., Gao, C. & Wang, Q. (2019). Generating Reliable Friends via Adversarial Training to Improve Social Recommendation. *2019 IEEE International Conference on Data Mining (ICDM).* IEEE Computer Society.

Zhang, S., Yao, L., Sun, A. & Tay, Y. (2019). Deep Learning Based Recommender System: A Survey and New Perspectives. 52**,** Article 5.