POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

Reproducing work for Simple-HGN and proposing a novel algorithm named attention-based matrix factorization

Laurea Magistrale in Computer Science and Engineering - Ingegneria Informatica

Author: Zhitao He

Advisor: Prof. Maurizio Ferrari Dacrema

Academic year: 2023-2024

1. Introduction

This thesis builds upon the Simple-HGN model presented in the article "Are we really making much progress? Revisiting, benchmarking, and refining heterogeneous graph neural networks"[4] and introduces a new model called Attention-based Matrix Factorization(AMF). The intention behind this is to investigate whether shallow methods, which have shown superiority over deep ones for traditional collaborative filtering tasks in previous evaluation work[1–3, 5–7] within the recommendation systems field, also apply to Simple-HGN. Moreover, it is also based on a assumption that the main predictive power of the Simple-HGN model comes from the simpler part.

2. Simple-HGN

The Simple-HGN model comprises two components: heterogeneous graph neural networks (HGNN) and pre-trained matrix factorization BPR (MF BPR) embeddings. HGNN is a deep learning model based on graph structures as shown in Figure 1, utilizing embedding representations for different node and edge types. At the same time, the model's learning capacity is enhanced by incorporating learnable edge-type embeddings, residual connections, and L2 normalization. Attention mechanisms are employed to aggregate and weigh different types of nodes and edges, generating learned node embeddings as output. On the other hand, pre-trained MF BPR embeddings are pre-computed embeddings obtained by training the MF BPR model on a large dataset. The final prediction score is calculated by combining these two sets of embeddings.

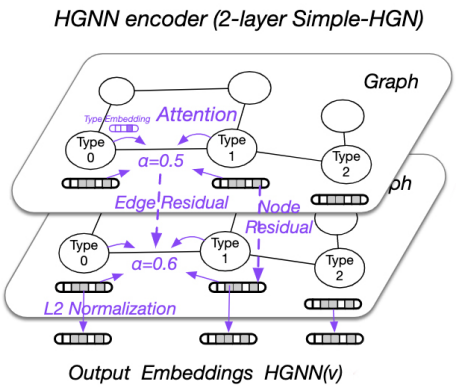

Figure 1: The architecture of HGNN in the Simple-HGN model

3. Attention-based Matrix Factorization

3.1. Model description

Given that HGNN is a deep learning model, our aim is to replace it with a non-deep learning algorithm while retaining the idea of pre-trained MF BPR embeddings, resulting in a new model. Specifically, for the pre-trained MF BPR embeddings, we introduce a modification: substituting the pre-trained embeddings with learnable embeddings that are trained by the MF model during the training process. We refer to the matrix obtained from these learnable embeddings as the user-item weight matrix, which consists of user embeddings and item embeddings. Within this matrix, each weight corresponds to a predicted score α_{ui}.

As for the HGNN component, we replace it with a non-deep learning model composed of two identical structured sub-components: one based on the user-user similarity matrix and the other on the item-item similarity matrix. The similarity matrices are calculated using the user embeddings and item embeddings derived from the user-item weight matrix, respectively. The user-user similarity matrix stores the similarity values s_{uv} between each user u and other users v, while the item-item similarity matrix stores the similarity values s_{ij} between each item i and other items j. Furthermore, the similarity matrices are combined with the corresponding attention weight matrices obtained through the MF technique. The attention weight matrices store the learned weights for the similarity matrices, ranging from 0 to 1. A weight of zero signifies that a particular similarity rating is unimportant and should be disregarded.

From the above, it can be inferred that both sub-components can obtain predicted scores for each user u and item i. One sub-component calculates the score by multiplying the similarity s_{uv} between user u and each other user v by the learned attention weight $\alpha_v i$ for each similarity, and then summing them up. The other sub-component calculates the score by multiplying the similarity s_{ij} between item i and each other item j by the learned attention weight α_{uj} for each similarity, and then summing them up. As a result of this process, each sub-component generates its own predicted score. The final prediction score $\widetilde{r}_{ui}$ of the AMF model is obtained by summing the user-item weight matrix with the scores from two sub-components. Give a user set U and a item set I, $\widetilde{r}_{ui}$ can be expressed as follows:

$$\widetilde{r}_{ui} = \alpha_{ui} + \sum_{v \in U} s_{uv} \cdot \alpha_{vi} + \sum_{j \in I} s_{ji} \cdot \alpha_{uj} \quad (1)$$

In addition, from the overall structure of the model, it mainly consists of three MF components, with each MF corresponding to a latent factor used to decompose and obtain embeddings. Therefore, it has a total of three latent factors.

3.2. Solution learning

In order to train the model, we first perform BPR sampling from URM in a uniform and random way, and each sample is composed of three elements <u,a,b>:

- **u**: an user who have at least an interaction in their user profile.
- **a**: a positive sample which is an item the user u interacted with.
- **b**: a negative sample which is an item the user u did not interact with.

During the training process, BPR algorithms uses these samples to continuously optimize the parameters by stochastic gradient descent following the rule below where $r_{u,ab}$ represents the difference between predicted ratings for positive item a and negative item b:

$$\theta = \theta + \alpha\left(\frac{1}{1 + e^{\widetilde{r}_{u,ab}}} \cdot \frac{\partial \widetilde{r}_{u,ab}}{\partial \theta} + \lambda\theta\right) \quad (2)$$

In the scenario of AMF, the predicted rating difference $r_{u,ab}$ can be expressed as:

$$\widetilde{r}_{u,ab} = \alpha_{u,ab} + \sum_{v \in U} s_{uv} \cdot \alpha_{v,ab} + \sum_{j \in I} s_{j,ab} \cdot \alpha_{uj} \quad (3)$$

4. Datasets

The results of Simple-HGN in the original article, were based on four datasets: MovieLens, Yelp-2008, LastFM, and Amazon-book. These datasets were provided by the original authors as open-source. Additionally, since AMF was derived from Simple-HGN, in order to ensure comparability between the models' performances, we continued to use these datasets to train the various models in our experiments.

4.1. MovieLens

The GroupLens research group at the University of Minnesota developed the MovieLens dataset, which is widely used in the field of recommender system(RS). The authors of Simple-HGN model used a subset of the 20M version, where they transformed explicit ratings into implicit ones by only retaining the interaction records between users and items, while discarding the specific rating data.

4.2. Yelp-2018

The Yelp-2018 dataset is a large-scale dataset of user ratings from the Yelp platform which is an online platform that allows users to discover and review local businesses which are viewed as items, such as restaurants, cafes, bars. Yelp-2018 is adopted from the 2018 edition of the Yelp challenge.

4.3. LastFM

The LastFM dataset records user listening sequences collected and published by Last.fm which is a music website, founded in the United Kingdom in 2002. It is also often used in the field of RS. The LastFM dataset contains information about users, and the songs they have listened to, therefore in this scenario, a rating is obtained when a user listens to a song. We extract a subset with the timestamp from January,2015 to June,2015.

4.4. Amazon-book

Amazon-book is another popular dataset that is commonly used to train the models in RS. It is a collection of book ratings and reviews obtained from Amazon.com. The original dataset comprises a vast collection of over 22 million ratings for nearly 2.8 million books from more than 900,000 users, here we use the subset of it.
In order to evaluate model performance and guarantee the generalization ability of the model, we split the above four datasets into three parts: the training set, the validation set, and the test set.

5. Our work

Our forthcoming work encompasses three primary tasks. Initially, we undertake two essential preliminary tasks, namely reproduction work and ablation study. Subsequently, we develop the AMF model and conduct a series of experiments to compare its performance with other models.

5.1. Reproduction work

We reproduce the results of the Simple-HGN model on four different datasets from the original paper to verify the reliability of those results. This step is crucial to address inconsistencies between the source code and the paper or to uncover important details that may be insufficiently described or explained.

5.2. Ablation study

We perform an ablation study on the Simple-HGN model to assess the individual contributions of each component in the model structure and visually demonstrate the extent to which the pre-trained MF BPR embeddings contribute to its performance. Specifically, this analysis involves training and evaluating the Simple-HGN model after removing the pre-trained MF BPR embeddings completely. In this scenario, only the HGNN component remains, and the objective is to learn node embeddings. Additionally, we also need to evaluate the performance of the pre-trained embeddings. This can be easily accomplished by replacing the initial embeddings of the MF BPR model with the pre-trained embeddings, eliminating the need for the model training process in this case.

5.3. Experiments

After completing these tasks, we develop the new model AMF and trained it along with the Simple-HGN model and a collection of well-optimized baselines on all four datasets. In order to comprehensively evaluate the performance of AMF and compare with other algorithms, we perform the following experiments:

- **Performance analysis**: in this analysis, we measure the performance of AMF and other algorithms in terms of accuracy, coverage, and diversity based on the recommendation results obtained from the final test set. Specifically, Recall measures classification accuracy. Normalized Discounted Cumulative Gain(NDCG) measures ranking accuracy. Item-space Coverage(IC) measures coverage. Mean Inter-List (MIL)

measures diversity.

- **Carousel analysis**: In this task, our goal is not to find the single best model for recommendations, but to develop a model that complements the existing recommendations generated by other algorithms. The model that best complements the existing recommendations will have the highest accuracy. To assess the performance, we use the SLNDCG metric which is the extended version of NDCG. The model that best complements the TopPop recommendations will have the highest accuracy.
- **Popularity analysis**: this section our main focus is to analyze how different algorithms address the issue of popularity bias which refers to the phenomenon where popular items become even more popular, while less popular items are neglected. We evaluate their performance based on average popularity(AP), the Gini Index (GI) values of item popularity, and the ability to explore non-popular items.
- **Model sensitivity analysis**: AMF incorporates three different MFs, with each MF decomposing the URM into a pair of embeddings. The performance of MF models is often influenced by the size of these embeddings, which is directly determined by the size of the latent factors in the MF. We take dataset MovieLens as an example and primarily focus on the impact of different size of three latent factors in AMF on model performance.

5.4. Tables

Dataset	RECALL	NDCG
MovieLens	0.4618±0.0007	0.3090±0.0007
	0.4612	**0.3078**
Yelp-2018	0.0732±0.0003	0.0466±0.0003
	0.0729	**0.0464**
LastFm	0.0917±0.0006	0.0797±0.0003
	0.0914	**0.0795**
Amazon-book	0.1587±0.0011	0.0854±0.0005
	0.1593	**0.0855**

Table 1: The result of Recall and NDCG for reproducing work in four datasets with cutoff at 20, the reproduction results are shown in bold, the value deviates from the result range provided by the author is marked in red

Model Component	RECALL	NDCG
Complete model	0.4612	0.3078
Only HGNN remaining	0.3681	0.2285
Pre-trained embeddings	0.3992	0.2559

Table 2: The results of Recall and NDCG for ablation study in MovieLens dataset with cutoff at 20

Algorithm	Recall	NDCG	IC	MIL
Item KNN CF	**0.4377**	**0.3020**	**0.3661**	0.7489
User KNN CF	**0.4700**	**0.3301**	**0.1839**	**0.8429**
SLIM BPR	**0.4320**	0.2475	0.1568	0.8077
PureSVD	0.3527	0.2351	0.0113	0.6550
MF BPR	0.4092	0.2564	0.1393	0.8070
IALS	**0.4544**	**0.3023**	0.0759	**0.8937**
MF SVD++	**0.4224**	**0.2715**	0.0829	**0.8673**
MF ASYSVD	**0.4534**	**0.3017**	0.0834	0.7782
P^3_α	**0.4252**	**0.2891**	**0.2643**	0.7164
Simple-HGN	**0.4612**	**0.3078**	**0.2131**	**0.8299**
AMF	0.4125	0.2657	0.1664	0.8218

Table 3: The performance evaluation of algorithms on dataset MovieLens with cutoff at 20. The red font highlights the result of AMF, and the bold font indicates better performance than AMF.

Algorithm	Recall	NDCG	IC	MIL
Item KNN CF	**0.0747**	**0.0489**	**0.4001**	**0.9748**
User KNN CF	**0.0715**	**0.0477**	**0.2802**	**0.9724**
SLIM BPR	**0.0708**	**0.0466**	**0.3642**	**0.9561**
PureSVD	0.0552	0.0365	0.0643	**0.9725**
MF BPR	0.0497	0.0316	**0.3394**	**0.9736**
IALS	**0.0772**	**0.0503**	**0.2047**	**0.9905**
MF SVD++	**0.0625**	**0.0397**	**0.2523**	**0.9903**
MF ASYSVD	**0.0636**	**0.0409**	**0.1951**	**0.9871**
P_{α}^{3}	**0.0711**	**0.0467**	**0.2969**	**0.9521**
Simple-HGN	**0.0729**	**0.0464**	**0.3756**	**0.9884**
AMF	0.0559	0.0365	0.1254	0.9520

Table 4: The performance evaluation of algorithms on dataset Yelp-2018 with cutoff at 20. The red font highlights the result of AMF, and the bold font indicates better performance than AMF.

6. Conclusions

6.1. Reproduction work

Based on the results of the reproduction work shown in Table 1, we can observe that apart from a slight deviation in the NDCG result on MovieLens compared to the range provided by the author, all other results are consistent. Therefore, we can conclude that the results in the original article are reliable.

6.2. Ablation study

According to the outcomes on all four datasets(here we take the result of MovieLens as the example of result, see Table 2), the performance of the pre-trained MF BPR model is slightly superior to that of Simple-HGN with only the HGNN component remained. However, when the two models are combined, the overall performance is significantly enhanced across various datasets. Hence, it can be concluded that the HGNN alone is not particularly effective, the pre-trained MF BPR model performs relatively well, and the hybrid model successfully leverages the strengths of both approaches.

6.3. Analysis for Experiments

Here, we present the performance results of the MovieLens and Yelp-2018 datasets as examples, which can be found in Table 3 and Table 4, respectively. The evaluation of recommendation performance reveals that the AMF model consistently lags behind KNN algorithms in terms of accuracy metrics which are Recall and NDCG. The same results are also found on the other two datasets. This indicates that the recommendation quality of the AMF model is not competitive compared to KNN algorithms, which are widely recognized for their strong performance and consistent leading position among all algorithms. Meanwhile, AMF shows a significant performance gap compared to Simple-HGN in both accuracy metrics and non-accuracy metrics such as IC and MIL in all datasets, emphasizing the crucial role of the HGNN model in Simple-HGN's performance. Since the AMF model is primarily built using MF techniques, the AMF model shows comparable performance to mainstream MF algorithms like MF BPR in terms of accuracy on the three datasets other than MovieLens. However, it performs poorly on non-accuracy metrics, indicating that it generally exhibits similar recommendation quality to these mainstream MF algorithms but sacrifices item coverage and diversity in the recommendation results.

In carousel analysis, the AMF model exhibited no significant difference compared to its individual performance on most datasets, but it lagged behind Simple-HGN and other MF models. Popularity analysis indicated that the AMF model did not provide notable improvements in mitigating popularity bias or enabling exploration of long-tail items. Unfortunately, AMF consistently underperformed in these aspects compared to Simple-HGN and other MF models. In the model sensitivity analysis, we assessed the AMF model's sensitivity to latent factor sizes in the three MF methods. The results revealed that the AMF model's performance was sensitive to the sizes of latent factors in the user-item weight matrix but was relatively insensitive to the sizes of the other two latent factors in two sub-components.

7. Acknowledgements

First and foremost, I extend my sincere appreciation to my advisor, Maurizio Ferrari Dacrema, for his invaluable guidance and prompt assistance whenever I faced questions or encountered obstacles during the research process. His ad-

vice has consistently proven to be timely and highly effective. I would also like to express my heartfelt gratitude to my mother, grandmother, and friends, Yihan, Wang Hui, Luca, and Roberto. While they may not have provided academic solutions directly, their unwavering support throughout the lengthy journey of completing this paper has been immeasurable. During moments of self-doubt or when I felt stuck, they offered consolation and encouragement in their own unique ways. Lastly, I wish to acknowledge and appreciate my own determination and perseverance. It is through my continuous efforts that I have been able to complete my master's thesis, and for this, I am grateful to myself.

References

[1] Maurizio Ferrari Dacrema, Simone Boglio, Paolo Cremonesi, and Dietmar Jannach. A troubling analysis of reproducibility and progress in recommender systems research. *CoRR*, abs/1911.07698, 2019.

[2] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In Toine Bogers, Alan Said, Peter Brusilovsky, and Domonkos Tikk, editors, *Proceedings of the 13th ACM Conference on Recommender Systems, RecSys 2019, Copenhagen, Denmark, September 16-20, 2019*, pages 101–109. ACM, 2019.

[3] Maurizio Ferrari Dacrema, Federico Parroni, Paolo Cremonesi, and Dietmar Jannach. Critically examining the claimed value of convolutions over user-item embedding maps for recommender systems. 07 2020.

[4] Qingsong Lv, Ming Ding, Qiang Liu, Yuxiang Chen, Wenzheng Feng, Siming He, Chang Zhou, Jianguo Jiang, Yuxiao Dong, and Jie Tang. Are we really making much progress? revisiting, benchmarking and refining heterogeneous graph neural networks. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 1150–1160, 2021.

[5] Fernando Benjamín Pérez Maurera, Maurizio Ferrari Dacrema, and Paolo Cremonesi. An evaluation study of generative adversarial networks for collaborative filtering. In Matthias Hagen, Suzan Verberne, Craig Macdonald, Christin Seifert, Krisztian Balog, Kjetil Nørvåg, and Vinay Setty, editors, *Advances in Information Retrieval - 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10-14, 2022, Proceedings, Part I*, volume 13185 of *Lecture Notes in Computer Science*, pages 671–685. Springer, 2022.

[6] Steffen Rendle, Walid Krichene, Li Zhang, and John R. Anderson. Neural collaborative filtering vs. matrix factorization revisited. In Rodrygo L. T. Santos, Leandro Balby Marinho, Elizabeth M. Daly, Li Chen, Kim Falk, Noam Koenigstein, and Edleno Silva de Moura, editors, *RecSys 2020: Fourteenth ACM Conference on Recommender Systems, Virtual Event, Brazil, September 22-26, 2020*, pages 240–248. ACM, 2020.

[7] Harald Steck. Embarrassingly shallow autoencoders for sparse data. In Ling Liu, Ryen W. White, Amin Mantrach, Fabrizio Silvestri, Julian J. McAuley, Ricardo Baeza-Yates, and Leila Zia, editors, *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 3251–3257. ACM, 2019.