
Evaluation of Collaborative Prediction Algorithm

Kensuke Ohta

Department of Computer Science
University of California, Irvine
Irvine, CA 92697
kohta@uci.edu

Abstract

This project investigates the collaborative prediction algorithm focusing on the recently proposed promising method Maximum-Margin Matrix Factorization (MMMMF). Several experiments were conducted with actual datasets, and quantitative results showed a good prediction performance of MMMMF. Also, the significant variance of computational time were observed in the experiments.

1 Introduction

Collaborative prediction is a common technique to predict unknown ratings of *items* for particular *users* based on the past rating data. This can be formulated as a matrix completion task: given a *users* by *items* matrix, whose non-zero entries represent known ratings, predict the rating of any given user.

The estimation of a *fully-observed* target matrix Y by minimizing the sum-squared error with a low-rank matrix X can be efficiently computed via SVD of Y . However, for the estimation of a *partially-observed* matrix, like in a collaborative prediction setting, SVD is no longer applicable. In fact, the problem of estimating a low-rank matrix for a *partially-observed* matrix is a difficult non-convex problem. Also, it is often desirable to minimize other loss functions such as the hinge loss or a loss corresponding to a specific probabilistic model [1].

Low-rank matrix factorization constrains the dimensionality of the factorization $X = UV^T$. Maximum-Margin Matrix Factorization (MMMMF) [2, 3], on the other hand, constrains the norm of U and V , and can be formulated as either Semi-Definite Programming (SDP) or Conjugate Gradients (CG) based problem, and the SDP formulation leads to a convex optimization problem.

This project investigates MMMMF problem by conducting several experiments on the public datasets focusing on the SDP formulation.

2 Maximum-Margin Matrix Factorization

MMMF seeks for a low-norm matrix factorization $X = UV^T$ by minimizing the trace-norm of X . In fact, the trace-norm $\|X\|_\Sigma$ which is given by the sum of singular values of X has been studied as a convex surrogate to the rank in rank optimization problems [4].

2.1 Low-Norm Matrix Factorization

[2, 3] characterize the matrices with factorization $X = UV^T$, where U and V have low Frobenius norm, in several ways as follows.

Lemma 1. *For any matrix X the following are all equal:*

1. $\min_{\substack{U, V \\ X=UV^T}} \|U\|_{Fro} \|V\|_{Fro}$
2. $\min_{\substack{U, V \\ X=UV^T}} \frac{1}{2} (\|U\|_{Fro}^2 + \|V\|_{Fro}^2)$
3. *The sum of the singular values of X , i.e. $\text{tr}\Lambda$ where $X = U\Lambda V^T$ is the singular value decomposition of X .*

Therefore, minimizing a trace-norm with any convex loss is a convex optimization problem. Also, this optimization is same as SVM with hinge loss. Here, the hinge loss is generalized for discrete ordinal ratings as in MovieLens¹ datasets.

2.2 Problem Formulation

Given a set of observed entries $ij \in S$, the problem can be formulated as *soft-margin matrix factorization* using a trade off constant C and the hinge loss $h(z) = \max(0, 1 - z)$. For the generalization of ordinal ratings, thresholds $\theta_1, \dots, \theta_{R-1}$ are introduced as follows.

$$\theta_{Y_{ij}-1} + 1 \leq X_{ij} \leq \theta_{Y_{ij}} - 1$$

where $Y_{ij} \in \{1, \dots, R\}$. The resulting optimization problem is

$$\min \|X\|_\Sigma + C \sum_{ij \in S} \sum_{r=1}^{R-1} h(T_{ij}^r(\theta_r - X_{ij})) \quad (1)$$

$$\text{where } T_{ij}^r = \begin{cases} +1 & \text{for } r \geq Y_{ij} \\ -1 & \text{for } r \leq Y_{ij} \end{cases}.$$

3 Optimization

Lemma 1 allows the trace norm to be bounded by $\frac{1}{2} (\|U\|_{Fro}^2 + \|V\|_{Fro}^2)$, and the optimization problem of learning a low-norm factorization $X = UV^T$ can be formulated as SDP.

Lemma 2. *For any matrix $X \in R^{n \times m}$ and $t \in R$. $\|X\|_\Sigma \leq t$ iff there exists $A \in R^{n \times n}$ and $B \in R^{m \times m}$ such that $\begin{bmatrix} A & X \\ X^T & B \end{bmatrix} \succeq 0$ and $\text{tr}A + \text{tr}B \leq 2t$.*

Lemma 2 can be used to formulate the minimization as SDP. Soft-margin matrix factorization (1) can be written as follows.

¹<http://www.grouplens.org/>

$$\min \frac{1}{2} (\text{tr}UU^T + \text{tr}VV^T) + C \sum_{ij \in S} \xi_{ij} \quad \text{s.t.} \quad \begin{bmatrix} UU^T & X \\ X^T & VV^T \end{bmatrix} \succeq 0, \quad (2)$$

$$T_{ij}^r X_{ij} \geq 1 - \xi_{ij}, \quad \xi_{ij} \geq 0 \quad \forall ij \in S.$$

4 Implementations and Experiments

For the evaluation of MMMF in the SDP formulation, I conducted several experiments in both hard-margin and soft-margin settings, with allowing different thresholds for each user, on two types of subsets of 100K MovieLens dataset. This dataset consists of 100,000 ratings from 1 to 5 for 1682 movies by 943 users, in which approximately 6% of entries are filled. Because generic SDP solvers are known to handle MMMF problems on matrices with up to a few hundred dimensionality, 100 users and 100 movies were chosen for the experiments in two ways. First, one subset was just chosen avoiding there exists all-zero rows or columns. This subset 1 has 1915 ratings (approximately 20% are filled). Subset 2 was selected by choosing 100 users and 100 movies with most ratings as [2] did, in which 7086 (approximately 70%) entries are filled. Test data were chosen by selecting one observed rating at random for each user.

In implementing experiments programs, the available MATLAB code² by Srebro was utilized, and CSDP³ was used for the SDP solver as [2] did. Also, YALMIP⁴ was utilized in bridging the MATLAB code and CSDP.

Experiments were conducted in four cases, i.e. soft-margin setting (with trade off constant $C = 0.24$ as provided in [2] by cross validation) and hard-margin settings, on both subset 1 and 2 with running the program five times for each case. All the experiments were performed on a laptop computer with $1.06\text{GHz} \times 2$ Intel Core2 Duo processor.

5 Results

The followings are the results of four experiments. The time for computation, accuracy of prediction, and the mean absolute error (MAE) were measured. Here, the accuracy is the rate of the correct predictions, and the MAE is computed by dividing the sum of absolute error by the total number of test data.

Table 1: Hard-Margin MMMF on Subset 1

	Time	Accuracy	MAE
1st	4m10s	0.32	0.88
2nd	4m45s	0.35	0.89
3rd	3m54s	0.36	0.82
4th	3m1s	0.36	0.83
5th	3m6s	0.36	0.82
Average	3m47s	0.35	0.848

²<http://people.csail.mit.edu/nati/mmmf>

³<https://projects.coin-or.org/Csdp/>

⁴<http://control.ee.ethz.ch/~joloef/wiki/pmwiki.php>

Table 2: Hard-Margin MMMF on Subset 2

	Time	Accuracy	MAE
1st	2h57m33s	0.45	0.65
2nd	2h43m24s	0.45	0.69
3rd	2h39m26s	0.44	0.64
4th	2h37m3s0	0.45	0.67
5th	3h9m48s	0.42	0.72
Average	2h49m14s	0.442	0.674

Table 3: Soft-Margin MMMF on Subset 1

	Time	Accuracy	MAE
1st	3m22s	0.39	0.86
2nd	3m7s	0.33	0.93
3rd	3m14s	0.34	0.97
4th	3m10s	0.34	0.90
5th	2m56s	0.34	0.88
Average	3m10s	0.348	0.908

Table 4: Soft-Margin MMMF on Subset 2

	Time	Accuracy	MAE
1st	4h10m4s	0.51	0.57
2nd	4h15m7s	0.49	0.63
3rd	5h9m40s	0.47	0.64
4th	7h23m19s	0.51	0.66
5th	7h10m25s	0.51	0.58
Average	5h37m43s	0.498	0.616

6 Discussion

The results of conducted experiments showed that soft-margin MMMF achieved almost same MAEs on the subset 2 as those described in [2], and slightly worse results were obtained in hard-margin MMMF as expected. Less observed entries of the subset 1 made the prediction more difficult, and both the accuracies and MAEs became worse than those on the subset 2. This clearly showed the number of observations greatly affects on the prediction quality. The soft-margin MMMF didn't achieve better predictions on subset 1. This is considered to be because of the trade off constant which was chosen in [2] for the subset 2 by cross validation.

It is worth noting that there was a significant difference in computational time between subset 1 and 2. The computation on the subset 1 took a few minutes and the subset 2 took several hours. Although the number of observed entries of subset 2 (7086) is just about 3.7 times as many as that of subset 1 (1915), the computational time for the subset 2 took about 45 times as much in hard-margin setting and 107 times in soft-margin setting on average. Because subset 2 is an artificially made 70% filled matrix, it is far from the matrices in real problems. Real matrices typically have a huge dimension (e.g. millions by millions). However, this significant difference implies the difficulty of computing MMMF in a distributed way.

In this project, it was confirmed that MMMF achieves a better performance than previous state-of-the-art methods. However, the SDP formulation is limited to the small scale of the datasets, and it was found that computational time greatly depends on the number of observed entries in the dataset. Although there is a problem of local minima, CG based formulation is reported to work well [3,5]. [5] reported an approach with ensembles in prediction. In order to achieve a better prediction quality and less computational time, further investigations will be needed. One approach would be building computationally distributable ensembles. Finding a good formulation of such ensembles needs further researches.

Acknowledgments

I would like to appreciate Nathan Srebro for making his MMMF code publicly available.

References

- [1] T. Hofmann, Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning Journal*, 2001.
- [2] N. Srebro, J. Rennie, T. Jaakola, Maximum-margin matrix factorization. *Advances In Neural Information Processing Systems*, 2004.
- [3] J. Rennie, N. Srebro, Fast Maximum-Margin Factorization for Collaborative Prediction. *International Conference on Machine Learning*, 2005.
- [4] M. Fazel, H. Hindi, S. Boyd, A rank minimization heuristic with application to minimum order system approximation. *American Control Conference*, 2001.
- [5] D. DeCoste, Collaborative Prediction Using Ensembles of Maximum Margin Matrix Factorizations. *International Conference on Machine Learning*, 2006.