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

Chao Chen*, Dongsheng Li†, Yingying Zhao*, Qin Lv*, Li Shang**

* Tongji University, China
† 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
  \[ O(kmn) \text{ per iteration for k-means clustering} \]

**Bregman co-clustering**
- Efficient and scalable
  \[ O(mkl + nkl) \text{ per iteration} \]
- Able to detect diverse inner structures
  \[ \text{Different distance function + constraint set} \implies \text{different co-clustering} \]
- Low-parameter structure of the generated submatrices
  \[ \text{Mostly uneven distribution of generated submatrices} \]

Matrix size: \( 4 \times 4 \)
Co-clustering size: \( 2 \times 2 \)
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

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

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

### Table: Rating Distribution and RMSE

<table>
<thead>
<tr>
<th>Rating</th>
<th>Distribution</th>
<th>RMSE without Weighting</th>
<th>RMSE with Weighting</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>17.44%</td>
<td>1.2512</td>
<td>1.2533</td>
</tr>
<tr>
<td>2</td>
<td>25.39%</td>
<td>0.6750</td>
<td>0.6651</td>
</tr>
<tr>
<td>3</td>
<td>35.35%</td>
<td>0.5260</td>
<td>0.5162</td>
</tr>
<tr>
<td>4</td>
<td>18.28%</td>
<td>1.1856</td>
<td>1.1793</td>
</tr>
<tr>
<td>5</td>
<td>3.54%</td>
<td>2.1477</td>
<td>2.1597</td>
</tr>
<tr>
<td>Overall accuracy</td>
<td></td>
<td>0.9517</td>
<td>0.9479</td>
</tr>
</tbody>
</table>

*Case study on synthetic dataset*
Step (3) – Ensemble of Local Models

Observations
- User rating distribution $$\rightarrow$$ User rating preferences
- Item rating distribution $$\rightarrow$$ Item quality

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

$$\tilde{M}_{ui} = \sum_t \frac{Q_{ui}(t)}{\sum_s Q_{ui}(s)} \tilde{M}_{ui}(t)$$

- Ensemble weight

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

<table>
<thead>
<tr>
<th>Model</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1 + 0.05 + 0.05 = 1.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.1 + 0.3 + 0.6 = 1.9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.1 + 0.5 + 0.2 = 1.7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$$\frac{1.1 \times 1 + 1.9 \times 5 + 1.7 \times 4}{1.1 + 1.9 + 1.7} = 3.70 > 3.33 = \frac{1 + 5 + 4}{3}$$
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 $M \in \mathbb{R}^{m \times n}$ 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

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

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

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

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

<table>
<thead>
<tr>
<th></th>
<th>MovieLens 1M</th>
<th>MovieLens 10M</th>
<th>Netflix</th>
</tr>
</thead>
<tbody>
<tr>
<td>#users</td>
<td>6,040</td>
<td>69,878</td>
<td>480,189</td>
</tr>
<tr>
<td>#items</td>
<td>3,706</td>
<td>10,677</td>
<td>17,770</td>
</tr>
<tr>
<td>#ratings</td>
<td>$10^6$</td>
<td>$10^7$</td>
<td>$10^8$</td>
</tr>
</tbody>
</table>

*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.

Optimal weighting parameter on uneven dataset is smaller than that on even dataset.

Rating Distribution of Three Synthetic Datasets

<table>
<thead>
<tr>
<th>Rating</th>
<th>D1 (uneven)</th>
<th>D2 (medium)</th>
<th>D3 (even)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.98%</td>
<td>3.44%</td>
<td>18.33%</td>
</tr>
<tr>
<td>2</td>
<td>3.14%</td>
<td>9.38%</td>
<td>26.10%</td>
</tr>
<tr>
<td>3</td>
<td>15.42%</td>
<td>29.25%</td>
<td>35.27%</td>
</tr>
<tr>
<td>4</td>
<td>40.98%</td>
<td>37.86%</td>
<td>16.88%</td>
</tr>
<tr>
<td>5</td>
<td>39.49%</td>
<td>20.06%</td>
<td>3.43%</td>
</tr>
</tbody>
</table>
Sensitivity Analysis – Ensemble Method

The 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

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

<table>
<thead>
<tr>
<th>Method</th>
<th>MovieLens 10M</th>
<th>Netflix</th>
</tr>
</thead>
<tbody>
<tr>
<td>NMF</td>
<td>0.8832 ± 0.0007</td>
<td>0.9396 ± 0.0002</td>
</tr>
<tr>
<td>RSVD</td>
<td>0.8253 ± 0.0009</td>
<td>0.8534 ± 0.0001</td>
</tr>
<tr>
<td>BPMF</td>
<td>0.8195 ± 0.0006</td>
<td>0.8420 ± 0.0003</td>
</tr>
<tr>
<td>APG</td>
<td>0.8098 ± 0.0005</td>
<td>0.8476 ± 0.0028</td>
</tr>
<tr>
<td>DFC</td>
<td>0.8064 ± 0.0006</td>
<td>0.8451 ± 0.0005</td>
</tr>
<tr>
<td>LLORMA</td>
<td>0.7851 ± 0.0007</td>
<td>0.8275 ± 0.0004</td>
</tr>
<tr>
<td><strong>WEMAREC</strong></td>
<td><strong>0.7769 ± 0.0004</strong></td>
<td><strong>0.8142 ± 0.0001</strong></td>
</tr>
</tbody>
</table>
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

The graph shows the relationship between the number of row and column clusters and the Recommendation Error (RMSE) and Execution Time. As the number of clusters increases, the RMSE generally increases, indicating a decrease in accuracy. Conversely, the execution time increases, suggesting a lower scalability. This illustrates the trade-off between accuracy and scalability in the context of clustering.
Detailed Implementation

Algorithm 1 Co-clustering-based Matrix Approximation

Input: All co-clustering submatrices $M^{(t)} \subseteq M$ ($t \in [k])$, rank $r$, learning rate $\nu$, regularization coefficient $\lambda$.

Output: Approximated user-item rating matrix $\hat{M}$.

1: for each $t \in \{1, \ldots, k\}$ in parallel do
2:     // Computing weights
3:     Compute the rating distribution on $F$ in $M^{(t)}$.
4:     for each observed entry $(u, i)$ in $M^{(t)}$ do
5:         $W_{ui} = p(x)$, if $M_{ui} = x$.
6:     end for
7:     // Updating model
8:     Initialize $U^{(t)} \in \mathbb{R}^{m \times r}, V^{(t)} \in \mathbb{R}^{n \times r}$ randomly
9:     while not converged do
10:        for each observed entry $(u, i)$ in $M^{(t)}$ do
11:            $\Delta_{ui} = M_{ui}^{(t)} - U_{u}^{(t)}(V_{i}^{(t)})^T$
12:            for each $z \in \{1, \ldots, r\}$ do
13:                $U_{uz}^{(t)} = U_{uz}^{(t)} + \nu \ast (\Delta_{ui} \ast V_{iz}^{(t)} \ast W_{ui} - \nu U_{uz}^{(t)})$
14:                $V_{iz}^{(t)} = V_{iz}^{(t)} + \nu \ast (\Delta_{ui} \ast U_{uz}^{(t)} \ast W_{ui} - \nu V_{iz}^{(t)})$
15:            end for
16:        end for
17:     end while
18: end for
19: for each $(u, i) \in [m] \times [n]$ do
20:     Locate $(u, i)$ in its corresponding submatrix and let the index of the submatrix be $\xi$.
21:     $\hat{M}_{ui} = U_{u}^{(\xi)}(V_{i}^{(\xi)})^T$
22: end for
23: 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 $i$: $\hat{M}_{ui}$.

1: // Computing weights
2: for $t \in [z]$ do
3:     $Q_{ui}^{(t)} = q(\hat{M}_{ui}^{(t)})$
4: end for
5: return $\hat{M}_{ui} = \sum_{i=1}^{z} \frac{Q_{ui}^{(t)}}{\sum_{i=1}^{z} Q_{ui}^{(t)}} \hat{M}_{ui}^{(t)}$