**Problem**

Neighbor selection in Collaborative Filtering suffers from data sparsity and the cold-start problem.

Trust networks can be used to alleviate the problem, but are often also sparse.

**Experimental Setup**

Dataset: Gathered from epinions.com with 49,290 users, 139,738 items, 664,824 ratings, and 487,181 trust connections.

Trust-graph density = 0.0002.

**Baselines:**

Most Popular (MP)

Naïve trust-based CF (Trust

Jaccard trust-based CF

Adapted Katz (KSa,b,c,d) approaches:

(a) Use Trust Propagation with lmax or Not

(b) Use Combined, In-Degree or No Degree Normalization

(c) Use L1, L2, Max or No Row Normalization

(d) Boosting of propagated trust values or Not

**Setting:**

Simulating the cold-start problem by recommending n = [1, 10] items for all users which have rated at least 10 items (= 25,393 users)

**Contribution**

Explore the application of the Katz similarity (KS) measure for cold-start users in a trust-based CF approach.

Evaluate the resulting similarity matrix with different normalization techniques for a better recommendation accuracy.

Introduce an adapted KS measure that gives higher similarity values to node pairs with path lengths of 2.

**Future Work**

Investigate the impact of trust-based networks on beyond accuracy metrics such as novelty, diversity and coverage.

Explore the recently popularized node embedding techniques (e.g., Node2Vec or GraphSAGE) for trust networks.

**Approach**

**Step 1: Calculating Katz Similarity with a chosen lmax.**

By using the iterative approach:

\[ \sigma^{(l_{max}+1)} = \sum_{l=0}^{\max} (\alpha A)^l, \text{ where } \sigma^{(0)} = 0 \text{ and } \sigma^{(1)} = I \]

(1)

In the conducted experiments, we used values 1 and 2 for lmax, which means that we either have not propagated similarities through the network at all or that we propagated them through the network using a maximum path length of 2.

**Step 2: Degree normalization.**

KS as defined in Eq. (1), tends to give high similarity to nodes that have a high degree. In some cases this might be desirable but if we want to get rid of this bias, we can apply a degree normalization on \( \sigma \):

\[ \sigma^{(l_{max}+1)} = D^{-1} \left( \sum_{l=0}^{\max} (\alpha A)^l \right) D^{-1} \]

(2)

**Step 3: Row normalization.**

We introduced an additional step where we individually scale rows of the final resulting matrix using one of the three vector norms: L1, L2 or max.

**Step 4: Boosting propagated similarities.**

One of the contributions of this paper was to increase the impact of propagated trust values generated with KS for lmax = 2. Our proposed approach for doing this consists of the following four steps: (i) calculate \( \sigma^{(3)} \) as described above using the trust network as \( A \), (ii) create a new similarity matrix \( \hat{\sigma} \) such that:

\[ \hat{\sigma}_{i,j} = \begin{cases} \sigma^{(3)}_{i,j}, & \text{if } A_{i,j} = 0 \\ 0, & \text{otherwise} \end{cases} \]

(3)

(iii) create \( \sigma_{\text{norm}} \) matrix by individually scaling rows of \( \hat{\sigma} \) using L1, L2 or \( \text{max vector norm} \) and lastly, (iv) create a similarity matrix \( \sigma_{\text{boost}} \) such that:

\[ \sigma_{\text{boost}} = A + \sigma_{\text{norm}} \]

(4)

**Evaluation**

Evaluation results for n = 10. The reported subset of the 33 evaluated KS-based approaches are additionally labeled for an easier result comparison between different step combinations (i.e., columns 2 to 5).

<table>
<thead>
<tr>
<th>Approach</th>
<th>lmax</th>
<th>Degree normalization</th>
<th>Row normalization</th>
<th>Boost</th>
<th>nDCG</th>
<th>Recall</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trustexp</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>.0224</td>
<td>.0296</td>
<td>.0110</td>
</tr>
<tr>
<td>Trustjac</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>.0176</td>
<td>.0219</td>
<td>.0087</td>
</tr>
<tr>
<td>MP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>.0134</td>
<td>.0202</td>
<td>.0070</td>
</tr>
<tr>
<td>KS_PCBMB</td>
<td>2</td>
<td>Combined</td>
<td>Max</td>
<td>Yes</td>
<td>.0303</td>
<td>.0425</td>
<td>.0117</td>
</tr>
<tr>
<td>KS_PCNM</td>
<td>2</td>
<td>Combined</td>
<td>Max</td>
<td>No</td>
<td>.0295</td>
<td>.0422</td>
<td>.0113</td>
</tr>
<tr>
<td>KS_PCLB</td>
<td>2</td>
<td>Combined</td>
<td>L1</td>
<td>Yes</td>
<td>.0273</td>
<td>.0358</td>
<td>.0106</td>
</tr>
<tr>
<td>KS_PNNB</td>
<td>2</td>
<td>No degree</td>
<td>L2</td>
<td>Yes</td>
<td>.0257</td>
<td>.0340</td>
<td>.0106</td>
</tr>
<tr>
<td>KS_NCMM</td>
<td>1</td>
<td>Combined</td>
<td>Max</td>
<td>No</td>
<td>.0213</td>
<td>.0289</td>
<td>.0106</td>
</tr>
<tr>
<td>KS_NINN</td>
<td>1</td>
<td>In degree</td>
<td>N/A</td>
<td>No</td>
<td>.0161</td>
<td>.0243</td>
<td>.0087</td>
</tr>
<tr>
<td>KS_PNNB</td>
<td>2</td>
<td>No degree</td>
<td>N/A</td>
<td>No</td>
<td>.0036</td>
<td>.0057</td>
<td>.0020</td>
</tr>
</tbody>
</table>

![Graph showing evaluation results](image)

**References**


We implemented and evaluated our approach using ScaR [2], a scalable recommendation framework which is easily adaptable for a multi-domain environment.