Aplicação de Modelos Neurais de Recomendação ao Contexto dos Jogos Eletrônicos

João Lucas Rodrigues Constantino
jlconstantino@usp.br
Instituto de Ciências Matemáticas e de Computação (ICMC-USP)
São Carlos, São Paulo, Brasil

RESUMO

Este artigo visa a comparar diferentes modelos de recomendação no contexto de jogos eletrônicos. Para tanto, foram empregadas as arquiteturas ItemKNN, ItemAttributeKNN, BPR-MF e NCF.

KEYWORDS

recommender systems, neural networks, eletronic games

ACM Reference Format:
João Lucas Rodrigues Constantino. 2018. Aplicação de Modelos Neurais de Recomendação ao Contexto dos Jogos Eletrônicos. In *Proceedings of Trabalho de Finalização de Disciplina (SCC0284).* ACM, New York, NY, USA, 2 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUÇÃO

Com o sucesso das aplicações de aprendizagem profunda aos mais variados contextos, como visão computacional e processamento de linguagem natural, essas passaram a ser empregadas também aos sistemas de recomendação – com o intuito de promover recomendações mais fidedignas aos diferentes perfis de usuários existentes em diversos domínios de aplicação. Desse modo, ao longo do final da década de 2010, alguns algoritmos, tais como Neural Collaborative Filtering (NCF) [6] e xDeepFM [7], foram desenvolvidos com base em redes neurais artificiais e, em geral, apresentaram resultados mais satisfatórios do que as abordagens mais tradicionais. Assim sendo, este artigo visa a aplicar tais algoritmos em problemas de recomendação de jogos eletrônicos, verificando e comparando suas respectivas métricas de desempenho e, além disso, averiguando também o ganho de desempenho sobre o uso de técnicas mais tradicionais.

2 PROCEDIMENTO METODOLÓGICO

Para consulta, foram selecionados artigos escritos ou em linguagem inglesa, ou em linguagem portuguesa, que abordam o uso de técnicas de aprendizagem profunda no contexto dos sistemas de recomendação. Desse modo, tais técnicas serão utilizadas em conjuntos de dados de histórico de interação de usuários com jogos eletrônicos, assim como de metadados associados a esses mesmos jogos. A saber, os conjuntos de dados foram obtidos da plataforma Kaggle, sendo eles o *Steam Reviews Dataset* [3] – contendo ambos os tipos de interação, implícita e explícita – e o *Steam Store Games* [2], em que este apresenta metadados acerca de diversos jogos eletrônicos, além de métricas gerais do público que podem ser utilizadas para validação.

Para implementação dos modelos, foi utilizada a linguagem Python 3 [8], em que pacotes de ciências de dados foram empregados para a obtenção das métricas de desempenho – os quais serão apresentados e comparados nos resultados adiante. Para tanto, o pacote *recommenders* [4], da Microsoft, foi empregado a fim de agilizar a aplicação dos algoritmos neurais nos conjuntos de dados apontados. Por fim, os resultados coletados serão comparados com aqueles obtidos por meio de técnicas mais tradicionais – como ItemKNN e BPR-MF – implementadas no pacote Case Recommender [1].

SCC0284, October 16, 2022, São Carlos, SP

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/XXXXXXX.XXXXXXX

3 TRATAMENTO DOS DADOS

Para as *reviews* dos usuários, foram mantidas somente as colunas "steamid", "appid"e "voted_up", que foram renomeadas, respectivamente, para "userID", "itemID"e "rating-- passo necessário para a adequação de alguns dos modelos neurais utilizados, que foram desenvolvidos para lidar, especificamente, com o conjunto de dados da MovieLens [5]. Em seguida, quantos aos metadados dos jogos, foram mantidos somente as colunas de identificação e de gêneros dos jogos, em que este foi transformado em *one-hot encoding*.

Posteriormente, devido a limitações de recursos de RAM e de poder de processamento computacional, foi descartado parte dos dados, de maneira a manter somente as avaliações de usuários que realizam 48 ou mais interações e, dessas, foram mantidas somente aquelas referentes a jogos com 32 ou mais avaliações. Por fim, uma vez que vários dos métodos usados não ofereciam suporte adequado para o uso de um conjunto de validação, todos os dados de interação explícita dos usuários foram divididos entre os conjuntos de treinamento e de teste, que mantiveram – respectivamente – 80% e 20% dos dados, isto é, 286616 e 71654 exemplos nessa ordem.

4 MÉTODOS TRADICIONAIS

Para a filtragem colaborativa padrão, foi utilizada aquela baseada em vizinhança de itens, denominada ItemKNN, com o hiperparâmetro K definido como 10; seus resultados foram $NDCG@10 = 0.44$ e $MAP@10 = 0.32$. Por sua vez, para a filtragem baseada em conteúdo, foi empregado o ItemAttributeKNN, com $K = 10$, obtendo $NDCG@10 = 0.052$ e $MAP@10 = 0.033$. Por fim, para o ranqueamento, foi utilizado o algoritmo orientado a pares BPR-MF, que obteve $NDCG@10 = 0.26$ e $MAP@10 = 0.17$; seus hiperparâmetros, quais sejam, quantia de fatores latentes e taxa de aprendizado, foram utilizados com os valores, respectivamente, 10 e 0.001.

5 MÉTODOS DE APRENDIZAGEM PROFUNDA

Para a metodologia NCF, foram treinadas 48 épocas, cada qual com um tamanho de lote definido como 256. Quanto aos resultados, foram obtidos $NDCG@10 = 0.1$ e $MAP@10 = 0.031$. Entretanto, devido a problemas de falta de documentação e de padronização da API da Microsoft Recommenders, não foi possível testar outros algoritmos. Ainda em relação a essa biblioteca, houve problemas quanto ao uso de código deprecado do tensorflow, que não funciona nas versões atuais.

6 COMPARATIVO

Todos os resultados quantitativos foram reunidos na *figura 1*. Note que o algoritmo de filtragem colaborativa baseada em vizinhança, ItemKNN, obteve as melhores métricas – apesar de se tratar do modelo menos sofisticado dentre os utilizados. Ainda, o único algoritmo funcional de aprendizagem profunda, NCF, obteve péssimos resultados; em geral, houve treino até a convergência, então o problema provavelmente não provém de falta de épocas de treino. Logo, as hipóteses mais prováveis são a possível presença significativa de vieses no corte de dados realizado, a insuficiência de dados e a presença de problemas de código – tais como os apresentados no do modelo Wide&Deep.

Ainda, o ItemKNN perfomou melhor do que o ItemAttributeKNN, o que é inesperado, dado que este deveria ter modelado as categorias dos itens para um mesmo hiperparâmetro K de vizinhança. Nesse sentido, também obteve resultados melhores do que o BPR-MF, porém, aqui, tais resultados provém, provavelmente, de ajuste de hiperparâmetros, tais quais a quantia de fatores latentes.

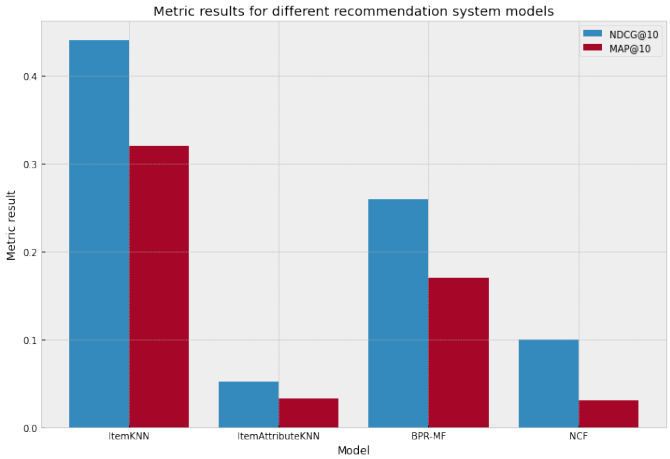

Figura 1: Resultados obtidos para as métricas NDCG@10 e MAP@10 para os diferentes modelos de recomendação utilizados.

ACKNOWLEDGMENTS

Repositório virtual para o código desenvolvido para os resultados deste artigo: https://github.com/jlrconstantino/steam_recommenders_comparison.

Acesso à gravação da apresentação do código desenvolvido para este artigo: https://drive.google.com/file/d/1psWo4i_zQKfhRQQU9W4S3BST6QJgR view?usp=sharing.

REFERÊNCIAS

[1] Arthur da Costa, Eduardo Fressato, Fernando Neto, Marcelo Manzato, and Ricardo Campello. 2018. Case Recommender: A Flexible and Extensible Python Framework for Recommender Systems. In *Proceedings of the 12th ACM Conference on Recommender Systems* (Vancouver, British Columbia, Canada) *(RecSys '18)*. ACM, New York, NY, USA, 494–495. https://doi.org/10.1145/3240323.3241611

[2] Nik Davis. 2019. *Steam Store Games, Version 3.* Retrieved December 03, 2022 from https://www.kaggle.com/datasets/nikdavis/steam-store-games/versions/3

[3] forgemaster. 2021. *Steam Reviews Dataset, Version 1.* Retrieved October 15, 2022 from https://www.kaggle.com/datasets/forgemaster/steam-reviews-dataset/versions/1

[4] Scott Graham, Jun-Ki Min, and Tao Wu. 2019. Microsoft recommenders. In *Proceedings of the 13th ACM Conference on Recommender Systems*. ACM. https://doi.org/10.1145/3298689.3346967

[5] grouplens. 2016. *MovieLens 20M Dataset.* Retrieved December 05, 2022 from https://www.kaggle.com/datasets/grouplens/movielens-20m-dataset

[6] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. https://doi.org/10.48550/ARXIV.1708.05031

[7] Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen, Xing Xie, and Guangzhong Sun. 2018. xDeepFM. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery &amp Data Mining*. ACM. https://doi.org/10.1145/3219819.3220023

[8] Guido Van Rossum and Fred L. Drake. 2009. *Python 3 Reference Manual.* CreateSpace, Scotts Valley, CA.