SLIM algorithm

- The goal is learning a similarity matrix such that it can be used to reconstruct the UI matrix

$S \in \mathbb{R}^{|I| \times |I|}$

Item-Item similarity matrix

UI matrix

$R \in \mathbb{R}^{|U| \times |I|}$

$\hat{R} \in \mathbb{R}^{|U| \times |I|}$

Reconstructed UI matrix

$$\hat{R} = R \times S$$

SLIM algorithm

- Similarity matrix is learned by minimizing the difference between actual rating and predicted rating

Original UI matrix $=$ **Predicted UI matrix**

R $\quad$ $\hat{R}$ $= R \times S$

$$R = R \times S \quad \textbf{or} \quad \underbrace{R - R \times S = 0}$$

Ideal, but not achievable. Therefore, $R - R \times S \approx 0$

$$\underset{S}{\text{minimize}} \quad R - R \times S$$

SLIM algorithm

- Similarity matrix is learned by minimizing the difference between actual rating and predicted rating

Original UI matrix $=$ **Predicted UI matrix**

R $\quad$ $\hat{R}$ $= R \times S$

$$R = R \times S \quad \textbf{or} \quad \underbrace{R - R \times S = 0}$$

Ideal, but not achievable. Therefore, $R - R \times S \approx 0$

$$\underset{S}{\text{minimize}} \quad (R - R \times S)^2 + \textbf{regularization}$$

SLIM algorithm

- Therefore,

Regularization terms for avoiding overfitting

$$\min_{S} \|R - R \times S\|_F^2 + \lambda_1 \|S\|_1 + \lambda_2 \|S\|_F^2$$

$$\text{subject to } diag(S) = 0 \text{ and } S \geq 0$$

Constrain to avoid self-recommendation

Constrain to emphasize similar items (than dissimilar items)

- **Matrix $S \in \mathbb{R}^{n \times n}$ is learned to approximately represent the item-item similarity matrix**

Learning process

1) **Randomly initialize matrix S**

   For each available rating provided by user u on item i in R, iteratively update S as follows:

   2) **Predict the rating value: $\widehat{R}_{ui} = R_u \times S_i$**

   3) **Compute *error* by comparing $\widehat{R}_{ui}$ and R_{ui}**

   4) **Update matrix S based on observed *error***

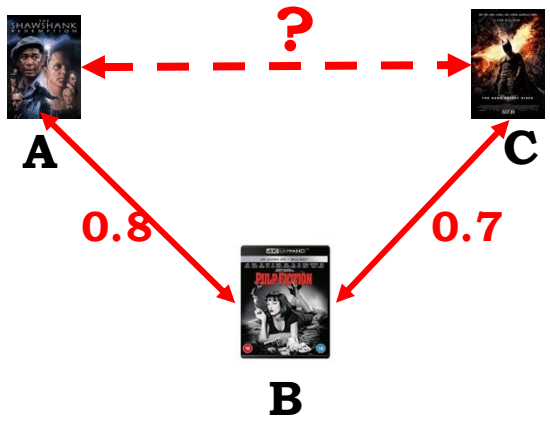
Results on Netflix data

Ning and Karypis, ICDM 2011

	ItemKNN		SLIM
Accuracy (Hit ratio)	0.178	<	0.200
Time	24.53	<	9.84

Why is SLIM more effective?

- Less computation is needed for building similarity matrix
- SLIM enables computing similarity value between all items
  - Capturing hidden relationships/similarities between items
    - Assume we want to compute the similarity between item A and item C, but there is no neighbor between them
    - However, we know the similaity between A-B and C-B
    - Then, can we approximately estimate the similarity between A and B?



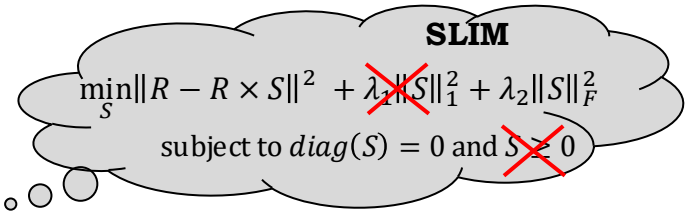
Extensions (additional resources)

- Embarrassingly Shallow Auto-Encoder ($EASE^R$)
  - *Embarrassingly shallow autoencoders for sparse data* (*WWW 2019*)
  - *Admm slim: Sparse recommendations for many users* (*WSDM 2020*)
- Incorporating side-information
  - *Closed-form models for collaborative filtering with side-information (RecSys 2020)*
- Higher Order SLIM (HOLISM)
  - *Hoslim: Higher-order sparse linear method for top-n recommender systems* (*PAKDD 2014*)
- Higher order $EASE^R$
  - *Negative Interactions for Improved Collaborative Filtering: Don't go Deeper, go Higher* (*RecSys 2021*)

Embarrassingly Shallow Auto-Encoder

Steck, WWW 2019

- SLIM's objective function is simplified for more efficient optimization
  - S can be mathematically derived instead of iterative optimization process



$$\min_S \|R - R \times S\|^2 + \lambda\|S\|_F^2$$

$$\text{subject to } diag(S) = 0$$

Closed-form solutions for Ordinary Least Squares (OLS) and ridge regression

$$\boldsymbol{S = \mathbf{I} - P.diagMat(\mathbf{1} \oslash diag(P))}$$

$$P = (X^T X + \lambda . I)^{-1}$$

Limitations of Neighborhood-Based Methods

✓ Similarity computation is sometimes not accurate/possible due to *sparsity* issue

✓ Computation of item-item similarity matrix is expensive

- Maintaining similarity matrix in memory is not possible and practical
  - Model parameter, item-item similarity matrix, is too big
  - Assuming $n = 1M$, then similarity matrix will be in $1M \times 1M$ dimension which requires ~4000GB memory