MPMA: Mixture Probabilistic Matrix Approximation for Collaborative Filtering

Chao Chen[※], Dongsheng Li[ŧ], Qin Lv[*], JunChi Yan[ŧς], Stephen M. Chu[ŧ], Li Shang[*※]

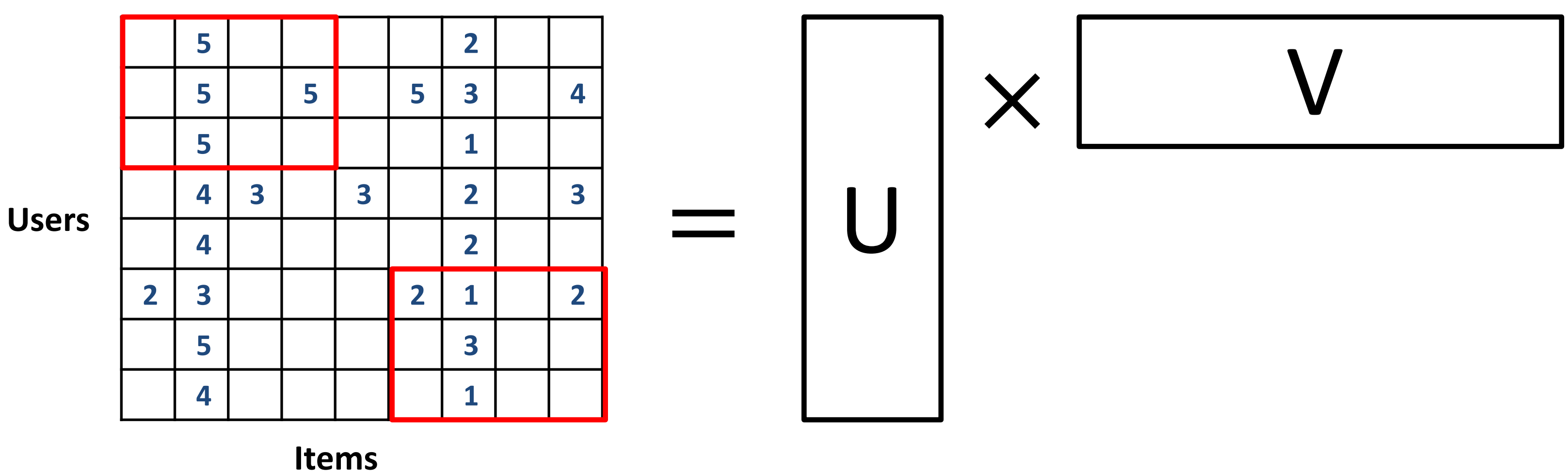
[※]Tongji University, China
[ŧ] IBM Research, China
[ς] East China Normal University, China
[*] University of Colorado Boulder, USA

Introduction

❑ Matrix approximation based collaborative filtering

- Better recommendation accuracy
- High computation complexity: *O(rMN) per iteration*
- Effectively estimate overall structures
- **Poorly detect strong local associations**

$$\widehat{\mathrm{M}} = \underset{\mathrm{X}=\mathrm{UV}'}{\operatorname{argmin}} \|\mathrm{M} - \mathrm{UV}'\|$$



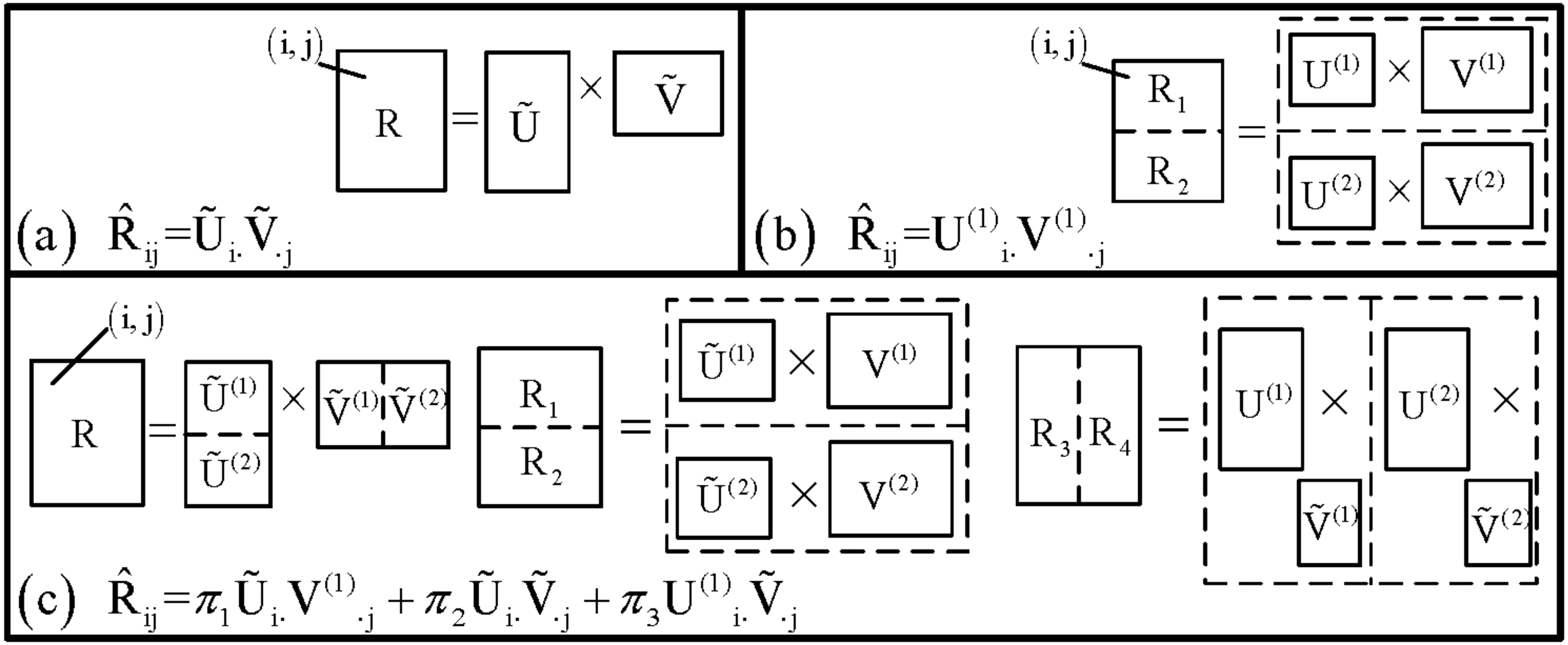
Introduction

❑ Challenge

- How to utilize both global and local information

❑ Intuition

a) Standard low-rank model *ignoring local associations*
b) Clustering-based model *ignoring global structure*
c) Proposed MPMA model *automatically fuse global and local information*

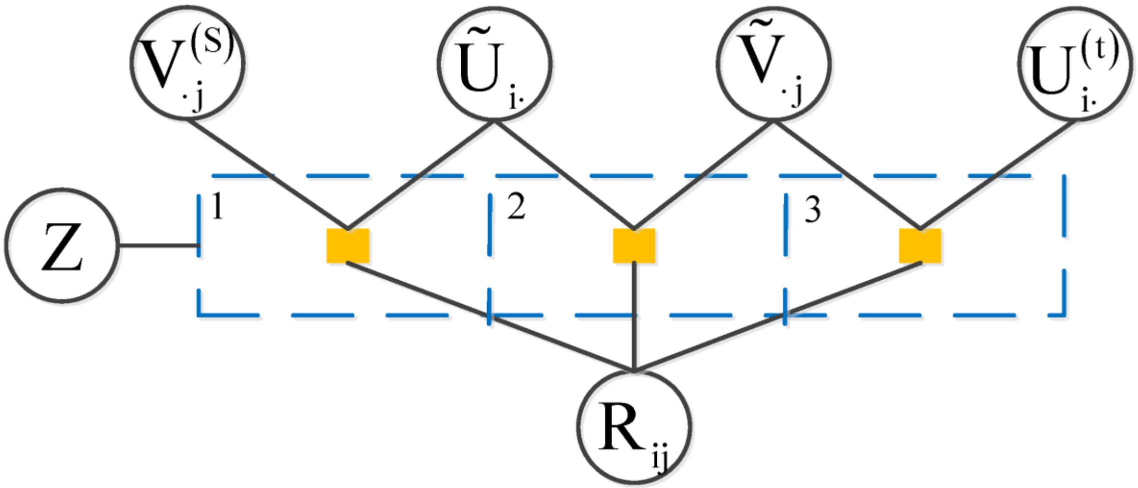


Outline

- ❑ Introduction
- ❑ MPMA Design
  - ❑ Problem Formulation
  - ❑ Efficient Pipeline-based Learning Algorithm
  - ❑ Recommendation Prediction
- ❑ Performance Analysis
  - ❑ Sensitivity Analysis
  - ❑ Performance Comparison
- ❑ Conclusion

MPMA Design – Problem Formulation

❑ Mixture Model



1. $\mathrm{N}\left(R_{ij}\middle|\tilde{U}_{i\cdot}V^{(s)}_{\cdot j},\sigma_1^2\right)$
2. $\mathrm{N}\left(R_{ij}\middle|\tilde{U}_{i\cdot}\tilde{V}_{\cdot j},\sigma_2^2\right)$
3. $\mathrm{N}\left(R_{ij}\middle|U^{(t)}_{i\cdot}\tilde{V}_{\cdot j},\sigma_3^2\right)$

❑ Loss Function

$$\ln p\left(\tilde{U},\tilde{V},U^{(1)},\dots,U^{(g)},V^{(1)},\dots,V^{(f)}\middle|R\right)$$
$$=\sum_{s=1}^{f}\sum_{t=1}^{g}\sum_{\rho(i)=s}\sum_{\rho(j)=t}\ln\{I_{ij}(\pi_1 p(R_{ij},\tilde{U},V^{(s)})$$
$$+\pi_2 p(R_{ij},\tilde{U},\tilde{V})+\pi_3 p(R_{ij},U^{(t)},\tilde{V}))\}+C$$

$$\zeta\left(\tilde{U},\tilde{V},U^{(1)},\dots V^{(f)}\right)$$
$$=\left\|I\otimes\left(R-\tilde{U}\tilde{V}\right)\right\|+\lambda_1\|\tilde{U}\|+\lambda_2\|\tilde{V}\|$$
$$+\sum_{s\epsilon[f]}\alpha_s\left\|I_U^{(s)}\otimes\left(R_U^{(s)}-\tilde{U}^{(s)}V^{(s)}\right)\right\|$$
$$+\sum_{t\epsilon[g]}\beta_t\left\|I_V^{(t)}\otimes\left(R_V^{(t)}-U^{(s)}\tilde{V}^{(s)}\right)\right\|$$
$$+\sum_{s\epsilon[f]}\lambda_3\|V^{(s)}\|+\sum_{t\epsilon[g]}\lambda_4\|U^{(t)}\|$$

MPMA Design – Problem Formulation

❑ Transformation and Variants

$$\min\left\|I\otimes\left(R-\tilde{U}\tilde{V}\right)\right\|+\lambda_1\left\|\tilde{U}\right\|+\lambda_2\left\|\tilde{V}\right\|$$
$$+\sum_{s\epsilon[f]}\alpha_s\left\|I_U^{(s)}\otimes\left(R_U^{(s)}-\tilde{U}^{(s)}V^{(s)}\right)\right\|$$

local item features

$$+\sum_{t\epsilon[g]}\beta_t\left\|I_V^{(t)}\otimes\left(R_V^{(t)}-U^{(s)}\tilde{V}^{(s)}\right)\right\|$$

local User features

$$+\sum_{s\epsilon[f]}\lambda_3\left\|V^{(s)}\right\|+\sum_{t\epsilon[g]}\lambda_4\left\|U^{(t)}\right\|$$

i-MPMA: only local item features are applied

u-MPMA: only local user features are applied

MPMA: both local item and user features are applied

⇩

$$\min\left\|I\otimes\left(R-\tilde{U}\tilde{V}\right)\right\|+\lambda_1\left\|\tilde{U}\right\|+\lambda_2\left\|\tilde{V}\right\|$$
$$+\sum_{s\epsilon[f]}\lambda_3\left\|V^{(s)}\right\|+\sum_{t\epsilon[g]}\lambda_4\left\|U^{(t)}\right\|$$

$s.t.$

$$\sum_{s\epsilon[f]}\alpha_s\left\|I_U^{(s)}\otimes\left(R_U^{(s)}-\tilde{U}^{(s)}V^{(s)}\right)\right\|\leq\delta$$
$$\sum_{t\epsilon[g]}\beta_t\left\|I_V^{(t)}\otimes\left(R_V^{(t)}-U^{(s)}\tilde{V}^{(s)}\right)\right\|\leq\varepsilon$$

Minimizing the overall error, while guaranteeing the performance in each submatrices

MPMA Design – Efficient Pipeline-based Learning Algorithm

❑ Challenge

- High computational overheads

$$\frac{\partial \zeta}{\partial \tilde{U}^{(s)}} = \lambda_1 \tilde{U}^{(s)} + I_U^{(s)} \otimes \left(\tilde{U}^{(s)}\tilde{V} - R_U^{(s)}\right)\tilde{V}' + \alpha_s I_U^{(s)} \otimes \left(\tilde{U}^{(s)} V^{(s)} - R_U^{(s)}\right)\left[V^{(s)}\right]'$$

$$\frac{\partial \zeta}{\partial \tilde{V}^{(s)}} = \lambda_2 \tilde{V}^{(t)} + I_V^{(t)} \otimes \left(\tilde{U}\tilde{V}^{(t)} - R_V^{(t)}\right)'\tilde{U} + \beta_t I_V^{(t)} \otimes \left(U^{(t)}\tilde{V}^{(t)} - R_V^{(t)}\right)' U^{(t)}$$

$$\frac{\partial \zeta}{\partial U^{(t)}} = \beta_t I_V^{(t)} \otimes \left(U^{(t)}\tilde{V}^{(t)} - R_V^{(t)}\right)\left[\tilde{V}^{(t)}\right]' + \lambda_3 U^{(t)}$$

$$\frac{\partial \zeta}{\partial V^{(s)}} = \alpha_s I_U^{(s)} \otimes \left(\tilde{U}^{(s)} V^{(s)} - R_U^{(s)}\right)'\tilde{U}^{(s)} + \lambda_4 V^{(s)}$$

For 100 items, the running time is reduced from 200T to 101T (very closed to SVD's 100T)

❑ Pipeline-based Learning Algorithm

S_1			
	S_2		
		S_3	
			S_4

Item Seq.					
1	Glo_1	Loc_1			
2		Glo_2	Loc_2		
3			Glo_3	Loc_3	
4				Glo_4	Loc_4
Time	T_1	T_2	T_3	T_4	T_5

MPMA Design – Recommendation Prediction

❑ Problem

Given global and local features $\widetilde{U}, \widetilde{V}, U^{(1)}, \ldots, U^{(g)}, V^{(1)}, \ldots, V^{(f)}$,

estimate (π_1, π_2, π_3) to produce prediction by

$$\hat{R}_{ij} = \pi_1 \widetilde{U}_{i.} V_{.j}^{(s)} + \pi_2 \widetilde{U}_{i.} \widetilde{V}_{.j} + \pi_3 U_{i.}^{(t)} \widetilde{V}_{.j}$$

❑ EM-based Estimation Method

E-Step

$$\gamma\left(Z_{ij}^{k}\right) = \frac{\pi_k N\left(R_{ij} \middle| R_{ij}^{(k)}, \sigma_k^2\right)}{\sum_{l \epsilon [1,3]} \pi_l N\left(R_{ij} \middle| R_{ij}^{(l)}, \sigma_l^2\right)}$$

$$R_{ij}^{(1)} = \widetilde{U}_{i.} V_{.j}^{(s)}$$

$$R_{ij}^{(2)} = \widetilde{U}_{i.} \widetilde{V}_{.j}$$

$$R_{ij}^{(3)} = U_{i.}^{(t)} \widetilde{V}_{.j}$$

M-Step

$$\sigma_k^2 = \frac{\sum_{ij} \gamma\left(Z_{ij}^{k}\right)\left(R_{ij} - R_{ij}^{(k)}\right)^2}{N_k}$$

$$\pi_k = \frac{N_k}{N}$$

$$N_k = \sum_{ij} \gamma\left(Z_{ij}^{k}\right)$$

$$N = \sum_{k} N_k$$

Outline

- Introduction
- MPMA Design
  - Problem Formulation
  - Efficient Pipeline-based Learning Algorithm
  - Recommendation Prediction
- Performance Analysis
  - Sensitivity Analysis
  - Performance Comparison
- Conclusion

Empirical Analysis – Experimental Setup

	MovieLens 1M	MovieLens 10M	Netflix
#users	6,040	69,878	480,189
#items	3,706	10,677	17,770
#ratings	10^6	10^7	10^8

Benchmark datasets

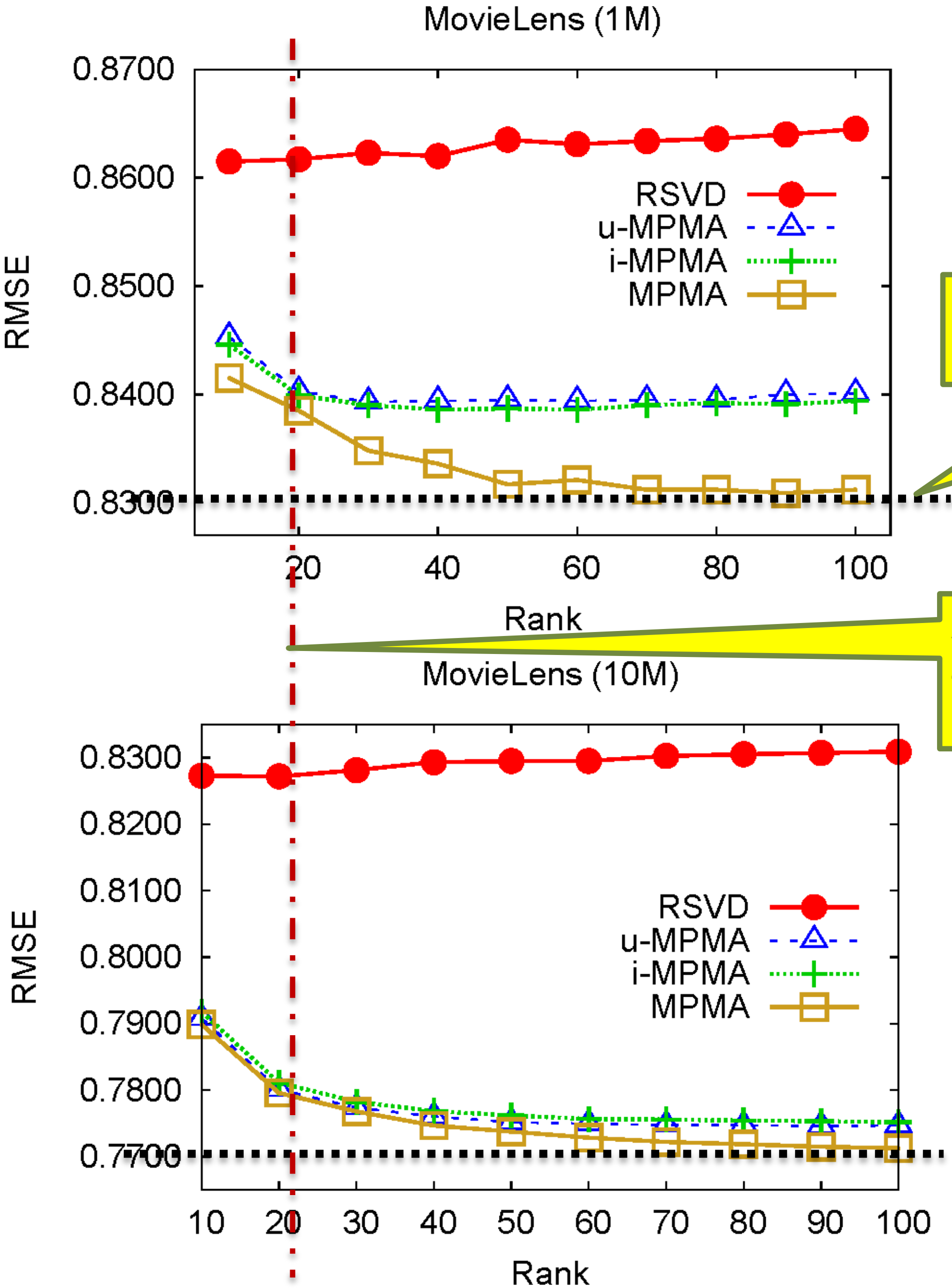
❑ Sensitivity analysis

1. Effect of the latent factor
2. Effect of the clustering

❑ Comparison to state-of-the-art methods

1. Recommendation accuracy
2. Computation efficiency

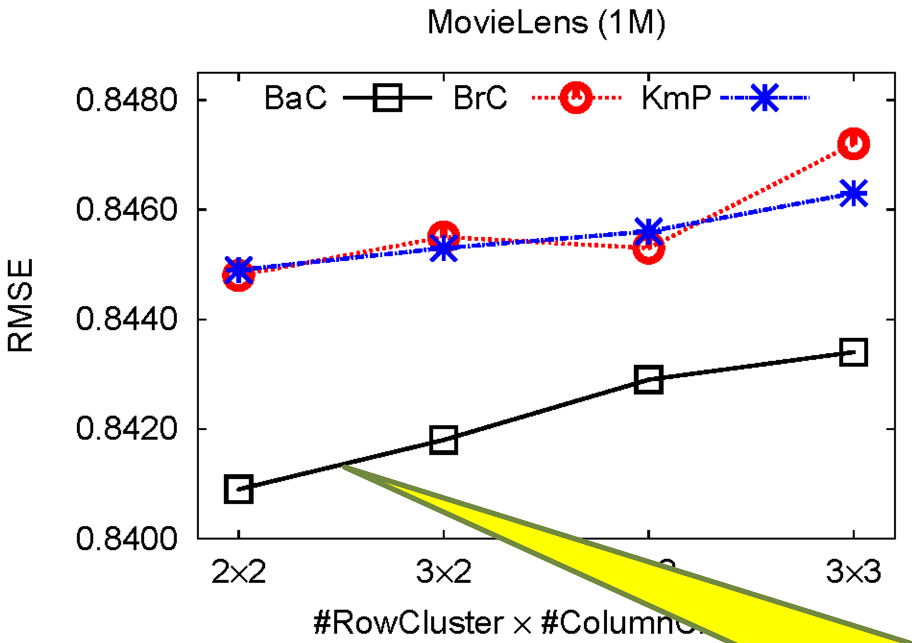
Sensitivity Analysis –Latent Factor

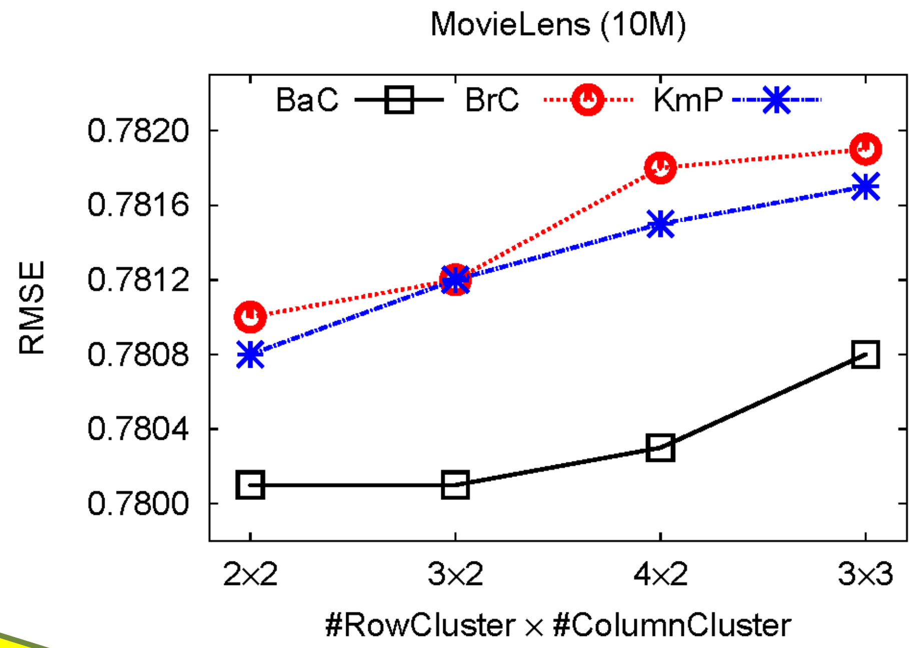


Both local user features and local item features contribute to performance improvement.

In contrast to RSVD, MPMA's recommendation accuracy increases as rank increases, even when rank is quite large.

Sensitivity Analysis – Clustering





MPMA with Balance Clustering(Bac) outperforms the one with Bregman Co-clustering(BrC) and with K-mean Plus(KmP).

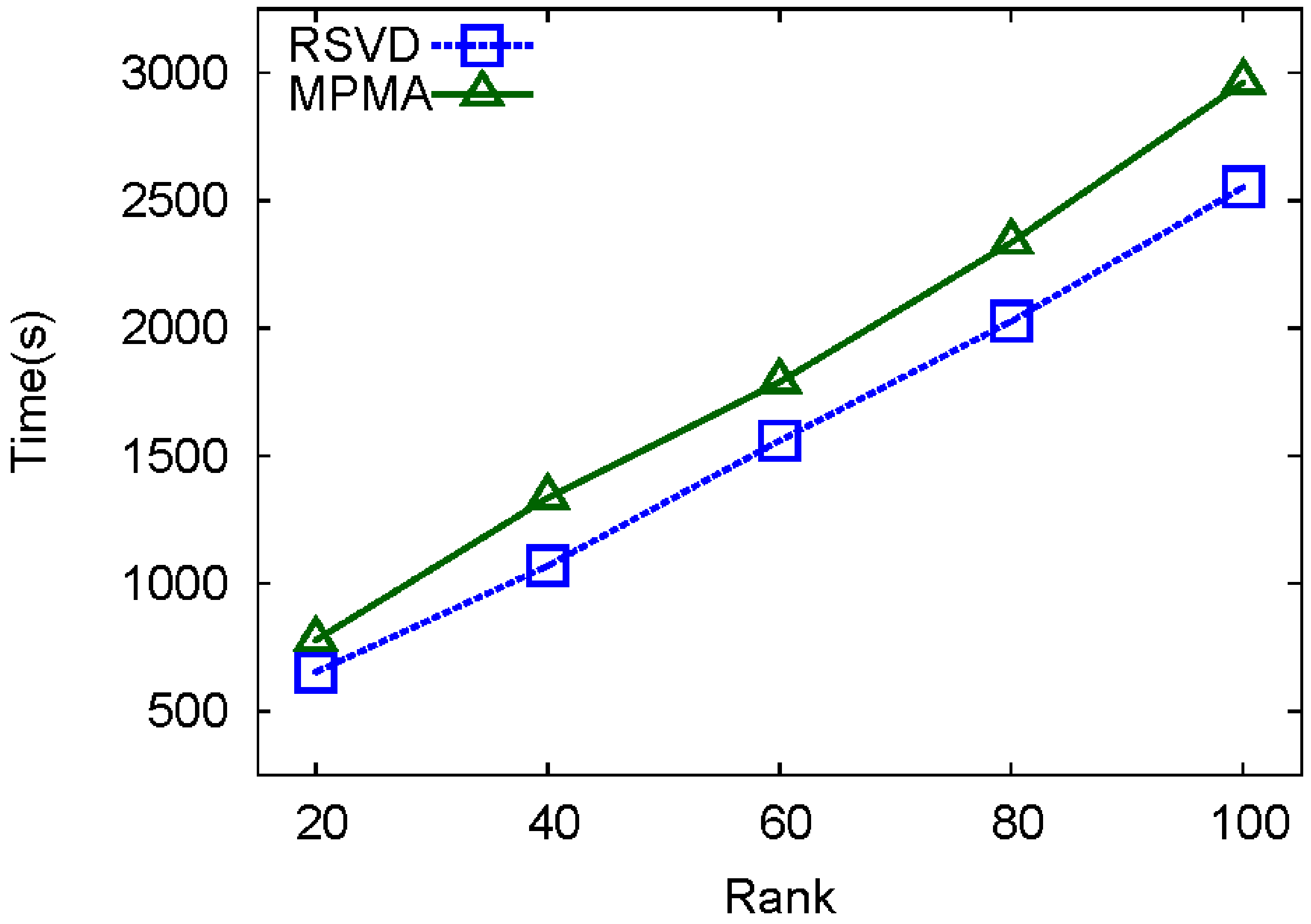
The recommendation accuracy decreases as the clustering size increases.

The recommendation accuracy is maintained as the clustering size increases.

Performance Comparison(1) – Recommendation Accuracy

	MovieLens 10M	Netflix
NMF	0.8832 ± 0.0007	0.9396 ± 0.0002
RSVD	0.8253 ± 0.0009	0.8534 ± 0.0001
BPMF	0.8195 ± 0.0006	0.8420 ± 0.0003
APG	0.8098 ± 0.0005	0.8476 ± 0.0028
GSMF	0.8012 ± 0.0011	0.8420 ± 0.0006
MPMA	$\mathbf{0.7712 \pm 0.0002}$	$\mathbf{0.8139 \pm 0.0003}$

Performance Comparison (2) – Computation Efficiency



Outline

- Introduction
- MPMA Design
  - Problem Formulation
  - Efficient Pipeline-based Learning Algorithm
  - Recommendation Prediction
- Performance Analysis
  - Sensitivity Analysis
  - Performance Comparison
- **Conclusion**

Conclusion

- MPMA – Mixture Probabilistic Matrix Approximation
  - Mixture probabilistic model
  - Efficient pipeline-based learning algorithm
  - EM-based recommendation prediction
- Empirical analysis on three benchmark datasets
  - Sensitivity analysis
  - Improvement in accuracy with good efficiency