Kernelized Probabilistic Matrix Factorization: Exploiting Graphs and Side Information

T. Zhou, H. Shan, A. Banerjee, G. Sapiro

presented by Jorge Silva

Problem definition

- Matrix factorization
  - Given a matrix R of size $N \times M$
  - Find matrices U, V such that $R \approx UV^T$
  - $U \in \mathbb{R}^{N \times D}$ and $U \in \mathbb{R}^{M \times D}$
- Main ideas of the paper
  - Take advantage of existing side information, such as graphs
  - Define a Gaussian Process prior over *all* rows of U (same for columns of V^T)
  - The GP covariances K_U and K_V are given by *a priori* chosen kernels

Related approaches

- Probabilistic Matrix Factorization (PMF)[1] defines a zero-mean spherical Gaussian prior for each row of U and each column of V^T
- Bayesian Probabilistic Matrix Factorization (BPMF)[2] defines a full hierarchical prior for each row of U and each column of V^T (means and covariances have Gaussian-Wishart hyperpriors)

[1] R. Salakhutdinov and A. Mnih, Probabilistic matrix factorization, in NIPS 2007.

[2] R. Salakhutdinov and A. Mnih, Bayesian Probabilistic Matrix Factorization using Markov Chain Monte Carlo, in ICML 2008.

Notation

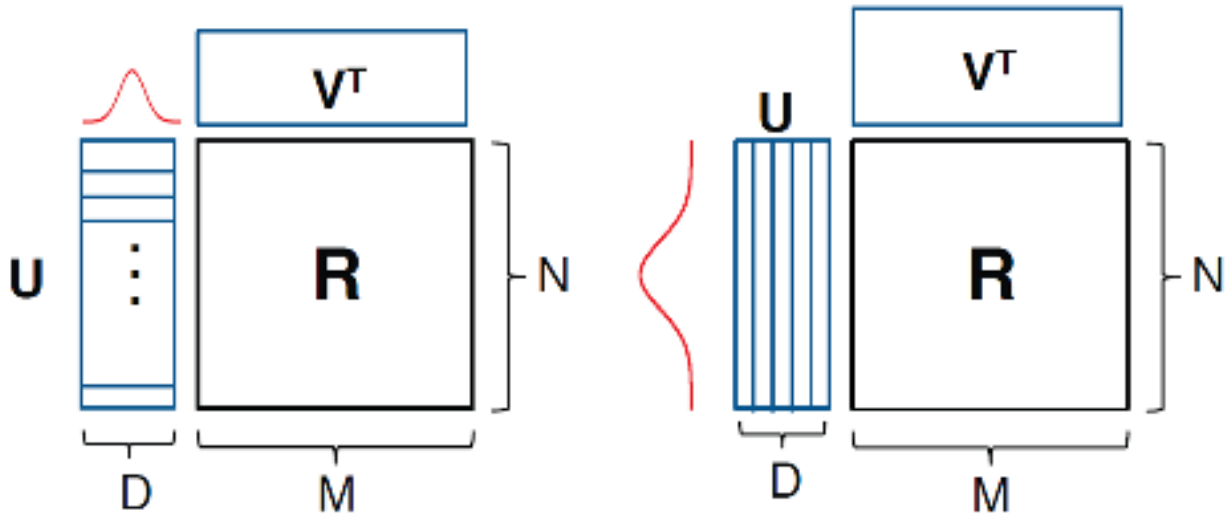
R – $N \times M$ data matrix.
$R_{n,:}$ – n^{th} row of R.
$R_{:,m}$ – m^{th} column of R.
N – Number of rows in R.
M – Number of columns in R.
D – Dimension of the latent factors.
U – $N \times D$ latent matrix for rows of R.
V – $M \times D$ latent matrix for columns of R.
$U_{n,:} \in \mathbb{R}^D$ – Latent factors for $R_{n,:}$.
$V_{m,:} \in \mathbb{R}^D$ – Latent factors for $R_{:,m}$.
$U_{:,d} \in \mathbb{R}^N$ – d^{th} latent factor for all rows of R.
$V_{:,d} \in \mathbb{R}^M$ – d^{th} latent factor for all columns of R.
$K_U \in \mathbb{R}^{N \times N}$ – Covariance matrix for rows.
$K_V \in \mathbb{R}^{M \times M}$ – Covariance matrix for columns.
$S_U \in \mathbb{R}^{N \times N}$ – Inverse of K_U.
$S_V \in \mathbb{R}^{M \times M}$ – Inverse of K_V.
$[n]_1^N$ – $n = \{1, 2, \ldots, N\}$.

KPMF generative process

1. Generate $U_{:,d} \sim GP(\mathbf{0}, K_U)$, $[d]_1^D$.

2. Generate $V_{:,d} \sim GP(\mathbf{0}, K_V)$, $[d]_1^D$.

3. For each non-missing entry $R_{n,m}$, generate $R_{n,m} \sim \mathcal{N}(U_{n,:}V_{m,:}^T, \sigma^2)$, where σ is a constant.



(a) PMF and BPMF (b) KPMF

Note: all figures taken from the original paper

Graphical model

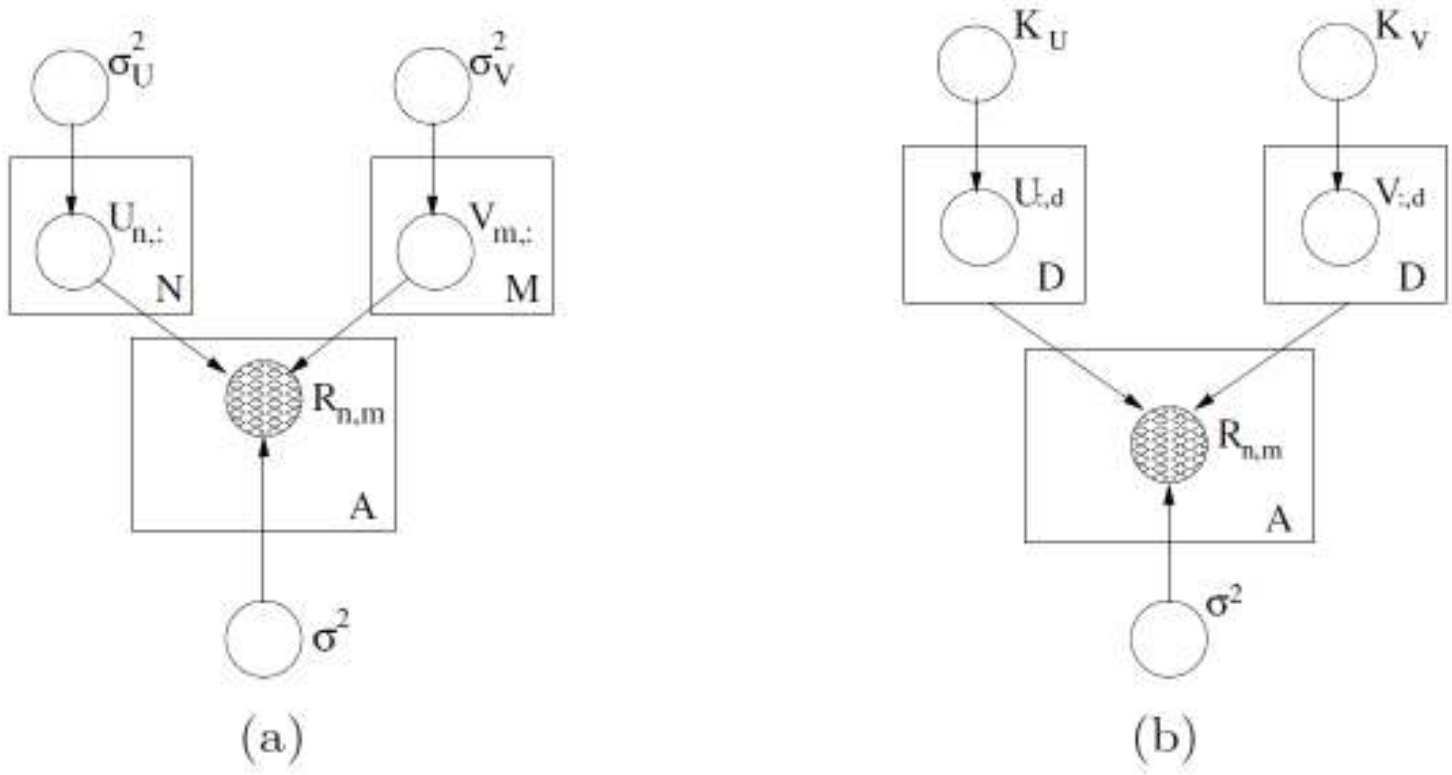


Figure 1: (a) The generative process of R in PMF; (b) The generative process of R in KPMF. A is the total number of non-missing entries in the matrix.

KPMF model equations

- Let $\delta_{n,m} = 1$ if $R_{n,m}$ is observed, and zero otherwise
- Likelihood

$$p(R|U, V, \sigma^2) = \prod_{n=1}^{N} \prod_{m=1}^{M} [\mathcal{N}(R_{n,m}|U_{n,:}, V_{m,:}, \sigma^2)]^{\delta_{n,m}}$$

- Priors

$$p(U|K_U) = \prod_{d=1}^{D} GP(U_{:,d}|0, K_U)$$

$$p(V|K_V) = \prod_{d=1}^{D} GP(V_{:,d}|0, K_V)$$

Log-posterior of KPMF

$$
\begin{aligned}
&\log p(U, V | R, \sigma^2, K_U, K_V) \\
&= -\frac{1}{2\sigma^2} \sum_{n=1}^{N} \sum_{m=1}^{M} \delta_{n,m} (R_{n,m} - U_{n,:} V_{m,:}^T)^2 \\
&\quad - \frac{1}{2} \sum_{d=1}^{D} U_{:,d}^T S_U U_{:,d} - \frac{1}{2} \sum_{d=1}^{D} V_{:,d}^T S_V V_{:,d} \\
&\quad - A \log \sigma^2 - \frac{D}{2} (\log |K_U| + \log |K_V|) + C,
\end{aligned}
$$

where A is the total number of observed elements of R and C is a constant

Gradient descent

$$\frac{\partial E}{\partial U_{n,d}} = -\frac{1}{\sigma^2}\sum_{m=1}^{M}\delta_{n,m}(R_{n,m} - U_{n,:}V_{m,:}^T)V_{d,m} + \boldsymbol{e}_{(n)}^T S_U U_{:,d},$$

$$\frac{\partial E}{\partial V_{m,d}} = -\frac{1}{\sigma^2}\sum_{n=1}^{N}\delta_{n,m}(R_{n,m} - U_{n,:}V_{m,:}^T)U_{d,n} + \boldsymbol{e}_{(m)}^T S_V V_{:,d}\ ,$$

where E is the log-posterior and $e_{(n)}$ is a vector with the n-th element equal to one and all others to zero

Gradient descent

- Updates

$$U_{n,d}^{(t+1)} = U_{n,d}^{(t)} - \eta \frac{\partial E}{\partial U_{n,d}},$$

$$V_{m,d}^{(t+1)} = V_{m,d}^{(t)} - \eta \frac{\partial E}{\partial V_{m,d}}$$

- KPMF yields estimates even when there are completely unobserved rows or columns

$$\begin{aligned} U_{n,d}^{(t+1)} &= U_{n,d}^{(t)} - \eta \boldsymbol{e}_{(n)}^T S_U U_{:,d} \\ &= U_{n,d}^{(t)} - \eta \sum_{n'=1}^{N} S_U(n,n') U_{n',d} \\ V_{m,d}^{(t+1)} &= V_{m,d}^{(t)} - \eta \boldsymbol{e}_{(m)}^T S_V V_{:,d} \\ &= V_{m,d}^{(t)} - \eta \sum_{m'=1}^{M} S_V(m,m') V_{m',d} \end{aligned}$$

Stochastic gradient descent

- The objective can be rewritten as

$$E = \sum_{n=1}^{N}\sum_{m=1}^{M} \delta_{n,m}\left[\frac{1}{\sigma^2}(R_{n,m} - U_{n,:}V_{m,:}^T)^2 + \frac{1}{\tilde{M}_n}U_{n,:}^T\sum_{n'=1}^{N} S_U(n,n')U_{n',:} + \frac{1}{\tilde{N}_m}V_{m,:}^T\sum_{m'=1}^{M} S_V(m,m')V_{m',:}\right]$$

$$= \sum_{n=1}^{N}\sum_{m=1}^{M} \delta_{n,m}E_{n,m}$$

where $\tilde{M}_n$ is the number of observed entries in row n (and similarly for $\tilde{N}_m$)

Stochastic gradient descent

$$\frac{\partial E_{n,m}}{\partial U_{n,:}} = -\frac{2}{\sigma^2}(R_{n,m} - U_{n,:}^T V_{m,:})V_{m,:}$$
$$+ \frac{1}{\tilde{M}_n}\left[\sum_{n'=1}^{N} S_U(n,n')U_{n',:} + S_U(n,n)U_{n,:}\right],$$
$$\frac{\partial E_{n,m}}{\partial V_{m,:}} = -\frac{2}{\sigma^2}\delta_{n,m}(R_{n,m} - U_{n,:}^T V_{m,:})U_{n,:}$$
$$+ \frac{1}{\tilde{N}_m}\left[\sum_{m'=1}^{M} S_V(m,m')V_{m',:} + S_V(m,m)V_{m,:}\right]$$

Graphs and side information

- Social network of users represented as a graph G with adjacency matrix A [3]
- $A_{i,j} = 1$ if users i and j are connected, and zero otherwise
- Laplacian matrix $L = D - A$, where D is a diagonal matrix containing the node degrees[4]

[3] Do not confuse with the number of observed elements of R, also denoted A

[4] Do not confuse with the number of factors, also denoted D

Graph kernels

- Diffusion kernel
  $K_D = e^{-\beta L}$, with β a bandwith parameter
- Commute time kernel
  $K_{CT} = L^{\dagger}$ (pseudoinverse)
- Regularized Laplacian kernel
  $K_{RL} = (I + \gamma L)^{-1}$, with γ a regularization parameter

Collaborative filtering experiments

	Flixster	Epinion
# Users	2000	2000
# Items	3000	3000
# Ratings	173,172	60,485
# Relations	32,548	74,575
Rating Density	2.89%	1.00%

Table 1: Statistics of the datasets used.

Test RMSE and running times

(a) 20% training data used

	Flixster		Epinion	
	D = 5	D = 10	D = 5	D = 10
Item Average	0.2358	0.2358	0.3197	0.3197
KPMF(Diffusion)	0.2183	0.2180	0.2424	0.2436
KPMF(CT)	0.2184	0.2180	**0.2375**	**0.2378**
KPMF(RL)	**0.2182**	**0.2179**	0.2422	0.2433

(b) 80% training data used

	Flixster		Epinion	
	D = 5	D = 10	D = 5	D = 10
Item Average	0.2256	0.2256	0.2206	0.2206
KPMF(Diffusion)	0.2207	0.2209	0.2257	0.2269
KPMF(CT)	0.2209	0.2208	**0.2180**	**0.2180**
KPMF(RL)	**0.2206**	**0.2207**	0.2252	0.2263

Table 3: RMSE on users with no ratings for training.

	Flixster		Epinion	
	KPMF_{GD}	KPMF_{SGD}	KPMF_{GD}	KPMF_{SGD}
RMSE	**0.180**	0.184	**0.232**	0.240
Time (sec)	1353.6	**5.5**	1342.3	**7.8**

Table 4: Comparison of RMSE and running time for KPMF_{GD} and KPMF_{SGD}. KPMF_{SGD} is slightly worse than KPMF_{GD} in terms of RMSE, but significantly faster.

Comparisons

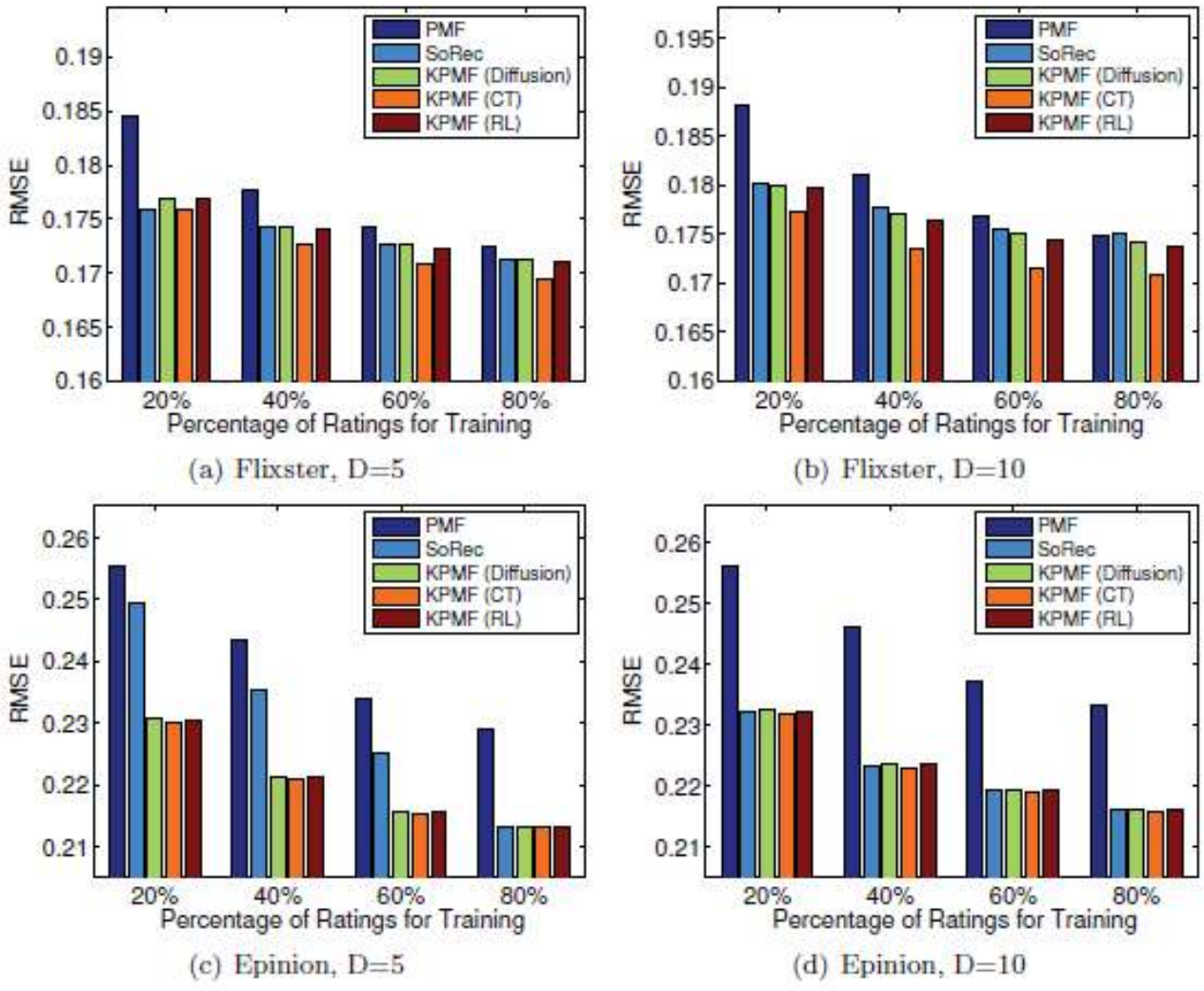
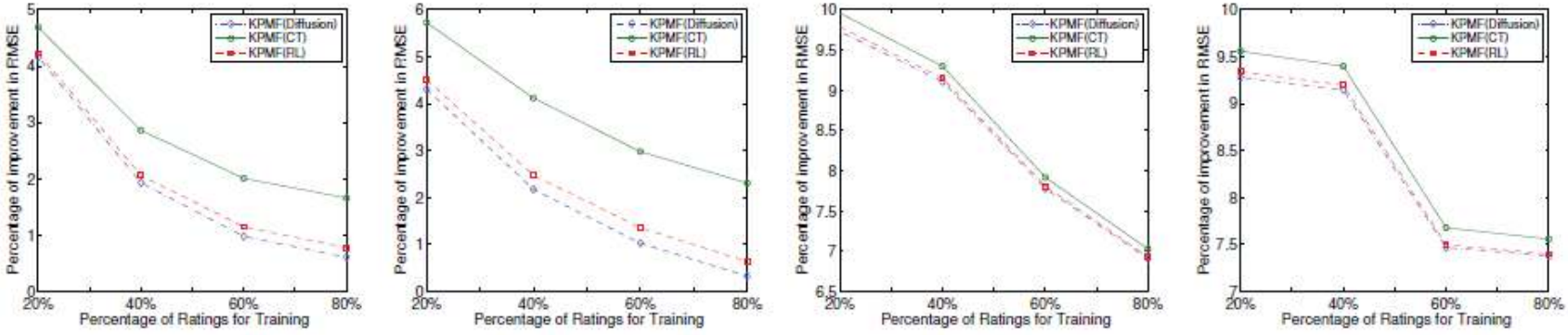


Figure 5: RMSE for different algorithms on Flixster and Epinion datasets (best viewed in color). Lower is better.

Comparisons



(a) Flixster, D=5 (b) Flixster, D=10 (c) Epinion, D=5 (d) Epinion, D=10

Figure 6: Performance improvement of KPMF compared to PMF on training sets with different number of observed ratings (best viewed in color). The improvement of KPMF versus PMF decreases as more training data are used. This is because for sparser datasets, PMF would have relatively more difficulty in learning users' preferences from fewer number of past ratings, while KPMF could still take advantage of the known social relations among users and utilize the observed ratings better.

Image restoration experiments

Figure 8: Image restoration results using PMF and KPMF (best viewed in color). From left to right: original images, corrupted images (regions to be restored are in black), images restored using PMF, and images restored using KPMF. For KPMF, Δ equals to 5 when constructing the row and column graphs, and Diffusion kernel with $\beta = 0.5$ is used to obtain the kernel matrices.

Image restoration RMSE

	KPMF			PMF			
Image	Channel R	Channel G	Channel B	Channel R	Channel G	Channel B	#Masked pixels
1	0.082	0.073	0.066	0.120	0.101	0.094	5426
2	0.160	0.164	0.156	0.173	0.177	0.170	5975
3	0.135	0.133	0.131	0.179	0.171	0.164	31902
4	0.100	0.106	0.124	0.149	0.141	0.173	32407
5	0.103	0.049	0.030	0.143	0.081	0.040	28025
6	0.109	0.091	0.081	0.141	0.109	0.105	23061

Table 5: RMSE comparison between PMF and KPMF on RGB channels of restored images. Smaller is better.