PushTrust: An Efficient Recommendation Algorithm by Leveraging Trust and Distrust Relations

Rana Forsati, Iman Barjasteh, Farzan Masrour, Abdol-Hossein Esfahanian, Hayder Radha

Michigan State University

RecSys’15
Outline

• Some background
• Problem statement
• Our Algorithm
• Results
Recommender Systems

Recommend **items** to **users** to maximize some **objective(s)**

**Latent feature models (e.g., matrix factorization):** The current most successful technique as demonstrated in KDD Cup and Netflix competition
Latent Feature Models

**Items:** \( \mathcal{I} = \{i_1, i_2, \ldots, i_m\} \)

**Users:** \( \mathcal{U} = \{u_1, u_2, \ldots, u_n\} \)

**Partially observed rating matrix:** \( \mathbf{R} \in \{1, \ldots, 5, ?\}^{n \times m} \)

**Intuition:** ratings are deeply influenced by a set of **non-obvious features** specific to the domain

**Goal:** Extract **latent features** from existing ratings for future (unknown) predictions
Latent Features Models...

Assume there are $k$ latent features for rating

$$\mathbf{u}_i \in \mathbb{R}^k \quad \mathbf{v}_j \in \mathbb{R}^k$$

$$\begin{pmatrix}
3.33 \\
0.61 \\
\vdots \\
2.31 \\
1
\end{pmatrix} \quad \begin{pmatrix}
0.43 \\
0.77 \\
\vdots \\
1.45 \\
0
\end{pmatrix}$$

$$R_{i,j} \approx \mathbf{u}_i^\top \mathbf{v}_j$$

How to find latent features for users and items?
Matrix Factorization

Model each user/item as a vector of features (learned from data)

\[ \mathbf{R} \in \mathbb{R}^{n \times m} \quad \approx \quad \mathbf{U} \in \mathbb{R}^{n \times k} \times \mathbf{V}^\top \in \mathbb{R}^{k \times m} \]

\[ \Omega_R : \text{observed ratings} \]

Solve the following optimization problem:

\[ \mathcal{F}(\mathbf{U}, \mathbf{V}) = \frac{1}{2} \sum_{(i,j) \in \Omega_R} \left( \mathbf{R}_{i,j} - \mathbf{u}_i^\top \mathbf{v}_j \right)^2 + \lambda_U \| \mathbf{U} \|_F^2 + \lambda_V \| \mathbf{V} \|_F^2 \]

- Training error on observed ratings
- Regularization to avoid overfitting
Challenges

**Sparsity** of rating matrix

Example: Epinions only 0.02% of matrix is observed.

Handling **cold-start** users or items

- **Cold-start users**: new users who have rated only a few items
- **Cold-start items**: new items without ratings

Huge number of unrated movies
**Side Information**

**Goal:** exploit other sources of information to cope with these challenges

---

**Rating Matrix**

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>?</td>
<td>3</td>
<td>3</td>
<td>?</td>
</tr>
<tr>
<td>?</td>
<td>2</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>5</td>
<td>?</td>
<td>?</td>
<td>3</td>
<td>?</td>
</tr>
</tbody>
</table>

---

- **Side Information about Items**
- **Side Information about Ratings**
- **Side Information about Users**
Side Information about Users

- Trust/Distrust Relations between users

- Attribute of users such as location, age,....
Social Information and Data Sparsity

Adding *side information* to resolve sparsity and cold-start problems.

Our focus: *trust/distrust* relations between users

**Trust** = agreement on ratings  
[Guo et al., 2014]

**Distrust** = disagreement on ratings  
[Forsati et al., 2014]

Growth of social networks

Opinion Aware Recommendation Systems

How to effectively exploit social relations of users in recommendation to boost the accuracy?
Matrix Factorization & PushTrust Algorithm
Trust versus Distrust

- **Trust** can be considered as a **transitive** relation and can propagate.

- **Distrust** is **not transitive**.

- **Distrust** propagates only one step [Guha et al., 2004]

- **Distrust** cannot be considered as negative of **trust**.
Social Regularization with Trust and Distrust

### Rating Matrix

<table>
<thead>
<tr>
<th></th>
<th>i1</th>
<th>i2</th>
<th>i3</th>
<th>i4</th>
<th>i5</th>
<th>i6</th>
</tr>
</thead>
<tbody>
<tr>
<td>u1</td>
<td>4</td>
<td>?</td>
<td>3</td>
<td>?</td>
<td>5</td>
<td>?</td>
</tr>
<tr>
<td>u2</td>
<td>?</td>
<td>2</td>
<td>?</td>
<td>?</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>u3</td>
<td>?</td>
<td>?</td>
<td>1</td>
<td>?</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>u4</td>
<td>?</td>
<td>?</td>
<td>3</td>
<td>?</td>
<td>?</td>
<td>1</td>
</tr>
<tr>
<td>u5</td>
<td>1</td>
<td>4</td>
<td>?</td>
<td>?</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>u6</td>
<td>5</td>
<td>?</td>
<td>2</td>
<td>1</td>
<td>?</td>
<td>4</td>
</tr>
<tr>
<td>u7</td>
<td>?</td>
<td>2</td>
<td>3</td>
<td>?</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

### Trust Network

### Distrust Network

- **Memory based methods:** distrust is used to either filter out or debug propagated trust network [Victor et al., 2011]

- **Factorization based methods:** regularization by ranking of latent features [Forsati et al., 2014]
Social Regularization with Trust and Distrust

Trust+Distrust enhanced MF: the latent features of trusted users are closer and distrusted users are more distant

trust network

Distrust network

users arranged based on similarity of their latent features
\[ F(U, V) = \frac{1}{2} \sum_{(i,j) \in \Omega_R} (R_{i,j} - u_i^T v_j)^2 + \lambda_U \|U\|_F^2 + \lambda_V \|V\|_F^2 + \frac{\lambda_S}{2} \sum_{(i,j,k) \in \Omega_S} \max(0, 1 - \|u_i - u_j\|^2 + \|u_i - u_k\|^2) \]

where \( \Omega_S = \{(i, j, k) \in [n] \times [n] \times [n] : S_{ij} = 1 \ \& \ S_{ik} = -1\} \).
Issues with Ranking

**Computationally expensive:** in large social graphs, optimization cost features can increase cubically in the number of users $O(n^3)$.

- Not scalable to large social networks!

The top portion of ranked list might include distrusted friends due to nature of pairwise ranking model.

- The latent features might affected by distrusted friends who appear at top of the ranked list!

Only consider the trusted and distrusted friends and ignores the neutral (users with no relation) users.

- The neutral friends might appear before the trust friends and be negatively influential!
PushTrust

A more complete approach:

- The **trustees** are PUSHED to the **top** of list as much as possible.

- The **foes** are PUSHED to the **bottom** of list as much as possible.

- The **neutral** friends are PUSHED to the **middle** of list.
PushTrust: A Convex Formulation

Rewrite the social-regularized matrix factorization

\[ F(U, V) = \frac{1}{2} \sum_{(i, j) \in \Omega_R} (R_{i, j} - u_i^T v_j)^2 + \lambda_U \|U\|_F^2 + \lambda_V \|V\|_F^2 + \lambda_s \sum_{i=1}^{n} \mathcal{P}(u_i) \]

Here \( \mathcal{P} : \mathbb{R}^k \rightarrow \mathbb{R}_+ \) is social regularization of latent features of individual users.
**PushTrust: A Convex Formulation**

$\mathcal{U}^+ :$ latent features of **trusted** friends

$\mathcal{U}^- :$ latent features of **distrusted** friends

$\mathcal{U}^\circ :$ latent features of **neutral** friends

$$
\mathcal{P}(u) = \frac{1}{p} \sum_{i=1}^{p} \mathbb{I} \left[ \langle u, u^+_i \rangle \leq \max_{1 \leq j \leq q} \langle u, u^-_j \rangle \right] 
+ \frac{1}{p} \sum_{i=1}^{p} \mathbb{I} \left[ \langle u, u^+_i \rangle \leq \max_{1 \leq j \leq r} \langle u, u^\circ_j \rangle \right] 
+ \frac{1}{r} \sum_{i=1}^{r} \mathbb{I} \left[ \langle u, u^\circ_i \rangle \leq \max_{1 \leq j \leq q} \langle u, u^-_j \rangle \right]
$$

where $\mathbb{I}[\cdot]$ is indicator function.

$$
\langle u, v \rangle = u^T v = \sum_{i=1}^{k} u_i v_i
$$

**Put trusted friends above distrusted ones**

**Put trusted friends above neutral ones**

**Put neutral friends above distrusted ones**

**non-convex indicator function**
Convex Loss Function

Indicator function:

\[ P(u) = \frac{1}{p} \sum_{i=1}^{p} \left[ \langle u, u_i^+ \rangle \leq \max_{1 \leq j \leq q} \langle u, u_j^- \rangle \right] \]

\[ + \frac{1}{p} \sum_{i=1}^{p} \left[ \langle u, u_i^+ \rangle \leq \max_{1 \leq j \leq r} \langle u, u_j^o \rangle \right] \]

\[ + \frac{1}{r} \sum_{i=1}^{r} \left[ \langle u, u_i^o \rangle \leq \max_{1 \leq j \leq q} \langle u, u_j^- \rangle \right] \]

\[ \mathbb{I}[x] := \begin{cases} 1 & \text{if } x \in A, \\ 0 & \text{if } x \notin A. \end{cases} \]
Convex Loss Function

Hinge loss:

\[
P(u) = \frac{1}{p} \sum_{i=1}^{p} \ell \left( \langle u, u_i^+ \rangle - \max_{1 \leq j \leq q} \langle u, u_j^- \rangle \right) \\
+ \frac{1}{p} \sum_{i=1}^{p} \ell \left( \langle u, u_i^+ \rangle - \max_{1 \leq j \leq r} \langle u, u_j^o \rangle \right) \\
+ \frac{1}{r} \sum_{i=1}^{r} \ell \left( \langle u, u_i^o \rangle - \max_{1 \leq j \leq q} \langle u, u_j^- \rangle \right)
\]

\[
\ell(x) = \max(0, 1 - x)
\]
Rewrite the social regularized matrix factorization

\[ F(U, V) = \frac{1}{2} \sum_{(i, j) \in \Omega_R} (R_{i,j} - u_i^T v_j)^2 + \frac{\lambda_V}{2} \| V \|_F + \frac{\lambda_U}{2} \| U \|_F^2 \]

\[ + \lambda_s \sum_{i=1}^{n} \left( \frac{1}{p} \sum_{j \in N^+(i)} \ell \left( \langle u_i, u_{i,j}^+ \rangle - \| U_i^- u_i \|_\infty \right) \right) \]

\[ + \lambda_s \sum_{i=1}^{n} \left( \frac{1}{p} \sum_{j \in N^+(i)} \ell \left( \langle u_i, u_{i,j}^\circ \rangle - \| U_i^\circ u_i \|_\infty \right) \right) \]

\[ + \lambda_s \sum_{i=1}^{n} \left( \frac{1}{r} \sum_{j \in N^\circ(i)} \ell \left( \langle u_i, u_{i,j}^\circ \rangle - \| U_i^- u_i \|_\infty \right) \right) , \]

where

\[ U_i^- = [u_{i,1}^-, \cdots, u_{i,q}^+]^T \]

\[ U_i^\circ = [u_{i,1}^\circ, \cdots, u_{i,r}^\circ]^T \]
Optimization Procedure

The problem is non-convex jointly in both U and V.

Solution: The standard gradient descent method
1. Updating \( U \) while keeping \( V \) fixed:

Set \( \mathbf{u}_{i}^{t+1} = \mathbf{u}_{i}^{t} - \eta_{t} \mathbf{g}_{\mathbf{u}_{i}}^{t} \), where

\[
\mathbf{g}_{\mathbf{u}_{i}} = \frac{\partial \mathcal{F}}{\partial \mathbf{u}_{i}} = \sum_{j=1}^{m} \mathbf{I}_{ij}(\mathbf{R}_{ij} - \mathbf{u}_{i}^{\top} \mathbf{v}_{j})\mathbf{v}_{j} + \lambda_{U} \mathbf{u}_{i}
\]

\[
+ \frac{\lambda_{S}}{p} \sum_{j=1}^{p} \mathbb{I} \left[ \langle \mathbf{u}_{i}, \mathbf{u}_{i,j}^{+} \rangle - \| \mathbf{U}_{i}^{-} \mathbf{u}_{i} \|_{\infty} \leq 1 \right] \left( \partial \| \mathbf{U}_{i}^{-} \mathbf{u}_{i} \|_{\infty} - \mathbf{u}_{i,j}^{+} \right)
\]

\[
+ \frac{\lambda_{s}}{p} \sum_{j=1}^{p} \mathbb{I} \left[ \langle \mathbf{u}_{i}, \mathbf{u}_{i,j}^{+} \rangle - \| \mathbf{U}_{i}^{o} \mathbf{u}_{i} \|_{\infty} \leq 1 \right] \left( \partial \| \mathbf{U}_{i}^{o} \mathbf{u}_{i} \|_{\infty} - \mathbf{u}_{i,j}^{+} \right)
\]

\[
+ \frac{\lambda_{S}}{r} \sum_{j=1}^{r} \mathbb{I} \left[ \langle \mathbf{u}_{i}, \mathbf{u}_{i,j}^{o} \rangle - \| \mathbf{U}_{i}^{-} \mathbf{u}_{i} \|_{\infty} \leq 1 \right] \left( \partial \| \mathbf{U}_{i}^{-} \mathbf{u}_{i} \|_{\infty} - \mathbf{u}_{i,j}^{o} \right)
\]
2. Updating $V$ while keeping $V$ fixed:

Set $v_{i}^{t+1} = v_{i}^{t} - \eta_{t}g_{v_{i}}^{t}$, where

$$g_{v_{j}} = \frac{\partial F}{\partial v_{i}} = \sum_{i=1}^{n} I_{ij} (R_{ij} - u_{i}^{\top} v_{j}) u_{i} + \lambda v v_{j}.$$
Conventional: The neutral friends of users are also incorporated in ranking the latent features.

Computational: The number of constraints increases quadratically $O(n^2)$
Experiments
Experimental Evaluation

Two evaluation metrics:

Root Mean Squared Error:

$$\text{RMSE} = \sqrt{\frac{\sum_{(i,j) \in \mathcal{T}} (R_{ij} - \hat{R}_{ij})^2}{|\mathcal{T}|}}$$

Mean Absolute Error:

$$\text{MAE} = \frac{1}{|\mathcal{T}|} \sum_{(i,j) \in \mathcal{T}} |R_{ij} - \hat{R}_{ij}|,$$

where $\mathcal{T}$ is the set of rating that should be predicted.
The Epinions Dataset

- The only social rating network dataset publicly available.
- User trust and distrust information is included in this dataset.
- The social network in Epinions is directed.

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Users</th>
<th>Items</th>
<th>Ratings</th>
<th>Trust Relations</th>
<th>Distrust Relations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>121,240</td>
<td>685,621</td>
<td>12,721,437</td>
<td>481,799</td>
<td>96,823</td>
</tr>
</tbody>
</table>

80% of the rating data was selected as the training set with 20% as the test data.
Baseline Algorithms & Experimental Design

- **MF**: Matrix Factorization
- **MF-T**: Matrix Factorization with Trust information
- **MF-D**: Matrix Factorization with Distrust information
- **MF-DT**: Matrix Factorization with Trust and Distrust
- **PushTrust**: The proposed algorithm

![Diagram showing performance metrics (RMSE and MAE) across different latent vector dimensions for various algorithms. The x-axis represents latent vector dimensions (5 and 10). The y-axis represents the metric values. The diagram compares PushTrust, MF, MF-T, MF-D, and MF-DT across these dimensions.]
Experiment on handling cold-start users

To evaluate different algorithms:

- select 30%, 20%, and 10% of the users, to be cold-start users.
- For cold-start users, no rating is included in the training data.
- consider all ratings made by cold-start users as testing data.
Thank You for your attention!

Questions?