FISM: Factored Item Similarity Models for Top-N Recommender Systems

Santosh Kabbur
Computer Science & Engineering, UMN

Present by Feng Xie

October 16, 2013
Outline

- Background
- Motivation
- FISM
  - Basic FISM
  - FISMrmse
  - FISMAuc
- Evaluation
- Conclusion
What is recommender system?
Information Overload
How to solve information overload?
Search Engine VS. Recommender System

- User will try search engine if
  - they have specific needs
  - they can use keywords to describe needs

- User will try recommender system if
  - they do not know what they want now
  - they can not use keywords to describe needs
Mission

• Help user find item of their interest
• Help item provider deliver their item to right user
• Help website improve user’s loyalty
Recommender System
Background

- Content filtering
  - Music Genome Project (Pandora.com)
- Collaborative filtering
  - Neighborhood methods
    - User based
    - Item based
  - Latent factor models
Collaborative Filtering

- User based
  - Users with similar history selections will share same future interest
- Item based
  - Users will like items similar to what they consumed before
The Process

Input (ratings table) → CF-Algorithm → Prediction

Item for which prediction is sought

P_{aj} (prediction on item j for the active user)

{T_{i1}, T_{i2}, ..., T_{in}} Top-N list of items for the active user

Output interface
**Item based CF**

Item-item similarity is computed by looking into co-rated items only. In the case of items $i$ and $j$ the similarity $s_{ij}$ is computed by looking into them. Note: each of these co-rated pairs are obtained from different users, in this example they come from users $1$, $u$ and $m-1$. 
Item based CF

Ranking of the items similar to the $i$-th item
\[ \tilde{r}_u = r_u S \]

where \( r_u \) is the rating vector of \( u \) on all items and \( S \) is a \( m \times m \) sparse matrix of aggregation coefficients.

\[
\min_{S} \frac{1}{2} \| R - RS \|_F^2 + \frac{\beta}{2} \| S \|_F^2 + \lambda \| S \|_1
\]

subject to \( S \geq 0 \), \( \text{diag}(S) = 0 \),
**NSVD**

In this method, an item-item similarity was learned as a product of two low-rank matrices, $P$ and $Q$, where $P \in \mathbb{R}^{m \times k}$, $Q \in \mathbb{R}^{m \times k}$, and $k \ll m$.

$$
\hat{r}_{ui} = b_u + b_i + \sum_{j \in \mathcal{R}_u^+} p_j q_i^T
$$

where $b_u$ and $b_i$ are the user and item biases and $\mathcal{R}_u^+$ is the set of items rated by $u$. The parameters of this model are estimated as the minimizer to the following optimization problem:

$$
\min_{P,Q} \frac{1}{2} \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{R}_u^+} \|r_{ui} - \hat{r}_{ui}\|_F^2 + \frac{\beta}{2} (\|P\|_F^2 + \|Q\|_F^2)
$$
Motivation

- Sparse user-item rating matrix results in Item based and SLIM, which rely on learning similarities between items, fail to capture the dependencies between items that have not been co-rated by at least one user.
- Methods based on matrix factorization, alleviate this problem by projecting the data onto a low dimensional space, thereby implicitly learning better relationships between the users and items (including items which are not co-rated). However, such methods are consistently out-performed by SLIM.
- NSVD does not exclude the diagonal entries while estimating the ratings during learning and prediction phases. In this case it can lead to rather trivial estimates, in which an item ends up recommending itself.
Basic FISM

Estimated value

\[ \tilde{r}_{ui} = b_u + b_i + (n_u^+)^{-\alpha} \sum_{j \in R_u^+} p_j q_i^T \]

where \( R_u^+ \) is the set of items rated by user \( u \), \( p_j \) and \( q_i \) are the learned item latent factors, \( n_u^+ \) is the number of items rated by \( u \), and \( \alpha \) is a user specified parameter between 0 and 1.
FISMrmse

Loss function: RMSE

\[ \mathcal{L}(\cdot) = \sum_{i \in \mathcal{D}} \sum_{u \in \mathcal{C}} (r_{ui} - \hat{r}_{ui})^2 \]

Estimated value

\[ \hat{r}_{ui} = b_u + b_i + (n_u^+ - 1)^{-\alpha} \sum_{j \in \mathcal{R}_u^+ \setminus \{i\}} p_j q_i^T \]

Regularized optimization problem

\[
\begin{aligned}
\text{minimize}_{P, Q} & \quad \frac{1}{2} \sum_{u, i \in \mathcal{R}} \| r_{ui} - \hat{r}_{ui} \|_F^2 + \frac{\beta}{2} (\| P \|_F^2 + \| Q \|_F^2) \\
& \quad + \frac{\lambda}{2} \| b_u \|_2^2 + \frac{\gamma}{2} \| b_i \|_2^2
\end{aligned}
\]
Algorithm 1 FISMrmse:Learn

1: procedure FISMrmse_LEARN
2: \( \eta \leftarrow \) learning rate
3: \( \beta \leftarrow \ell_F \) regularization weight
4: \( \rho \leftarrow \) sample factor
5: \( \text{iter} \leftarrow 0 \)
6: Init \( P \) and \( Q \) with random values in \((-0.001, 0.001)\)
7:
8: while \( \text{iter} < \text{maxIter} \) or error on validation set decreases do
9: \( \mathcal{R}' \leftarrow \mathcal{R} \cup \text{SampleZeros}(\mathcal{R}, \rho) \)
10: \( \mathcal{R}' \leftarrow \text{RandomShuffle}(\mathcal{R}') \)
11:
12: for all \( r_{ui} \in \mathcal{R}' \) do
13: \( x \leftarrow (n_u^+ - 1)^{-\alpha} \sum_{j \in \mathcal{R}_u^+ \setminus \{i\}} p_j \)
14: \( \tilde{r}_{ui} \leftarrow b_u + b_i + q_i^T x \)
15: \( e_{ui} \leftarrow r_{ui} - \tilde{r}_{ui} \)
16: \( b_u \leftarrow b_u + \eta \cdot (e_{ui} - \lambda \cdot b_u) \)
17: \( b_i \leftarrow b_i + \eta \cdot (e_{ui} - \gamma \cdot b_i) \)
18: \( q_i \leftarrow q_i + \eta \cdot (e_{ui} \cdot x - \beta \cdot q_i) \)
19: for all \( j \in \mathcal{R}_u^+ \setminus \{i\} \) do
20: \( p_j \leftarrow p_j + \eta \cdot (e_{ui} \cdot (n_u^+ - 1)^{-\alpha} \cdot q_i - \beta \cdot p_j) \)
21: end for
22: end for
23: \( \text{iter} \leftarrow \text{iter} + 1 \)
24: end while
25: return \( P, Q \)
26: end procedure
FISMauc

Loss function: AUC

Given user’s rated items in $\mathcal{R}_u^+$ and unrated items in $\mathcal{R}_u^-$

$$\mathcal{L}(\cdot) = \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{R}_u^+, j \in \mathcal{R}_u^-} ((r_{ui} - r_{uj}) - (\hat{r}_{ui} - \hat{r}_{uj}))^2$$

Estimated value

$$\hat{r}_{ui} = b_u + b_i + (n_u^+ - 1)^{-\alpha} \sum_{j \in \mathcal{R}_u^+ \setminus \{i\}} p_j q_i^T$$

Regularized optimization problem

$$\minimize_{P, Q} \frac{1}{2} \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{R}_u^+, j \in \mathcal{R}_u^-} \left\| (r_{ui} - r_{uj}) - (\hat{r}_{ui} - \hat{r}_{uj}) \right\|_F^2 + \frac{\beta}{2} (\|P\|_F^2 + \|Q\|_F^2) + \frac{\gamma}{2} (\|b_i\|_2^2)$$
Algorithm 2 FiSMauc:Learn.

1: procedure FiSMauc\_Learn
2: \(\eta \gets\) learning rate
3: \(\beta \gets \ell_F\) regularization weight
4: \(\rho \gets\) number of sampled zeros
5: \(\text{iter} \gets 0\)
6: Init \(P\) and \(Q\) with random values in \((-0.001, 0.001)\)
7: \(\text{while iter} < \text{maxIter}\) or error on validation set decreases do
8: \(\text{for all} \ u \in C \ \text{do}\)
9: \(\text{for all} \ i \in \mathcal{R}_u^+ \ \text{do}\)
10: \(x \gets 0\)
11: \(t \gets (n_u^+ - 1)^{-\alpha} \sum_{j \in \mathcal{R}_u^+ \setminus \{i\}} p_j\)
12: \(\mathcal{Z} \gets \text{SampleZeros}(\rho)\)
13: \(\text{for all} \ j \in \mathcal{Z} \ \text{do}\)
14: \(\hat{r}_{ui} \gets b_i + t \cdot q_i^T\)
15: \(\hat{r}_{uj} \gets b_j + t \cdot q_j^T\)
16: \(r_{uj} \gets 0\)
17: \(e \gets (r_{ui} - r_{uj}) - (\hat{r}_{ui} - \hat{r}_{uj})\)
18: \(b_i \gets b_i + \eta \cdot (e - \gamma \cdot b_i)\)
19: \(b_j \gets b_j - \eta \cdot (e - \gamma \cdot b_j)\)
20: \(q_i \gets q_i + \eta \cdot (e \cdot t - \beta \cdot q_i)\)
21: \(q_j \gets q_j - \eta \cdot (e \cdot t - \beta \cdot q_j)\)
22: \(x \gets x + e \cdot (q_i - q_j)\)
23: \(\text{end for}\)
24: \(\text{end for}\)
25: \(\text{for all} \ j \in \mathcal{R}_u^+ \setminus \{i\} \ \text{do}\)
26: \(p_j \gets p_j + \eta \cdot \left(\frac{1}{\rho} \cdot (n_u^+ - 1)^{-\alpha} \cdot x - \beta \cdot p_j\right)\)
27: \(\text{end for}\)
28: \(\text{end for}\)
29: \(\text{iter} \gets \text{iter} + 1\)
30: \(\text{end while}\)
31: \(\text{return} \ P, Q\)
32: \(\text{end procedure}\)
Evaluation

- Data set
  - ML100K: ML100K-1, ML100K-2, ML100K-3
  - Netflix: Netflix-1, Netflix-2, Netflix-3
  - Yahoo Music: Yahoo-1, Yahoo-2, Yahoo-3
  - Only ML100K-3, Netflix-3, Yahoo-2 are used

<table>
<thead>
<tr>
<th>Table 1: Datasets.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset</td>
</tr>
<tr>
<td>---------</td>
</tr>
<tr>
<td>ML100K-1</td>
</tr>
<tr>
<td>ML100K-2</td>
</tr>
<tr>
<td>ML100K-3</td>
</tr>
<tr>
<td>Netflix-1</td>
</tr>
<tr>
<td>Netflix-2</td>
</tr>
<tr>
<td>Netflix-3</td>
</tr>
<tr>
<td>Yahoo-1</td>
</tr>
<tr>
<td>Yahoo-2</td>
</tr>
<tr>
<td>Yahoo-3</td>
</tr>
</tbody>
</table>
Evaluation

- Methodology
  - 5-fold Leave-One-Out-Cross-Validation (LOOCV)

- Metrics
  - HR (Hit Rate)
    \[ HR = \frac{\# \text{hits}}{\# \text{users}} \]
  - ARHR
    \[ ARHR = \frac{1}{\# \text{users}} \sum_{i=1}^{\# \text{hits}} \frac{1}{\text{pos}_i} \]
Evaluation

• Comparison Algorithms
  • ItemKNN(cos)
  • ItemKNN(cprob)
  • ItemKNN(log)
  • PursSVD
  • BPRkNN
  • BPRMF
  • SLIM
  • Basic FISM
  • FISMrmse
  • FISMauc
Experimental Results

\[ \hat{r}_{ui} = b_u + b_i + (n_u^+ - 1)^{-\alpha} \sum_{j \in R_u^+ \setminus \{i\}} p_j q_i^T \]

<table>
<thead>
<tr>
<th>Scheme</th>
<th>ML100K</th>
<th></th>
<th>Yahoo</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Beta</td>
<td>Lambda</td>
<td>Gamma</td>
<td>HR</td>
</tr>
<tr>
<td>NoBias</td>
<td>8e-4</td>
<td>-</td>
<td>-</td>
<td>0.1281</td>
</tr>
<tr>
<td>UserBias</td>
<td>6e-4</td>
<td>0.1</td>
<td>-</td>
<td>0.1336</td>
</tr>
<tr>
<td>ItemBias</td>
<td>2e-4</td>
<td>-</td>
<td>0.01</td>
<td>0.1401</td>
</tr>
<tr>
<td>User&amp;ItemBias</td>
<td>6e-4</td>
<td>0.1</td>
<td>1e-4</td>
<td>0.1090</td>
</tr>
</tbody>
</table>
Experimental Results

\[ \hat{r}_{ui} = b_u + b_i + (n_i - 1)^{-\alpha} \sum_{j \in R_u \setminus \{i\}} p_j q_i^T \]

Figure 1: Effect of neighborhood agreement on performance.
Experimental Results

\[ S = \hat{P}Q^T \]

Figure 2: Performance of induced Sparsity on S.
Experimental Results

(a) ML100K dataset.

(b) Netflix dataset.

(c) Yahoo dataset.

Figure 3: Effect of estimation approach on performance.
Experimental Results

Figure 4: Non-negative and negative entries in S.
Experimental Results

Figure 5: Performance for different values of $N$. 
# Experimental Results

Table 3: Comparison of performance of top-$N$ recommendation algorithms with FISM.

<table>
<thead>
<tr>
<th>Method</th>
<th>ML100K-1</th>
<th></th>
<th></th>
<th>ML100K-2</th>
<th></th>
<th></th>
<th>ML100K-3</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Params</td>
<td>HR</td>
<td>ARHR</td>
<td></td>
<td>HR</td>
<td>ARHR</td>
<td></td>
<td>HR</td>
<td>ARHR</td>
</tr>
<tr>
<td>ItemKNN (cos)</td>
<td>100</td>
<td></td>
<td></td>
<td>100</td>
<td></td>
<td></td>
<td>100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ItemKNN (log)</td>
<td>100</td>
<td></td>
<td></td>
<td>100</td>
<td></td>
<td></td>
<td>100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ItemKNN (cprob)</td>
<td>500</td>
<td>0.6</td>
<td>0.1604</td>
<td>0.1604</td>
<td>0.0578</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PureSVD</td>
<td>10</td>
<td></td>
<td>0.1700</td>
<td></td>
<td></td>
<td>0.1700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BPRkNN</td>
<td>1e-4</td>
<td>0.01</td>
<td>0.1621</td>
<td></td>
<td></td>
<td>0.1621</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BPRMF</td>
<td>400</td>
<td>0.1</td>
<td>0.1711</td>
<td></td>
<td></td>
<td>0.1711</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLIM</td>
<td>0.1</td>
<td>20</td>
<td>0.1782</td>
<td></td>
<td>0.1782</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FISMrmse</td>
<td>96</td>
<td>2e-5</td>
<td>0.001</td>
<td>0.1908</td>
<td></td>
<td>0.1908</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FISMauc</td>
<td>64</td>
<td>0.001</td>
<td>1e-4</td>
<td>0.1518</td>
<td></td>
<td>0.1518</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Method          | Netflix-1 |          |          | Netflix-2 |          |          | Netflix-3 |          |          |
|                 | Params    | HR       | ARHR     |          | HR       | ARHR     |          | HR       | ARHR     |
| ItemKNN (cos)   | 100       |          |          | 100       |          |          | 100       |          |          |
| ItemKNN (log)   | 100       |          |          | 100       |          |          | 100       |          |          |
| ItemKNN (cprob) | 20       | 0.5      | 0.1555   |          | 0.1555   | 0.0678   |          |          |          |
| PureSVD         | 600       |          | 0.1783   |          |          | 0.1783   |          |          |          |
| BPRkNN          | 1e-3     | 1e-4     | 0.1678   |          | 0.1678   | 0.0781   |          |          |          |
| BPRMF           | 800      | 0.1      | 0.1638   |          | 0.1638   | 0.0719   |          |          |          |
| SLIM            | 1e-3     | 8        | 0.2025   | 0.2025   | 0.1008   |          |          |          |          |
| FISMrmse        | 192      | 2e-5     | 0.001    | 0.2118   |          | 0.2118   |          |          |          |
| FISMauc         | 192      | 1e-5     | 1e-4     | 0.2095   |          | 0.2095   |          |          |          |

| Method          | Yahoo-1  |          |          | Yahoo-2  |          |          | Yahoo-3  |          |          |
|                 | Params    | HR       | ARHR     |          | HR       | ARHR     |          | HR       | ARHR     |
| ItemKNN (cos)   | 100       |          |          | 100       |          |          | 100       |          |          |
| ItemKNN (log)   | 100       |          |          | 100       |          |          | 100       |          |          |
| ItemKNN (cprob) | 500      | 0.6      | 0.1387   |          | 0.1387   | 0.0510   |          |          |          |
| PureSVD         | 50        |          | 0.1229   |          | 0.1229   | 0.0459   |          |          |          |
| BPRkNN          | 1e-3     | 1e-4     | 0.1432   |          | 0.1432   | 0.0528   |          |          |          |
| BPRMF           | 700      | 0.1      | 0.1337   |          | 0.1337   | 0.0473   |          |          |          |
| SLIM            | 0.1      | 12       | 0.1454   |          | 0.1454   | 0.0542   |          |          |          |
| FISMrmse        | 192      | 1e-4     | 0.001    | 0.1522   |          | 0.1522   |          |          |          |
| FISMauc         | 144      | 8e-5     | 1e-4     | 0.1426   |          | 0.1426   |          |          |          |
Experimental Results

![Bar Chart]

Figure 6: Effect of sparsity on performance for various datasets.
Conclusion

- This paper proposed a factored item similarity based method (FISM), which learns the item similarities as the product of two matrices.
- FISM can well cope with data sparsity problem, and better estimators are achieved as the number of factors increases.
- FISM outperforms other state-of-the-art top-N recommendation algorithms.
Discussion

- Why not use RMSE as the metric?
- NSVD doesn’t use unrated entries for learning, while FISM does. The comparison seems bias?
- Is there a symmetric method to factor user similarity with two low dimensional matrices?
Questions?
Thank you!