Research on the Solutions to Cold-Start Problems

Yuehan Qin[1](✉), Wuji Chang[2], Songtao Zhang[3], and Yihuan Yan[4]

[1] School of Information Science and Technology, Hangzhou Normal University, Hangzhou, Zhejiang, China
qinyuehan585@gmail.com
[2] Shanghai United International School, Shanghai, China
[3] Shanghai HongRunBoYuan School, Shanghai, China
[4] Pennsylvania State University, University Park, Pennsylvania, USA

Abstract. The recommender system has seeped into each corner of the earth, but any recommender system has to experience the process of lacking data at the beginning. So, how to recommend things well in that condition becomes a cold start problem. The recommendation system cold start problem has always been a big problem in recommendation systems. Many different algorithms are developed in order to better solve the problem. The cold start problem is studied in this paper. The authors try to use neural co-filtering, autoRec and DropoutNet to study this. By comparing different methods to deal with data without project characteristics or user characteristics, it is concluded that DropoutNet has made some improvements to the dataset with new uses or projects.

Keywords: Cold-Start · NeuralCF · AutoRec · DropoutNet

1 Introduction

Nowadays, the recommender system has seeped into each corner of the earth, from garbage e-mail filtering to good recommendations in the e-economics. No matter its impact is good or not, It has developed its own unbreakable position in the big data era. Many kinds of recommender systems have been made, from Collaborative Filtering to logistic regression algorithms, etc. They are all chasing one goal, how to recommend things more precisely in a more efficient way. However, any recommender system has to experience the process of lacking data at the beginning. So, how to recommend things well in that condition becomes a cold start problem.

1.1 Standard Method for Cold Start

Cold starts are separated into three categories, user cold start, item cold start and system cold start. The user cold starts are recommended things for users when there is no history data. The item cold start problem. The item cold start is to focus on recommending the items which have no interaction records with any users, especially the new items. The

M. F. b. Sedon et al. (Eds.): SSHA 2023, ASSEHR 752, pp. 12–18, 2023.
https://doi.org/10.2991/978-2-38476-062-6_3

system cold start is recommended things when there is no or less related history data for items and users. At the beginning of a recommender system, the system can recommend the popular items to users or the system can asks user's information at the beginning, use their login data to do recommend or randomly choose items to judge the user's interest, etc.

1.2 Comparing Algorithm

In this essay, we try to compare and summary cold start algorithms with different frameworks, we mainly discuss Filtering, autoRec and Dropout-Net algorithm and reach a summary of their comprehensive ability and illustrating each advantage and disadvantages.

2 Common Frameworks of Cold-Start

2.1 Difference Between Warm-Start and Cold-Start Recommendation System

When we are dealing with warm-start data, we will have more choices. For example, when we are facing with more characteristic values, we can extract characteristic values to calculate items that users may be interested in. When we are facing with cold start data, we may face the lack of data, or even start with new data. In the case of enough data, we can use any recommendation algorithms. However, in the case of cold start, we need to recommend through the content information or the user's registration information. There are some new solutions to deal with the cold-start recommendations, and compared with normal ways, they pretend to use latent factors and new algorithms like DropoutNet to solve the problems.

2.2 Cold-Start Solutions in Collaborative Filtering Algorithm

When facing the cold-start problem of collaborative filtering algorithm, Golbandi et al. proposed to construct the decision tree of initial questionnaire through bootstrapping algorithm [1]. They constructed a decision tree for the initial questionnaire, each node is a question, allowing the recommender to query the user adaptive according to the user's previous answers. And the function matrix factorization proposed by Zhou et al. [2].

$$T, V = \min_{T \in H, V} \sum_{(i, j \in O)} (r_{ij} - v_j^T T(a_i))^2 + \lambda \|V\|^2 \tag{1}$$

In this formula, a_i is an answer set, the user profile is generated by $u_i = T(a_i)$. r_{ij} Represents the actual item rating and V_j is the latent representation of item profile. The potential feature is connected with each node of the decision tree. The user feature is a function of all possible questionnaire answers. In essence, it is an alternating iterative optimization algorithm between decision tree construction and potential profile feature generation. In addition, the deep Q network method proposed by hima et al. [3] dynamically generates form questions, passes the answer to the Multi-layer Perceptron model

(MLP), generates a predicted user embedding vector, and then is used by the cold-start user to generate the prediction rating of all movies.

$$\hat{r}_{ij} = f(u_i, m_j) \tag{2}$$

u_i is the user embedding for user i, m_j is the item embedding for movie j, and r_{ij} is the predicted rating for movie j by user i.

2.3 Cold-Start with Latent Factors

Schein A I et al. A cold start algorithm based on latent factors proposed [4]. They try not to recommend ratings in the movie lens data set, but to recommend with an implicit variable Z like the casts of the movie. Using implicit variables, content information can be used to recommend users when new users join. Compared with the recommendation algorithm based on items and users, the proposal of a pure collaborative filtering method is based on community preference and ignores user and project attributes. On the other hand, pure content-based filtering or information filtering methods usually match query words or other user data with item attribute information and ignore the data of other users. Some hybrid algorithms combine these two technologies. The same method is used in recommending music to people, such as finding a latent factor of the songs' type, by decomposing the low-rank matrix to calculate the actual scoring matrix, then removing songs already been heard and choosing the songs with the highest score in the scoring matrix to recommend to the users. Nowadays lots of recommend systems have been using the latent factor models to deal with the cold-start problem.

3 Experiments

3.1 Model Introductions

In this paper, three models of cold start recommendations are experimented upon: Neural Collaborative Filtering, autoRec, and DropoutNet.

Neural Collaborative Filtering. Neural Collaborative Filtering is composed of two parts: the Generalized Matrix Factorization (GMF), and the Multilayer Perceptron (MLP). Matrix Factorization model is a very popular method for recommendation systems. Under Neural Collaborative Filtering, the Matrix Factorization model is generalized, thus Generalized Matrix Factorization. In the GMF model, a non-linear function is used for the output, which generalizes the normal MF to a more expressive level. Multilayer Perceptron is a design that has been widely used in deep learning works. In Neural Collaborative Filtering, a standard MLP was used to add hidden layers are added upon concatenated vectors and learn the interactions. As for the fusion, the model allowed separate embeddings, and combined them by concatenating their last hidden layer.

AutoRec. AutoRec is an autoencoder framework used for collaborative filtering. It utilizes user's preferences towards items to create recommendations. It is a discriminative model based upon autoencoders, and it doesn't require much parameters [5].

DropoutNet. This model uses latent representations as the preference input in order to directly use rows and columns would be too large in their raw form. The model is aimed to achieve the goal that the model can still produce accurate representations when some inputs are missing. For training, the model simulated cold start by random drop out some of the input so that the model would perform well in cold start situations. This balances between the two extremes, one to only use content information, and the other to ignore the content and reproduce preference input [6].

3.2 Experimenting

In this section, the different models will be tested in terms of their performance.

Neural Collaborative Filtering. Using the Movielens 1M Dataset, the model Neural Collaborative Filtering is tested using code implemented with Pytorch. The result of the model is shown as follows (see Fig. 1).

AutoRec. Still using the Movielens 1M Dataset, AutoRec has the resulting error as shown in the graph below when implemented with Keras (see Fig. 2) [8].

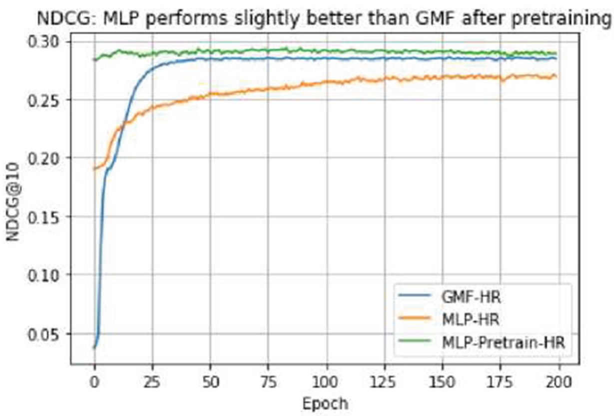

Fig. 1. The result of the model Neural Collaborative Filtering testing [7]

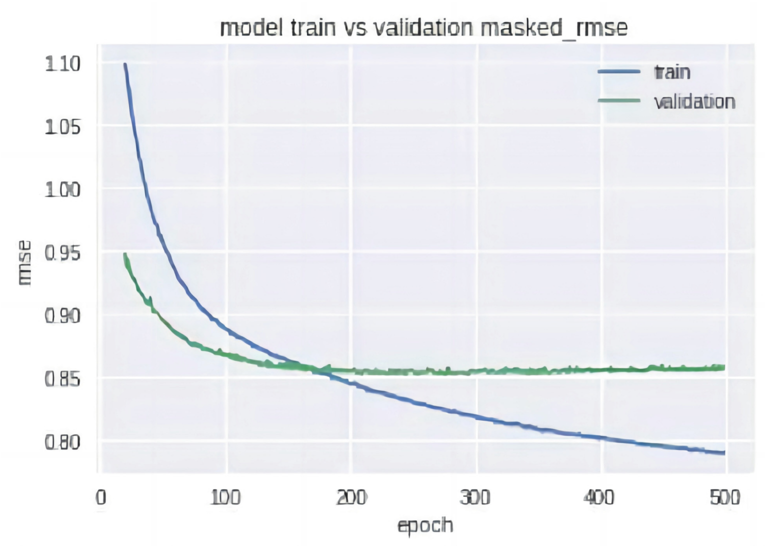

Fig. 2. Model train vs validation masked_rmse

DropoutNet. Using the Citeulike data, the DropoutNet Model presents a performance as the graph following (see Fig. 3) [9].

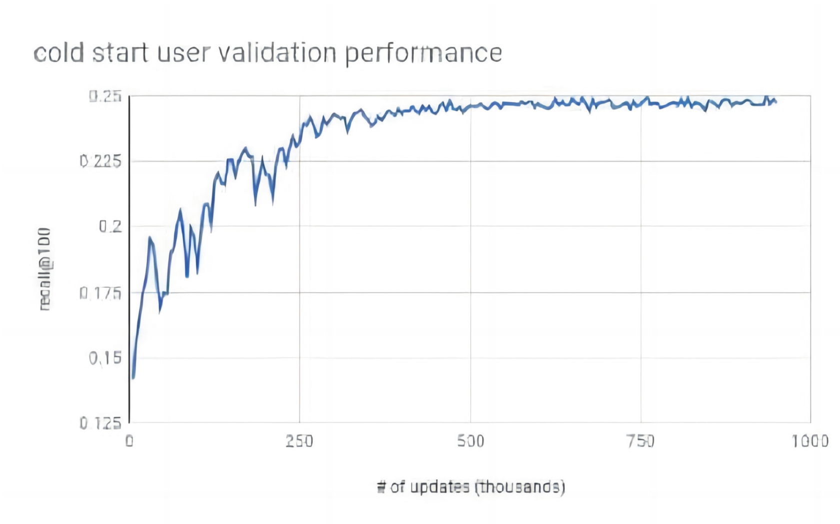

Fig. 3. Cold start user validation performance [9]

4 Conclusion

In this paper, we studied on the cold-start problem. And we tried to use Neural Collaborative Filtering, autoRec and DropoutNet to study on this. By comparing different ways of dealing with data without item features or user features, we have concluded that the DropoutNet have certain improvement to the dataset which has new usesr or items. However, by using Neural Collaborative Filtering and autoRec, the neuralcf model does not introduce more other types of features and wastes other valuable information in practical application. The advantage of u-autorec over i-autorec is that the user's score vector of all items can be reconstructed by inputting the user vector of the target user only once, but the sparsity of the user vector may affect the effect of the model.

Acknowledgments. I wish to thank Professor Woodruff, who helped us a lot with the paper and gave us many valuable suggestions. Thanks are also given to assistant Hudson, who gave us lots of beneficial suggestions on our project.

References

1. Nadav Golbandi, Yehuda Koren, and Ronny Lempel. Adaptive bootstrapping of recommender systems using decision trees. In Proceedings of the fourth ACM international conference on Web search and data mining, pages 595–604. ACM, 2011.
2. Ke Zhou, Shuang-Hong Yang, and Hongyuan Zha. Functional matrix factorizations for cold-start recommendation. In Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval, pages 315–324. ACM, 2011.
3. Dureddy H V, Kaden Z. Handling ColdStart Collaborative Filtering with Reinforcement Learning[J]. 2018.
4. Schein A I, Popescul A, Ungar L H , et al. Methods and Metrics for Cold-Start Recommendations[C] SIGIR 2002: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 11–15, 2002, Tampere, Finland. ACM, 2002.
5. Suvash Sedhain, Aditya Krisna Menon, Scott Sanner, Lexing Xie. AutoRec: Autoencoders Meet Collaborative Filtering, 2015.
6. Maksims Volkovs, Guangwei Yu, Tomi Poutanen. DropoutNet: Addressing Cold Start in Recommender Systems, December 4, 2017.
7. Kuchaiev, Oleksii, and Boris Ginsburg. "Training deep autoencoders for collaborative filtering." arXiv preprint arXiv:1708.01715 (2017).
8. Wu, Yao, et al. "Collaborative denoising auto-encoders for top-n recommender systems." Proceedings of the Ninth ACM International Conference on Web Search and Data Mining. ACM, 2016.
9. Strub, Florian, Jeremie Mary, and Ro- maric Gaudel. "Hybrid collaborative filtering with autoencoders." arXiv preprint arXiv:1603.00806 (2016).