Factored Proximity Models for Top-N Recommendations

Athanasios N. Nikolakopoulos[1], Vassilis Kalantzis[2], Efstratios Gallopoulos[3] and John D. Garofalakis[3]

[1]Digital Technology Center, University of Minnesota, USA.
[2]Dept. of Computer Science & Engineering, University of Minnesota, USA.
[3]Dept. of Computer Engineering & Informatics, University of Patras, Greece.

August 08, 2017

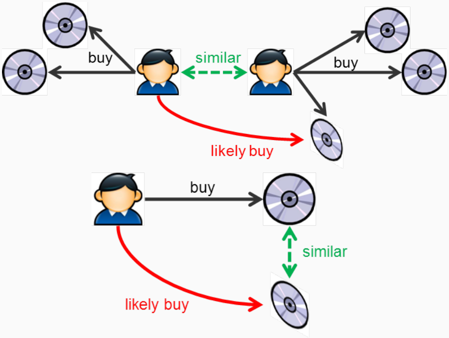
IEEE International Conference on Big Knowledge, IEEE ICBK 2017

Recommender Systems

- Widely Applicable Technology
  - Value for Customers
  - Value for Companies

Collaborative Filtering

- **Model:** Ratings!
- Recommendation task
  - Rating Prediction
  - **Top-N Lists**



Sparsity Problem

- Limits the quality recommendations; especially for **Long-Tail Items**.
- Intrinsic RS Characteristic
  - **Cold-Start Problem**

Promising Approaches

- **Graph-based Models**
  - (+) Good Performance
  - (-) Scalability Issues
- **Latent-Factor Models**.

Our Focus: *Efficient and High-Quality Top-N Recommendations Even under Extreme Sparsity*

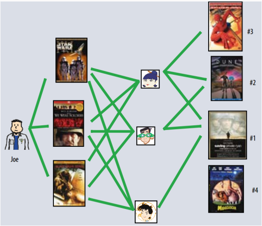


Figure 1: Graph-Based Idea

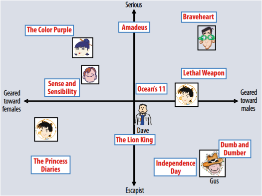


Figure 2: Latent-Factor Idea

EigenRec Framework

EigenRec

- Build a symmetric $m \times m$ **Inter-Item Proximity Matrix** $\mathbf{A}$, each element of which is defined to be a product of a *Scaling* and a *Similarity* component.

$$[\mathbf{A}]_{ij} \triangleq \xi(i,j) \cdot \kappa(i,j)$$

- Build a **Lower Dimensional Model** using the principal eigenvectors of $\mathbf{A}$
- Project the Users's feedback vectors onto the Latent Subspace:

$$\mathbf{\Pi} \triangleq \mathbf{R}\mathbf{V_f}\mathbf{V_f}^{\intercal}$$

Simple Baseline Choices

Scaling function

$$\xi(i,j) \triangleq f(i,j;d) = (\|\mathbf{r_i}\|\|\mathbf{r_j}\|)^d$$

Similarity functions

$$\kappa(i,j) \triangleq \begin{cases} \cos(v_i, v_j) \\ \mathsf{pc}(v_i, v_j) \\ \mathsf{jaccard}(v_i, v_j) \end{cases}$$

PureSVD within EigenRec

PureSVD

$$\boldsymbol{\Pi}_{\text{PureSVD}} \triangleq \mathbf{U_f \Sigma_f Q_f^\intercal} \equiv \cdots \equiv \mathbf{R Q_f Q_f^\intercal}$$

Where $\mathbf{Q_f}$, the matrix containing the f principal eigenvectors of:

$$\mathbf{R}^\intercal\mathbf{R} = \begin{array}{cc} & \textit{users} \\ \textit{items} & \begin{bmatrix} - & \mathbf{r}_{\mathbf{v_i}}^\intercal & - \end{bmatrix} \end{array} \times \begin{array}{cc} & \textit{items} \\ \textit{users} & \begin{bmatrix} | \\ \mathbf{r}_{\mathbf{v_j}} \\ | \end{bmatrix} \end{array}$$

$$= \begin{array}{cc} & \textit{items} \\ \textit{items} & \begin{bmatrix} \boxed{\cdot} \end{bmatrix} \end{array} \qquad \underbrace{\|\mathbf{r}_{\mathbf{v_i}}\|\|\mathbf{r}_{\mathbf{v_j}}\|}_{\textit{scaling}} \cdot \underbrace{\cos\theta_{ij}}_{\textit{similarity}},$$

- PureSVD $\equiv$ EigenRec with Cosine similarity and $f(i, j; 1)$

Computing EigenRec

EigenRec:

Input: Inter-Item proximity matrix $\mathbf{A} \in \mathfrak{R}^{m \times m}$. Rating Matrix $\mathbf{R} \in \mathfrak{R}^{n \times m}$. Latent Factors f.
Output: Matrix $\mathbf{\Pi} \in \mathfrak{R}^{n \times m}$ whose rows are the recommendation vectors for every user.

$\mathbf{q_j} = 0$, set $\mathbf{r} \leftarrow \mathbf{q}$ as a random vector
$\beta_0 \leftarrow \|\mathbf{r}\|_2$
for $j \leftarrow 1, 2, \ldots,$ **do**
    $\mathbf{q_j} \leftarrow \mathbf{r}/\beta_{j-1}$
    $\mathbf{r} \leftarrow \mathbf{A}\mathbf{q_j}$
    $\mathbf{r} \leftarrow \mathbf{r} - \mathbf{q_{j-1}}\beta_{j-1}$
    $\alpha_j \leftarrow \mathbf{q_j^\intercal}\mathbf{r}$
    $\mathbf{r} \leftarrow \mathbf{r} - \mathbf{q_j}\alpha_j$
    $\mathbf{r} \leftarrow (\mathbf{I} - \mathbf{Q_j}\mathbf{Q_j^\intercal})\mathbf{r}$,
    $\beta_j \leftarrow \|\mathbf{r}\|_2$
    Solve the tridiag problem $(\mathbf{Q_j^\intercal}\mathbf{A}\mathbf{Q_j})\mathbf{\Xi_j} = \mathbf{\Theta_j}\mathbf{\Xi_j}$
    Form the j approximate eigenvectors $\mathbf{Q_j}\mathbf{\Xi_j}$ of $\mathbf{A}$
    If the f top eigenvectors have converged, stop.
end for
Compute latent factors $\mathbf{V} = \mathbf{Q_f}\mathbf{\Xi}$
return $\mathbf{\Pi} \leftarrow \mathbf{R}\mathbf{V}\mathbf{V}^\intercal$

Computational Aspects:

- The MV product in j Lanczos steps is $\mathcal{O}(j \cdot nnz)$
- Making the j-th vector orthogonal to the previous ones costs $\mathcal{O}(jm)$

Parallel Implementation:

	MovieLens20M				
	f=50	100	150	200	300
8 cores	11.0	18.0	24.2	28.7	36.8
16 cores	6.9	11.6	15.6	18.9	24.3
32 cores	4.8	8.1	11.1	13.5	17.6
64 cores	3.5	6.0	8.2	9.9	12.5
EigenRec					

The Code is available here:
https://github.com/nikolakopoulos/EigenRec

Qualitative Evaluation

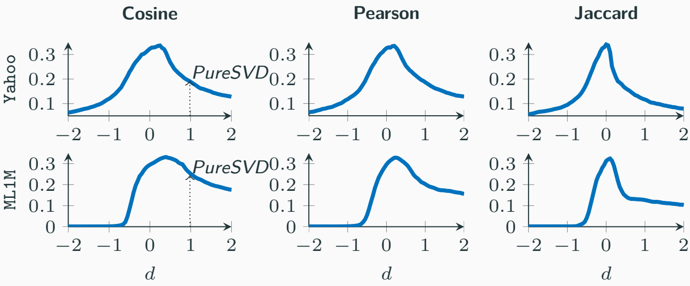
Effects of Prior Popularity

Methodology

- Randomly sample 1.4% of the ratings of the dataset ⇒ probe set $\mathcal{P}$
- Use each item v_j, rated with 5 stars by user u_i in $\mathcal{P}$ ⇒ test set $\mathcal{T}$
- Randomly select another 1000 unrated items of the same user for each item in $\mathcal{T}$
- Form ranked lists by ordering all the 1001 items

Metrics

- Recall
- Precision
- R-Score
- NDCG@k
- **MRR**



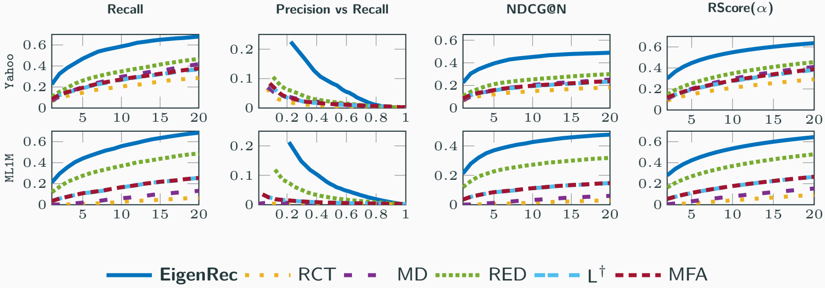
Standard Top-N Recommendations

Methodology

- Randomly sample 1.4% of the ratings of the dataset $\Rightarrow$ probe set $\mathcal{P}$
- Use each item v_j, rated with 5 stars by user u_i in $\mathcal{P} \Rightarrow$ test set $\mathcal{T}$
- Randomly select another 1000 unrated items of the same user for each item in $\mathcal{T}$
- Form ranked lists by ordering all the 1001 items

Metrics

- **Recall**
- **Precision**
- **R-Score**
- **NDCG@k**
- **MRR**



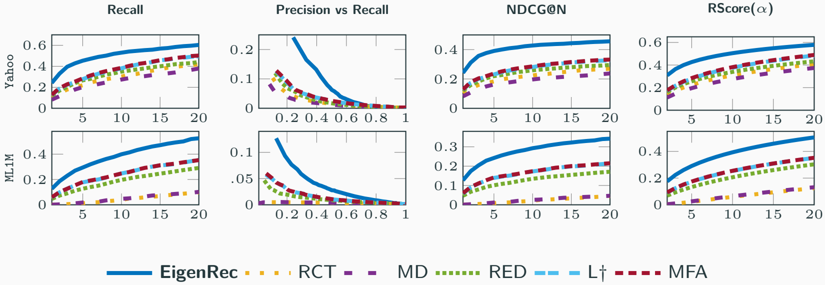
Long-Tail Recommendations

Methodology

- We order the items according to their popularity (measured in terms of number of ratings)
- We further partition the test set $\mathcal{T}$ into two subsets, $\mathcal{T}_{\text{head}}$ and $\mathcal{T}_{\text{tail}}$
- We discard the popular items and we evaluate EigenRec and the other algorithms on the $\mathcal{T}_{\text{tail}}$ test set, using the procedure explained previously.

Metrics

- **Recall**
- **Precision**
- **R-Score**
- **NDCG@k**
- MRR



Cold-Start Recommendations I

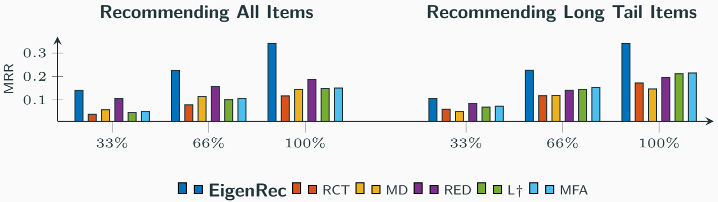
Cold-Start Problem

- Difficulty of making reliable recommendations **due to an initial lack of ratings**
- In beginning stages, when there is not sufficient number of ratings for the collaborative filtering algorithms to uncover similarities $\Rightarrow$ **New Community Problem**
- Introduction of new users to an existing system where they have not rated many items $\Rightarrow$ **New Users Problem**

Cold-Start Recommendations II

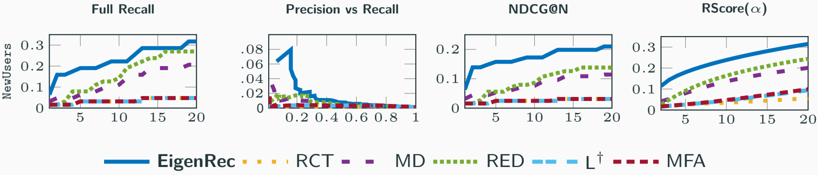
New Community:

- **Methodology:** Randomly select to include 33%, 66%, and 100% of the overall ratings on two new artificially sparsified versions of the dataset.



New Users:

- **Methodology:** Randomly select 50 users having rated at least 100 items and randomly delete 95% of each users' ratings.



Conclusions and Future Work

Conclusions and Future Work

EigenRec

- Computationally Efficient framework for Top-N Recommendations
- Allows for flexible modeling and control of the effects of prior popularity
- Natural generalization of PureSVD
  - (+) Optimize its Top-N recommendation performance
  - (+) Alleviate its inherent popularity bias
  - (+) Compute it more efficiently
- Good Top-N Recommendation Performance

Future Directions

- Explore more elaborate Similarity and Scaling functions
- Explore the Hierarchical structure of the Itemspace

References

A. N. Nikolakopoulos, V. Kalantzis, E. Gallopoulos, and J. Garofalakis.
EigenRec: Generalizing PureSVD for Effective and Efficient Top-N Recommendations.
Knowledge and Information Systems, 2018.

A. N. Nikolakopoulos, V. Kalantzis, E. Gallopoulos, and J. Garofalakis.
Factored Proximity Models for Top-N Recommendations.
In *2017 IEEE International Conference on Big Knowledge (ICBK)*, pp. 80-87, Hefei, 2017.

P. Cremonesi, Y. Koren, and R. Turrin.
Performance of recommender algorithms on top-n recommendation tasks.
In *Proceedings of the fourth ACM conference on Recommender systems*, RecSys '10, pages 39–46. ACM, 2010.

F. Fouss, K. Francoisse, L. Yen, A. Pirotte, and M. Saerens.
An experimental investigation of kernels on graphs for collaborative recommendation and semisupervised classification.
Neural Netw., 31:53–72, July 2012.

Thank you for your Attention!

Questions?