WEMAREC: Accurate and Scalable Recommendation through Weighted and Ensemble Matrix Approximation

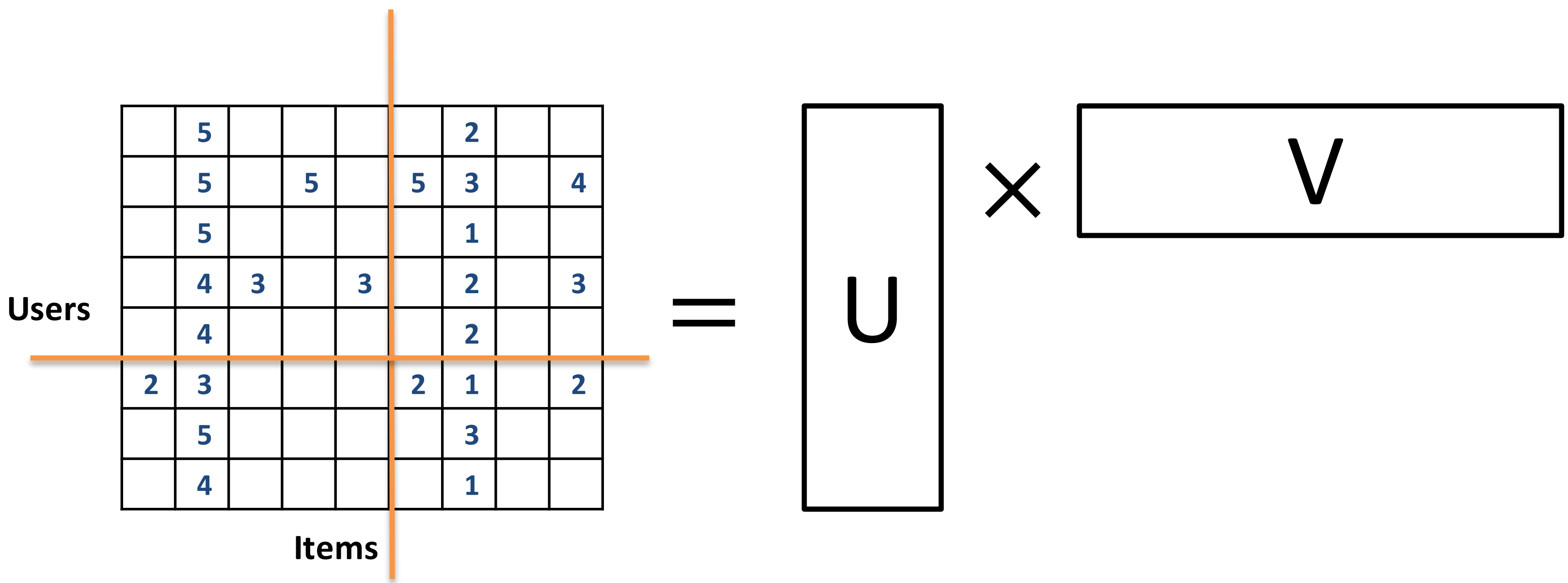
Chao Chen[※], Dongsheng Li[ƚ], Yingying Zhao[※], Qin Lv[*], Li Shang[*※]

[※]Tongji University, China
[ƚ] IBM Research, China
[*] University of Colorado Boulder, USA

Introduction

❑ Matrix approximation based collaborative filtering

- Better recommendation accuracy
- High computation complexity: *O(rMN) per iteration*
- **Clustering based matrix approximation**
    - Better efficiency but lower recommendation accuracy



Outline

- Introduction
- WEMAREC design
  - Submatrices generation
  - Weighted learning on each submatrix
  - Ensemble of local models
- Performance analysis
  - Theoretical bound
  - Sensitivity analysis
  - Comparison with state-of-the-art methods
- Conclusion

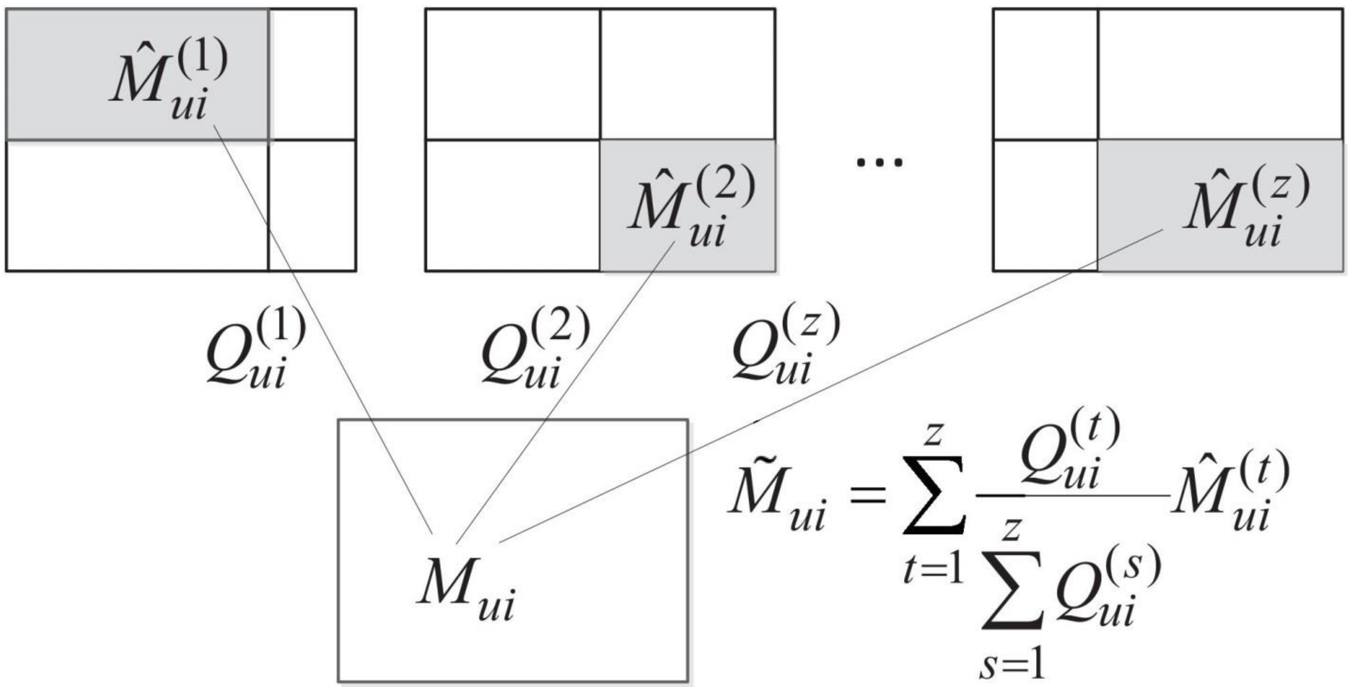
WEMAREC Design

❑ Divide-and-conquer using submatrices

- Better efficiency
- Localized but limited information

❑ Key components

- Submatrices generation
- Weighted learning on each submatrix
- Ensemble of local models



$$\tilde{M}_{ui} = \sum_{t=1}^{z} \frac{Q_{ui}^{(t)}}{\sum_{s=1}^{z} Q_{ui}^{(s)}} \hat{M}_{ui}^{(t)}$$

Step (1) – Submatrices Generation

❑ Challenge

- Low efficiency

e.g., O(kmn) per iteration for k-means clustering

❑ Bregman co-clustering

- Efficient and scalable

O(mkl + nkl) *per iteration*

- Able to detect diverse inner structures

Different distance function + constraint set => different co-clustering

- Low-parameter structure of the generated submatrices

Mostly uneven distribution of generated submatrices

1	2	1	2
3	4	3	4
1	2	1	2
3	4	3	4

After clustering →

1	1	2	2
1	1	2	2
3	3	4	4
3	3	4	4

Matrix size: 4×4

Co-clustering size: 2×2

Step (2) – Weighted Learning on Each Submatrix

❑ Challenge

- Low accuracy due to limited information

❑ Improved learning algorithm

- Larger weight for high-frequency ratings such that the model prediction is closer to high-frequency ratings

$$\widehat{\mathrm{M}} = \underset{X}{\mathrm{argmin}} \| W \otimes (M - X) \| \text{ s.t.,} \quad rank(X) = r, \quad W_{ij} \propto \Pr[M_{ij}]$$

To train a biased model which can produce better prediction on partial ratings

Rating	Distribution	RMSE without Weighting	RMSE with Weighting
1	17.44%	1.2512	1.2533
2	**25.39%**	0.6750	**0.6651**
3	**35.35%**	0.5260	**0.5162**
4	**18.28%**	1.1856	**1.1793**
5	3.54%	2.1477	2.1597
Overall accuracy		0.9517	**0.9479**

Case study on synthetic dataset

Step (3) – Ensemble of Local Models

❑ Observations

- User rating distribution → User rating preferences
- Item rating distribution → Item quality

❑ Improved ensemble method

- Global approximation considering the effects of user rating preferences and item quality

$$\widetilde{\mathrm{M}}_{ui} = \sum_t \frac{Q_{ui}{}^{(t)}}{\sum_s Q_{ui}{}^{(s)}} \widehat{M}_{ui}{}^{(t)}$$

- Ensemble weight

$$Q_{ui}{}^{(t)} = 1 + \beta_1 \Pr\left[\widehat{M}_{ui}{}^{(t)} | M_u\right] + \beta_2 \Pr\left[\widehat{M}_{ui}{}^{(t)} | M_i\right]$$

	1	2	3	4	5
Probabilities of M_u	0.05	0.05	0.1	**0.5**	0.3
Probabilities of M_i	0.05	0.05	0.1	0.2	**0.6**

Model 1 → **1** → 1 + 0.05 +0.05 = 1.1

Model 2 → **5** → 1 + 0.3 +0.6 = 1.9

Model 3 → **4** → 1 + 0.5 +0.2 = 1.7

$$\frac{1.1 \times 1 + 1.9 \times 5 + 1.7 \times 4}{1.1 + 1.9 + 1.7} = \mathbf{3.70} > 3.33 = \frac{1 + 5 + 4}{3}$$

Outline

- ❑ Introduction
- ❑ WEMAREC
  - ❑ Submatrices generation
  - ❑ Weighted learning on each submatrix
  - ❑ Ensemble of local models
- ❑ Performance analysis
  - ❑ Theoretical bound
  - ❑ Sensitivity analysis
  - ❑ Comparison with state-of-the-art methods
- ❑ Conclusion

Theoretical Bound

❑ Error bound

- [Candés & Plan, 2010] If $\mathrm{M} \in \mathbb{R}^{m\times n}$ has sufficient samples ($|\Omega| \geq C\mu^2 nr \log^6 n$), and the observed entries are distorted by a bounded noise Z, then with high probability, the error is bounded by

$$\left\|M - \widehat{M}\right\|_F \leq 4\delta\sqrt{\frac{(2+\rho)m}{\rho}} + 2\delta$$

- Our extension: Under the same condition, with high probability, the global matrix approximation error is bounded by

$$\mathrm{D}\left(\widehat{M}\right) \leq \frac{\alpha(1+\beta_0)}{\sqrt{mn}}\left(4\sqrt{\frac{2+\rho}{\rho}(klm)} + 2kl\right)$$

❑ Observations

- When the matrix size is small, a greater co-clustering size may reduce the accuracy of recommendation.
- When the matrix size is large enough, the accuracy of recommendation will not be sensitive to co-clustering size.

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

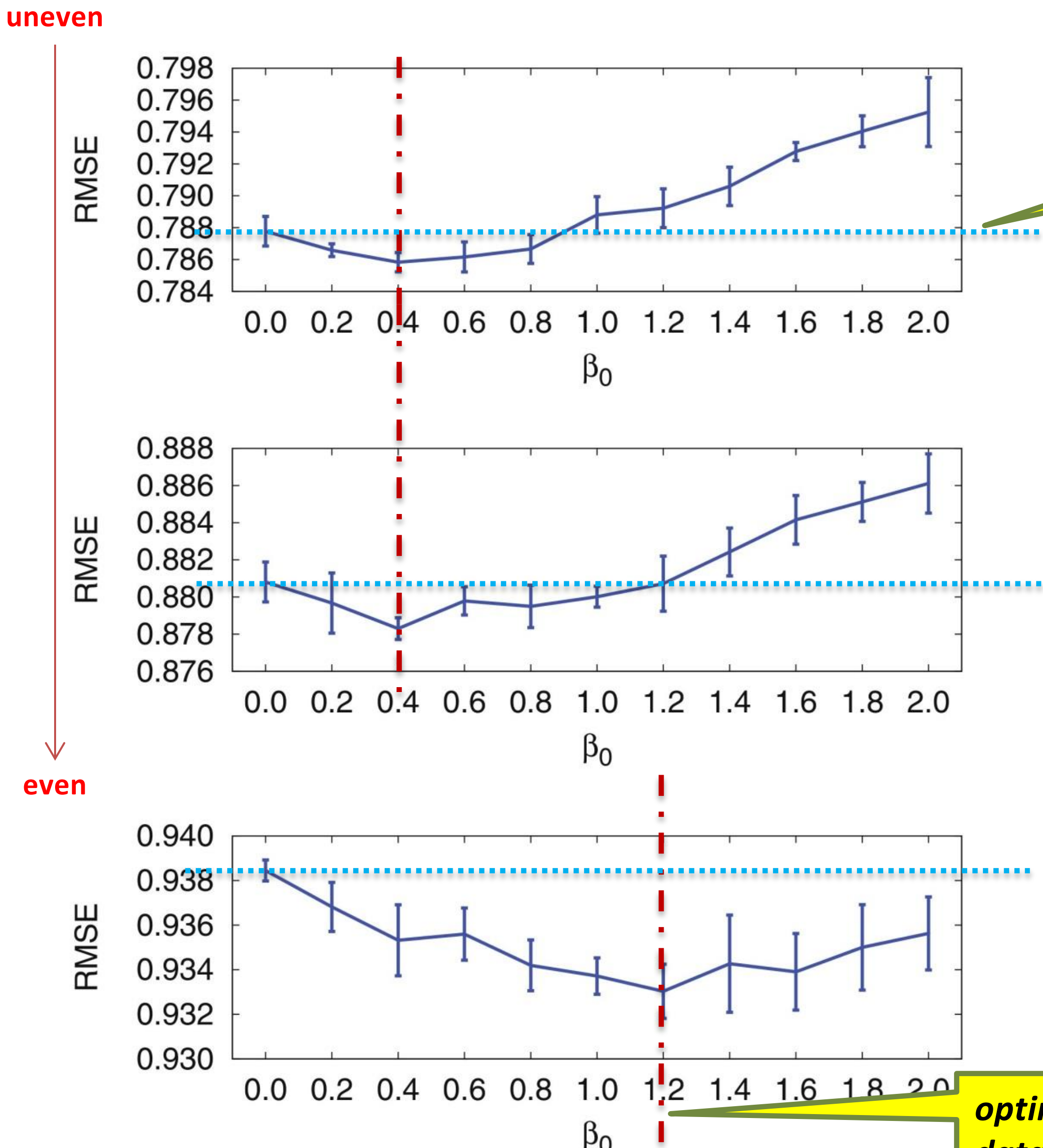
❑ Sensitivity analysis

1. Effect of the weighted learning
2. Effect of the ensemble method
3. Effect of Bregman co-clustering

❑ Comparison to state-of-the-art methods

1. Recommendation accuracy
2. Computation efficiency

Sensitivity Analysis – Weighted Learning



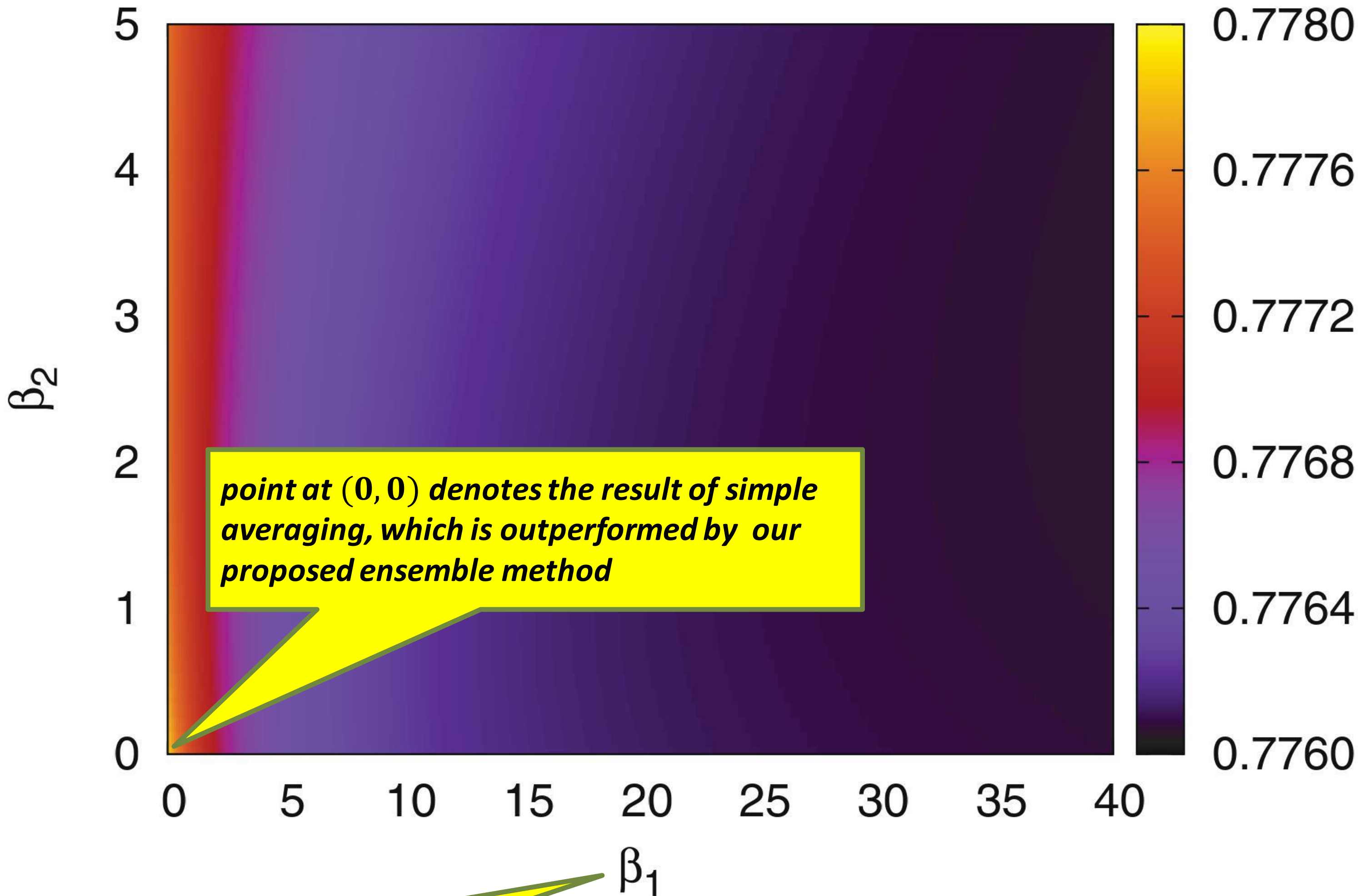
weighted learning algorithm can outperform no-weighting methods

Rating	D1 (uneven)	D2 (medium)	D3 (even)
1	0.98%	3.44%	18.33%
2	3.14%	9.38%	26.10%
3	15.42%	29.25%	35.27%
4	40.98%	37.86%	16.88%
5	39.49%	20.06%	3.43%

Rating Distribution of Three Synthetic Datasets

optimal weighting parameter on uneven dataset is smaller than that on even dataset

Sensitivity Analysis – Ensemble Method

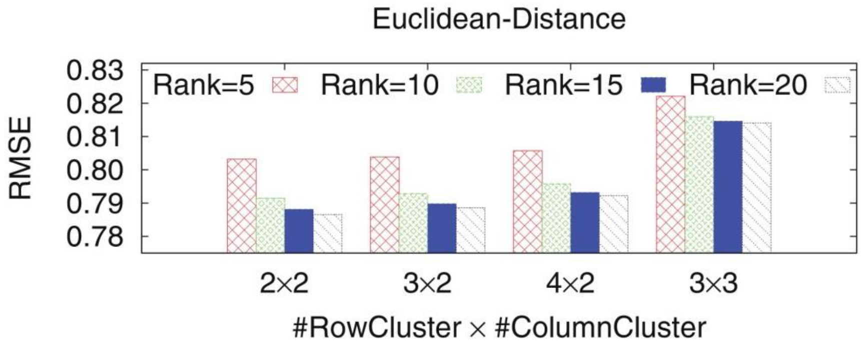


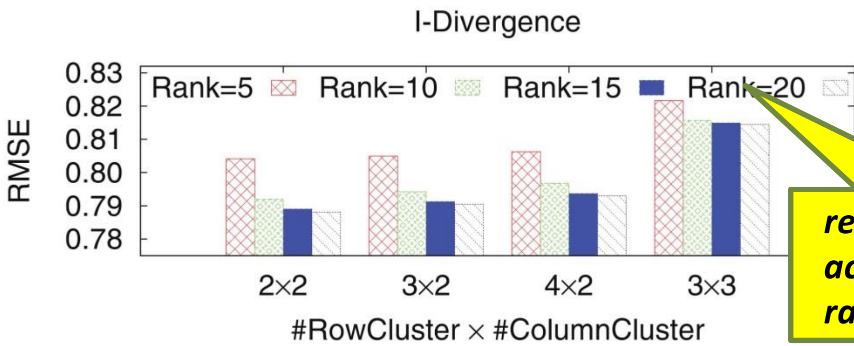
point at $(0, 0)$ denotes the result of simple averaging, which is outperformed by our proposed ensemble method

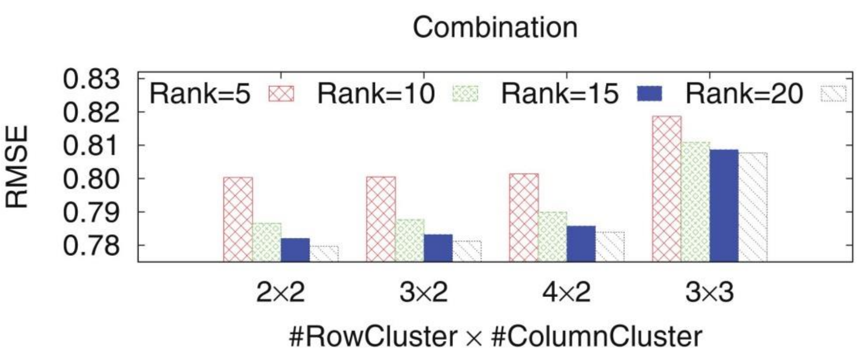
information about user rating preferences is more valuable than that of item quality

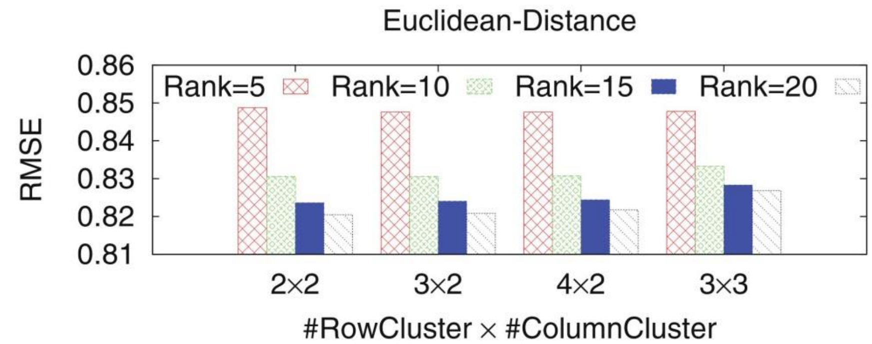
Sensitivity Analysis – Bregman Co-clustering

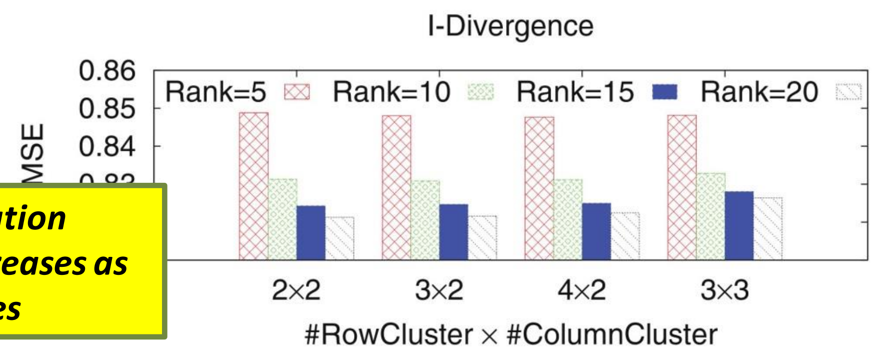
MovieLens 10M | **Netflix**

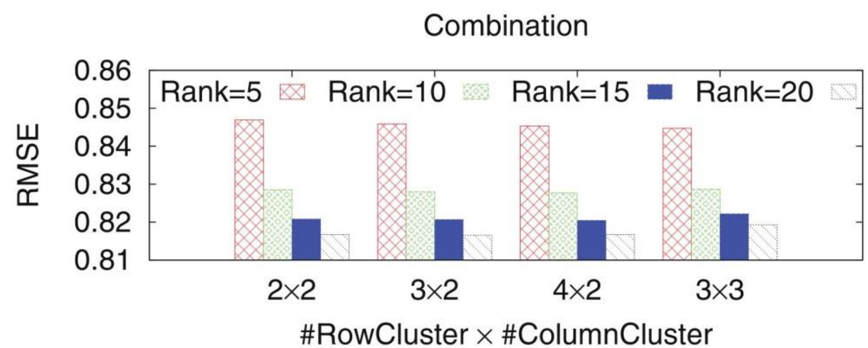












recommendation accuracy increases as rank increases

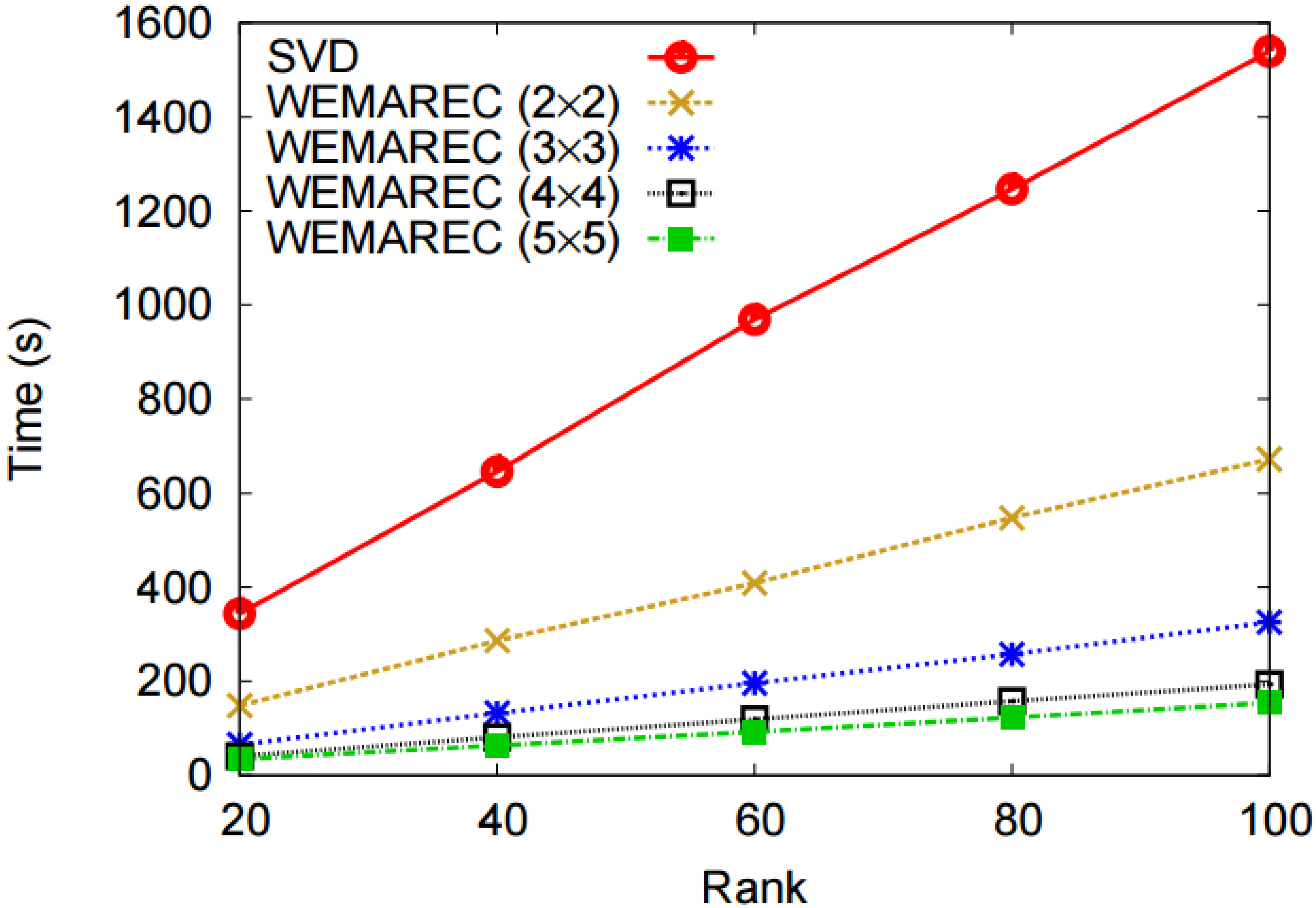
recommendation accuracy decreases as co-clustering size increases

recommendation accuracy is maintained as co-clustering size increases

Comparison with State-of-the-art Methods (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
DFC	0.8064 ± 0.0006	0.8451 ± 0.0005
LLORMA	0.7851 ± 0.0007	0.8275 ± 0.0004
WEMAREC	$\mathbf{0.7769 \pm 0.0004}$	$\mathbf{0.8142 \pm 0.0001}$

Comparison with State-of-the-art Methods (2) – Computation Efficiency

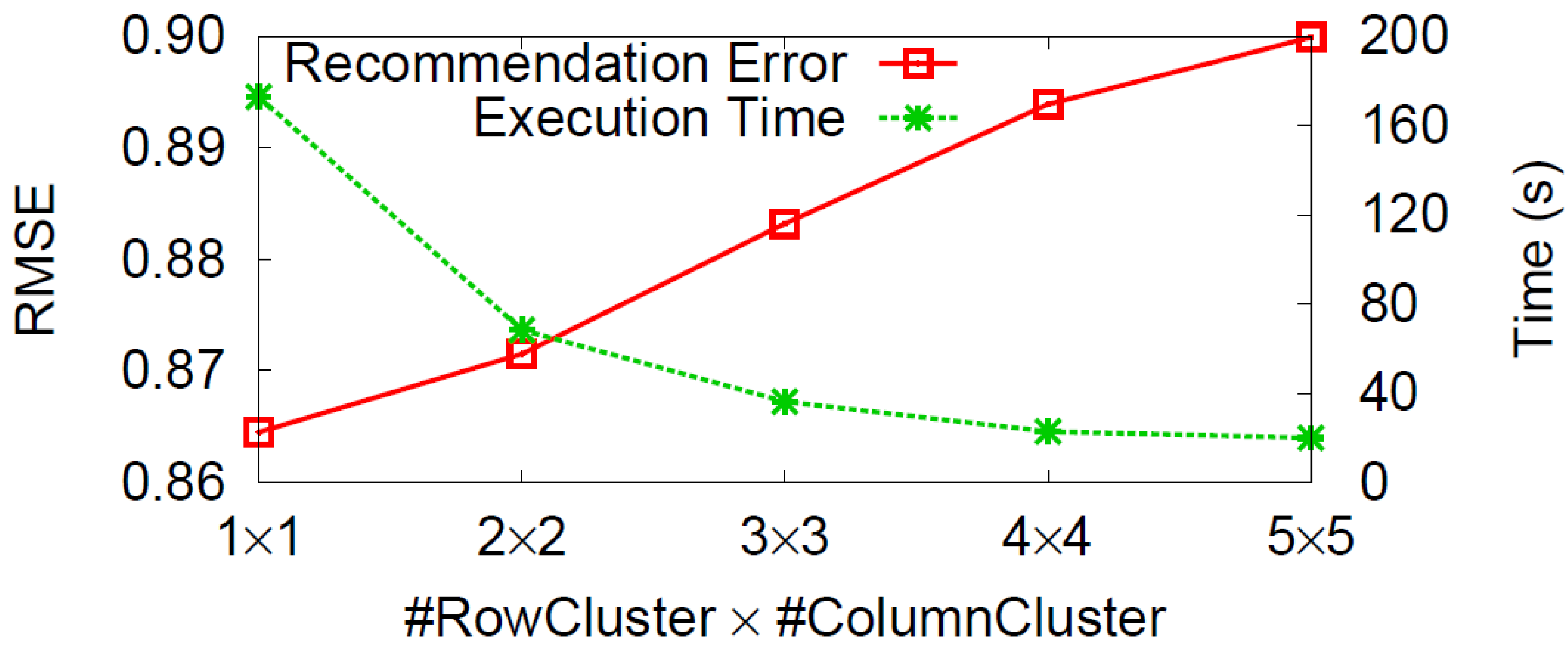


Execution time on the MovieLens 1M dataset

Conclusion

- ❑ WEMAREC – Accurate and scalable recommendation
  - Weighted learning on submatrices
  - Ensemble of local models
- ❑ Theoretical analysis in terms of sampling density, matrix size and co-clustering size
- ❑ Empirical analysis on three benchmark datasets
  - Sensitivity analysis
  - Improvement in both accuracy and efficiency

Trade-off between Accuracy and Scalability



Detailed Implementation

Algorithm 1 Co-clustering-based Matrix Approximation

Input: All co-clustering submatrices $\mathcal{M}^{(t)} \subseteq M$ $(t \in [kl])$, rank r, learning rate v, regularization coefficient λ.
Output: Approximated user-item rating matrix $\hat{M}$.

for each $t \in \{1, \ldots, kl\}$ **in parallel do**
    // Computing weights
    Compute the rating distribution on $\mathbb{F}$ in $\mathcal{M}^{(t)}$.
    for each observed entry (u, i) in $\mathcal{M}^{(t)}$ **do**
        $W_{ui} = p(x)$, if $M_{ui} = x$.
    end for
    // Updating model
    Initialize $U^{(t)} \in \mathbb{R}^{m \times r}, V^{(t)} \in \mathbb{R}^{n \times r}$ randomly
    while not converged **do**
        for each observed entry (u, i) in $\mathcal{M}^{(t)}$ **do**
            $\triangle_{ui} = \mathcal{M}_{ui}^{(t)} - U_u^{(t)} (V_i^{(t)})^T$
            for each $z \in \{1, \ldots, r\}$ **do**
                $U_{uz}^{(t)} = U_{uz}^{(t)} + v * (\triangle_{ui} * V_{iz}^{(t)} * W_{ui} - \lambda U_{uz}^{(t)})$
                $V_{iz}^{(t)} = V_{iz}^{(t)} + v * (\triangle_{ui} * U_{uz}^{(t)} * W_{ui} - \lambda V_{iz}^{(t)})$
            end for
        end for
    end while
end for
for each $(u, i) \in [m] \times [n]$ **do**
    Locate (u, i) in its corresponding submatrix and let the index of the submatrix be ξ.
    $\hat{M}_{ui} = U_u^{(\xi)} (V_i^{(\xi)})^T$
end for
return $\hat{M}$

Algorithm 2 WEMAREC_Ensemble (u, i)

Input: Resulting matrix approximations $\hat{M}^{(t)}$ $(t \in [z])$ from z different co-clusterings, u and i are the targeted user and item, respectively.
Output: The predicted rating of user u on item i: $\bar{M}_{ui}$.

// Computing weights
for $t \in [z]$ **do**
    $Q_{ui}^{(t)} = q(\hat{M}_{ui}^{(t)})$
end for
return $\bar{M}_{ui} = \sum_{t=1}^{z} \frac{Q_{ui}^{(t)}}{\sum_{s=1}^{z} Q_{ui}^{(s)}} \hat{M}_{ui}^{(t)}$