Trust-aware Recommender Systems

Paolo Massa and Paolo Avesani
University of Trento / ITC irsT - Italy
Summary

1. Recommender Systems (RSs)
   – Weaknesses
2. Solution: trust-awareness
   – Trust and trust metrics
3. Experiments on Epinions.com
   – Evidence trust-awareness improves RSs
   – (~50.000 users!)
4. Future works
Collaborative Filtering (CF)

RSs suggest to users items they may like (books, movies, songs, ...)

CF Input: ratings given by users to items

Example: I like “Titanic” as 4/5

1. I ask recommendation

2. RS computes the similarity of me against every other user
   - Pearson correlation coefficient

3. RS finds similar users and suggests to me items liked by them.
It does not consider the content of the items, only the ratings given by users.
It works independently of the domain (also jokes)

**BUT**

Overlapping of rated items required!

<table>
<thead>
<tr>
<th></th>
<th>Item1</th>
<th>Item2</th>
<th>Item3</th>
<th>Item4</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ME</strong></td>
<td>2</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>User2</td>
<td>5</td>
<td>1</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>User3</td>
<td>5</td>
<td>5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>User4</td>
<td>2</td>
<td>5</td>
<td>5</td>
<td>4</td>
</tr>
</tbody>
</table>

Sim(ME, User2) = -0.2
Sim(ME, User3) = -0.4
Sim(ME, User4) = +0.9
RSs weaknesses

1. User Similarity often not computable
   - Ratings Matrix sparseness (95-99%) -> Low or No overlapping

2. Cold start
   - New users have 0 ratings (-) not comparable

3. Easy Attacks by Malicious Users
   - Copy profile and become the most similar
   - Even easier on the Semantic Web

4. Hard to understand and control
   - Black box (bad recs -> user gives up)

Solution? Trust-awareness!
Trust-awareness

Trust = explicit rating of a user on another user about the perceived quality of the user's characteristics (in RSs about ratings quality)

E-marketplaces: Ebay.com, Epinions.com, Amazon.com
News sites: Slashdot.org, Kuro5hin.org
P2P networks: eDonkey, Gnutella, JXTA
Jobs sites: LinkedIn, Ryze
Opensource Developers communities: Advogato.org
Network of personal weblogs (some millions blogrolls!)
Semantic Web: FOAF (Friend-Of-A-Friend) is an RDF format that allows to express social relationships (~10 millions files)
Trust networks

Aggregate all the trust statements to produce a trust network.

A node is a user.
A direct edge is a trust statement.

Properties of Trust:
- weighted (0=distrust, 1=max trust)
- subjective
- asymmetric
- context-dependent

Trust Metric (TM):
Uses existing edges for predicting values of trust for non-existing edges, thanks to trust propagation (if you trust someone, then you have some degree of trust in anyone that person trusts).
Trust-aware RSs

Instead of computing $UserSim$ of other users, compute $Trust$ in other users.

Instead of recommending items liked by similar users, recommends items liked by “trustable” users.
Trust solves RS problems

1. Trust solves CF sparseness problem
   - trust propagation and “6 degrees” -> reach many

2. Trust solves Cold Start problem
   - “just add 1 friend”

3. Trust metrics resistant to copy-profile-attack.
   - “you can be similar but if no trust path to you ...”

4. Trust easier to understand and control
   - trust nets supports Explanation (HCI tests needed)

EVIDENCE of 1 and 2 provided by analyzing a REAL, VAST community (Epinions.com)
Epinions.com dataset

1. Epinions.com users can
   - Review and rate items (from 1 to 5)
   - Keep web of trust (trust=1) and block list (trust=0). [Epinions FAQ says to put in Web of Trust “Reviewers whose reviews and ratings you have consistently found to be valuable”]

2. Dataset (collected by crawling site):
   - ~50K users, ~140K items, ~660K ratings.
   - ~500K trust statements.
     - No block list (not shown on site, kept hidden)
Experiment design

Compare performances of CF (1) and trust-aware (2) algorithm

(1) - use CF on ratings and compute “similarity” of other users
(2) - use Trust Metric and compute “trustworthiness” of other users

Then we can predict ratings based on similar OR trustable users.

Leave-one-out: hide one rating, predict it and compute the error (660,000 ratings)
Used “trust metric”

...not really a trust metric.

Linear decay based on distance from ME: closer users are more trustable.

Parameter: max propagation distance (mpd)

\[
\text{Trust}_{\text{ME}(B)} = \frac{(\text{mpd} - \text{dist}_{\text{ME}(B)} + 1)}{\text{mpd}}
\]

If \( \text{dist}_{\text{ME}(B)} > \text{mpd} \) then \( \text{Trust}_{\text{ME}(B)} = \text{null} \)

Experiments with mpd=1, 2, 3, 4 called \textit{Trust-1, Trust-2, Trust-3, Trust-4}
Used “trust metric”

Example: max propagation distance=4

\[ \text{Dist}_{ME}(B) = 1 \]
\[ T_{ME}(B) = (4 - 1 + 1)/4 = 1 \]

\[ \text{Dist}_{ME}(B) = 2 \]
\[ T_{ME}(B) = (4 - 2 + 1)/4 = 0.75 \]

\[ \text{Dist}_{ME}(B) = 3 \]
\[ T_{ME}(B) = (4 - 3 + 1)/4 = 0.5 \]

\[ \text{Dist}_{ME}(B) = 4 \]
\[ T_{ME}(B) = (4 - 4 + 1)/4 = 0.25 \]

\[ \text{Dist}_{ME}(B) > 4 \]
\[ T_{ME}(B) = \text{null} \]
### Experimental Results

<table>
<thead>
<tr>
<th># Expressed Ratings</th>
<th>ALL</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>User population size</td>
<td>40169</td>
<td>3937</td>
<td>2917</td>
<td>2317</td>
</tr>
<tr>
<td>Mean Web of Trust Size</td>
<td>9.88</td>
<td>2.54</td>
<td>3.15</td>
<td>3.64</td>
</tr>
<tr>
<td>Ratings UserSim</td>
<td>51%</td>
<td>N/A</td>
<td>4%</td>
<td>8%</td>
</tr>
<tr>
<td>Coverage Trust-1</td>
<td>28%</td>
<td>10%</td>
<td>11%</td>
<td>12%</td>
</tr>
<tr>
<td></td>
<td>60%</td>
<td>23%</td>
<td>26%</td>
<td>31%</td>
</tr>
<tr>
<td></td>
<td>74%</td>
<td>39%</td>
<td>45%</td>
<td>51%</td>
</tr>
<tr>
<td></td>
<td>77%</td>
<td>45%</td>
<td>53%</td>
<td>59%</td>
</tr>
<tr>
<td>Users UserSim</td>
<td>41%</td>
<td>N/A</td>
<td>6%</td>
<td>14%</td>
</tr>
<tr>
<td>Coverage Trust-1</td>
<td>45%</td>
<td>17%</td>
<td>25%</td>
<td>32%</td>
</tr>
<tr>
<td></td>
<td>56%</td>
<td>32%</td>
<td>43%</td>
<td>53%</td>
</tr>
<tr>
<td></td>
<td>61%</td>
<td>46%</td>
<td>57%</td>
<td>64%</td>
</tr>
<tr>
<td></td>
<td>62%</td>
<td>56%</td>
<td>59%</td>
<td>66%</td>
</tr>
<tr>
<td>Mean UserSim</td>
<td>0.843</td>
<td>N/A</td>
<td>1.244</td>
<td>1.027</td>
</tr>
<tr>
<td>Absolute Trust-1</td>
<td>0.837</td>
<td>0.929</td>
<td>0.903</td>
<td>0.840</td>
</tr>
<tr>
<td>Error Trust-2</td>
<td>0.829</td>
<td>1.050</td>
<td>0.940</td>
<td>0.927</td>
</tr>
<tr>
<td>(MAE)</td>
<td>0.811</td>
<td>1.046</td>
<td>0.940</td>
<td>0.918</td>
</tr>
<tr>
<td></td>
<td>0.805</td>
<td>1.033</td>
<td>0.926</td>
<td>0.903</td>
</tr>
<tr>
<td>Mean UserSim</td>
<td>0.939</td>
<td>N/A</td>
<td>1.319</td>
<td>1.095</td>
</tr>
<tr>
<td>Absolute Trust-1</td>
<td>0.855</td>
<td>0.942</td>
<td>0.891</td>
<td>0.847</td>
</tr>
<tr>
<td>User Trust-2</td>
<td>0.881</td>
<td>1.041</td>
<td>0.935</td>
<td>0.905</td>
</tr>
<tr>
<td>Error Trust-3</td>
<td>0.862</td>
<td>1.033</td>
<td>0.942</td>
<td>0.915</td>
</tr>
</tbody>
</table>

**Rows:**
- **UserSim**=CollaborativeFiltering
- **Trust-x**=Trust propagation up to distance x

**RatingsCoverage**=how many hidden ratings are predictable.

**UsersCoverage**=how many users get at least a prediction

**MAE**=|real_rating-pred_rating| averaged over all the ratings.

**MAUE**=|real_rating-pred_rating| averaged over the ratings of one user, then averaged over all users.

**Columns:**
- Views over users.
- **ALL**=all the users (with 1 rating)
- **2**=only users that gave 2 ratings (they are 3937)
Experimental Results

On average, Trust-\(x\) achieves better coverage without loss of accuracy.

<table>
<thead>
<tr>
<th># Expressed Ratings</th>
<th>ALL</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>User population size</td>
<td>40169</td>
<td>3937</td>
<td>2917</td>
<td>2317</td>
</tr>
<tr>
<td>Mean Web of Trust Size</td>
<td>9.88</td>
<td>2.54</td>
<td>3.15</td>
<td>3.64</td>
</tr>
</tbody>
</table>

| Ratings | UserSim | 51% | N/A | 4%  | 8%  |
| Coverage | Trust-1 | 28% | 10% | 11% | 12% |
|          | Trust-2 | 60% | 23% | 26% | 31% |
|          | Trust-3 | 74% | 39% | 45% | 51% |
|          | Trust-4 | 77% | 45% | 53% | 59% |

| Users | UserSim | 41% | N/A | 6%  | 14% |
| Coverage | Trust-1 | 45% | 17% | 25% | 32% |
|          | Trust-2 | 56% | 32% | 43% | 53% |
|          | Trust-3 | 61% | 46% | 57% | 64% |
|          | Trust-4 | 62% | 56% | 59% | 66% |

| Mean | UserSim | 0.843 | N/A | 1.244 | 1.027 |
| Absolute Error | Trust-1 | 0.837 | 0.929 | 0.903 | 0.840 |
|             | Trust-2 | 0.829 | 1.050 | 0.940 | 0.927 |
|             | Trust-3 | 0.811 | 1.046 | 0.940 | 0.918 |
|             | Trust-4 | 0.805 | 1.033 | 0.926 | 0.903 |

| Mean | UserSim | 0.939 | N/A | 1.319 | 1.095 |
| Absolute Error | Trust-1 | 0.855 | 0.942 | 0.891 | 0.847 |
|             | Trust-2 | 0.881 | 1.041 | 0.935 | 0.905 |
|             | Trust-3 | 0.862 | 1.033 | 0.942 | 0.915 |
On average, *Trust-x* achieves better coverage without loss of accuracy.

*UserSim* performs well with heavy raters and poorly with cold start users.
Experimental Results

On average, Trust-x achieves better coverage without loss of accuracy.

UserSim performs well with heavy raters and poorly with cold start users.

For cold start users (50% of the total!), Trust-x achieves also better accuracy.

For bootstrapping RSs, asking one trust statement is better than asking one rating.

(experiments on 660,000 ratings)
Contribution

Experimental evidence that:
- CF is ineffective in real world scenarios (and in Semantic Web)
  - Especially for Cold Start users (50%!)
- Trust-awareness *can* solve CF problems
  - Sparseness
  - Cold Start
  - (Attacks)
Future works

1. UserSim and Trust correlate? Contradict?
2. Distrust?
   - Propagation? Properties?
3. Design a Trust Metric (for RS)
   - Compare different trust metrics (local and global)
4. Create and evaluate a Trust-aware RS
   - Technique for combining Ratings and Trusts
5. Integrate Trust-awareness in www.Moleskiing.it and Cocoa.itc.it
6. Study RS attacks and attack-resistance
Thanks for your attention!


Questions?

Paolo Massa
Email: massa@itc.it
Homepage: http://sra.itc.it/people/massa/