Improving Top-N Recommendations using Non-negative Matrix Factorization with Divergence

Md. Enamul Haque [1] and SM Zobaed [1]

[1]School of Computing and Informatics
University of Louisiana, Lafayette, LA 70504, USA

December 2, 2017

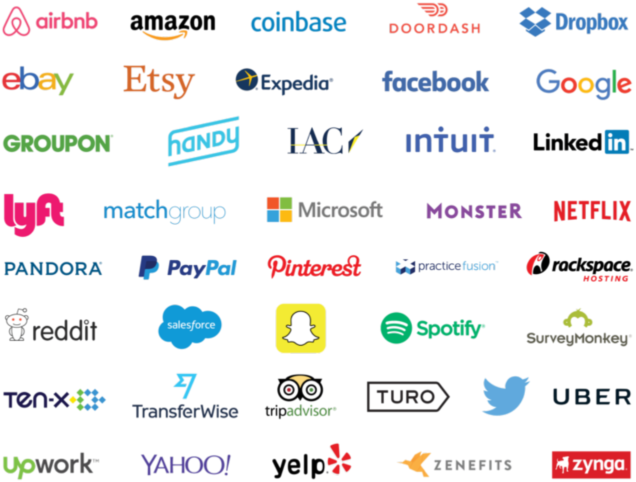
Overview

Yet another recommender system?

In short YES!, but there's more!

Introduction

What is a Recommendation Systems?

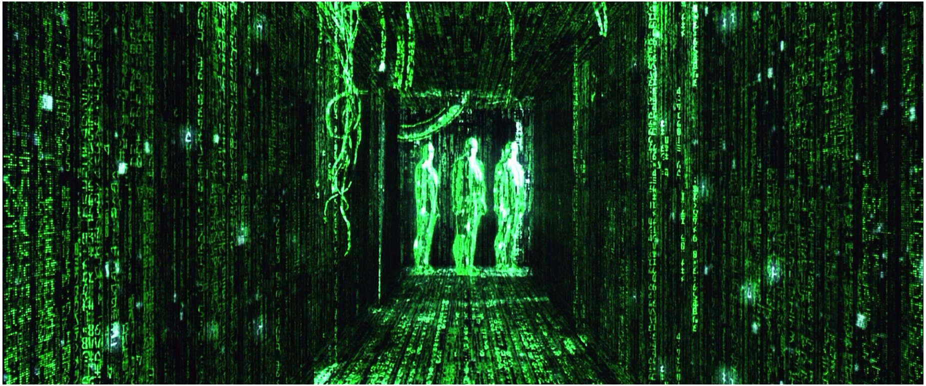


Rating systems

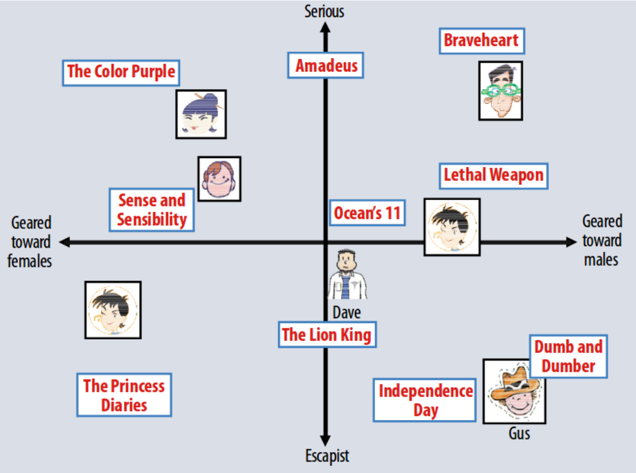
Recommendation Systems

- Content-based systems.
- Collaborative filtering systems.
- Hybrid recommender systems.

Matrix Factorization



A latent space representation



[1]

---

[1]The picture is taken from Y. Koren et al. (2009). *Matrix Factorization Techniques for Recommender Systems.* Computer 42 (8)

Known factorization models (1/5)

Principal Component Analysis(PCA)

- transform data to a new coordinate system.
- variances by any projection of the data lies on coordinates in decreasing order

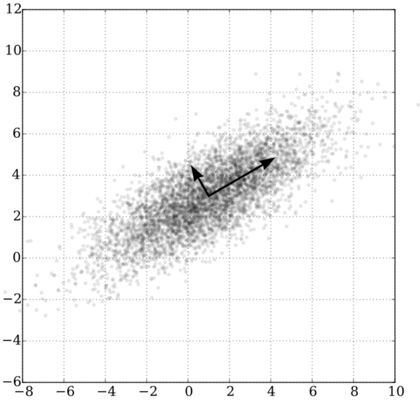


Figure: By Nicoguaro - Own work, CC BY 4.0, https://commons.wikimedia.org/

Known factorization models (2/5)

Singular Value Decomposition(SVD)

$$\Phi = W^{n\times k}\Sigma^{k\times k}H^{n\times k^{\top}}$$

- $W^{\top}W = I,\ H^{\top}H = I.$
- column vectors of W are orthonormal eigenvectors of $\Phi\Phi^{\top}$.
- column vectors of H are orthonormal eigenvectors of $\Phi\Phi^{\top}$.
- Σ contains eigenvalues of W in descending order.

PCA, SVD computed algebraically

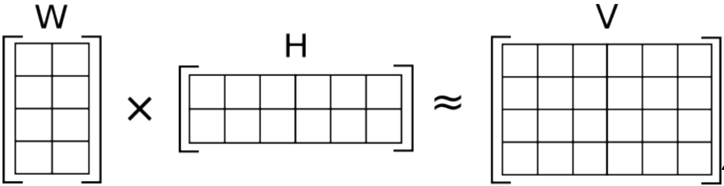
- Φ is a **BIG** and **SPARSE** matrix.
- Approximations of PCA[2] and SVD[3].

[2] T.Raiko et al. (2007). Principal Component Analysis for Sparse High-Dimensional Data.Neural Information Processing, LNCS. 4984

[3] A.K. Menon and Ch. Elkan (2011). Fast Algorithms for Approximating the Singular Value Decomposition. ACM Trans. Knowl. Discov. Data 5 (2).

Known factorization models (3/5)

Matrix Factorization(MF)



[4]

[4] By Qwertyus - Own work, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=29114677

Known factorization models (4/5)

MF - rating prediction

Recommendation task

- to find $\hat{\Phi} : \mathcal{U}, \mathcal{I} \rightarrow \mathbb{R}$ such that the accuracy $acc(\hat{\Phi}, \Phi, \mathcal{T})$ is maximal.
- Training $\hat{\Phi}$ on $\mathcal{D}$ such that empirical loss $err(\hat{\Phi}, \Phi, \mathcal{D})$ is minimal.

A simple, approximative MF model

- k is the number of latent factors.
- User component, $\mathcal{U} = X^{m \times k}$
- Item component, $\mathcal{I} = Y^{n \times k}$
- $\Phi^{m \times n} \approx \hat{\Phi}^{m \times n} = XY^{\top}$
- Predicted rating, $\hat{\Phi}_{ij} = x_i y_j^{\top}$

Known factorization models (5/5)

MF - rating prediction

The **loss** $err(\hat{\Phi}, \Phi, \mathcal{D})$ function

- square loss: $err(\hat{\Phi}, \Phi, \mathcal{D}) = \sum_{(i,j) \in \mathcal{D}} (\Phi_{ij} - \hat{\Phi}_{ij})^2$

the **objective** function

- **Regularization** term $\lambda \geq 0$ to prevent overfitting.
- Penalize the magnitude of the parameters.
- $F(\hat{\Phi}, \Phi, \mathcal{D}) = \sum_{(i,j) \in \mathcal{D}} (\Phi_{ij} - x_i y_j^\top)^2 + \lambda(||X||^2 + ||Y||^2)$

The **task** is to to find parameters X and Y such that, given λ, the objective function $F(\hat{\Phi}, \Phi, \mathcal{D})$ is minimal.

Gradient Descent

How to find a minimum of an "objective" function $F(\Theta)$?

- In case of MF, $\Theta = X \cup Y$, and
- $F(\Theta)$ refers to the error of approximation of Φ by $XY^\top$

Gradient Descent

Input: F, α, Σ^2, *stopping criteria*

initialize $\Theta \sim \mathcal{N}(0, \Sigma^2)$

repeat

$\Theta \leftarrow \Theta - \alpha \frac{\delta F}{\delta \Theta}(\Theta)$

unitl approximate minimum is reached

return Θ

Stopping criteria

- $\Theta^{old} - \Theta \leq \epsilon$
- maximum number of iterations reached
- a combination of both

Stochastic Gradient Descent

When

$$F(\Theta) = \sum_{i=1}^{n} F(\Theta)$$

Stochastic Gradient Descent

Input: $F_i, \alpha, \Sigma^2, \textit{stopping criteria}$
initialize $\Theta \sim \mathcal{N}(0, \Sigma^2)$
repeat
  for all i in random order **do**
    $\Theta \leftarrow \Theta - \alpha \frac{\delta F_i}{\delta \Theta}(\Theta)$
  end for
unitl approximate minimum is reached
return Θ

MF with Stochastic Gradient Descent

Updating parameters iteratively for each data point Φ_{ij} in the opposite direction of the gradient of the objective function at the given point until a convergence criterion is fulfilled.

MF with Stochastic Gradient Descent - Example

Lets have the following hyper-parameters:
$K = 3, \alpha = 0.0002, \beta = 0.02, iter = 5000$

$\Phi =$

5	3	0	1
4	0	0	1
1	1	0	5
1	0	0	4
0	1	5	4

MF with Stochastic Gradient Descent - Example

Results are:

$X =$

-0.04934113	1.34410185	1.77343084
0.03978801	1.18810803	1.3230008
2.01185337	0.51518384	0.49810045
1.58449972	0.64733736	0.27476875
1.54911873	0.41921957	1.0073149

$Y^\top =$

1.80019693	1.18141528	0.38356536	-0.2664426
-0.39075386	0.32691031	2.12877112	1.65417156
1.29524055	0.42334401	1.59878863	1.42570338

MF with Stochastic Gradient Descent - Example

$$\Phi = \begin{array}{|c|c|c|c|}\hline 5 & 3 & 0 & 1 \\ \hline 4 & 0 & 0 & 1 \\ \hline 1 & 1 & 0 & 5 \\ \hline 1 & 0 & 0 & 4 \\ \hline 0 & 1 & 5 & 4 \\ \hline \end{array}$$

$$\hat{\Phi} = \begin{array}{|c|c|c|c|}\hline 4.99678094 & 2.93347608 & 4.44734609 & 0.99887879 \\ \hline 3.96372815 & 2.38151041 & 3.67910781 & 0.99783584 \\ \hline 1.0673425 & 0.84007158 & 5.02504173 & 4.96497519 \\ \hline 0.96686521 & 0.8206766 & 4.0581564 & 3.97526862 \\ \hline 1.95325792 & 1.19655369 & 4.91835471 & 4.02525881 \\ \hline \end{array}$$

Our Proposed Approach

What Matrix Factorization Assumes?

Matrix factorization assumes that:

- Each user can be described by k attributes or features. For example, feature 1 might be a number that says how much each user likes sci-fi movies.
- Each item (movie) can be described by an analogous set of k attributes or features. To correspond to the above example, feature 1 for the movie might be a number that says how close the movie is to pure sci-fi.
- If we multiply each feature of the user by the corresponding feature of the movie and add everything together, this will be a good approximation for the rating the user would give that movie.

Nonnegative Matrix Factorization

A typical NMF solves the following optimization problem:

$$
\begin{aligned}
\min_{X\in\mathbb{R}^{m\times r}, Y\in\mathbb{R}^{n\times r}} \quad & f(X,Y) \simeq \frac{1}{2}\|R - XY^\top\|_F^2 \\
\text{s.t.} \quad & X \geq 0, Y \geq 0
\end{aligned}
$$

Top-N Nonnegative Matrix Factorization (TNMF)

$$\arg\min_{X,Y} \sum_{(i,j)\in\Upsilon} \mathcal{L}_{rank}\left(R_{ij} \log \frac{R_{ij}}{X^\top Y_{ij}} - R_{ij} + X^\top Y_{ij} \right) + \frac{\beta}{2}\left(\|X\|_F^2 + \|Y\|_F^2 \right) \quad (1)$$

Optimization Algorithm

Optimization Algorithm

- Alternating least square
- Stochastic gradient descent

Top-N Nonnegative Matrix Factorization (TNMF)

$$X_u{}^{new} \leftarrow X_u{}^{old} - \eta\left(\sum_{i=1}^{n} y_{ui}\frac{R_{ui}}{(X^\top Y)_{ui}} + \sum_{i=1}^{n} Y_{ui} - \beta X_u^\top\right) \tag{2}$$

$$Y_v{}^{new} \leftarrow Y_v{}^{old} - \eta\left(\sum_{i=1}^{m} x_{vi}\frac{R_{vi}}{(X^\top Y)_{vi}} + \sum_{i=1}^{m} X_{vi} - \beta Y_v^\top\right) \tag{3}$$

Projected Stochastic Gradient descent

Input: $\eta > 0, \beta > 0$, initialize $X_{k \times m} = \mathbf{0}$ and $Y_{k \times n} = \mathbf{0}$
Set: latent factors array, k
repeat

1. Randomly select i, j
2. Update X_u by using (2)
3. Project the updated X_u onto the feasible set:
   $X_u = \max(0, X_u)$
4. Update Y_v by using (3)
5. Project the updated Y_v onto the feasible set:
   $Y_v = \max(0, Y_v)$

until *converge*;
return X, Y

Algorithm 1: Projected Stochastic Gradient Descent

Why not SVD?

What is the issue with Singular Value Decomposition?

- Usually user-product matrix is very large. (m, n are large numbers)
- SVD computation is too expensive.
- Complexity: $\mathcal{O}(mn \min(m, n))$

Experimental Evaluations

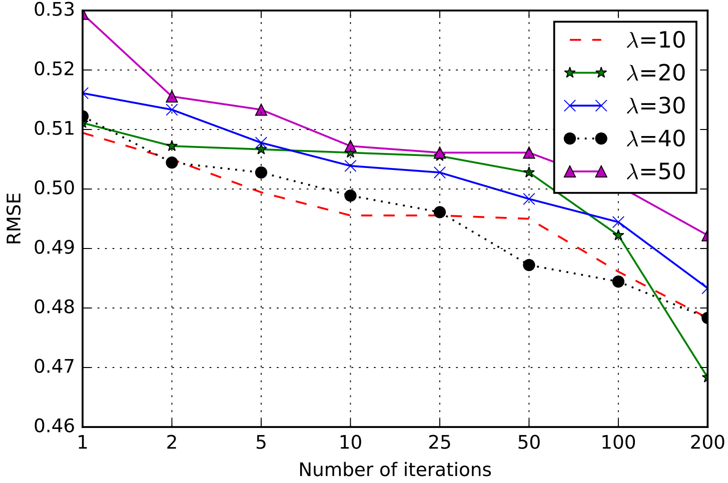
Dataset Statistics

Table: Dataset statistics

Sl.	Dataset	User	Item	Sparsity	Transact
1	*Yahoo!* music	2689	994	3.95%	6738
2	*Yahoo!* movie	2309	2380	0.18%	10136
3	ml-100k	943	1682	6.30%	100000
4	Netflix	83539	22	5.44%	100000
5	Book crossing	2582	24009	0.12%	12102
6	Jester	3000	100	71.01%	213037

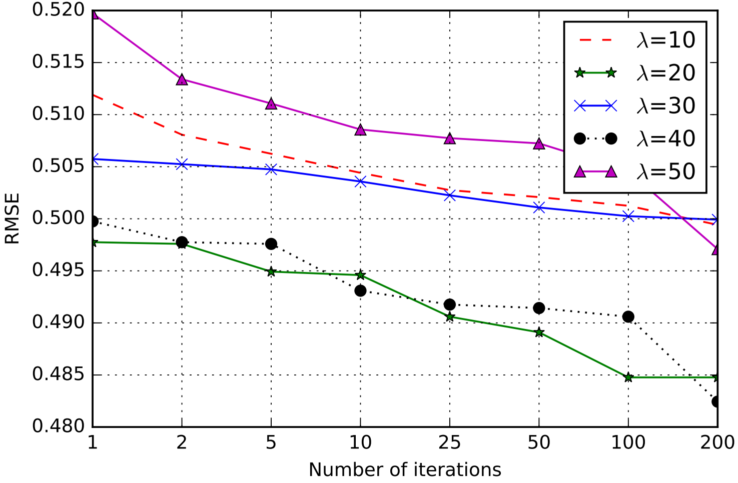
Experimental Evaluation of TNMF (contd.)

Yahoo music dataset



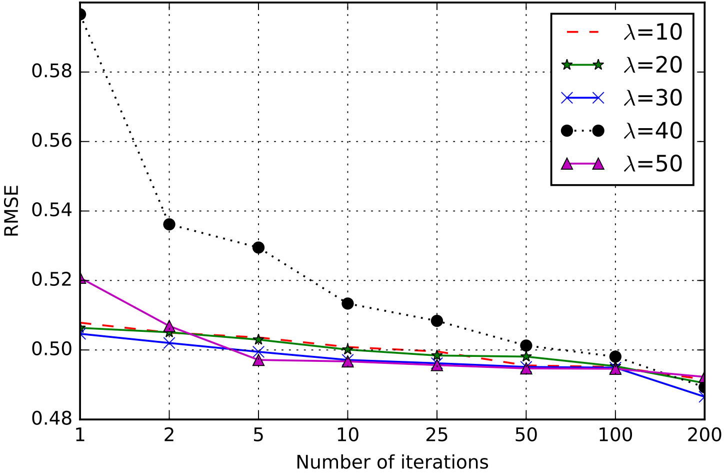
Experimental Evaluation of TNMF (contd.)

Yahoo movies dataset



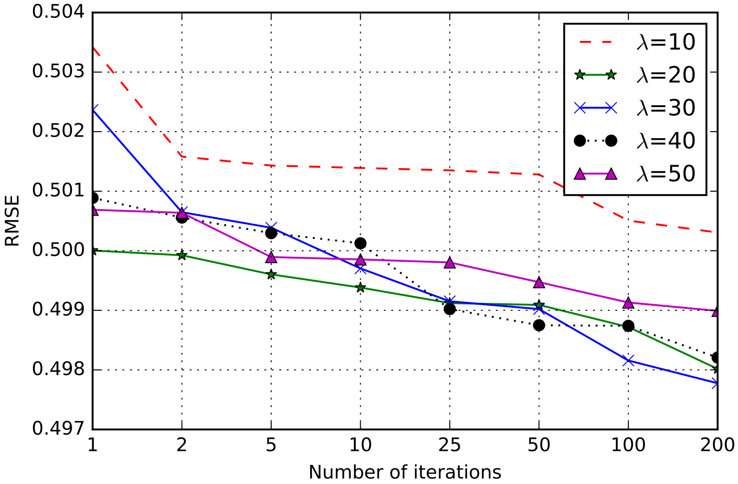
Experimental Evaluation of TNMF (contd.)

Movie lens dataset



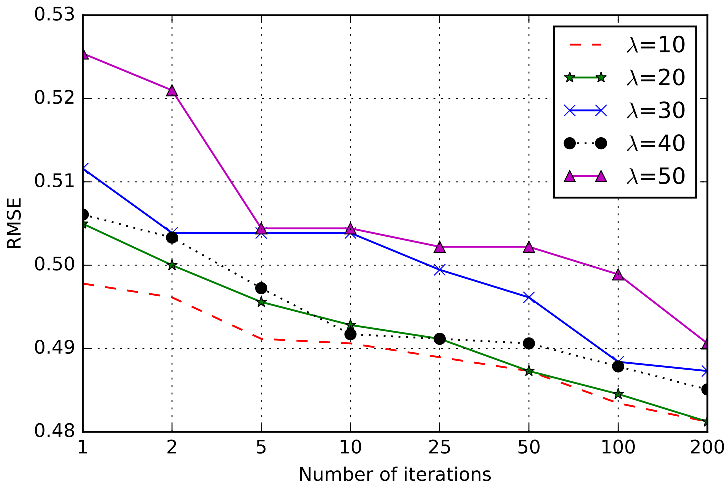
Experimental Evaluation of TNMF (contd.)

Netflix dataset



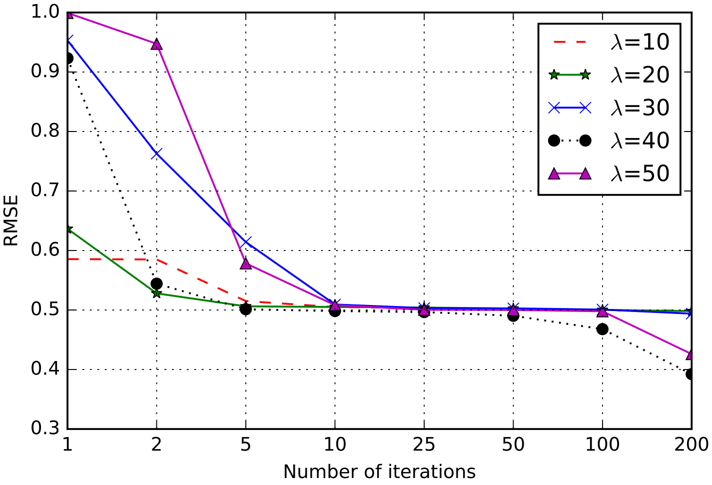
Experimental Evaluation of TNMF (contd.)

Book crossing dataset



Experimental Evaluation of TNMF (contd.)

Jester dataset



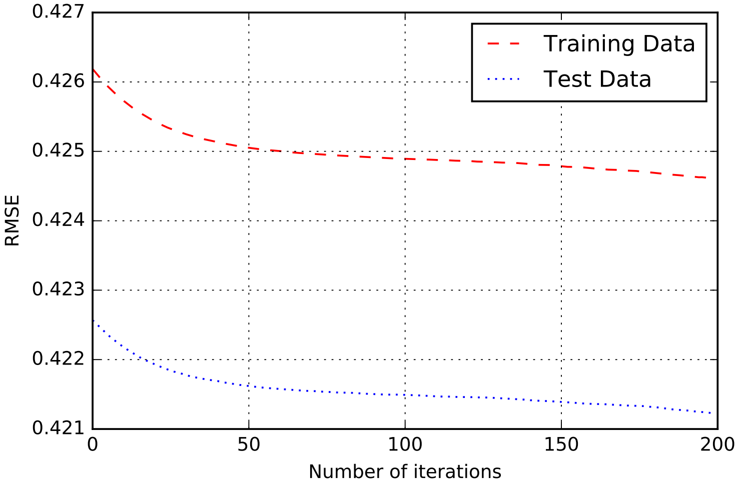
Experimental Evaluations of PMF

Probabilistic Matrix Factorization

- Performs well on the large, sparse dataset.
- Extend PMF model to make interactive model capacity.
- According to their claim, designed method is nearly 7% better than Netflix.
- Use logistic function to bound the prediction range unlike simple linear Gaussian model.

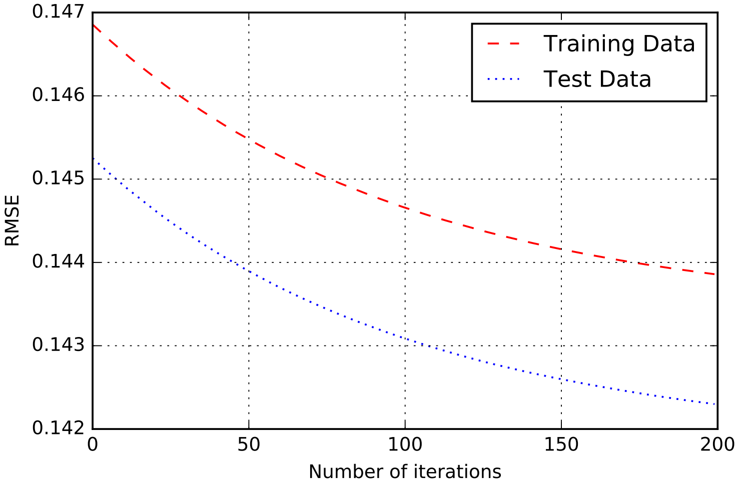
Experimental Evaluation PMF (contd.)

Yahoo music dataset



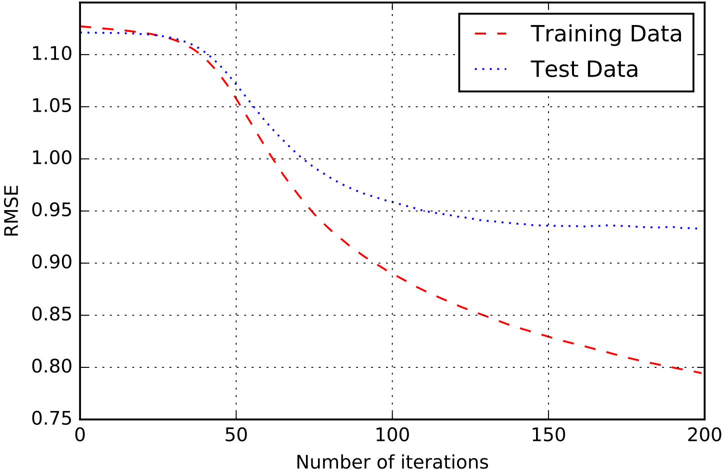
Experimental Evaluation PMF (contd.)

Yahoo movie dataset



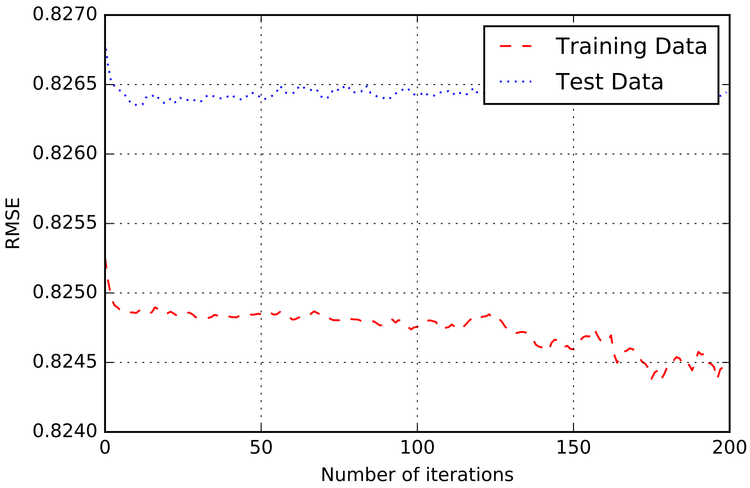
Experimental Evaluation PMF (contd.)

Movie lens dataset



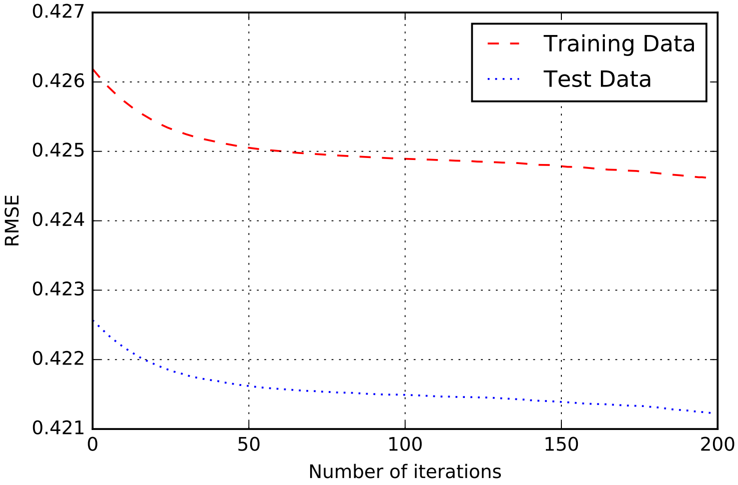
Experimental Evaluation PMF (contd.)

Netflix dataset



Experimental Evaluation PMF (contd.)

Yahoo music dataset



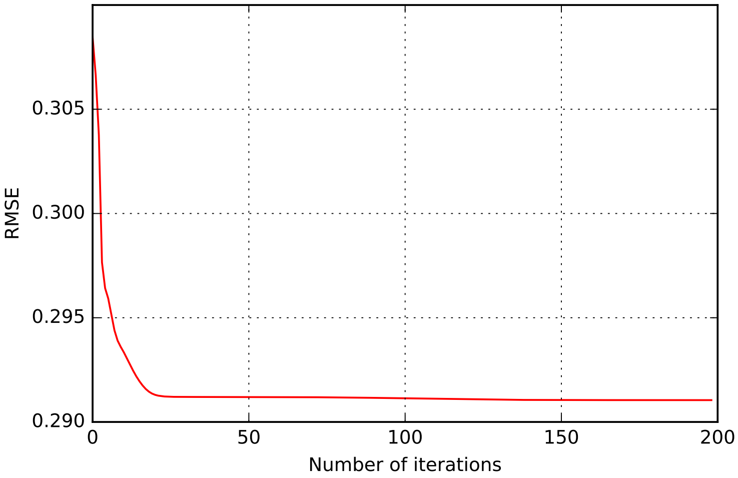
Experimental Evaluations of ALS-WR

Large-scale Parallel Collaborative Filtering for the Netflix Prize

- Start work with low-rank approximation of the user-item matrix.
- Problem: R contains many zeros or unknown ratings.
- Algorithm
  - Initialize item matrix M.
  - Fix M, solve user matrix U by minimizing objective function e.g., Loss function (L^2)
  - Fix U, solve M by minimizing similar function
  - Repeat steps until stopping criterion is satisfied
- Start work with low-rank approximation of the user-item matrix.
- Performance of algorithm increases with number of features and number of ALS iterations

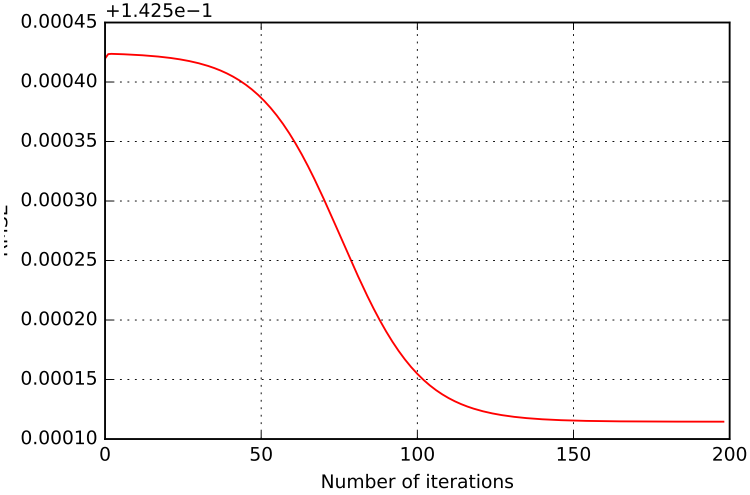
Experimental Evaluation of ALS-WR (contd.)

Yahoo music dataset



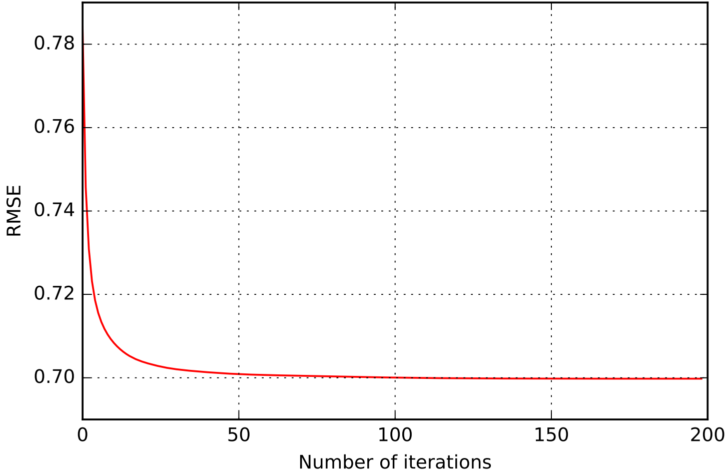
Experimental Evaluation of ALS-WR (contd.)

Yahoo movies dataset



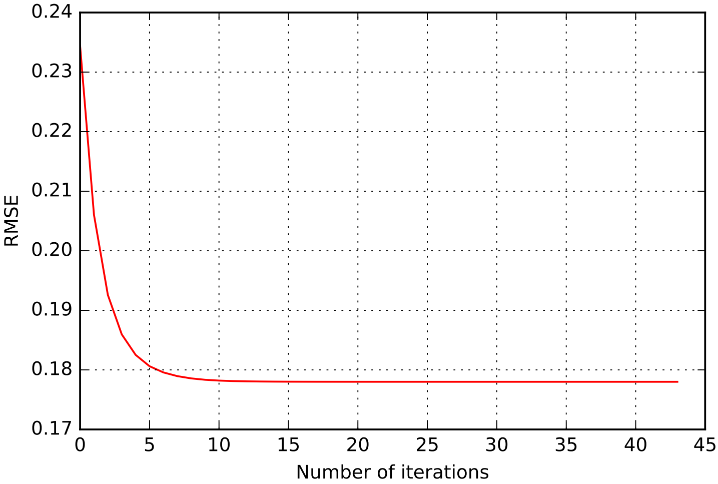
Experimental Evaluation of ALS-WR (contd.)

Movie lens dataset



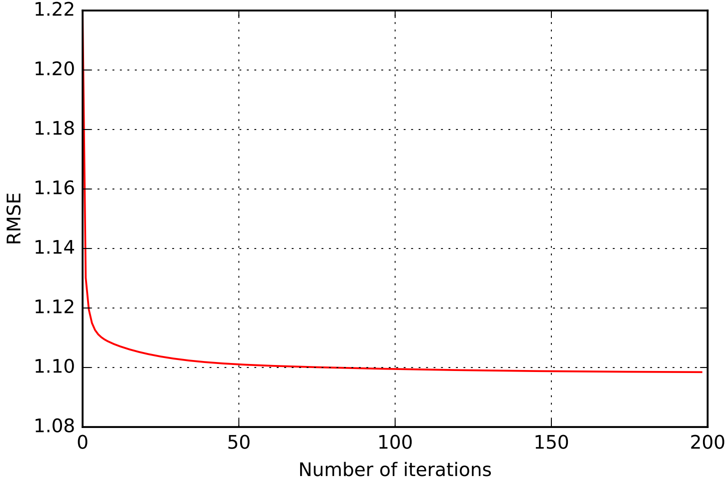
Experimental Evaluation of ALS-WR (contd.)

Netflix dataset



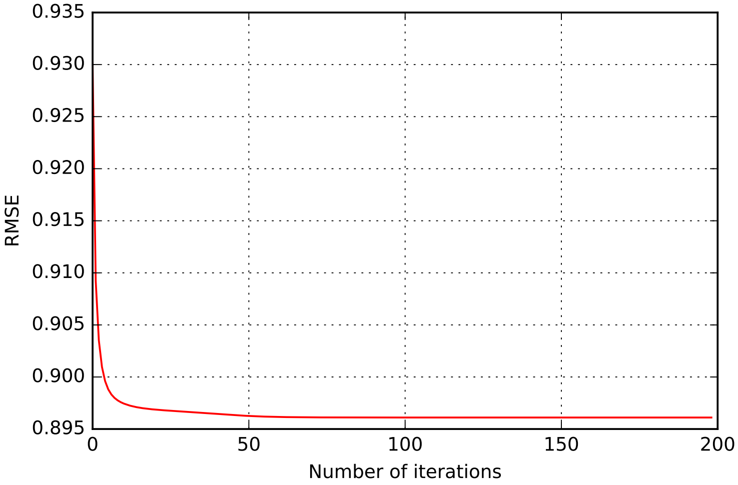
Experimental Evaluation of ALS-WR (contd.)

Book crossing dataset



Experimental Evaluation of ALS-WR (contd.)

Jester dataset



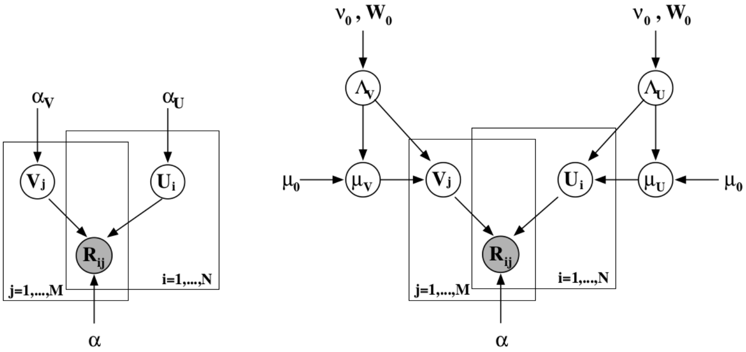
Experimental Evaluations of BPMF

Bayesian Probabilistic Matrix Factorization using Markov Chain Monte Carlo

- Model capacity is fitted automatically by integrating all parameters
- Efficiency improvement: By integrating Markov chain Monte Carlo Methods.
  - Result: higher prediction accuracy.

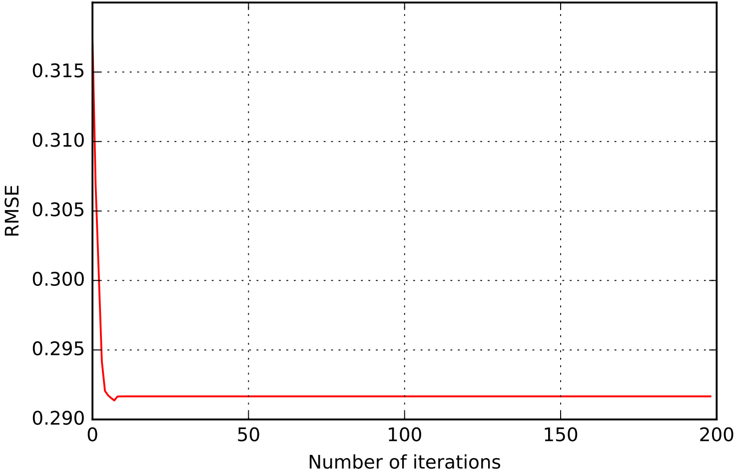
  Challenges: Hard to determine when Markov chain is converged

Bayesian Probabilistic Matrix Factorization using Markov Chain Monte Carlo



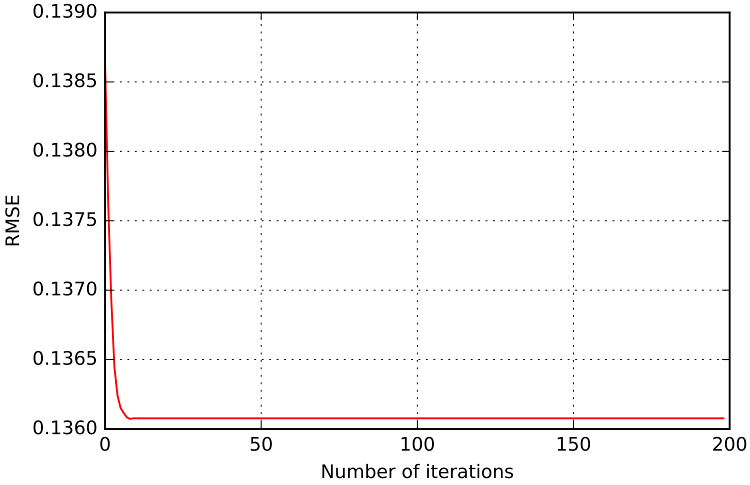
Experimental Evaluation of BPMF (contd.)

Yahoo music dataset



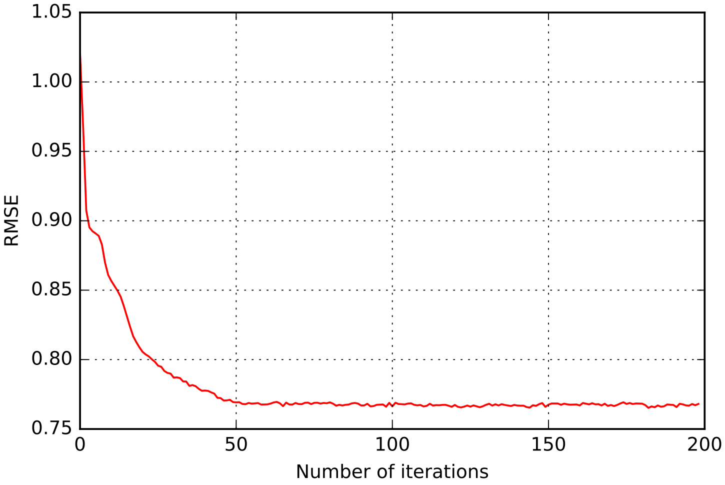
Experimental Evaluation of BPMF (contd.)

Yahoo movies dataset



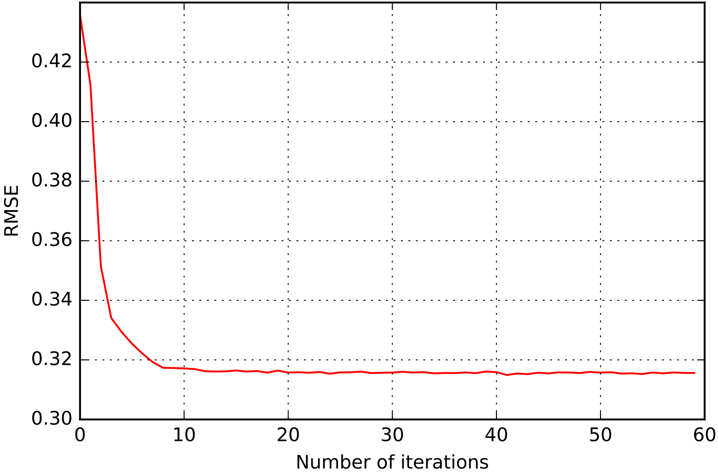
Experimental Evaluation of BPMF (contd.)

Movie lens dataset



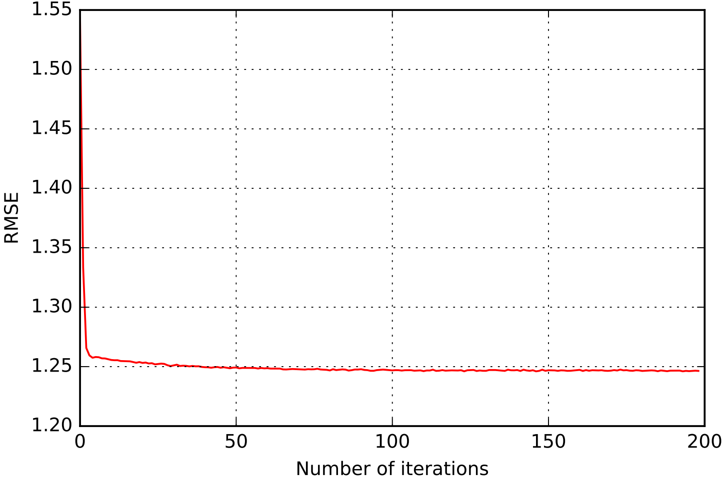
Experimental Evaluation of BPMF (contd.)

Netflix dataset



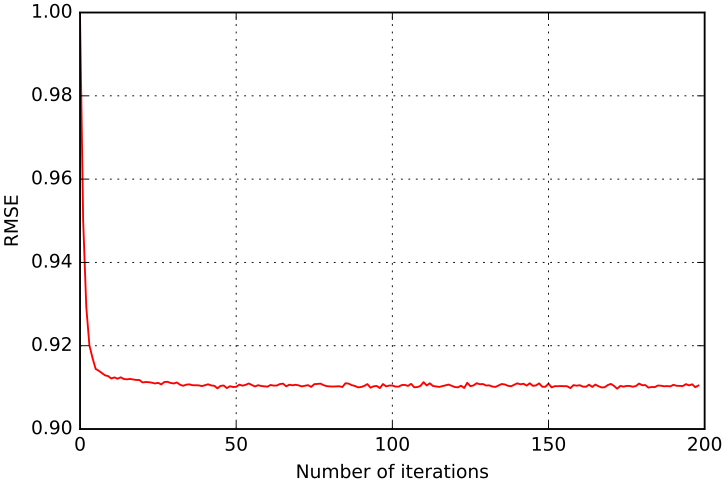
Experimental Evaluation of BPMF (contd.)

Book crossing dataset



Experimental Evaluation of BPMF (contd.)

Jester dataset



Comparison Results

Results and Discussion (contd)

Table: RMSE with standard deviation for different datasets and methods.

Methods	ml-100k	yahoo! music	yahoo! movies	Netflix	Jester	Book X
TNMF	**0.522±0.032**	0.496±0.011	0.493±0.005	0.500±0.001	**0.539±0.151**	**0.494±0.007**
PMF	1.013±0.071	0.424±0.001	0.142±0.001	0.825±0.010	1.270±0.223	1.098±0.001
ALS	0.703±0.022	**0.292±0.003**	0.143±0.001	**0.186±0.034**	0.898±0.021	0.898±0.021
BPMF	0.782±0.040	0.299±0.011	**0.137±0.002**	0.323±0.025	0.913±0.023	0.913±0.023

Future Improvement

- Add other evaluation measures, e.g., hit rate and precision.
- Use pairwise ranking model.
- Use grid search to optimize the latent factor parameter.

Many thanks to

- all the image sources (funny images, graphs, . . .) ...and last
- **YOU for your attention!**

Questions?

https://github.com/enamul-haque/TNMF

Contributions

Task(s)	Performed by
Analyzing recommendation problem	Enamul, Zobaed
Background study	Enamul
Problem formulation	Enamul
System model	Enamul, Zobaed
Implement toy TNMF	Zobaed
Data preprocessing	Zobaed
Implement TNMF	Enamul(70%), Zobaed(30%)
Adapt ALS-WR and BPMF	Zobaed
Adapt PMF	Enamul
Evaluation	Enamul(60%), Zobaed(40%)
Presentation preparation	Enamul(60%), Zobaed(40%)
Report preparation	ongoing..