Matrix Co-factorization for Recommendation with Rich Side Information and Implicit Feedback

Yi Fang
Department of Computer Science
Purdue University
West Lafayette, IN 47907, USA
fangy@cs.purdue.edu

Luo Si
Department of Computer Science
Purdue University
West Lafayette, IN 47907, USA
lsi@cs.purdue.edu

ABSTRACT

Most recommender systems focus on the areas of leisure activities. As the Web evolves into omnipresent utility, recommender systems penetrate more serious applications such as those in online scientific communities. In this paper, we investigate the task of recommendation in online scientific communities which exhibit two characteristics: 1) there exists very rich information about users and items; 2) The users in the scientific communities tend not to give explicit ratings to the resources, even though they have clear preference in their minds. To address the above two characteristics, we propose matrix factorization techniques to incorporate rich user and item information into recommendation with implicit feedback. Specifically, the user information matrix is decomposed into a shared subspace with the implicit feedback matrix, and so does the item information matrix. In other words, the subspaces between multiple related matrices are jointly learned by sharing information between the matrices. The experiments on the testbed from an online scientific community (i.e., Nanohub) show that the proposed method can effectively improve the recommendation performance.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information Filtering*

General Terms

Algorithms, Performance, Experimentation

Keywords

Matrix co-factorization, Implicit feedback, Rich side information

1. INTRODUCTION

Recommender systems attempt to analyze user preferences over items, and model the relationship between users and items in order to generate meaningful recommendations to the users. Such systems have been ubiquitously adopted in many applications such as e-commerce, social bookmarking, and subscription based services. They provide personalized recommendations which are especially important in markets where the variety of choices is large and the taste of the customer is important. Most recommender systems focus on the areas of leisure activities such as art (e.g., movies and books), fashion (e.g., music and gaming), and food (e.g., restaurants). As the Web evolves into omnipresent utility, recommender systems penetrate more serious applications such as those in online scientific communities.

In this paper, we investigate the task of recommendation in online scientific communities. In particular, our study is based on the Nanohub[1] website hosted by Purdue University. Nanohub is an online scientific community for research, education and collaboration in nanotechnology. It comprises numerous resources with an active user base. These resources include lectures, seminars, tutorials, publications, events and so on. The task is to recommend relevant resources to the users. The scientific communities such as Nanohub exhibit two characteristics: 1) there exists very rich information about resources and users. Most resources contain detailed information such as titles, abstracts and tags. Many registered users also provide detailed profiles about themselves such as research interest, education and affiliation. This information is very indicative for recommendation and thus needs to be taken into consideration. 2) The users in the scientific communities tend not to give explicit ratings to the resources, even though they have clear preference in their minds. There only exists implicit user feedback such as the user clicks on resources. These two characteristics may also be noticeable in many other real-world applications, while they are more prominent in online scientific communities such as Nanohub.

This paper proposes matrix co-factorization techniques to incorporate rich user and resource information into recommendation with implicit feedback. Specifically, the user information matrix is decomposed into a shared subspace with the implicit feedback matrix, and so does the item information matrix. In other words, the subspaces between multiple related matrices are jointly learned by sharing information between the matrices. To reflect the confidence level on the implicit feedback, the binary elements in the implicit feedback matrix are weighted according to the frequency of the feedback (for 1) or the user-resource content similarity (for 0). In sum, our main contribution is to factorize implicit

[1]http://www.nanohub.org

feedback, user, and item content matrices into shared subspaces so that the rich side information can be exploited for recommendation with implicit feedback. The experiments on Nanohub show that the proposed method can effectively improve the recommendation performance.

2. RELATED WORK

In this section, we briefly review the main works in the context. The list of references is not exhaustive due to the page limit. One main class of approaches is Collaborative Filtering (CF) [2] which has been successfully applied to several real world problems, such as Netflix's movie recommendation [5]. CF methods are popular because they do not require domain knowledge, but it suffers from the Cold Start problem, in which few ratings can be obtained when a new item enters to the system. Content-based filtering (CBF) systems make recommendations by analyzing the content of textual information [1], but they do not incorporate the information in preference similarity across individuals. Both content-based recommender systems and CF systems have limitations. Hybrid methods such as [9], combine CF and CBF, hoping to avoid the limitations of either approach and thereby improve recommendation performance. Pilászy and Tikk [8] link CF and CBF by finding a linear transformation that transforms user or item descriptions. Singh et al. [11] exploit relational information for recommendation by solving multiple matrix factorization tasks simultaneously. Gunawardana and Meek [3] describe unified Boltzmann machines, which are probabilistic models that encode content information into collaborative filtering. The vast majority of the literature is focused on explicit user ratings. However, in many real world systems, these explicit ratings are hard to collect and user feedback can be implicitly expressed by user behaviors such as clicks, bookmark, purchase history and even mouse movement.

To address the implicit feedback, Hu et al. [4] proposed to treat the data as indication of positive and negative preference associated with vastly varying confidence levels. Recently, several other methods formulated the problem as One-Class Collaborative Filtering (OCCF) [7, 10]. Li et al [6] attempted to further improve OCCF by assigning the weights to the negative examples based on user information. To the best of our knowledge, there is no prior work on incorporating both user and item information in a principled manner.

3. NOTATION

Throughout this paper we use the following notations:
n: number of users;
m: number of resources (or items);
k: number of latent factors;
l: size of word vocabulary;
R: implicit feedback matrix where $R = (R_{ij})_{n\times m} \in \{0,1\}^{n\times m}$;
W: Weight matrix;
U: User information matrix where $U = (U_{iw})_{n\times l}$;
T: resource information matrix where $T = (U_{jw})_{m\times l}$;
P, Q, X, Y: low-rank matrices; $P = \{P_1, P_2, ..., P_n\}^T$ where the i^{th} row P_i is a k-dimension vector representing a user's preference over latent factors. Similar notations hold for Q_j, X_i and Y_j;
I: denotes the identity matrix of the appropriate size;
λ_1, λ_2: regularization parameters;
$||.||_F$: Frobenius norm of a matrix;

4. THE APPROACH

4.1 One Class Collaborative Filtering

With explicit feedback users tell us both what they like and what they dislike, but with implicit user feedback, there is no negative examples. This setting is referred to as one-class collaborative filtering in [7]. In these problems, the training data usually consist simply of binary data reflecting a user's action or inaction. A naive approach is to treat all missing values as negative examples (i.e., AMAN) and then directly apply matrix factorization techniques. A better method proposed in [7] is to treat all missing values as negative, but with weights controlling their relative contribution to the loss function (i.e., wAMAN). Specifically, weighted low-rank matrix approximation [12] can be adapted as follows:

$$J(P,Q) = \sum_{i=1}^{n}\sum_{j=1}^{m} W_{i,j}(R_{ij} - P_iQ_j)^2$$

In wAMAN, the weights for positive examples are set to 1, i.e., $W_{ij} = 1$ if $R_{ij} = 1$. The weights are lowered on missing examples, because we have less confidence on these unobserved examples. We discuss different weighting schemes in Section 4.3. Different from wAMAN and most other OCCF methods, we vary weights for both positive and negative examples. In the experiments, we use AMAN and wAMAN as two baselines.

4.2 Matrix Co-factorization for Embedding User and Item Information

The above OCCF models do not consider the rich side information that are available in many real-world systems. It is natural to represent the user information as a vector space model. For user i and word w, U_{iw} is the TFIDF weight calculated from user profiles. Similarly, the matrix T_{jw} encodes the item information. Our method is motivated by the assumption that the latent features that determine whether a user likes a given item, and the latent features that determine the content of that item, can be mapped into a shared space in which they are likely to be similar. Thus, we constrain our factorizations to use a common matrix to model the features of each item. In other words, the feedback matrix $R \approx PQ$, and the item content matrix $T \approx YQ$, with the latent feature matrix Q contributing to both matrices. Similarly, the user content matrix can also be approximated by $U \approx PX$ with the coupled factor P. Mathematically, the weighted matrix co-factorization is as follows:

$$\begin{aligned} J(P,Q,X,Y) =& ||W \otimes (R - PQ)||_F^2 + \lambda_1(||U - PX||_F^2 \\ &+ ||T - YQ||_F^2) \end{aligned}$$

where $\otimes$ denotes element-wise product and W is used for weighting implicit feedback. To prevent overfitting, a regularization term can be appended to the objective function J. We then aim to find a solution by minimizing the following loss function:

$$\begin{aligned} J(P,Q,X,Y) = ||W \otimes (R - PQ)||_F^2 + \lambda_1(||U - PX||_F^2 \\ + ||T - YQ||_F^2) + \lambda_2(||P||_F^2 + ||Q||_F^2 + ||X||_F^2 + ||Y||_F^2) \end{aligned} \quad (1)$$

4.3 Weighting Scheme for Implicit Feedback

As discussed above, the weight matrix W is crucial to handle the implicit feedback. In wAMAN, three global weighting schemes are used, i.e., uniform, user oriented, and item oriented [7]. To encode the side information, a more specific way to assign the weight for each negative example is to look at the similarity between the user and the item [6]. The more similar they are, the less weight we should assign to that negative example. This similarity is measured by the content features of user i and item j, i.e., $W_{ij} = 1 - sim(i,j)$ where $sim(i,j)$ is the cosine similarity between U_i and T_j.

For positive examples, there also exists various confidence levels. For example, the large number of clicks of a user on an item indicates the strong interest of the user on the item. We denote the observation of implicit feedback by f_{ij} for user i and item j (e.g., f_{ij} is the observed number of clicks). In general, as f_{ij} grows, we have a stronger indication that the user indeed likes the item. In consequence, a plausible choice for W_{ij} in the case of positive instances is

$$W_{ij} = 1 + \beta f_{ij}$$

where β controls the increase rate of confidence. In the experiment, we set it to be 0.1.

4.4 Parameter Estimation

The low-rank matrices P, Q, X and Y can be solved by weighted Alternative Least Square. In order to solve P, we first fix the other matrices, and take derivatives of J in Eqn. (1) with respect P.

$$\frac{\partial J}{2\partial P} = \big(W \otimes (PQ - R)\big)Q^T + \lambda_1 (PX - U)X^T + \lambda_2 ||P||_F$$

Let the partial derivative $\frac{\partial J}{\partial P_i} = 0$, we get

$$P_i = (R_i \breve{W}_i Q^T + \lambda_1 U_i X^T)\big(Q W_i Q^T + \lambda_1 X X^T + \lambda_2 (\sum_j W_{ij}) I\big)^{-1} \quad (2)$$

where $\breve{W}_i$ is a diagonal matrix with entries of i^{th} row in W on the diagonal.

Similarly, let $\frac{\partial J}{\partial Q_j} = 0$, $\frac{\partial J}{\partial X_j} = 0$ and $\frac{\partial J}{\partial Y_i} = 0$, we get

$$Q_j = (R_j^T \tilde{W}_j P + \lambda_1 T_j^T Y)\big(P^T W_j P + \lambda_1 Y^T Y + \lambda_2 (\sum_i W_{ij}) I\big)^{-1} \quad (3)$$

$$X_j = U_j^T P (P^T P + \lambda_2/\lambda_1 I)^{-1} \quad (4)$$

$$Y_i = T_i Q^T (Q Q^T + \lambda_2/\lambda_1 I)^{-1} \quad (5)$$

where $\tilde{W}_j$ is a diagonal matrix with entries of j^{th} column in W on the diagonal. The update is then repeated until convergence. The computational complexity of the algorithm is $O(Nk^2mn)$ where N is the number of iterations. We refer the algorithm as Matrix Co-factorization for Rich side information and Implicit feedback (MCRI) and summarize it in Table 1.

Table 1: The Matrix Co-factorization for Rich side information and Implicit feedback algorithm

```
Algorithm MCRI
Input: $R$, $U$, $T$, $W$, $k$
Output: $P$, $Q$, $X$, $Y$
1: Initialize $P$, $Q$, $X$, $Y$
2: Initialize $W_{ij}$
3: if $R_{ij} = 0$ then
4: $W_{ij} = 1 - sim(i,j)$
5: else
6: $W_{ij} = 1 + \beta f_{ij}$
7: end if
8: repeat
9: Eqn. (2), Eqn. (3), Eqn. (4) and Eqn. (5)
10: until $P$, $Q$, $X$, $Y$ converge
11: return $P$, $Q$, $X$, $Y$
```

5. EXPERIMENTS

5.1 Experimental Setup

We test our proposed method on the Nanohub dataset. We split the data into three parts: the data from year 2001 to 2008 is used for training, the data in 2009 is a validation set, and the 2010 data is for testing. We remove the users who have no click history in either of the datasets. The training and validation set includes 10,013 users and 4,430 resources, and the test set contains 6,029 users and 3,673 resources.

The baseline models include AMAN and wAMAN described in Section 4.1. These two are collaborative filtering based approaches. One content based (CB) method is also compared: to compute the ranked list of resources based on the cosine similarity between the resource description and the user profile. In addition, a hybrid model (wAMAN+CB) is also adopted as a baseline, which as a linear combination of wAMAN and CB.

We use Mean Percentage Ranking (MPR) [4, 6] to evaluate the prediction accuracy, which is a typical evaluation metric for recommendation with implicit feedback. MPR is recall-oriented because precision based metrics are not very appropriate as they require knowing which resources are undesired to a user. Lower values of MPR are more desirable. The expected value of MPR for random predictions is 50%, and thus MPR $>$ 50% indicates an algorithm no better than random.

5.2 Effect of Number of Latent Factors

We investigate our model with different number of latent factors k, ranging from 10 to 100. Figure 1 shows the results. In general, both wAMAN and MCRI improve in MPR as k increases. When k is small (e.g., $k = 10$), wAMAN performs better than MCRI. When k becomes larger, MCRI gets bigger improvement than wAMAN. These results suggest for MCRI to work with the highest number of factors feasible within computational limitations. Both of them become flattened in MPR when k reaches 100. Thus, we choose $k = 100$ in the other experiments.

5.3 Effect of Side Information

In this subsection, we investigate the effect of user and resource information. Table 2 shows the results of MCRI with incorporating various side information. We can see that MPR gains 2.6% by including user information and gains 5.8% by adding resource information. When incorporating both information, the performance gains 8.1% in MPR. These results show the effectiveness of side informa-

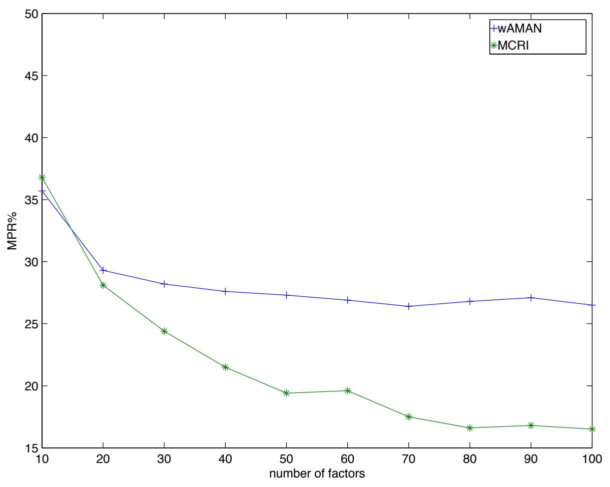

Figure 1: Impact of varying the number of latent factors in wAMAN and MCRI

tion for improving the recommendation performance.

Table 2: Comparison of MCRI with different configurations. MCRI$_0$ incorporates no side information. MCRI$_U$ incorporates only user information. MCRI$_T$ incorporates only resource information. MCRI$_{UT}$ incorporates both user and resource information. MCRI$_0$ is the baseline in Gain.

	MCRI$_0$	MCRI$_U$	MCRI$_T$	MCRI$_{UT}$
MPR	0.245	0.219	0.187	0.164
Gain%	-	2.6	5.8	8.1

5.4 Effect of Weighting Schemes

In this subsection, we study the effect of different weighting schemes in the MCRI model. Table 3 shows the results. We can see that user-oriented (UO) weighting performs the best among the three weighting schemes for the negative examples. By combining UO with PO for the positive instances, the performance is further improved. These results show the effectiveness of weighting on both positive and negative examples.

Table 3: Comparison of different weighting schemes in MCRI. UNI denotes uniform weighting, UO denotes user-oriented weighting, and IO is item-oriented [7]. UNI, UO and IO only weight on negative instances. PO denotes only weighting on positive instances. "Both" denotes the weighting that combines UO and PO. UNI is the baseline in Gain.

	UNI	UO	IO	PO	Both
MPR	0.201	0.187	0.194	0.213	0.164
Gain%	-	1.4	0.7	-1.2	3.7

5.5 Comparison with Other Methods

In this subsection, we compare MCRI$_{UT}$ with other methods. From Table 4, we can see that MCRI$_{UT}$ perform substantially better than the baseline. In addition, we can find that AMAN does not perform well and wAMAN can improve it by weighting the implicit feedback. The pure content based method (CB) performs similarly with wAMAN. By combining these two, the results can get further improvement.

Table 4: Comparison of MCRI$_{UT}$ with other methods. AMAN is the baseline in Gain.

	AMAN	wAMAN	wAMAN+CB
MPR	0.320	0.267	0.227
Gain%	-	5.3	9.3
		CB	MCRI$_{UT}$
MPR		0.268	0.164
Gain%		5.2	15.6

6. CONCLUSION AND FUTURE WORK

This paper presents a principled approach to exploiting rich user and item information with implicit feedback. The experiments are conducted on an online scientific community dataset, which has been rarely investigated in the prior work. The experimental results have shown the proposed model can effectively incorporate the side information and improve the recommendation performance. In the future work, we will conduct more comprehensive experiments on large-scale recommender systems.

7. ACKNOWLEDGMENT

We thank the anonymous reviewers for many valuable comments. This research was partially supported by the NSF research grants IIS-1017837 and EEC-0634750. Any opinions, findings, conclusions, or recommendations expressed in this paper are the authors', and do not necessarily reflect those of the sponsors.

8. REFERENCES

[1] M. Balabanović and Y. Shoham. Fab: content-based, collaborative recommendation. *Communications of the ACM*, 40(3):66–72, 1997.

[2] D. Goldberg, D. Nichols, B. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.

[3] A. Gunawardana and C. Meek. A unified approach to building hybrid recommender systems. In *3rd ACM conference on Recommender systems*, pages 117–124. ACM, 2009.

[4] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *8th IEEE International Conference on Data Mining*, pages 263–272. IEEE, 2008.

[5] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[6] Y. Li, J. Hu, C. Zhai, and Y. Chen. Improving one-class collaborative filtering by incorporating rich user information. In *19th ACM International Conference on Information and Knowledge Management*, pages 959–968. ACM, 2010.

[7] R. Pan, Y. Zhou, B. Cao, N. Liu, R. Lukose, M. Scholz, and Q. Yang. One-class collaborative filtering. In *8th IEEE International Conference on Data Mining*, pages 502–511. IEEE, 2008.

[8] I. Pilászy and D. Tikk. Recommending new movies: even a few ratings are more valuable than metadata. In *3rd ACM conference on Recommender systems*, pages 93–100. ACM, 2009.

[9] A. Popescul, L. Ungar, D. Pennock, and S. Lawrence. Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In *17th Conference in Uncertainty in Artificial Intelligence*, pages 437–444, 2001.

[10] V. Sindhwani, S. Bucak, J. Hu, and A. Mojsilovic. One-class matrix completion with low-density factorizations. In *10th IEEE International Conference on Data Mining*, pages 1055–1060. IEEE, 2010.

[11] A. Singh and G. Gordon. Relational learning via collective matrix factorization. In *14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 650–658. ACM, 2008.

[12] N. Srebro and T. Jaakkola. Weighted low-rank approximations. In *20th International Conference on Machine Learning*, volume 20, page 720, 2003.