Goal-Directed Inductive Matrix Completion

Si Si
Department of Computer Science
University of Texas at Austin

KDD 2016 @ San Francisco

Joint work with Kai-Yang Chiang, Cho-Jui Hsieh, Nikhil Rao, and Inderjit S. Dhillon
Outline

1. Background: Matrix Completion
2. Background: Inductive Matrix Completion
3. Our Method: Goal-directed Inductive Matrix Completion
4. Applications of Our Method
Matrix Completion (MC)

- Goal: given *partially observed* entry set, estimate *missing* entries.
- Applications: recommender system, multi-label learning, etc.

**Rating Matrix**

<table>
<thead>
<tr>
<th>Users</th>
<th>Movie 1</th>
<th>Movie 2</th>
<th>Items</th>
<th>Movie 10</th>
<th>Movie 11</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>?</td>
<td>5</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>5</td>
<td>5</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>1</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>
Low-rank Matrix Factorization Approach

- Low-rank matrix factorization:

\[
\min_{W \in \mathbb{R}^{m \times k}, \ H \in \mathbb{R}^{n \times k}} \sum_{(i, j) \in \Omega} (A_{ij} - (WH^T)_{ij})^2 + \lambda \left( \|W\|_F^2 + \|H\|_F^2 \right),
\]

- \( \Omega = \{(i, j) \mid A_{ij} \text{ is observed}\} \).
- \( W \) and \( H \) represent the model.

\[ A \approx WH^T \]

\[ A_{ij} \approx w_i^T h_j \]
Challenges for MC

- What if we also have some explicit features?
  - movie type, cast, IMDb score, etc.
  - user’s demographic information, etc.
- What if we also need to make prediction for new items or users?
Inductive Matrix Completion [Jain and Dhillon, 2013]:

\[
\min_{W,H} \sum_{(i,j) \in \Omega} (A_{ij} - (XWH^T Y^T)_{ij})^2 + \lambda(\|W\|_F^2 + \|H\|_F^2),
\]

where \(X \in \mathbb{R}^{n \times d}\) and \(Y \in \mathbb{R}^{m \times k}\) are the features; \(A\) is a rating matrix.
Matrix Completion vs. Inductive Matrix Completion

- **Difference:**
  - use features or not?
  - can make prediction for new users and items or not?

Matrix Completion

\[ A_{ij} \approx w_i^T h_j \]

Inductive Matrix Completion

\[ A_{ij} \approx x_i^T W H^T y_j \]
Challenges for Inductive Matrix Completion

- **Inductive Matrix Completion:**
  \[
  \min_{W,H} \sum_{(i,j) \in \Omega} (A_{ij} - (XWH^T Y^T)_{ij})^2 + \lambda(\|W\|_F^2 + \|H\|_F^2).
  \]

- Features $X$ and $Y$ are important to the model.
- Necessary condition for exact recovery [Jain and Dhillon, 2013]: \(\text{Col}(A) \subseteq \text{Col}(X) \) and \(\text{Col}(A^T) \subseteq \text{Col}(Y)\)

Can we learn features that can benefit the model?
Our Solution: Goal-directed Inductive Matrix Completion

Key ideas for GIMC:
- **Non-linear** feature mapping.
  - \( X \rightarrow \varphi(X) \) using non-linear mapping.
  - approximate kernel mapping.
- Learn the model and non-linear mapping **simultaneously**.
  - learn model and features jointly in a framework.
  - alternating minimization.
Non-linear Mapping

- $X \rightarrow \varphi(X)$ using non-linear mapping.

$$\min_{W,H} \sum_{(i,j) \in \Omega} (A_{ij} - (\varphi(X)WH^T \varphi(Y)^T)_{ij})^2 + \lambda (\|W\|_F^2 + \|H\|_F^2).$$

- kernel mappings used as non-linear mapping.
Approximate Kernel Mappings

- **Approximate kernel mappings:**
  - Nyström feature mapping from Nyström approximation [Williams and Seeger, 2001].
  - Random feature mapping from random feature approximation [Rahimi and Recht, 2007].

- **Parameter $U$ controls the mapping:**
  - $U = \{u_1, \cdots, u_m, M\}$ for Nyström feature mapping.
  - $U$ is the projection direction for random feature mapping.

### Nyström Feature Mapping

$$
\begin{align*}
\phi(.) & \xrightarrow{U} \frac{1}{\sqrt{m}} \begin{bmatrix}
\cos(u_1^T x_i) \\
\vdots \\
\cos(u_m^T x_i) \\
\sin(u_1^T x_i) \\
\vdots \\
\sin(u_m^T x_i)
\end{bmatrix}
\end{align*}
$$

### Random Feature Mapping

$$
\begin{align*}
\phi(.) & \xrightarrow{U} \begin{bmatrix}
\phi(u_i) \\
\vdots \\
\phi(u_m)
\end{bmatrix}
\end{align*}
$$

$$
\begin{align*}
\phi_U(x) &= \begin{bmatrix}
\phi(u_1) \\
\vdots \\
\phi(u_m)
\end{bmatrix}
\end{align*}
$$
Our Goal-directed Inductive Matrix Completion framework:

\[
\min_{W,H,U_x,U_y} \sum_{(i,j) \in \Omega} (A_{ij} - (\varphi_{U_x}(X) WH^T \varphi_{U_y}(Y)^T)_{ij})^2 + \lambda(\|W\|_F^2 + \|H\|_F^2).
\]

- \(U_x\) and \(U_y\) are parameters for the non-linear feature mapping.
- \(W, H\) are the model.

Optimization: alternating minimization

- Fix feature parameters \(U_x\) and \(U_y\), solve for model \(W, H\):
  - using traditional IMC solver.
- Fix \(W, H\), solve for feature parameters \(U_x\) and \(U_y\):
  - gradient descent \(U^{t+1} \leftarrow U^t - \eta \frac{\partial L_{U,W,H}}{\partial U}\).
Goal-directed Inductive Matrix Completion

1. \( X \rightarrow \varphi_{Ux^0}(X) \) and \( Y \rightarrow \varphi_{Uy^0}(Y) \).
1. \( X \rightarrow \varphi_{Ux^0}(X) \) and \( Y \rightarrow \varphi_{Uy^0}(Y) \).
2. Solve for model \( W \) and \( H \) using traditional IMC.

\[
\min_{W,H} \sum_{(i,j)\in\Omega} (A_{ij} - (\varphi(X)WH^T \varphi(Y)^T)_{ij})^2 + \lambda(\|W\|_F^2 + \|H\|_F^2).
\]
1. \( X \rightarrow \varphi_{Ux^0}(X) \) and \( Y \rightarrow \varphi_{Uy^0}(Y) \).
2. Solve for model \( W \) and \( H \) using traditional IMC.
3. Solve for feature parameters \( U_x \) and \( U_y \) using gradient descent.
Goal-directed Inductive Matrix Completion

1. \( X \rightarrow \varphi_{Ux^0}(X) \) and \( Y \rightarrow \varphi_{Uy^0}(Y) \).
2. Solve for model \( W \) and \( H \) using traditional IMC.
3. Solve for feature parameters \( U_x \) and \( U_y \) using gradient descent.
4. Repeat step 2 and 3.

\[
A \approx \varphi_{Ux}(X)^T WH^T \varphi_{Uy}(Y)
\]
Comparison among MC, IMC, and GIMC

- Whether use features?
- Whether use non-linear features?
- Whether learn features?

Matrix completion
- Rating matrix → training → model
- $\mathbf{A} \approx \mathbf{WH}^T$

Inductive Matrix completion
- Rating matrix → training → model → $\mathbf{A} \approx \mathbf{XWH}^T\mathbf{Y}^T$

Goal-directed Inductive Matrix completion
- Rating matrix → training → model → $\mathbf{A} \approx \varphi_{Ux}(\mathbf{X})\mathbf{WH}^T\varphi_{Uy}(\mathbf{Y})^T$
Application 1: Toy Example

- **Synthetic data:**
  - $A = XY^T + N$; $X, Y, N$ drawn from Gaussian.
  - Measure relative approximation error $\frac{\|\hat{A} - A\|_F}{\|A\|_F}$.
  - Vary the number of observed entries from 5% to 30%.

- **Methods compared in the experiments:**
  1. MC: Matrix Completion.
  2. IMC: Inductive Matrix Completion.
  3. IMC-RFF: non-linear Inductive Matrix Completion.
  4. GIMC-LFF: our proposed method.
Application 1: Toy Example

- Observations:
  1. IMC is better than MC: features are useful.
  2. IMC-RFF is worse than IMC: directly using non-linear feature mapping might not be good.
  3. GIMC-LFF is the best: learning both model and features are beneficial.
Application 2: Multi-label Learning

Problem:
- $X \in \mathbb{R}^{n \times d}$: each $x_i$ is a $d$-dimensional feature of the item $i$.
- $A \in \mathbb{R}^{n \times L}$: $A_{ij} = \{0, 1\}$ represents whether item $i$ belongs to label $j$.
- Applications: image/video annotation; query/keyword suggestions.

multi-label Learning

GIMF with row features

X: feature matrix; A: label matrix
 Goal: Predict $X_t$'s labels
Algorithm:
- Learning model: $f(x) = \varphi_U(x)WH^T$, parameterized by $U$, $W$, $H$.
- Special case of GIMC: only has row features.
  - Fix $\varphi_{Uy}(Y) = I$.
  - $\Omega = \text{all } nL \text{ elements (fully observed)}$.

**multi-label Learning**

$$A \approx \varphi_U(X)WH^T$$

$X$: feature matrix; $A$: label matrix
$X_t$'s labels: $\varphi_U(x_t)WH^T$
Data Set Statistics

- Three state-of-the-art multi-label learning methods:
  - **LEML** [Yu et al. 2014]: an embedding based technique.
  - **FastXML** [Prabhu and Varma, 2014]: a random forest approach.
  - **SLEEC** [Bhatia et al. 2015]: an ensemble of local distance preserving embeddings.

Data set statistics

<table>
<thead>
<tr>
<th>Dataset</th>
<th># training samples</th>
<th># testing samples</th>
<th># features</th>
<th># labels</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delicious</td>
<td>12,920</td>
<td>3,185</td>
<td>500</td>
<td>983</td>
</tr>
<tr>
<td>NUS-WIDE</td>
<td>161,789</td>
<td>107,859</td>
<td>1,134</td>
<td>1,000</td>
</tr>
<tr>
<td>Delicious-large</td>
<td>196,606</td>
<td>100,095</td>
<td>782,585</td>
<td>205,443</td>
</tr>
</tbody>
</table>
Experimental Results

- Three state-of-the-art extreme multi-label learning methods:
  - LEML [Yu et al. 2014]: an embedding based technique.
  - FastXML [Prabhu and Varma, 2014]: a random forest approach.
  - SLEEC [Bhatia et al. 2015]: an ensemble of local distance preserving embeddings.

- Results (Precision @ 1, 3, 5):

<table>
<thead>
<tr>
<th></th>
<th>Delicious</th>
<th></th>
<th>NUS-WIDE</th>
<th></th>
<th>Delicious-large</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P1 (%)</td>
<td>P3 (%)</td>
<td>P5 (%)</td>
<td>P1 (%)</td>
<td>P3 (%)</td>
<td>P5 (%)</td>
</tr>
<tr>
<td>GIMC</td>
<td>71.40</td>
<td>65.16</td>
<td>59.79</td>
<td>22.49</td>
<td>17.40</td>
<td>14.70</td>
</tr>
<tr>
<td>SLEEC</td>
<td>68.38</td>
<td>61.50</td>
<td>56.35</td>
<td>17.67</td>
<td>14.20</td>
<td>12.07</td>
</tr>
<tr>
<td>FastXML</td>
<td>69.65</td>
<td>63.93</td>
<td>59.36</td>
<td>21.00</td>
<td>16.32</td>
<td>13.66</td>
</tr>
<tr>
<td>LEML</td>
<td>65.66</td>
<td>61.15</td>
<td>56.08</td>
<td>20.76</td>
<td>16.00</td>
<td>13.11</td>
</tr>
</tbody>
</table>

- On Delicious and NUS-WIDE, GIMC performs the best.
- On Delicious-large dataset, GIMC has similar accuracy with SLEEC, but takes much less time: 4,724 vs 25,289 seconds.
More Multi-label Learning Results

- Number of projection dimensions vs. precision:

(a) Delicious

(b) NUS-WIDE
Application 3: Semi-Supervised Clustering

- Goal: find a clustering of $n$ items given:
  - Feature matrix $X \in \mathbb{R}^{n \times d}$.
  - Pairwise constraints $\{A_{ij} \mid (i, j) \in \Omega\}$ describing similar/dissimilar pairs.

- Example: interactive image segmentation.
Application 3: Semi-Supervised Clustering

- Goal: find a clustering of $n$ items given:
  - Feature matrix $X \in \mathbb{R}^{n \times d}$.
  - Pairwise constraints $\{A_{ij} \mid (i,j) \in \Omega\}$ describing similar/dissimilar pairs.
- Example: interactive image segmentation.
Experimental Results

- **MCCC algorithm** [Yi et al. 13]:
  1. Learn a low rank similarity matrix $\bar{A}$ by conducting IMC on $\Omega$.
  2. Do $k$-means clustering on top-$k$ eigenvectors of $\bar{A}$.

- Propose: replace completion step (step 1) in MCCC with GIMC.

- Results on clustering error rate:

| Dataset  | $n$  | $d$  | $k$ | $\#$ constraints $|\Omega|$ | $k$-means | MCCC   | IMC-RFF | GIMC-LFF |
|----------|------|------|-----|-----------------------------|-----------|--------|---------|----------|
|          |      |      |     | $n$                         | 0.1433    | 0.0891 | 0.0683  | 0.0724   |
|          |      |      |     | $5n$                        | 0.1347    | 0.0800 | 0.0580  | 0.0570   |
| Segment  | 2319 | 19   | 7   |                             | 0.1363    | 0.0809 | 0.0650  | 0.0446   |
|          |      |      |     | $10n$                       | 0.1362    | 0.0872 | 0.0678  | 0.0402   |
|          |      |      |     | $15n$                       | 0.1330    | 0.0837 | 0.0564  | 0.0380   |
|          |      |      |     | $20n$                       | 0.2523    | 0.2498 | 0.1840  | 0.1898   |
|          |      |      |     |                             | 0.2112    | 0.1772 | 0.1930  | 0.1592   |
|          |      |      |     |                             | 0.2068    | 0.1708 | 0.1722  | 0.1388   |
|          |      |      |     |                             | 0.2203    | 0.1677 | 0.1687  | 0.1262   |
|          |      |      |     |                             | 0.2124    | 0.1607 | 0.1561  | 0.1078   |
| Covtype-sub | 1711 | 54   | 7   |                             | 0.2523    | 0.2498 | 0.1840  | 0.1898   |
|          |      |      |     |                             | 0.2112    | 0.1772 | 0.1930  | 0.1592   |
|          |      |      |     |                             | 0.2068    | 0.1708 | 0.1722  | 0.1388   |
|          |      |      |     |                             | 0.2203    | 0.1677 | 0.1687  | 0.1262   |
|          |      |      |     |                             | 0.2124    | 0.1607 | 0.1561  | 0.1078   |
Conclusions

- Matrix completion (MC) and its challenges
- Inductive matrix completion (IMC) and its challenges
- Goal-directed Inductive Matrix Completion (GIMC)
  - Non-linear feature mapping
  - Learning feature mapping and model simultaneously

- Applications
  - Multi-label learning
  - Semi-supervised clustering

Questions?
More multi-label Learning Results

- number of projection dimensions vs. prediction accuracy

(c) Delicious

(d) NUS-WIDE