Gaussian Iteration: A Novel Way to Collaborative Filtering

Xiaochun Li[1], Fangqi Li[1], Ying Guo[1] and Jinchao Huang[1]

[1] Department of Electronic Engineering, Shanghai Jiao Tong University,800 Dongchuan Road, Min Hang, Shanghai 200240,P.R. China
{xiaochun_lee,solour_lfq}@sjtu.edu.cn

Abstract. Based on the missing not at random assumption and central limit theorem, this paper presents a novel way to accelerate the iteration speed in the collaborative filtering models called Gaussian iteration. In the proposed model, adding the Gaussian distribution to the estimation error makes the falling direction more credible, which significantly reduces the running time with the ideal accuracy. For evaluation, we compare the performance of the proposed model with three existing collaborative filtering models on two kinds of Movielens datasets. The results indicate that the novel method outperforms the existing models and it is easy to implement and faster. Moreover, the proposed model is scalable to the analogous objective function in other models.

Keywords: collaborative filtering, Gaussian iteration, recommender system

1 Introduction

With the rapid development of computers and computer networks, we are facing the revolution of big data [1]. People get stuck by the mass data when they surf the Internet. Customers cannot find out their favorite items effectively and the qualified commodity can't be known by more person. In the recent decades, thousands of scholars and researchers have focused on solving the problem. Generally speaking, recommender systems have been one of effective technologies to address such challenges [2].

Recommender systems can be divided into two different strategies, the content based approach [3] and the collaborative filtering (CF) [4]. The former needs a profile for each user or item to describe its feature. Researchers analyze the content and figure out the matching products or users. However, the shortcoming about the content based approach is that it requires gathering further information that might not be easily collected. The latter, CF, is the subject that we focus on, which analyses the relationship between users and products through users' historical record. On the whole, neighborhood models and latent factor models are the major approach in CF. Neighborhood models mean to locate the most parallel users set for the target user. In order to identify the neighborhood, researchers can calculate the similarity based on user or item. The user-oriented approach and item-oriented approach have a lot of practices, see [5, 6] for more details. The neighborhood models have many ways to

calculate the similarity, e.g. locate the group/community [7, 8], and modify the correlation coefficient [9, 10]. Latent factor models comprise an alternative approach to CF with the more holistic goal to uncover latent features that explain observed ratings; examples include pLSA [11], neural networks [12], and Latent Dirichlet Allocation [13]. The main models in latent factor models were induced by Singular Value Decomposition (SVD) [14] on the user-item ratings matrix. A typical model of latent factor models could be decomposed into user-factors matrix and item-factors matrix. We get the prediction by taking an inner product with a user-factors vector and an item-factors vector. More information can be imported into the model, such as the user/item bias, explicit feedback data [14].

Neighborhood models and latent factor models can be integrated into one integrated model. Paper [15] reveals the outperformance. The integrated model introduces more parameters into the model, therefore, represents the observations in more details. The integrated models, or ensemble learning have been a hot topic in the academia.

In this paper, we concentrate upon the collaborative filtering approach, including neighborhood models, latent factor models and the integrated models [15].

The rest of this paper is organized as follows. Section II reviews the related work including global neighborhood models, SVD++ and an integrated model. Section III introduces the proposed Gaussian iteration model. Experiments are demonstrated in Section IV to verify the effectiveness of the proposed model. Last section concludes this paper.

2 Previous Work

In order to give the readers clear understanding of our proposed model, we make a brief introduction about the main collaborative filtering. At first, we reserve special indexing letters for distinguishing users from items: for users u, v and for items i, j . A rating r_{ui} indicates the preference by user u of item i , where high values mean stronger preference. For example, values can be integers ranging from 1 (star) indicating no interest to 5 (star) indicating a strong interest. We distinguish predicted ratings from known ones, by using the notation r_{ui} for the predicted value of r_{ui} . The (u,i) pairs for which r_{ui} is known are stored in the set $\kappa = \{(u,i) \mid r_{ui} \text{ is known}\}$. Usually the vast majority of ratings are unknown.

2.1 Neighborhood Models

The most common approach to CF is based on neighborhood models. The used models are limited of the local neighborhood, which cause to the deviation about the prediction. In this part, we introduce the global neighborhood model [16], which uses the all users in the data set. First, let us make a definition of the baseline estimates:

$$b_{ui} = \mu + b_u + b_i$$

* MERGEFORMAT (1)

The parameters b_u and b_i indicate the observed biases of user u and item i, respectively, from average. Here, μ denotes the overall average rating. Such a simple model would not represent the data set entirely. In this case, Y.Koren [16] came up with the global neighborhood model, and he considered the effect of the overall user in the data set. An improved model can be expressed with the formula as the following:

$$r_{ui} = \mu + b_u + b_i + |R(u)|^{-\frac{1}{2}} \sum_{j \in R(u)} (r_{uj} - b_{uj}) w_{ij} + |N(u)|^{-\frac{1}{2}} \sum_{j \in N(u)} c_{ij} \quad \text{* MERGEFORMAT (2)}$$

Here, r_{ui} means the prediction of user u to the item i, and $R(u)$ is the set that users give their ratings to the target item i. On the meantime, |R(u)| is the size of the set, in other words, is the number of the users that have given their preferences to the item i. Compared with the baseline estimates, the $(r_{uj} - b_{uj})$ can be regarded as the bias of user j. Then, w_{ij} reveals the correlation coefficient of the bias. And c_{ij} is the implicit feedback, which provides an alternative way to learn user preferences. In order to weaken the dichotomy between heavy raters and those rare raters, we use $|N(u)|^{-\frac{1}{2}}$ to moderate the behavior. Here, $N(u)$ is the set that other users show their preferences to the target item i, and $|N(u)|$ is the size of the set. In our experiments, mostly, c_{ij} would indicate the rated users' preferences for the target item. In other words, |R(u)| is equal to $|N(u)|$.

2.2 Latent Factor Models

A popular approach to latent factor models is induced by a SVD-like lower rank decomposition of the ratings matrix. Each user u is associated with a user-factors vector $p_u \in$ ^{f}, and each item i with an item-factors vector $q_i \in$ ^{f}. Prediction is done by the rule:

$$r_{ui} = b_{ui} + p_u{}^T q_i \quad \text{* MERGEFORMAT (3)}$$

Y.Koren [16] suggested the SVD++ model, which takes the implicit feedback into consideration. With the modification of the implicit feedback, we could get more accurate prediction about the missing ratings. The model can be described as the following formula:

$$r_{ui} = b_{ui} + q_i^T \left(p_u + |N(u)|^{-\frac{1}{2}} \sum_{j \in N(u)} y_j \right) \quad \text{* MERGEFORMAT (4)}$$

Formula (4) adds the implicit feedback y_j into the influences of the user-factors vector p_u, where y_j denotes the preference of the user who had given his rating to the target item.

2.3 An Integrated Models

As mentioned in [15], there is no perfect model. Instead, the best results came from combining predictions of models that complemented each other. Y.Koren [16] came up with the integrated model that combined neighborhood models and latent factor models, as following:

$$r_{ui} = \mu + b_u + b_i + q_i^T \left(p_u + |N(u)|^{-\frac{1}{2}} \sum_{j \in N(u)} y_j \right) + |R(u)|^{-\frac{1}{2}} \sum_{j \in R(u)} (r_{uj} - b_{uj}) w_{ij} + |N(u)|^{-\frac{1}{2}} \sum_{j \in N(u)} c_{ij} \quad \text{* MERGEFORMAT (5)}$$

In general, the parameters in the models mentioned above can be learnt by solving the regularized least squares problem with stochastic gradient descent. Taking model (5) as an example, the objective function can be described as formula (6). Here, the objective function can be divided into estimation errors $\left(r_{ui} - r_{ui}\right)^2$ and regularization term $\lambda\left(b_u^2 + \quad + c_{ij}^2\right)$. It is obvious that the learning process takes a long time and many iterations [16]. Further, the better accuracy can be learned from the observations.

$$\min_{b_*, q_*, p_*, y_*, w_*, c_*} \sum_{(u,i) \in \kappa} \left(r_{ui} - r_{ui}\right)^2 + \lambda\left(b_u^2 + b_i^2 + \| q_i \|^2 + \| p_u \|^2 + \| y_j \|^2 + w_{ij}^2 + c_{ij}^2\right) \quad \text{* MERGEFORMAT (6)}$$

3 Proposed Model

[17, 18, 19] proposed the missing not at random assumption, which means the missing/unknown ratings don't follow the same distribution about the observed ratings. The collaborative filtering approach split the observed data into training set and test set, using the known information to predict the unknown data. Inspired by the missing not at random assumption, we propose a novel approach to complement the bias between the observed data and the missing data. It is necessary to add some kinds of noise into the model, making the parameters more generalization. In the meanwhile, we would like to reduce the iterations and running time of approach, because of the massive data. Hence, we come up with the Gaussian iteration collaborative filtering, an approach to revise the loss function. Next section would give unambiguous presentation on the proposed method.

3.1 Revised Objective Function

As mentioned in section 2, the objective function is combined with two parts: the estimation errors and the regularization term. The regularization term can prevent the model from overfitting. Considering the biases between observed data and unknown

data, our proposed approach adds the Gaussian distribution to the estimation errors. The probability density of the normal distribution is:

$$f(x \mid \mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad \text{* MERGEFORMAT (7)}$$

Here, μ is the mean or expectation of the distribution. The parameter σ is its standard deviation with its variance then σ^2 . Let us denote the prediction error, $r_{ui} - r_{ui}$, by e_{ui} .In our model, the estimation errors in the objective function become:

$$\left(1 + f(r_{ui} \mid \mu, \sigma^2)\right)\left(r_{ui} - r_{ui}\right)^2 \quad \text{* MERGEFORMAT (8)}$$

The mean μ and the standard deviation σ^2 can be tuned by the cross validation. According to the heuristic algorithm, we can start with the mean and the standard deviation of the ratings in the training set. In this case, the objective function of the global neighborhood model (2) would change into the following formula:

$$\min_{b_*, w_*, c_*} \sum_{(u,i)\in\kappa} \left(1 + f(r_{ui} \mid \mu, \sigma^2)\right) \left(\begin{array}{l} r_{ui} - \mu - b_u - b_i - |R(u)|^{-\frac{1}{2}} \sum_{j\in R(u)} (r_{uj} - b_{uj}) w_{ij} + \\ |N(u)|^{-\frac{1}{2}} \sum_{j\in N(u)} c_{ij} \end{array} \right)^2 \quad \text{* MERGEFORMAT (9)}$$

$$+\lambda_1\left(b_u^{\,2} + b_i^2\right) + \lambda_2\left(\sum_{j\in R(u)} w_{ij}^2 + \sum_{j\in N(u)} c_{ij}^2\right)$$

In the meanwhile, the objective function of the SVD++ (4) and the integrated model (5) have been revised as (9) and (10), respectively.

$$\min_{b_*, q_*, p_*} \sum_{(u,i)\in\kappa} \left(1 + f(r_{ui} \mid \mu, \sigma^2)\right) \left(r_{ui} - \mu - b_u - b_i - q_i^{\,T}\left(p_u + |N(u)|^{-\frac{1}{2}} \sum_{j\in N(u)} y_j \right) \right)^2 \quad \text{* MERGEFORMAT (10)}$$

$$+\lambda_3\left(b_u^{\,2} + b_i^2\right) + \lambda_4\left(\|q_i\|^2 + \|p_u\|^2 + \|y_j\|^2\right)$$

$$\min_{b_*, q_*, p_*, y_*, w_*, c_*} \sum_{(u,i)\in\kappa} \left(1 + f(r_{ui} \mid \mu, \sigma^2)\right)\left(r_{ui} - r_{ui}\right)^2 + \lambda_5\left(b_u^{\,2} + b_i^2\right) \quad \text{* MERGEFORMAT (11)}$$

$$+\lambda_6\left(\|q_i\|^2 + \|p_u\|^2 + \|y_j\|^2\right) + \lambda_7\left(w_{ij}^{\,2} + c_{ij}^2\right)$$

3.2 Gaussian Iteration

In the revised objective function, the new parameters in the Gaussian distribution r_{ui}, μ, σ^2 are regarded as constants in each iteration. Therefore, the gradients of the collaborative filtering approaches would be described as shown in Table 1.

We still use the stochastic gradient descent to update the parameters, randomly select the user's rating to train the model.

Table 1. The gradients of the collaborative filtering approaches

The Gaussian iteration on Global Neighborhood Model:
$\bullet b_u \leftarrow b_u + \gamma_1 \left(\left(1 + f(r_{ui} \mid \mu, \sigma^2) \right) e_{ui} - \lambda_1 \ b_u \right)$
$\bullet b_i \leftarrow b_i + \gamma_1 \left(\left(1 + f(r_{ui} \mid \mu, \sigma^2) \right) e_{ui} - \lambda_1 \ b_i \right)$
$\bullet \forall j \in R(u):$
$w_{ij} \leftarrow w_{ij} + \gamma_2 \ (\mid R(u) \mid^{-\frac{1}{2}} \left(1 + f(r_{ui} \mid \mu, \sigma^2) \right) e_{ui} \ (r_{uj} - b_{uj}) - \lambda_2 \ w_{ij})$
$\bullet \forall j \in N(u):$
$c_{ij} \leftarrow c_{ij} + \gamma_2 \ (\mid N(u) \mid^{-\frac{1}{2}} \left(1 + f(r_{ui} \mid \mu, \sigma^2) \right) e_{ui} - \lambda_2 \ c_{ij})$

The Gaussian iteration on Latent Factor Model:
$\bullet b_u \leftarrow b_u + \gamma_3 \left(\left(1 + f(r_{ui} \mid \mu, \sigma^2) \right) e_{ui} - \lambda_3 \ b_u \right)$
$\bullet b_i \leftarrow b_i + \gamma_3 \left(\left(1 + f(r_{ui} \mid \mu, \sigma^2) \right) e_{ui} - \lambda_3 \ b_i \right)$
$\bullet q_i \leftarrow q_i + \gamma_4 \left(\left(1 + f(r_{ui} \mid \mu, \sigma^2) \right) e_{ui} \left(p_u + \mid N(u) \mid^{-\frac{1}{2}} \sum_{j \in N(u)} y_j \right) - \lambda_4 \ q_i \right)$
$\bullet p_u \leftarrow p_u + \gamma_4 \left(\left(1 + f(r_{ui} \mid \mu, \sigma^2) \right) e_{ui} \ q_i - \lambda_4 \ p_u \right)$
$\bullet \forall j \in N(u):$
$y_j \leftarrow y_j + \gamma_4 \left(\left(1 + f(r_{ui} \mid \mu, \sigma^2) \right) e_{ui} \ q_i \mid N(u) \mid^{-\frac{1}{2}} - \lambda_4 \ y_j \right)$

The Gaussian iteration on the integrated model:

$\bullet b_u \leftarrow b_u + \gamma_5 \left(\left(1 + f(r_{ui} \mid \mu, \sigma^2)\right) e_{ui} - \lambda_5 \; b_u \right)$

$\bullet b_i \leftarrow b_i + \gamma_5 \left(\left(1 + f(r_{ui} \mid \mu, \sigma^2)\right) e_{ui} - \lambda_5 \; b_i \right)$

$\bullet q_i \leftarrow q_i + \gamma_6 \left(\left(1 + f(r_{ui} \mid \mu, \sigma^2)\right) e_{ui} \left(p_u + |N(u)|^{-\frac{1}{2}} \sum_{j \in N(u)} y_j \right) - \lambda_6 \; q_i \right)$

$\bullet p_u \leftarrow p_u + \gamma_6 \left(\left(1 + f(r_{ui} \mid \mu, \sigma^2)\right) e_{ui} \; q_i - \lambda_6 \; p_u \right)$

$\bullet \forall j \in N(u):$

$y_j \leftarrow y_j + \gamma_6 \left(\left(1 + f(r_{ui} \mid \mu, \sigma^2)\right) e_{ui} \mid N(u) \mid^{-\frac{1}{2}} q_i - \lambda_6 \; y_j \right)$

$\bullet \forall j \in R(u):$

$w_{ij} \leftarrow w_{ij} + \gamma_7 \left(|R(u)|^{-\frac{1}{2}} \left(1 + f(r_{ui} \mid \mu, \sigma^2)\right) e_{ui} \left(r_{uj} - b_{uj} \right) - \lambda_7 \; w_{ij} \right)$

$\bullet \forall j \in N(u):$

$c_{ij} \leftarrow c_{ij} + \gamma_7 \left(|N(u)|^{-\frac{1}{2}} \left(1 + f(r_{ui} \mid \mu, \sigma^2)\right) e_{ui} - \lambda_7 \; c_{ij} \right)$

4 Experiments

In this section, we have conducted the experiments on three main collaborative filtering models to examine the performance of our proposed algorithm. Section 4.1 gives the description of the experiment datas, section 4.2 introduces the adjustment process of the parameters in each model and section 4.3 shows the experiment results.

4.1 Data Description

In order to verify the proposed approach, experiments were conducted on the three models mentioned before, which can make a valid conclusion. The datasets we used for the experiments are Movielens 1M and Movielens 10M [20]. In the experiments, we use the user ID, item ID and the ratings. Table 2 shows the characteristics of the datasets. The ratings in 10M dataset and 1M dataset are made on a 5-star scale, with half-star increments and one-star increments, respectively. Analyzing the 10M data, it is easy to find out that the number of the movies is not consequent, which wastes the space of the storage. For saving the memory of computer, we pretreat the dataset by mapping the movie number into the successive one ranged 1 to 10677. The number of rated users had the same treatment ranged 1 to 69878. The datasets are split randomly into training set and test set, 80% ratings on each user for training set and 20% ratings for test set.

Table 2. Datasets information

Dataset	Number of users	Number of movies	Number of ratings	Rating range
1M	6040	3952	1000209	[1,5]
10M	71567	10681	10000054	[1,5]

In this paper, we use Root Mean Square Error (RMSE) to compare the performance of the models. The RMSE is defined as (12). Here, T_κ is the set of known ratings of the test set and $n(T_\kappa)$ is the number of ratings in T_κ. For the dataset on three models, we ran the algorithm 5 times. In each experiment, the training set and test set were randomly selected. The final result of the dataset in the model is the average RMSE on the test set. In each model, we stop the updating of the parameters once the RMSE of the test set start increasing.

$$\sqrt{\frac{\sum_{(u,i)\in T_\kappa}\left(r_{ui}-r_{ui}\right)^2}{n(T_\kappa)}} \qquad \backslash^* \text{ MERGEFORMAT (12)}$$

4.2 Parameters Tuning

As mentioned above, there are some kinds of hyper parameter in the models. In general, there are learning rate, regularization coefficient, the mean and the standard deviation, which can be tuned by cross validation. The adjustment of the standard deviation σ is in reference to heuristic algorithm. For example, the mean of one Movielens 1M training set is 3.58177, and the standard deviation is 1.125. Keeping the mean of the training set in the model, σ_1 is started from the standard deviation of the training set. Through the experiment, we select $\sigma_1 = 1.8$ as the optional choice. Table 3 shows the settings of experiment in the validation models. In both of latent factor models and the integrated models, the dimension of the user-factors vector $p_u \in$ ^{f} and item-factor vector $q_i \in$ ^{f} were set to 50.

Table 3. The hyper parameters for each dataset

	Neighborhood model	Latent factor model	Integrated model
1M	$\gamma_1=\gamma_2=0.005$, $\lambda_1=\lambda_2=0.002$, $\sigma_1=1.8$	$\gamma_3=0.009, \gamma_4=0.008$, $\lambda_3=0.005, \lambda_4=0.02$, $\sigma_1=1.8$	$\gamma_5=0.007, \gamma_6=0.007$, $\gamma_7=0.005, \lambda_5=0.015$, $\lambda_6=0.015, \lambda_7=0.05$, $\sigma_1=1.8$
10M	$\gamma_1=\gamma_2=0.005$, $\lambda_1=\lambda_2=0.002$, $\sigma_1=1.3$	$\gamma_3=0.0045, \gamma_4=0.0045$, $\lambda_3=0.015, \lambda_4=0.015$, $\sigma_1=1.3$	$\gamma_5=0.007, \gamma_6=0.007$, $\gamma_7=0.005, \lambda_5=0.015$, $\lambda_6=0.05, \lambda_7=0.002$, $\sigma_1=1.3$

4.3 Experiment Results

Fig.1-3 show the experiment results of the Gaussian iteration on the Movielens datasets on three CF models, the lower RMSE the better, the less time the better and the fewer iteration number the better. All the experiments were done on a single processor 3.2 GHz Intel(R) Core(TM) PC. As shown in Fig.1-3, we can see that the RMSE on the Gaussian models is slightly worse than the original ones, but the running time and iteration number have significantly reduced. The decrease rate on the running time is more than 20%. Considering that the amount of information generated from the Internet is tremendous, it is necessary to cut down the running time.

Next, we make an explanation of the experiment results. First, how the proposed models can significantly reduce the running time? The central limit theorem (CLT) states that, given certain conditions, the arithmetic mean of a sufficiently large number of iterates of independent random variables, each with a well-defined expected value and well-defined variance, will be approximately normally distributed, regardless of the underlying distribution [21]. In this case, we can conclude that most of the user will give the same ratings on the certain item. In the process of the stochastic gradient descent, once the prediction r_{ui} is near to the mean of the training set, we have the confidence that the descent direction is trustable and the declined step can be increased. As we all know, the Gaussian distribution is bell-shaped, symmetrical about the mean and the larger the variance the steeper the curve. Hence, adding the factor $\left(1+f(r_{ui}\mid\mu,\sigma^2)\right)$ to the prediction error is the good way to accelerate the speed of the iteration. In each iteration, the nearer the prediction to the mean, the larger the falling gradient, vice versa. Secondly, why adding the new factor would not make the prediction become worse? As mentioned in section 3, our proposed models are based on the missing not at random assumption, which indicates the deviation between the missing data and the observed data. The observation is just a small part of the whole massive data. It is feasible to normalize the objective function with Gaussian distribution. In other words, the proposed approach has the ability to balance the noise among data, making the models more robust.

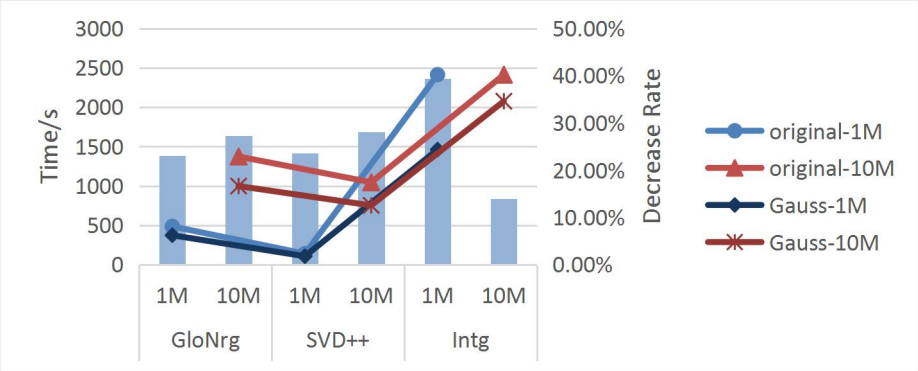

Fig.1. Comparison on running time on 1M dataset and 10M dataset

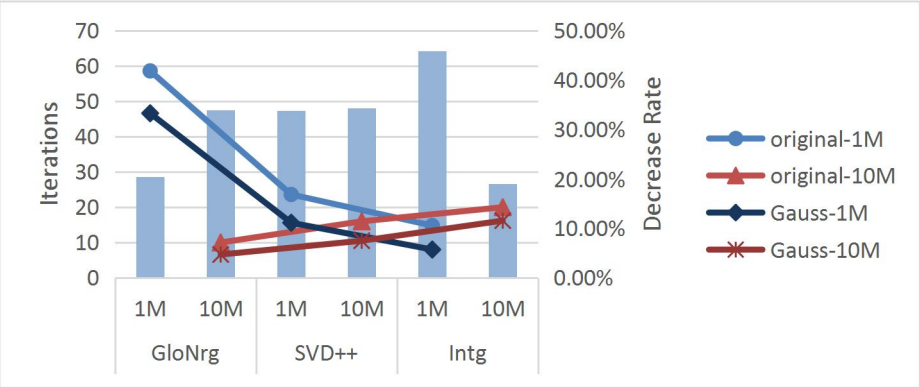

Fig.2. Comparison on iterations on 1M dataset and 10M dataset

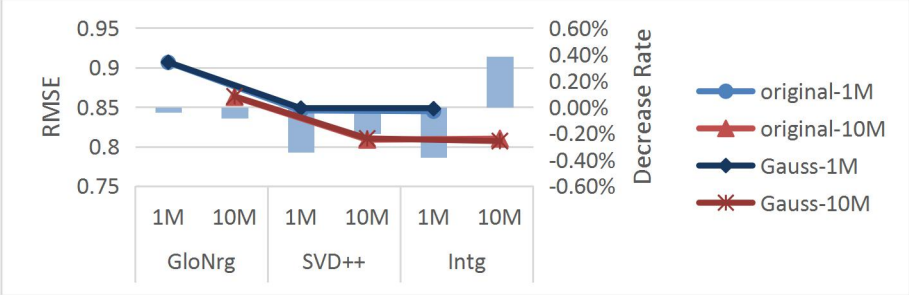

Fig.3 Comparison on RMSE on 1M dataset and 10M dataset

Fig.1-3 show the performance of the proposed model in the collaborative filtering models with the Movielens datasets. Here, we denote the global neighborhood model (2) as GloNrg, the SVD++ model (4) as SVD++ and the integrated model (5) as Intg. The line with circular and with diamond represent the original collaborative filtering approaches on the 1M dataset and 10 M dataset, respectively. The line with diamond and with star represent our proposed model on the 1M dataset and 10M dataset, respectively.

5 Conclusions

In this work, we proposed a new, simple and efficient way to accelerate the iteration speed and optimize the performance on the collaborative filtering approach called Gaussian iteration. In this method, a Gaussian factor was added to the prediction error in the objective function, based on the missing not at random assumption and the central limit theorem. The practical experiment results on three kinds of models with the Movielens datasets showed that our proposed approach was better than the original method in performance and convergence speed.

Further work is to apply the Gaussian iteration to other models with the similar objective function, not only in the recommender system. Because the missing not at random assumption and the central limit theorem are based on the attribution of the

data. In the experiment, parameter tuning is a thorny problem. Other future work will be to come up with an automatic tuning method.

References

1. James, M., Michael, C., Brad, B., et al: Big Data: The Next Frontier for Innovation, Competition, and Productivity. 2011
2. Ricci, F., Rokach, L., Shapira, B.: Introduction to Recommender Systems Handbook. Springer US. 2011
3. Melville, P., Mooney, R.J., Nagarajan, R.: Content-boosted Collaborative Filtering for Improved Recommendations. AAAI/IAAI. 187--192(2002)
4. Goldberg, D., Nichols, D., Oki, B.M., Terry, D.: Using Collaborative Filtering to Weave an Information Tapestry. Communications of the ACM. 35(12), 61--70 (1992)
5. Herlocker, J.L., Konstan, J.A., Borchers, A., Riedl, J.: An Algorithmic Framework for Performing Collaborative Filtering. In: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 230-237(1999)
6. Linden, G., Smith, B., York, J.: Amazon.com Recommendations: Item-to-item Collaborative Filtering. IEEE Internet Computing.7(1), 76--80(2003)
7. Chang, S., Harper, F.M., Terveen, L.: Using Groups of Items for Preference Elicitation in Recommender Systems. In: Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing, pp. 1258-1269. New York (2015)
8. Pujahari, A., Padmanabhan, V.: A New Grouping Method Based on Social Choice Strategies for Group Recommender System. Computational Intelligence in Data Mining. 1, 325-332(2015)
9. Chen, M.H., Teng, C.H., Chang, P.C.: Applying Artificial Immune Systems to Collaborative Filtering for Movie Recommendation. Advanced Engineering Informatics. 29(4), 830-839(2015)
10. Patra, B.K., Launonen, R., Ollikainen, V., et al.: A new Similarity Measure Using Bhattacharyya Coefficient for Collaborative Filtering in Sparse Data. Knowledge-Based Systems.82, 163-177(2015)
11. Hofmann, T.: Latent Semantic Models for Collaborative Filtering. ACM Transactions on Information Systems. 22(1), 89-115(2004)
12. Salakhutdinov, R., Mnih, A., Hinton, G.: Restricted Boltzmann Machines for Collaborative Filtering. In: Proceedings of the 24th International Conference on Machine Learning, pp. 791-798. ACM(2007)
13. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet Allocation. The Journal of Machine Learning Research. 3, 993-1022(2003)
14. Koren, Y., Bell, R., Volinsky, C.: Matrix Factorization Techniques for Recommender Systems. Computer. 42(8), 30-37(2009)
15. Bell, R.M., Koren, Y.: Lessons from the Netflix Prize Challenge. ACM SIGKDD Explorations Newsletter. 9(2), 75-79(2007)
16. Koren, Y.: Factorization Meets the Neighborhood: a Multifaceted Collaborative Filtering Model. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 426-434. ACM(2008)

17. Steck, H.: Training and Testing of Recommender Systems on Data Missing Not at Random. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 713-722. ACM(2010)
18. McAuley, J., Leskovec, J.: Hidden Factors and Hidden Topics: Understanding Rating Dimensions with Review Text. In: Proceedings of the 7th ACM Conference on Recommender Systems, pp. 165-172. ACM(2013)
19. Devooght, D., Kourtellis, N., Mantrach, A.: Dynamic Matrix Factorization with Priors on Unknown Values. In: Proceedings of the 21th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 189-198. ACM(2015)
20. Movielens Dataset. http://grouplens.org/datasets/movielens/1m/
21. Rice, J.A.: Mathematical Statistics and Data Analysis(Third Ed.). Duxbury Press, pp. 181-188. (2007)