Collaborative Filtering and Heavy-Tailed Degree Distributions

Maurizio Calo Caligaris
Department of Computer Science
Stanford University
Stanford, CA 94305
maurizio@cs.stanford.edu

Rafael Moreno Ferrer
Department of Computer Science
Stanford University
Stanford, CA 94305
rmferrer@stanford.edu

ABSTRACT
Common techniques in collaborative filtering rely on finding low-rank matrix approximations to the adjacency matrix (ratings that users assign to items), essentially representing users and items as a collection of a small number of latent features. One issue that arises in many real world datasets for collaborative filtering is that the number of observed entries per row/column follows a heavy-tail distribution. For instance, in the Amazon product ratings dataset, the maximum degree of a product is 12180 whereas the average number of ratings for each product is 4.68. We show that these over-represented rows/columns alter the spectrum of the adjacency matrix significantly, which negatively affects the performance of SVD based methods for low-rank approximations. Further, we present experimental evaluation to show that discarding rows/columns with high degree results in improved performance accross several different datasets (Amazon products ratings, movie ratings and book ratings).

Keywords: Collaborative filtering, recommendation systems, matrix factorization, Netflix prize, SVD, spectral analysis

INTRODUCTION

In this work we focus on the task of rating predictions: the task is to predict a numerical rating that a user u will assign to some item or product i. This is a core problem for many web companies: for instance, e-commerce websites such as Amazon would like to recommend its users products that they may be interested in purchasing; movie rental companies such as Netflix would like to recommend users movies they might be interested in watching.

A common approach for recommender systems is *collaborative filtering*, in which the system recommends to user u items i that users similar to u liked in the past. Many of these collaborative filtering techniques rely on finding low-rank matrix approximations to the adjacency matrix (ratings that users assign to items) – essentially representing users and items as a collection of a small number of latent features. The premise is that there is only a small number of factors influencing the preferences, and that a users preference vector is determined by how each factor applies to that user.

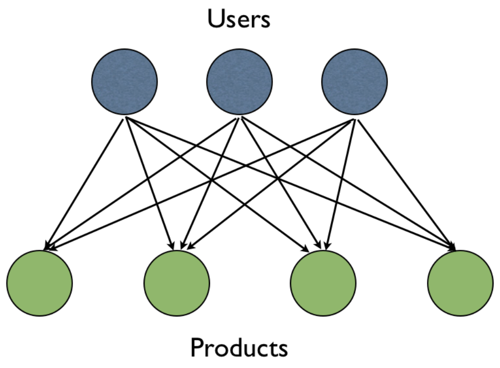


Figure 1: We can model the rating predictions task a bi-partite graph in which users u assign ratings to items i (modeled as edges in the graph), and the goal is to predict the weight of held-out edges based on the rest of the graph.

Most real-world datasets for collaborative filtering tend to have over-represented rows and columns. For example, in the Bookcrossings book ratings dataset, the highest item degree is 8524 whereas the average item degree is only 3.57. This results in a problem for SVD-based low-rank models, since theese over-represented rows/columns alter the spectrum of the adjacency matrix significantly.

We experiment with *trimming* as preprocessing, in which we discard nodes (corresponding to users and items) that have degrees above a certain threshold. This seems counter-intuitive – the common mantra in machine learning is that *more data beats better algorithms* [1] . Even though we are essentially throwing away valuable data, we show this prevents overfitting and thus leads to better rating prediction performance accross a variety of real-world datasets ranging from Amazon product recommendations, MovieLens movie recommendations to book recommendations.

RELATED WORK

The task of rating prediction, in which the goal is to predict a numerical rating that a user u will assign to some item or product i given previous ratings that users have assigned to items, is a core problem for many web service companies

and as such, recommender systems have been studied extensively. There are generally two approaches: *item-based*, which consists in recommending to user u items that are similar to u's past choices; and *collaborative filtering*: recommending to u items i that users similar to u liked in the past. The Netflix challenge [5] widely popularized collaborative filtering. One important component of many successful algorithms competing for the Netflix prize find low-rank matrix approximations to the ratings, which essentially represents users and items as a collection of a small number of latent features. The premise is that there is only a small number of factors influencing the preferences, and that a users preference vector is determined by how each factor applies to that user. This is the key idea behind matrix-factorization [**?**] methods: reconstruct a ratings matrix R as $R = XY^T$, where the columns of X are the latent features for each users preferences, and the columns of Y are latent features for each item. In this work, we adopt an approach similar to the popular maximum margin matrix factorization [6].
Montanari and Oh [3] proved theoretical guarantees for matrix completion based on low-rank optimization. A key assumption of the theorical guarantees rely on a trimming methodology in which degree above a certain threshold are removed. In this work we present experimental evaluation of the trimming methodology proposed by [3]. This seems counterintuitive, as seminal papers in machine learning have proven the *unreasonable effectiveness of data*, i.e. that increasing training set size leads to better performance on several tasks [1]. However, we provide an explanation of why why might want to discard data when degrees of the graph follow a heavy-tailed distribution and perform several experiments on different datasets showing the effectiveness of the heuristic. [2].

PROCEDURE

Problem Statement

We model the data as a bi-partite graph consisting of users $u \in \{1, 2, \dots m\}$ and items $i \in \{1, 2, \dots n\}$, and a set of edges $E \subset \{1, 2 \dots, m\} \times \{1, 2, \dots, n\}$ with associated weights R_{ui} denoting the rating given by user u to item i (which is 0 if user u did not assign a rating to item i).
The data can be thought of having been generated in the following manner

- Denote by M the complete adjacency matrix, of the "true" rating that each user u would theoretically give to each item i had all users truthfully rated all items. We assume M is low-rank.
- The actual rating are noisy: $R = M + Z$ (where Z denotes the noise, indicating that some users might have provided innacurate ratings to some items, either on purpose or accidentally).
- We only observe a subset R^E of the ratings, where missing entries are set to zero.

Our approach

Given an incomplete, noisy ratings adjacency matrix R, we

- Trim matrix R to remove columns and rows with degrees higher than a certain threshold to obtain trimmed matrix $\tilde{R}$.Specifically, we set to zero all edges to items with in-degree larger than $2|E|/n$ (columns in the matrix). We also set to zero all edges coming from users with degrees greater than $2|E|/m$ (rows in the matrix).
- Project $\tilde{R}$ into set of rank-r matrices (for some r obtained experimentally) using SVD: $PR(\tilde{R}) = U_r \Sigma_r V_r^T$.
- Clean residual errors using ALTERNATING LEAST SQUARES method described later, using U_r, Σ_r and V_r for initialization.

We compare performance obtained with and without trimming.

Experimental Evaluation

We use cross-validation to train on a subset $E \subset \{1, 2 \dots, m\} \times \{1, 2, \dots, n\}$ of ratings (which means a fraction $|E|/mn$ entries are revealed), and test on the remaining subset of ratings S. We use root-mean-squared error (RMSE) to measure performance, where RMSE is defined as:
$RMSE = \sqrt{\frac{1}{|S|}\sum_{i,j \in S}(\hat{R}_{ij} - R_{ij})^2}$, where $\hat{R}_{ij}$ is the prediction of our algorithm.

Baseline

We can obtain a reasonable baseline by using mean rating of each item as a predictor, i.e. predict $\hat{R}_{ui}$ to be the mean rating of item i accross all training examples. Similarly, we may also predict mean rating for each user: $\hat{R}_{ui}$ = mean rating for user u. Experiments show that these tend to be reasonably good predictors, so in practice we subtract each entry (rating) of R by the row mean.

Alternating Least Squares

We find low-rank matrices $X \in R^{m \times r}$ and $Y \in R^{n \times r}$ by minimizing the cost function: $\zeta(X, Y) = \sum_{(i,j) \in E}(R_{ij} - (XY^\star)_{ij})^2 + \lambda(\|X\|_F^2 + \|Y\|_F^2)$
$= \sum_{(i,j) \in E}(R_{ij} - < x_i, y_j >)^2 + \lambda(\sum_i \|x_i\|_2^2 + \sum_j \|y_j\|_2^2)$
Where $X^\star = \begin{bmatrix} x_1 \Big| x_2 \Big| \cdots \Big| x_m \end{bmatrix}$ and $Y^* = \begin{bmatrix} y_1 \Big| y_2 \Big| \cdots \Big| y_n \end{bmatrix}$
and R is the matrix containing the train set of ratings (minus item mean) with non-observed entries set to zero.
(Intuitively the x_u's are latent features for each user, and similarly y_i's are latent features for each item). The second term $\lambda(\sum_i \|x_i\|_2^2 + \sum_j \|y_j\|_2^2)$ is a weight-decay regularization penalty to prevent overfitting.
This is a non-convex objective, but it can be viewed as a quadratic program (QP) in X if we fix Y. Thus, we adopt an alternating optimization procedure [4] (ALTERNATE LEAST SQUARES) in which we iteratively optimize with respect to X while keeping Y fixed, then over Y, then again over X and so on.

Heuristic: Trimming

We set to zero all edges to items with in-degree larger than $2|E|/n$ (columns in the matrix). We also set to zero all edges coming from users with degrees greater than $2|E|/m$ (rows in the matrix). This may seem counter-intuitive, since we are essentially throwing out valuable information. However, since the bi-partitite graphs tend to follow power-law degree distributions, this effectively treats over-represented items and users. [3] proves theoretical guarantees that this matrix has spectral properties closer to the underlying matrix M (assuming ratings matrix $R =$ some "true" complete ma-

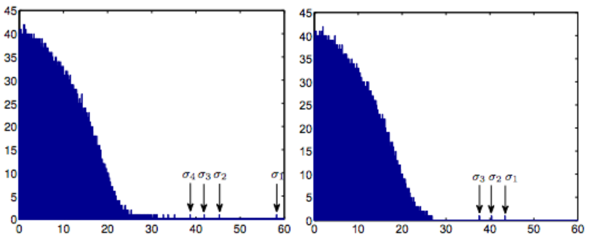


Figure 2: Motivation for trimming: Histogram of the singular values of a partially revealed matrix before trimming(left) and after trimming (right) for a rank 3 matrix M where a fraction $|E|/mn = 0.003$ of the entries are revealed, following a heavy tailed distribution $P(deg = k)\alpha k^{-3})$. After trimming, the underlying rank 3 structure becomes clear. Source:[3]

trix M + noise Z where only a fraction $|E$ of the entries are revealed). Therefore, a rank-R projection of the trimmed matrix is a good approximation of the original entries. Figure [1] shows a simulation demonstrating why trimming might be useful.

Heuristic: Initialization Using SVD

Since the optimization problem is non-convex, we initialize $X = U\Sigma^{1/2}$ and $Y = V\Sigma^{1/2}$, where U,Σ and V were obtained using SVD to find a low-rank projection of the matrix R.

The justification for this is as follows: The objective function we seek to optimize is: $\sum_{(i,j)\in E}(R_{ij} - (XY^\star)_{ij})^2$ (plus a weight decay term to prevent overfitting to the training set). This is a non-convex function that can't be optimized in close form. Instead, we seek low-rank X and Y so as to approximately optimize the quantity: $\tilde{\zeta}(X,Y) = \sum_{(i,j)\in[m]\times[n]}(R_{ij} - (XY^\star))^2$
$= \sum_{(i,j)\in E}(R_{ij} - (XY^\star)_{ij})^2 + \sum_{(i,j)\in \bar{E}}(0 - (XY^\star)_{ij})^2$
with the caveat that we also want $\|X\|_F^2 + \|Y\|_F^2$ to be small. It's worth noting that $\tilde{\zeta}(X,Y)$ is not exactly the objective we're interested in, and it could also be that such initialization results in a bad local minimum for the non-convex optimization problem we care about, but in practice this seems to work decently well nevertheless.

To minimize $\tilde{\zeta}(X,Y)$, we take X and Y so that $XY^* = U_r\Sigma_r V_r^T$, the best rank r approximation of R^E. Since we also want $\|X\|_F^2 + \|Y\|_F^2$ to be small subject to their product being constant, it is convenient to take X and Y so that they have equal Frobenius norm. Indeed, by setting $X = U_r\Sigma_r^{1/2}$ it follows that $\|X\|_F = \left\|\Sigma_r^{1/2}\right\|_F$ since U is orthogonal. Similarly, $\|Y\|_F = \left\|\Sigma_r^{1/2}\right\|_F$. The motivation for X and Y to have equal Frobenius norm is the following:
$\|X\|_F^2 + \|Y\|_F^2 \geq 2\,\|X\|_F^2\,\|Y\|_F^2$
$\geq 2\,\|XY^*\|_F^2$
$= 2\,\left\|U_r\Sigma_r V_r^T\right\|_F^2$ The first inequality is the arithmetic-geometric mean (AMGM) inequality, and equality holds iff the two additive terms are equal,i.e. iff X and Y have equal Frobenius norm.

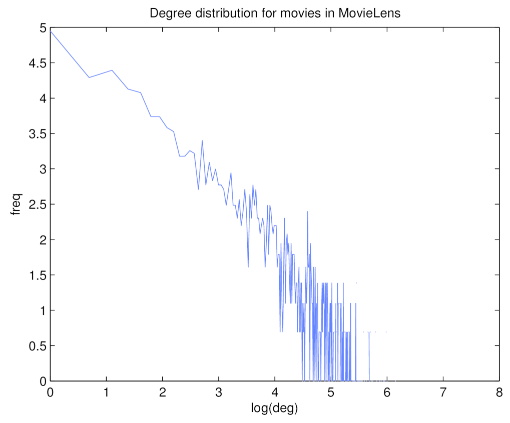
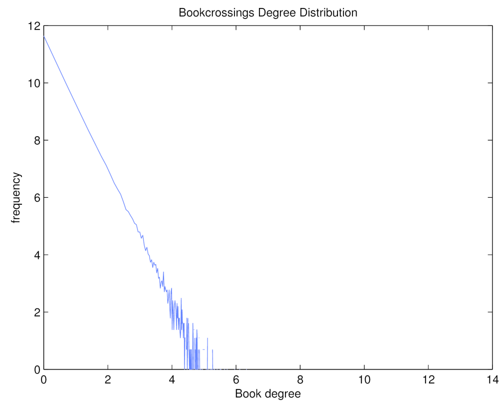
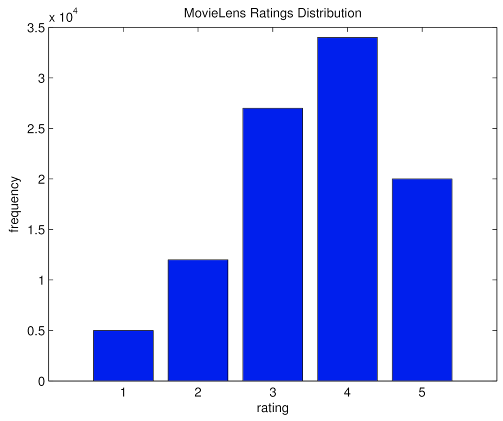



Figure 3: Power-law distribution of the degrees

EXPERIMENTS

We apply the above procedure on several datasets:

- MovieLens: This bipartite network consists of 100,000 usermovie ratings from http://movielens.umn.edu/. Left nodes are users and right nodes are movies. An edge between a user and a movie represents a rating of the movie by the user.
- Amazon: This bipartite network contains product ratings from the Amazon online shopping website. The rating scale ranges from 1 to 5, where 5 denotes the most positive rating. Nodes represent users and products, and edges represent individual ratings. For efficiency, we only use a fraction (0.10) of the rows and columns (selected at random). Data can be obtained here: http://liu.cs.uic.edu/download/data/
- Bookcrossings: This is the bipartite book rating network of the BookCrossing community (book reviews). Rating values are integers between one and ten, where ten represents the best score. An edge shows that a user has rated a book. Nodes in the left column are users; nodes in the right column are books. Data can be obtained here:http://www.informatik.uni-freiburg.de/ cziegler/BX/



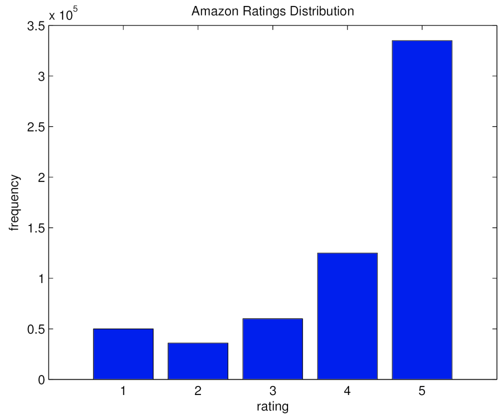


Figure 4: Ratings Distributions

Statistics

	MovieLens	Books	Amazon
m (num users)	943	77802	214605
n (num items)	1682	185955	123091
$\|E\|/mn$	6.3 x 10^{-2}	3 x 10^{-5}	2.21 x 10 $^{-6}$
max item degree	737	8524	1320
avg item degree	76.19	3.57	4.18
σ_1 (no trimming)	512.69	630.81	423.36
σ_1 (after trimming)	110.53	28.28	34.12

We see a heavy-tailed degree distribution different datasets. For example, in the Bookcrossings dataset, the highest item degree is 8524 whereas the average item degree is only 3.57. Furthermore,we observe that the overrepresented rows/columns significantly alter the spectral properties of the ratings matrix: after the trimming operation, the largest singular value σ_1 is shrunk by approximately an order of magnitude.

RESULTS

We use cross-validation: we leave out 20% of the dataset for testing, and train on the remaining 80%.

Baseline

As a baseline, for each item we predict the average rating (out of all the observed ratings) for that item, and similarly we predict the average user-rating for each user. We take whichever of these performs better as a baseline. We observe that the baseline scores are fairly good (For the sake of comparison, the Netflix prize was awarded to anyone who achieves an RMSE of 0.85 on their dataset).

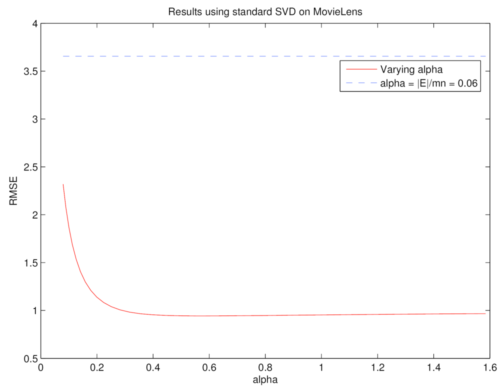


Figure 5: Standard SVD: Project ratings adjacency matrix R (no-trimming) onto rank-r matrices using SVD, $PR(R) = \alpha U_r \Sigma_r V_r^T$. Under the absence of noise, this projection should be weighted by $\alpha = |E|/mn$ but this results in poor performance since the over-represented rows/column alter the spectral properties of the matrix.

Results

We first evaluate performance of the standard SVD: project ratings adjacency matrix R (no-trimming) onto rank-r matrices using SVD, $PR(R) = \alpha U_r \Sigma_r V_r^T$. Under the absence of noise, this projection should be weighted by $\alpha = |E|/mn$ (fraction of revealed entries). This results in poor performance (worse than the baseline) since the over-represented rows/column alter the spectral properties of the matrix. This motivates once again the trimming heuristic.

We performed cross-validation to obtain the optimal value of λ (regularization coefficient in alternating least squares): for small values of λ we observe overfitting (small train error, but large test error) and for large values there is underftting. We show an example of convergence of ALTERNATING LEAST SQUARES for the optimal lambda (with and without trimming), and note that it converges fairly quickly. The numbers shown below correspond to the assymptotic training/testing error for the optimal λ.

	MovieLens	Books	Amazon
baseline	1.00	1.29	1.25
ALS training	0.82	0.82	0.81
ALS testing	0.92	0.89	0.88
ALS+trimming training	0.84	0.83	0.81
ALS+trimming testing	**0.91**	**0.88**	**0.86**

This seems to suggests that the trimming operation essentially prevents overfitting and leads to an improvement in performance (higher training error and lower testing error).

CONCLUSION

In this work, we have examined the effect of heavy-tailed degree distributions in real-world datasets used in collaborative filtering.

We have shown that over-represented rows/columns alter the spectrum of the adjacency matrix significantly, which may

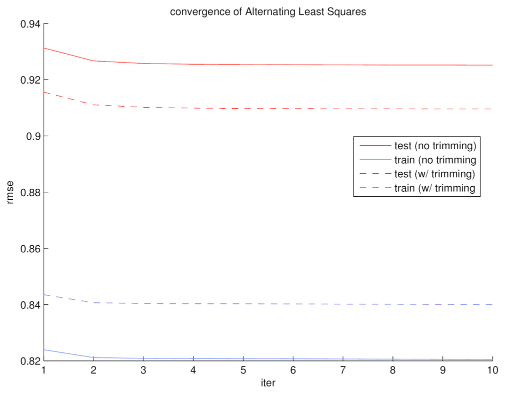


Figure 6: Convergence: Alternating Least Squares procedures seems to converge after just a few iterations (using SVD initialization). Trimming results in higher training error, but lower testing error.

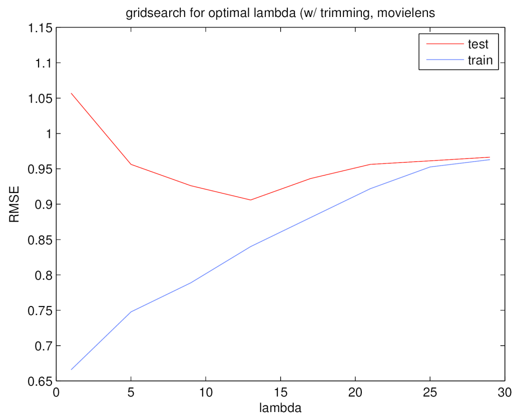


Figure 7: Lambda cross validation (using trimmed data): for small values of λ we observe overfitting (small train error, but large test error) and for large values there is underftting. All the results following correspond to the optimal value of lambda.

hurt performance of SVD-based methods for low-rank approximations.

We experimented with *trimming* as preprocessing, in which we discard nodes (corresponding to users and items) that have degrees above a certain threshold. This seems counter-intuitive – the common mantra in machine learning is that *more data beats better algorithms* [1] . Even though we are essentially throwing away valuable data, we've shown this prevents overfitting (lower testing error, even though training error is higher) and thus leads to better rating prediction performance accross a variety of real-world datasets ranging from Amazon product recommendations, MovieLens movie recommendations to book recommendations.

ACKNOWLEDGMENTS

The authors would like to thanks Prof. Jure Leskovec and the wonderful CS224W staff, as well as the providers of the datasets.

REFERENCES

1. Halevy, Alon, Peter Norvig, and Fernando Pereira. ”The unreasonable effectiveness of data.” Intelligent Systems, IEEE 24.2 (2009): 8-12.
2. Banko, Michele, and Eric Brill. ”Mitigating the paucity-of-data problem: Exploring the effect of training corpus size on classifier performance for natural language processing.” Proceedings of the first international conference on Human language technology research. Association for Computational Linguistics, 2001. APA
3. R.H. Keshavan, A. Montanari and S. Oh, *Matrix Completion from a Few Entries*,2009.
4. S. Boyd, *Sequential Convex Programming*, EE 364B Lecture Slides, 2011.
5. Netix prize, http://www.netflixprize.com/
6. N. Srebro, J. Rennie, and T. Jaakkola, Maximum margin matrix factorization, in Advances in Neural Information Processing Systems 17, 2005
7. Srebro, Nathan, and Tommi Jaakkola. ”Weighted low-rank approximations.” ICML. Vol. 3. 2003.
8. Adomavicius, Gediminas, and Alexander Tuzhilin. ”Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions.” Knowledge and Data Engineering, IEEE Transactions on 17.6 (2005): 734-749. [**?**]Koren, Yehuda, Robert Bell, and Chris Volinsky. ”Matrix factorization techniques for recommender systems.” Computer 42.8 (2009): 30-37.