Impact of Matrix Factorization and Regularization Hyperparameter on a Recommender System for Movies

Gess Fathan [1], Teguh Bharata Adji [2], and Ridi Ferdiana [3]
Department of Electrical and Information Technology Engineering
Universitas Gadjah Mada
Yogyakarta, Indonesia
[1]gess.fathan@mail.ugm.ac.id, [2]adji@ugm.ac.id, [3]ridi@ugm.ac.id

***Abstract*—Recommendation system is developed to match consumers with product to meet their variety of special needs and tastes in order to enhance user satisfaction and loyalty. The popularity of personalized recommendation system has been increased in recent years and applied in several areas include movies, songs, books, news, friend recommendations on social media, travel products, and other products in general. Collaborative Filtering methods are widely used in recommendation systems. The collaborative filtering method is divided into neighborhood-based and model-based. In this study, we are implementing matrix factorization which is part of model-based that learns latent factor for each user and item and uses them to make rating predictions. The method will be trained using stochastic gradient descent and optimization of regularization hyperparameter. In the end, neighborhood-based collaborative filtering and matrix factorization with different values of regularization hyperparameter will be compared. Our result shows that matrix factorization method is better than item-based collaborative filtering method and even better with tuning the regularization hyperparameter by achieving lowest RMSE score. In this study, the used functions are available from Graphlab and using Movielens 100k data set for building the recommendation systems.**

Keywords— Recommendation Systems, Collaborative Filtering, Matrix Factorization, Regularization.

I. INTRODUCTION

The numbers of data that are available in the internet are huge. Recommender Systems help to deal with this issue by giving item recommendations based on the user's preferences. There are many applications of recommendation systems in e-commerce products which consist of music, movies, travel and books.

Collaborative filtering methods are widely used in recommendation systems and is divided into neighborhood-based and model-based. Neighborhood-based collaborative filtering is also called memory-based algorithm. This algorithm works by discovering similarity patterns from users and items' past behavior towards the rating. The neighborhood-based collaborative filtering is divided into user-based collaborative filtering and item-based collaborative filtering. In this paper, the used method is item-based collaborative filtering method because it provides a better accuracy predictions rather than the user-based collaborative filtering [5].

In model-based approach, the models are built with supervised or unsupervised machine learning methods to predict the users' rating that is not rated yet based on the past rating. There are many methods in this approach including decision tree, naïve bayes, and latent factor models. The latent factor model is called by many researchers as the state-of-the art recommender system [8-10]. The latent factor method find the latent preferences of users and latent attributes of items from the ratings. Matrix factorization is the mathematical tool that can draw the latent features. Overfitting is one of the problems in recommendation systems. One way to fix this issue is by incorporating the regularization as a parameter in the Matrix factorization model [2].

In this research, GraphLab is used as a tool and Movielens 100k as the dataset. The dataset include the list of movies, users, and ratings given by users. From the dataset, the models will be trained using item-based collaborative filtering and matrix factorization with different values of regularization hyperparameter from GraphLab and compares the results in term of RMSE.

II. NEIGHBORHOOD-BASED COLLABORATIVE FILTERING

Neighborhood-based collaborative filtering method works by discovering similarity patterns from users and items' past behavior towards the rating and is divided into user-based and item-based collaborative filtering¬. Item-based collaborative filtering provides a better accuracy predictions rather than the user-based collaborative filtering, therefore item-based collaborative filtering is used [5]. The item-based collaborative filtering is shown in Fig. 1.

In Fig. 1, item 1 is chosen by three users on the left. To find which user this item should be recommend, the system finds similar items which are also chosen by those users, and then determines which other users that also choose those items. In this case, all three items are chosen by user 6. Therefore, item 1 is the best recommendation for user 6. The second best recommendation is to recommend item 1 to user 5, and so on.

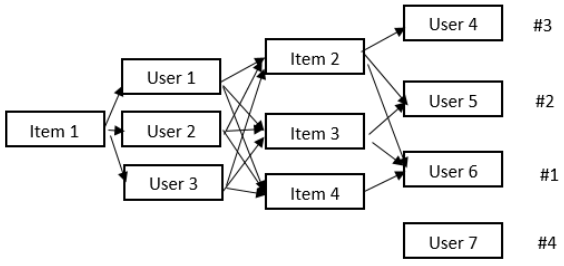

Fig. 1. Item-Based Collaborative Filtering

The models can learn user's behavior from implicit and explicit data. The examples of explicit data are rating of an item given by a user, searching of an item by a user, user preference towards two given items, and list of items that a user likes. While, examples of implicit data are what items a user views, how many times a user looks at an item, items that a user purchased, items that a user has watched or listened to, and items that a user likes or dislikes.

The collaborative filtering methods are suffering from some issues including cold start, scalability, and sparsity.

- Cold start: it requires a huge existing data to gives accurate recommendations.
- Scalability: in a big data era, a big computation power is often needed.
- Sparsity: most active users only rated a small amount of items.

III. Model-Based Collaborative Filtering

The main idea of model-based collaborative filtering approach is to predict the users' rating that is not rated yet based on the past rating. One of the main method in this approach is latent factor model. This method finds the latent preferences of users and latent attributes of items from the ratings [2].

Matrix factorization is the mathematical tool that can draw the latent features. Matrix factorization can help to overcomes the issues of neighborhood-based collaborative filtering with combination of good scalability and predictive accuracy. The recommendation items are given if there is high correspondence between item and user's factors. The illustration of matrix factorization is shown in Fig. 2. Fig. 2 characterizes user factors with male and female, and characterizes item features with serious and escapist. In matrix factorization model, the predicted rating is given by the dot product of the movie's and user's locations on the graph. For example, user 6 is expected to love movie 8 and to hate movie 1.

The basic matrix factorization method suffers from the data sparsity or missing values in user-item rating. Some works proposed to overcome this issue by minimizing the objective function or loss function and incorporate regularization to avoid overfitting. There are two approaches to minimize the objective function, stochastic gradient descent (SGD) and alternating least square (ALS).

According to [12], SGD works through iterative process of rating predictions and calculate the prediction error from each iterations. While, ALS minimizing the objective function in one time by fixing one of the parameter then compute the other parameter and vice versa. ALS is better than SGD for model that use implicit data and for parallezation process. However, SGD is generally easier and faster [11].

In the optimized matrix factorization method, regularization addresses the overfit problem. The main idea of regularization is by adding biases to control the magnitude of features to stay low. In this paper, stochastic gradient descent [5] is used to train the model. The optimization is done in parallel over multiple threads. The optimized objective function is shown in equation 1

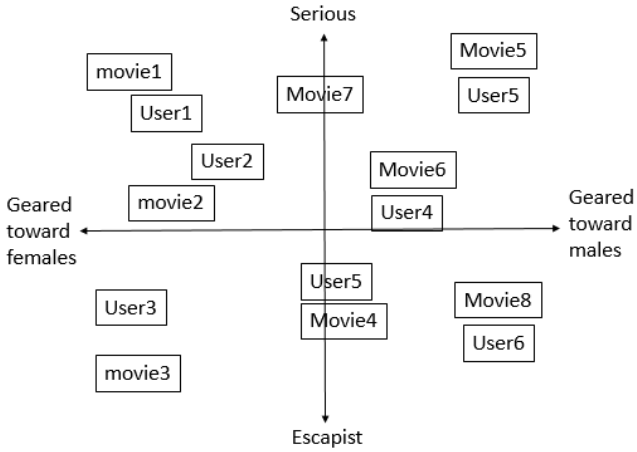

Fig. 2. Matrix Factorization approach

$$\min_{\mathbf{w},\mathbf{a},\mathbf{b},\mathbf{V},\mathbf{U}} \frac{1}{|\mathcal{D}|} \sum_{(i,j,r_{ij})\in\mathcal{D}} \mathcal{L}\left(\text{score}(i,j), r_{ij}\right) + \lambda_1(\|\mathbf{w}\|_2^2 + ||\mathbf{a}||_2^2 + ||\mathbf{b}||_2^2) + \lambda_2(\|\mathbf{U}\|_2^2 + \|\mathbf{V}\|_2^2) \quad (1)$$

where $\mathcal{D}$ is the observation dataset, r_{ij} is the rating of item j given by user i, $\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \dots)$ shows the latent factors of user, and $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \dots)$ shows the latent factors of item. $\mathcal{L}(\hat{y}, y)$ by default is the loss function of $(\hat{y} - y)^2$. There are two regularizations in this equation. The first one is λ_1 shows the *linear regularization* parameter, and the second one is λ_2 the *regularization* parameter.

IV. Related Work

Sarwar et. al. [11] from University Minnesota compared the effectiveness of recommender systems with collaborative filtering and singular value decomposition (SVD). The study showed that SVD is worse than the traditional collaborative filtering in e-commerce dataset due to the very sparse data. However, SVD works better in MovieLens dataset.

To overcome the sparsity problem, many methods have been applied. A method proposed by Ranjbar et. al. [15] try to overcomes this problem through imputation of values to the unknown rating in a matrix factorization-based method. The datasets that were used are MovieLens, Jester, and EachMovie. The proposed method outperformed another methods such as ALS, SGD, RSGD, SVD++ and MULT. Another way to improve the model performance is to incorporates regularization to make the model less likely to overfit [12,14].

A study about comparison of ALS and SGD was done by [13]. The study showed that SGD is faster, easy to implement, and less likely to overfit. However, ALS works better for a large datasets. A study by [9] compares the result between matrix factorization that incorporating regularization without tuning, biases, implicit feedback and temporal dynamic. The temporal dynamic achieved the lowest RMSE (0.8563).

Comparison between GraphLab library and MapReduce library showed that Graphlab works better in terms of accuracy and speed. However, MapReduce worked better for the model with parallelization process like ALS [16].

V. METHODOLOGY

This section will explains the dataset, tool, and implementations. Each steps are described below.

A. Dataset Description

In this study, Movie Lens – 100k is used as the dataset and it was published by Groupleans research group. The data can be retrieved from http://grouplens.org/datasets/movielens/100k/s. This dataset contains 90570 number of observation with 943 users and 1682 movies. Every user has rated minimally 20 movies. This dataset contains detail of users, ratings, and movies. The recommendation systems that are going to build consist of four systems, i.e. item-based collaborative filtering, matrix factorization with regularization hyperparameter = 1e-8, matrix factorization with regularization hyperparameter = 1e-6, and matrix factorization with regularization hyperparameter = 1e-5. We only use User_id, Movie_Id, and Rating from the dataset.

B. Tool

The tool that is going to be used is Graphlab. GraphLab is an open source project with Apache license and was started from Carnegie Mellon University by Prof. Carlos Guestrin in 2009. GraphLab is used for data mining and machine learning tasks. The main benefit of GraphLab is on dealing with big data. Graphlab also offers many methods on recommendation systems such as item-based collaborative filtering, matrix factorization, popularity-based recommender, etc. GraphLab is basically a paid package for python but free for educational purpose for 1 year. The dataset from MovieLens is loaded to GraphLab to build and evaluate the models.

C. Implementation

The implementation starts from dividing the data into training and testing data using the following codes:

```
train_data = graphlab.SFrame(ratings_base)
test_data = graphlab.SFrame(ratings_test)
```

The next step is to train our item-similarity collaborative filtering and matrix factorization models with different regularization (tuning) hyperparameters using the following codes:

```
item_sim_model = graphlab.item_similarity_recommender.create(train_data, user_id='user_id',
item_id='movie_id', similarity_type='pearson')

matrix_model = graphlab.factorization_recommender.create(train_data, user_id='user_id',
item_id='movie_id', target='rating')

matrix_model2 = graphlab.factorization_recommender.create(train_data, user_id='user_id',
item_id='movie_id', target='rating', regularization=1e-6)

matrix_model3 = graphlab.factorization_recommender.create(train_data, user_id='user_id',
item_id='movie_id', target='rating', regularization=1e-5)
```

The trained models is then used to make predictions of top 5 movies for first 5 users using the following codes:

```
item_sim_recomm = item_sim_model.recommend(users=range(1,6),k=5)
item_sim_recomm.print_rows(num_rows=25)

matrix_recomm = matrix_model.recommend(users=range(1,6),k=5)
matrix_recomm.print_rows(num_rows=25)

matrix_recomm2 = matrix_model2.recommend(users=range(1,6),k=5)
matrix_recomm2.print_rows(num_rows=25)

matrix_recomm3 = matrix_model3.recommend(users=range(1,6),k=5)
matrix_recomm3.print_rows(num_rows=25)
```

The recommendation items are shown in Table I, Table II, Table III, and Table IV and consecutively show the item-similarity collaborative filtering model, matrix factorization with default regularization = 1e-8, matrix factorization with regularization = 1e-6, and matrix factorization with regularization = 1e-5. The top 5 items are recommended for 5 users and will be based on the top prediction score of movie_id to the user_id.

The last step in implementations is evaluation. The evaluation method is using RMSE. RMSE evaluates how far the predicted rating is to the real rating in the test set.

TABLE I. RECOMMENDATION ITEMS FROM ITEM-SIMILARITY MODEL

user_id	movie_id	score	rank
1	286	0.807692307692	1
1	294	0.803643724696	2
1	288	0.779352226721	3
1	300	0.7004048583	4
1	117	0.645748987854	5
2	50	1.0	1
2	181	0.886639676113	2
2	121	0.775303643725	3
2	174	0.765182186235	4
2	98	0.706477732794	5
3	50	1.0	1
3	100	0.894736842105	2
3	286	0.807692307692	3
3	294	0.803643724696	4
3	1	0.791497975709	5
4	50	1.0	1
4	100	0.894736842105	2
4	181	0.886639676113	3
4	286	0.807692307692	4
4	294	0.803643724696	5
5	258	0.831983805668	1
5	286	0.807692307692	2
5	294	0.803643724696	3
5	1	0.791497975709	4
5	288	0.779352226721	5

TABLE II. RECOMMENDATION ITEMS FROM MF MODEL 1

user_id	movie_id	score	rank
1	641	5.98897409384	1
1	647	5.96105399673	2
1	1142	5.75104442184	3
1	513	5.68665328567	4
1	1143	5.54071626012	5
2	1084	5.20163165723	1
2	430	5.03280498777	2
2	408	5.00056316648	3
2	479	4.94514080082	4
2	1217	4.90031440687	5
3	1022	8.23701111917	1
3	1129	7.59797196035	2
3	956	7.11752150898	3
3	902	7.07752202396	4
3	896	6.81965915804	5
4	1512	7.15124506448	1
4	512	6.38825682496	2
4	537	6.29441139792	3
4	131	6.10380361174	4
4	960	6.01562090908	5
5	253	6.85750261639	1
5	835	6.34628666137	2
5	647	5.98012734149	3
5	1160	5.91109669898	4
5	1137	5.88876122926	5

TABLE III. RECOMMENDATION ITEMS FROM MF MODEL 2

user_id	movie_id	score	rank
1	580	5.99185414185	1
1	408	5.90874336232	2
1	647	5.81605891456	3
1	171	5.78312789966	4
1	661	5.75177876224	5
2	408	5.97054117118	1
2	345	5.80860196744	2
2	169	5.75896160994	3
2	1367	5.68278216515	4
2	513	5.67374794875	5
3	1065	6.77786777113	1
3	919	6.62959916835	2
3	640	6.60721805637	3
3	902	6.46627381389	4
3	390	6.31030798648	5
4	320	7.02297147308	1
4	543	6.7170946518	2
4	57	6.5523339728	3
4	1137	6.19249375854	4
4	887	6.17432463859	5
5	580	5.66781765912	1
5	640	5.55278543328	2
5	1022	5.44059067618	3
5	1065	5.4394948328	4
5	1240	5.22025699113	5

TABLE IV. RECOMMENDATION ITEMS FROM MF MODEL 3

user_id	movie_id	score	rank
1	313	4.61491753851	1
1	1449	4.60885021967	2
1	318	4.53699675147	3
1	483	4.48800209109	4
1	408	4.48317974052	5
2	1449	4.84868753181	1
2	318	4.76065751669	2
2	483	4.71997261924	3
2	408	4.71627475778	4
2	1524	4.71345534633	5
3	1449	4.81601162013	1
3	313	4.6885892287	2
3	169	4.68405101393	3
3	1524	4.68321508184	4
3	408	4.67280998538	5
4	1449	4.87368424135	1
4	313	4.77035822012	2
4	1524	4.73541900682	3
4	408	4.72595168094	4
4	114	4.71749826745	5
5	1449	4.1318826642	1
5	313	4.07762528156	2
5	318	3.99792723675	3
5	1524	3.99695160604	4
5	114	3.97113763206	5

VI. RESULT AND DISCUSSION

The RMSE evaluation is shown in Table V :

TABLE V. RECOMMENDATION ITEMS FROM MF MODEL 1

	Item-similarity model	MF model1	MF model2	MF model3
RMSE	3.43530307	1.06577743	1.05890407	0.997194837

The MF model1 is a matrix factorization method with regularization hyperparameter = 1e-8. In model2 and model3, the regularizations are 1e-6 and 1e-5. In table 5, MF model3 outperforms the other methods with the lowest RMSE score or highest accuracy. This explains that matrix factorization model with regularization (tuning) hyperparameter will result in better accuracy than that of the item-similarity collaborative filtering model.

VII. CONCLUSIONS

In this study, item-similarity collaborative filtering method and matrix factorization method with different values of regularization hyperparameters are used. The best recommender model is the one with the lowest RMSE score. The result in Table V shows that matrix factorization with regularization hyperparameter = 1e-5 outperforms the other methods in term of RMSE. Therefore, the matrix factorization with the smallest regularization hyperparameter results in better recommendations. For future work, we can incorporate the matrix factorization with more features such as user's gender, user's age and movie genre. The model can also be improved through preprocessing such as using imputation and removing extreme values.

REFERENCES

[1] V. Khadse, A. P, S. Basha, N. Iyengar and R. Caytiles, "Recommendation Engine for Predicting Best Rated Movies", International Journal of Advanced Science and Technology, vol. 110, pp. 65-76, 2018.

[2] Yehuda Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, New York, NY, USA, 2008, pp. 426-434

[3] Leon Bottou, "Stochastic Gradient Tricks," Neural Networks : Tricks of the Trade. Vol. 7700, pp. 430–445, 2012.

[4] Ilhami, Mirza & Suharjito, Suharjito, "Film Recommendation Systems using Matrix Factorization and Collaborative Filtering," International Conference on Information Technology Systems and Innovation (ICITSI), 2014.

[5] B. Sarwar, G. Karypsis, & J. Konstan, "Item-based Collaborative Filtering Recommendation Algorithms, " 10th International World Wide Web Conference (WWW10), 2001.

[6] P. Cremonesi, Y. Koren, & R. Turrin, . "Performance of recommender algorithms on top-n recommendation tasks, " In Proceedings of the fourth ACM conference on Recommender systems - RecSys '10 (p. 39)., 2010.

[7] G. Jawaheer., P. Weller, & P. Kostkova, "Modeling User Preferences in Recommender Systems: A Classification Framework for Explicit and Implicit User Feedback", ACM Transactions on Interactive Intelligent Systems, 2014.

[8] J. A. Konstan & J. Riedl, "Recommender systems: From algorithms to user experience", User Modelling and User-Adapted Interaction, 2012.

[9] Y. Koren, R. Bell, & C. Volinsky,"Matrix Factorization Techniques for Recommender Systems", Computer, vol. 42, no. 8, pp. 30-37, 2009.

[10] S. Funk, "Netflix Update: Try This at Home," Dec. 2006; http://sifter.org/~simon/journal/20061211.html.

[11] B.M. Sarwar. G. Karypis, J. Konstan, and J. Riedl, "Application of Dimensionality Reduction in Recommender System—A Case Study," Proc. KDD Workshop on Web Mining for e-Commerce: Challenges and Opportunities (WebKDD), ACM Press, 2000

[12] A. Paterek, "Improving Regularized Singular Value Decomposition for Collaborative Filtering", Proc. KDD Cup and Workshop, 2007.

[13] L. Zheng. "Performance evaluation of latent factor models for rating prediction". Master's thesis, Beijing University of Posts and Telecommunications, 2015.

[14] EV Cervantes, LC Quispe, and JO Luna, "Performance of alternating least squares in a distributed approach using graphlab and mapreduce", 2nd Annual International Symposium on Information Management and Big Data, 1478:122, 2015.

[15] M. Ranjbar, P. Moradi, M. Azami, M. Jalili, "An imputation-based matrix factorization method for improving accuracy of collaborative filtering systems", Engineering Applications of Artificial Intelligence, vol. 46, Issues PA, pp. 58-66, 2015.