Using Collective Matrix Factorization and Tags to improve a Recommender System

Cristopher Arenas
Universidad Técnica Federico Santa María
Vicuña Mackenna 3939
Santiago, Chile
cristopher.arenas@alumnos.usm.cl

ABSTRACT

In many recommender systems, there are not enough information to realize good predictions, mostly because poor information from data sources are used. Users rate few items in the dataset. This paper presents a improvement to recommender systems using Collective Matrix Factorization and tag information. A existing work will be used and some aditional contributions will be exposed.

Keywords

matrix factorization, tags, gradient descent method

1. INTRODUCTION

Recommender Systems are used to provide recommendation to users about items that they could like. The recommendation problem consist to predict for an user u the preference about an item i, tipically with a range of values r called rating.

Different approaches to resolve the problem have been proposed. There are methods based in Collaborative Filtering (CF) [6] that started with the works about this topic. CF can be based in items [5], users [1] or ratings [4]. This approach find similar entities to the user and propose a list with items of new items for that user. This items could be novelty and interesting for that user.

A common issue that will be considered is the problem of sparity in CF. The information of the datasets used in this cases shows that the most part of the users rate a low quantity of items. A litte quantity of users rate a considerably amount of items and the rest of them rate extremly few items. This produces problems to generate predictions in ratings because there are not enough information available.

Other approaches based in matrix factorizations [3] try to solve the sparsity problem using filling techniques in a matrix that conects users and items. The predicted value will be considered as a product of two vectors and will be represented in a matrix *user-item*.

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

This work will use an approach called Collective Matrix Factorization and will consider extra information derived from tags. A tag is commonly a label asigned by users to anitem that share a feature with others.

The paper is organized as follows. Section 2 present the matrix factorization used in recommender systems. Section 3 shows the method used in this work and aditional considerations used. Section 4 explains the experiments realized and section 5 shows the results obtained. Finally, section 6 shows the conclusions of this work.

2. RELATED WORK

The matrix factorization approaches [3] consider the factorization of a matrix U that relates users an items. A cell value of this matrix represent the rating that an user u assign to an item i. Traditional approaches try to minimize the error function:

$$E = \sum_{(u,i)\in K} (r_{ui} - q_i^T p_u)^2 + \lambda(||q_i||^2 + ||p_u||^2) \tag{1}$$

Where r_{ui} is the rating of the user u for the item i and the vector product $q_i^T p_u$ determine an aproximate rating for the previous pair. λ is used to avoid overfitting.

There are similar aproaches that adds bias terms to the error function, and others consider aditional imput sources.

3. PROPOSED METHOD

The base of this work is presented in [2]. There are three matrices that relates m users, n items and p tags:

- $U(u,i)$: user-item matrix. Shows the *rating* of user u for an item i.
- $T(u,t)$: user-tag matrix. Shows the *preference* of an user u for the tag t.
- $G(i,t)$: tag-item matrix. Shows *relevance* between item i and tag t.

The U matrix is constructed with traditional approaches of matrix factorization. G matrix is constructed in a similar way to U matrix, considering tags in place of items. T matrix is constructed using the other two matrices. The equation (2) shows how to determine the relation between the user u and tag t.

$$T(u,t) = \frac{1}{N}\sum_{k=1}^{n} U(u,k) \times G(k,t) \tag{2}$$

N is a normalized factor that consider the number of k that $U(u,k)$ and $G(k,t)$ are not zero.

The next step is construct two matrices U' and T' using users, items, tags and latent factors.

$$U' = XY^T \tag{3}$$

$$T' = XZ^T \tag{4}$$

Where X, Y and Z are submatrices. X is the relation between users and latent factors, Y is the relation between items and latent factors and Z is the relation between tags and latent factors.

Later, the Gradient Descent Method (GDM) is performed to minimize the error between the real values (U and T) and the aproximated values (U' and T'). The error function to minimize is showed in (5).

$$ER(X,Y,Z) = \frac{1}{2}||J \circ (U - XY^T)||_F^2 + \frac{\alpha}{2}||T - XZ^T||_F^2 + \frac{\beta}{2}\left(||X||_F^2 + ||Y||_F^2 + ||Z||_F^2\right) \tag{5}$$

J is an Indicator Matrix where $J(a,b) = 1$ if an user a has rated the item b and zero in other case. The GDM moves the values of the matrices X, Y and Z in direction of gradients $\nabla_X ER$, $\nabla_Y ER$ and $\nabla_Z ER$.

$$\nabla_X ER = \left[J \circ (XY^T - U)\right] Y + \alpha(XZ^T - T)Z + \beta X \tag{6}$$

$$\nabla_Y ER = \left[J \circ (XY^T - U)\right] X + \beta Y \tag{7}$$

$$\nabla_Z ER = \alpha(XZ^T - T)X + \beta Z \tag{8}$$

The GDM starts with random values in range (0,1) for X, Y and Z. Later, calculates a step γ to move this matrices in direction of the gradient. Finally, a value $U' = XY^T$ is returned and contain values that U matrix does not have. This new values correspond to predicted values to the user u rating the item i. The Algorithm 1 summarizes the GDM.

Algorithm 1 Gradient Descent Method. Adapted from [2]

```
1: Initialize $X$, $Y$, $Z$ with random number in range (0,1)
2: $t = 0$
3: while $t < max\_iteration$ do
4:     Get gradients $\nabla_X ER$, $\nabla_Y ER$ and $\nabla_Z ER$.
5:     $\gamma = 1$
6:     while $(ER(X_t - \gamma\nabla_{X_t}, Y_t - \gamma\nabla_{Y_t}, Z_t - \gamma\nabla_{Z_t}) > ER(X_t, Y_t, Z_t))$ do
7:         $\gamma = \gamma/2$
8:     end while
9:     $X_{t+1} = X_t - \gamma\nabla_{X_t}$
10:    $Y_{t+1} = Y_t - \gamma\nabla_{Y_t}$
11:    $Z_{t+1} = Z_t - \gamma\nabla_{Z_t}$
12:    $t = t + 1$
13: end while
```

4. EXPERIMENTS

The are two cases that [2] took for α and β. First, the baseline consider $\alpha = 0$ and $\beta = 1$. This case correspond to traditional matrix factorization approach, using only users and items. Later, the case $\alpha = 1$ and $\beta = 1$ was consider the best case that use tag information.

The dataset from MovieLens was considered to realize experiments. This dataset is composed by 1,000,209 ratings of 3,682 movies made by 6,040 users. Also, data from Tag-Genome, in the MovieLens site, was used for consider information about tags. Tag-Genome is composed by 9,734 movies and 1,128 tags. The total quantity of movies considered were those who was present in ratings and tags dataset simultaneously. This means that only 3,642 movies were considered to realize the experiments.

There were three experiments realized. First, a comparation about using different number of latent factors was considered. Later, a prediction of ratings were determinated using GDM with the minimization of the error function (5) MAE and RMSE metrics were considered to compare the use of tags and the traditional approach. Finally, a Top-N ranking was generated for users and average of nDCG metric was calculated.

The data was split into training and test. 80% of the data was used as training set and the other 20% was used as test set. The idea of prediction and Top-N ranking was predict the values of the test set using the training set to generate the matrices and execute the GDM.

5. RESULTS

The Figure 1 shows the error function using 1, 3 and 10 latent factors. The GDM was executed for 100 iterations and only the case $\alpha = 1$ and $\beta = 1$ was considered.

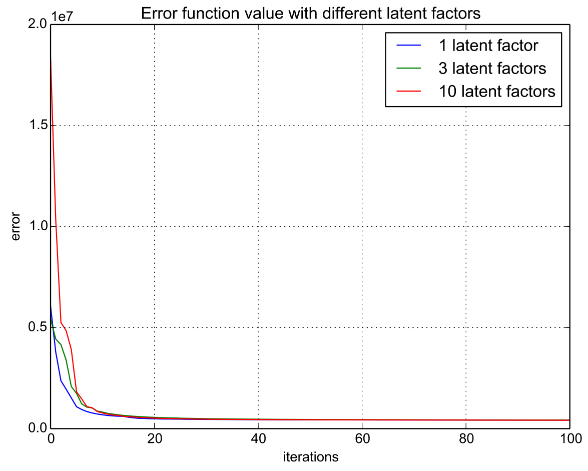

Figure 1: Error function using three different latent factors.

The Figure 5 shows that in the first moments of the algorithm the error value is high and in the next 10 iterations this value decreases faster. Since iteration 40, the error decreases slowly. Using more latent factors cause more error in the begining of the algorithm. There are not a big difference in the value of the error in the last iterations.

The Table 1 shows the results for the metrics MAE and RMSE for the two scenarios (with tags consider $\alpha = 1$ and $\beta = 1$ and without tags consider $\alpha = 0$ and $\beta = 1$). In both cases, GDM was excecuted with 200 iterations.

Table 1: MAE and RMSE considering two scenarios.

Metric	With tags	Without tags
MAE	0.7236	0.6961
$RMSE$	0.9264	0.8945

Table 2: nDCG@p metric for two scenarios and differents values of p

p	Best user		Worst user		Average	
	With tags	Without tags	With tags	Without tags	With tags	Without tags
1	0.7869	0.8354	0.7869	0.8354	0.7869	0.8354
3	0.7855	0.8323	0.6431	0.6581	0.7169	0.7337
5	0.7811	0.8256	0.4256	0.5164	0.6353	0.6740
10	0.9966	0.8666	0.4147	0.5002	0.6888	0.7157
20	0.9676	0.9658	0.3974	0.4682	0.6798	0.7066

The values obtained shows that there are not enough difference between use tags or not. The best values of MAE and RMSE are those who were determinated without use tags.

The Table 2 shows the values obtained for the $nDCG@p$ metric. Different values of p were used. The worst user, best user and average of this metric was calculated. The values shows that mostly of the cases are better when tags are not considered. In some cases ($p = 10$ and $p = 20$), the best user delivered a better value for nDCG when tags were takes into account.

6. CONCLUSIONS

There are different ways to do a matrix factorization to determine a prediction rating for users and items. In this work a method defined in a previous work was realized and aditional metrics and analisys were showed.

The purpouse of this work was show that using tags in a collective matrix factorization could be better than realize a simple matrix factorization. Considering the results obtained, there is not possible to conclude this proposition. The values of the metrics used where very similar considering the use of the tag information and only information of the traditional approach. Probably, this unespected results were obtained because the T matrix constructed was not completly full of values, and this factor impacted in lower performance. When more latent factors were used, initially were obtained worst values in the error function, but in the last iterations there will be not a significant difference between them.

Future work includes realize a more exaustive revision of other metrics and dataset used. The generation of the matrices will be an issue that will be necessary to review.

7. REFERENCES

[1] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 230–237. ACM, 1999.

[2] B. S. Kim, H. Kim, J. Lee, and J.-H. Lee. Improving a recommender system by collective matrix factorization with tag information. In *Soft Computing and Intelligent Systems (SCIS), 2014 Joint 7th International Conference on and Advanced Intelligent Systems (ISIS), 15th International Symposium on*, pages 980–984, Dec 2014.

[3] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37, 2009.

[4] D. Lemire and A. Maclachlan. Slope one predictors for online rating-based collaborative filtering. In *SDM*, volume 5, pages 1–5. SIAM, 2005.

[5] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295. ACM, 2001.

[6] J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen. Collaborative filtering recommender systems. In *The adaptive web*, pages 291–324. Springer, 2007.