ECS289: Scalable Machine Learning

Cho-Jui Hsieh
UC Davis

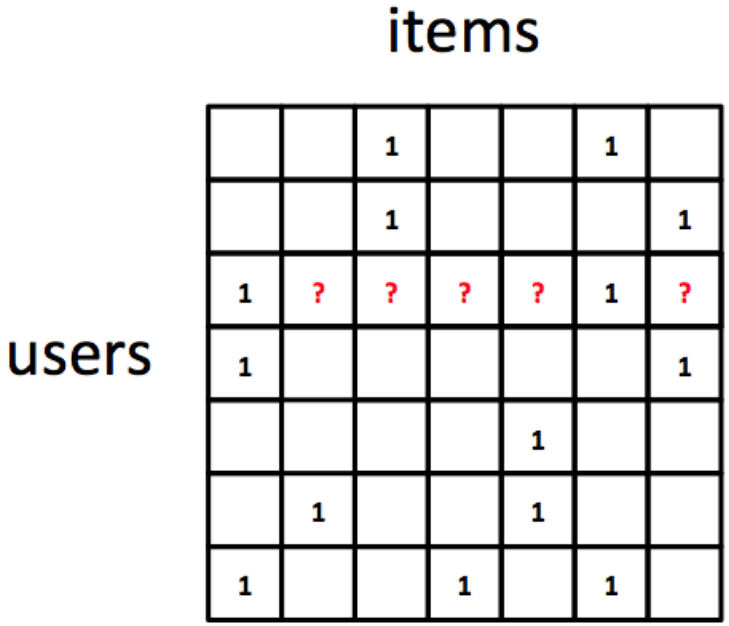
Oct 13, 2016

Outline

- Matrix completion with implicit feedback
- Matrix Completion with side information
  - Feature-based factorization
  - Graph-based Regularization

Example: Matrix Completion with 0/1 observations

- Amazon purchasing matrix: all the observed elements are 1
- Given: rating matrix $A \in \mathbb{R}^{m \times n}$, user feature matrix $X \in \mathbb{R}^{m \times d}$
- Goal: predict unknown ratings



0 (unobserved): user did not buy the item

1 (observed): user bought the item at some time point

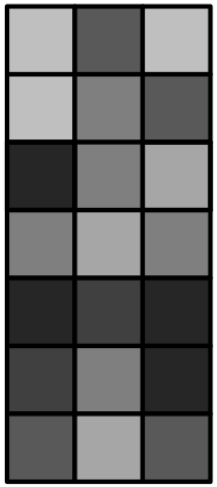
Example: Movie Recommendation with User Features

- Given: rating matrix $A \in \mathbb{R}^{m \times n}$, user feature matrix $X \in \mathbb{R}^{m \times d}$
- Goal: predict unknown ratings

Profile features

(Organization, location, behavior…)

users



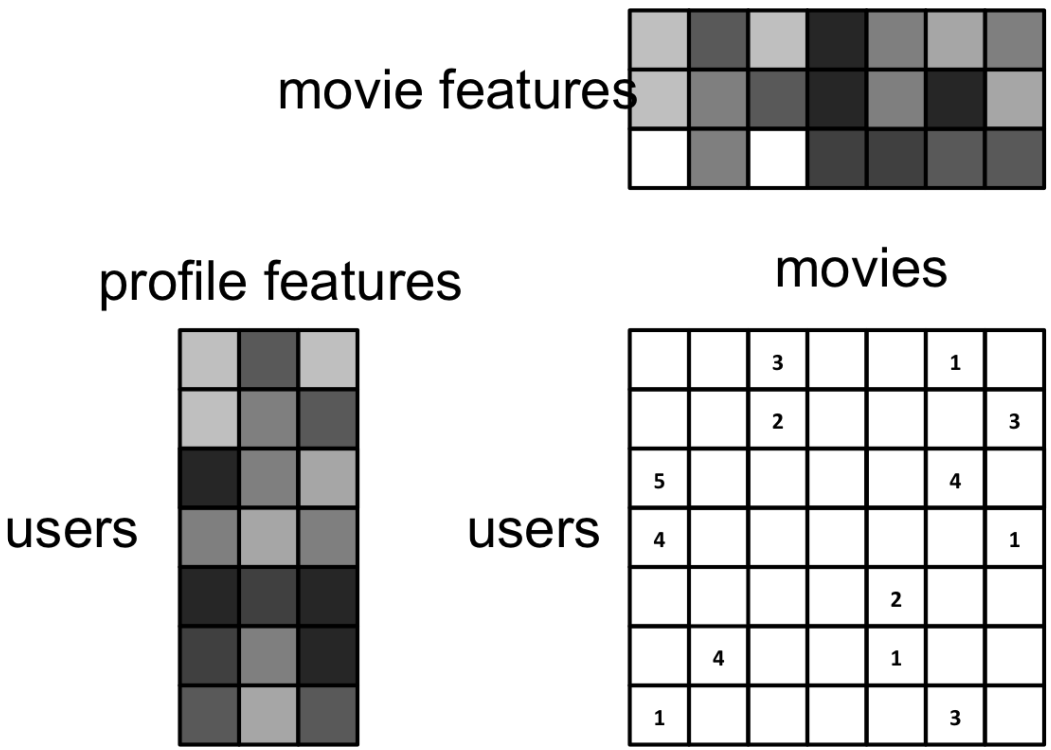
movies

users

		3			1	
		2				3
5					4	
4						1
				2		
	4			1		
1					3	

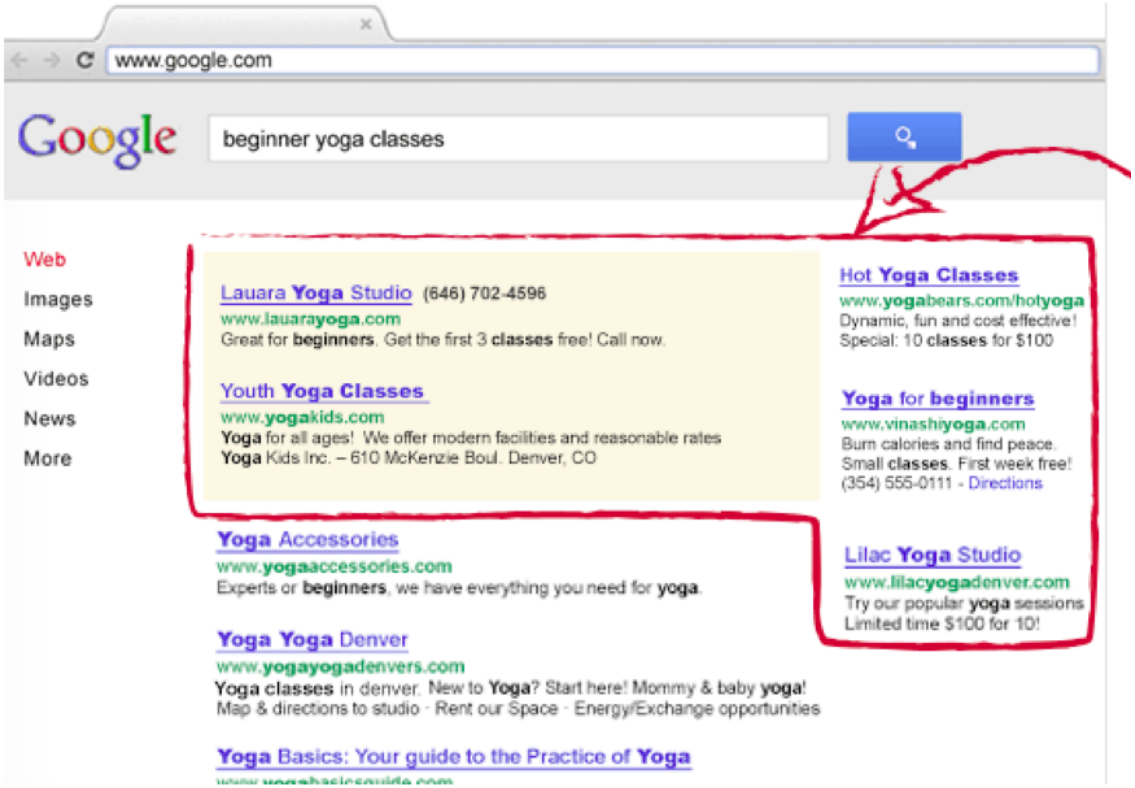
Example: Movie Recommendation with Features

- Given, rating matrix $A \in \mathbb{R}^{m \times n}$, user features $X \in \mathbb{R}^{m \times d_1}$, item features $Y \in \mathbb{R}^{n \times d_1}$
- Goal: predict unknown ratings



Applications

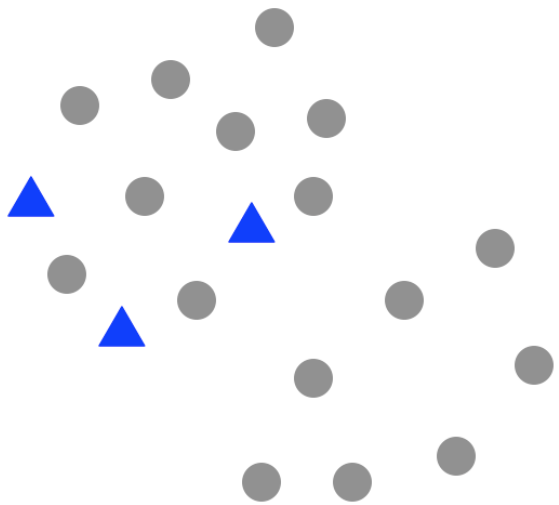
- Ads Recommendation
- Tag recommendation
- Click-through prediction
- ...



PU Matrix Completion

PU Matrix Completion

- Similar cases have been discussed in PU (Positive and Unlabeled) learning:
  - Classification, with very few positive labels
  - All the rest data points have unobserved labels (potentially can be $+1$ or -1)
- Simple but effective approaches:
  - Down-sampling & Bagging
  - Biased Loss

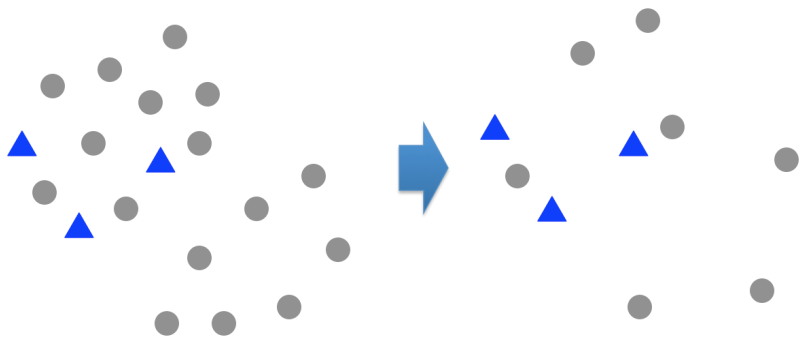


Down Sampling

- Given: positive samples $(\boldsymbol{u}_1, +1), \cdots, (\boldsymbol{u}_n, +1)$ and unlabeled samples $(\boldsymbol{z}_1, ?), \cdots, (\boldsymbol{z}_m, ?)$. Assume $m \gg n$
- Sample $O(n)$ data points $\boldsymbol{v}_1, \cdots, \boldsymbol{v}_s$ from unlabeled samples.
- Training the binary classification problem using

$$(\boldsymbol{u}_1, +1), (\boldsymbol{u}_2, +1), \cdots, (\boldsymbol{u}_n, +1)$$
$$(\boldsymbol{v}_1, -1), (\boldsymbol{v}_2, -1), \cdots, (\boldsymbol{v}_s, -1)$$

- Do this several times and average the results.

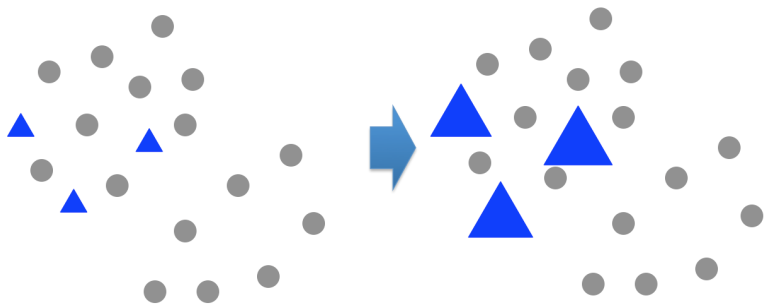


Biased Loss Function

- Given: positive samples $(\boldsymbol{u}_1, +1), \cdots, (\boldsymbol{u}_n, +1)$ and unlabeled samples $(\boldsymbol{z}_1, ?), \cdots, (\boldsymbol{z}_m, ?)$. Assume $m \gg n$
- Training the binary classification problem with smaller weights on the loss function of unlabeled samples:

$$\min_{\boldsymbol{w}} \sum_{i=1}^{n} \ell(\boldsymbol{w}^T \boldsymbol{u}_i, +1) + \alpha \sum_{j=1}^{m} \ell(\boldsymbol{w}^T \boldsymbol{z}_j, -1)$$

$\alpha \in (0, 1)$: smaller weight to the unlabeled samples (because they might be positive)



PU Matrix Completion

- Biased loss with all unlabeled data is much better than down-sampling in practice.
- PU matrix completion:
  - Given a partially observed $A \in \mathbb{R}^{m \times n}$, where

$$A_{ij} = \begin{cases} 1 & \text{if } (i,j) \in \Omega \\ ?(0) & \text{if } (i,j) \notin \Omega \end{cases}$$

- Biased Matrix Completion:

$$\min_{W,H} \sum_{(i,j)\in\Omega} ((WH^T)_{ij} - 1)^2 + \alpha \sum_{(i,j)\notin\Omega} ((WH^T)_{ij} - 0)^2$$

$\alpha \in (0,1)$: smaller weight to the unlabeled samples (because they might be positive)

PU Matrix Completion

- First proposed in (Hu et al., "Collaborative filtering for implicit feedback datasets", in ICDM 2008.)
- Naive implementation: $O(mn)$ time since there are mn terms in the loss function $\Rightarrow$ not scalable
- Theoretical guarantee and fast optimization algorithm: (Hsieh et al., "PU Learning for Matrix Completion", in ICML 2015.)
- $O(|\Omega|k + nk^2 + mk^2)$ for computing gradient:

$$f(W,H) = \alpha\|WH^T - A\|_F^2 + (1-\alpha)\sum_{(i,j)\in\Omega}((WH^T)_{ij} - 1)^2$$

$$\nabla_W f(W,H) = \underbrace{2\alpha(WH^TH - AH)}_{O(nk^2+mk^2+|\Omega|k)} + \underbrace{2(1-\alpha)\sum_{(i,j)\in\Omega}(\boldsymbol{w}_i^T\boldsymbol{h}_j - 1)\boldsymbol{h}_j}_{O(|\Omega|k)}$$

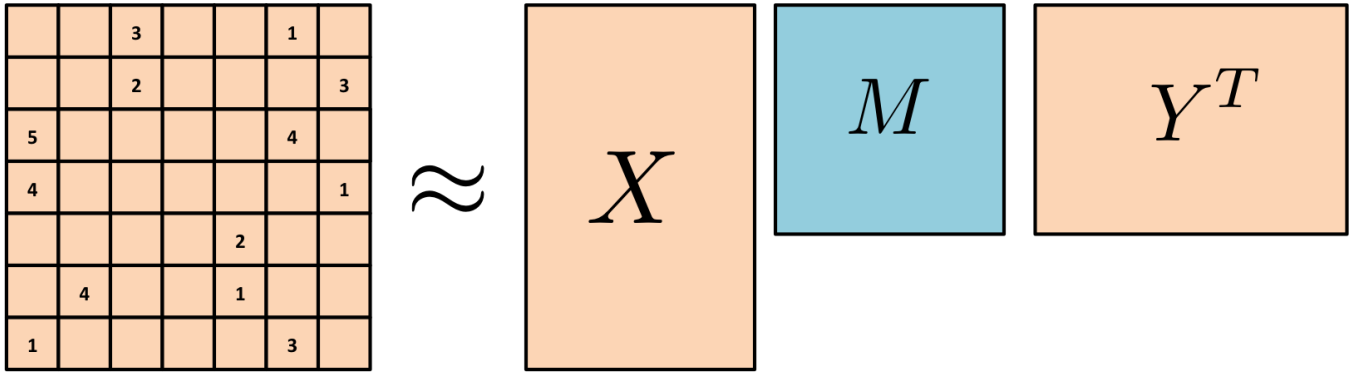
- The trick can be applied in ALS, coordinate descent (how about sgd?)

Inductive Matrix Completion

Matrix Completion with Features

Inductive Matrix Completion

- Assume the rating matrix A is generated from XMY^T, where X is the row feature matrix and Y is the column feature matrix
- Goal: Find the M matrix to minimize the error



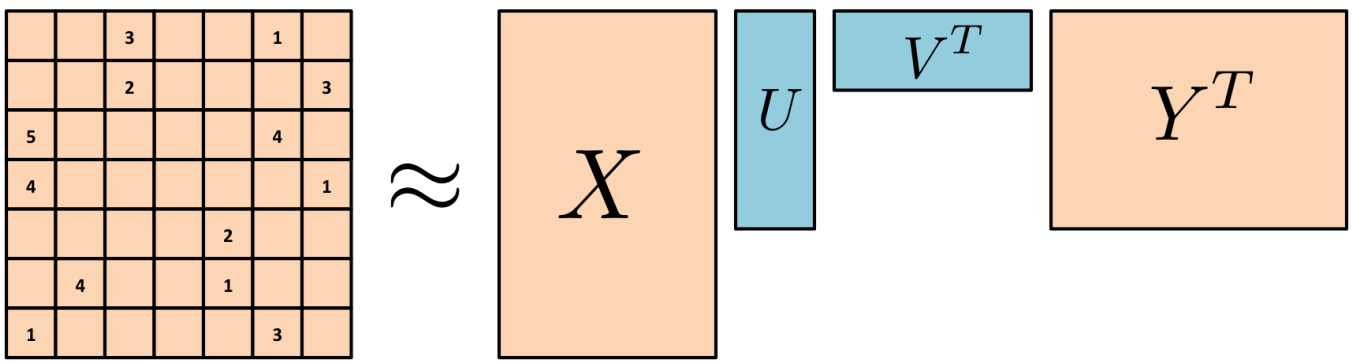
Inductive Matrix Completion

- $M \in \mathbb{R}^{d_1 \times d_2}$ may be large

  $\Rightarrow$ Further assume M is a rank k matrix
- Inductive matrix factorization:

$$\min_{U \in \mathbb{R}^{d_1 \times k}, V \in \mathbb{R}^{d_2 \times k}} \sum_{i,j \in \Omega} \left((XUV^T Y^T)_{ij} - A_{ij} \right)^2 + \lambda \|U\|_F^2 + \lambda \|V\|_F^2$$

where Ω is the observed set, A is the rating matrix, X, Y are feature matrices.



Theoretical Guarantees

- Matrix Completion (without features):

  If the rating matrix $A \in \mathbb{R}^{m \times n}$ is sampled from a rank k matrix, under certain condition we can recover A with $O(kn \log^2(n))$ observed entries.

- Inductive Matrix Completion:

  If the rating matrix $A = XMY^T$ and M is rank k, under certain condition we can recover A with $O(dk \log k \log n)$ observed entries.

- Proved by the following papers:
  - Xu et al., "Speedup Matrix Completion with Side Information: Application to Multi-Label Learning". In NIPS 2013.
  - Zhong et al., "Efficient Matrix Sensing using Rank-1 Gaussian Measurements". In ALT 2015.

Applications

- Multi-label classification
- Yahoo Music challenge
- Biological application
- Semi-supervised clustering

Drawbacks

- The assumption $A = XMY^T$ means:

$$\text{col}(A) \subseteq \text{col}(X) \text{ and } \text{col}(A^T) \subseteq \text{col}(Y)$$

  where $\text{col}(\cdot)$ denotes the column space of a matrix
- When features are not perfect:

$$\text{col}(A) \cap \text{col}(X)^T \neq \emptyset \text{ or } \text{col}(A^T) \cap \text{col}(Y)^T \neq \emptyset$$

  Recovery by inductive matrix completion is impossible.
- Can be resolved using nonlinear feature mappings:

$$X = \begin{bmatrix} \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_m \end{bmatrix} \Rightarrow \begin{bmatrix} \varphi(\boldsymbol{x}_1) \\ \vdots \\ \varphi(\boldsymbol{x}_m) \end{bmatrix}$$

(Si et al., "Goal Directed Inductive Matrix Completion", in KDD 2016.)

Factorization Machine

Factorization Machine

- Widely used for recommender systems in companies and in data mining competitions!
- Proposed by (S. Rendle, "Factorization Machines", in ICDM 2010.)
- Observation: Recommendation systems can be transformed to classification or regression

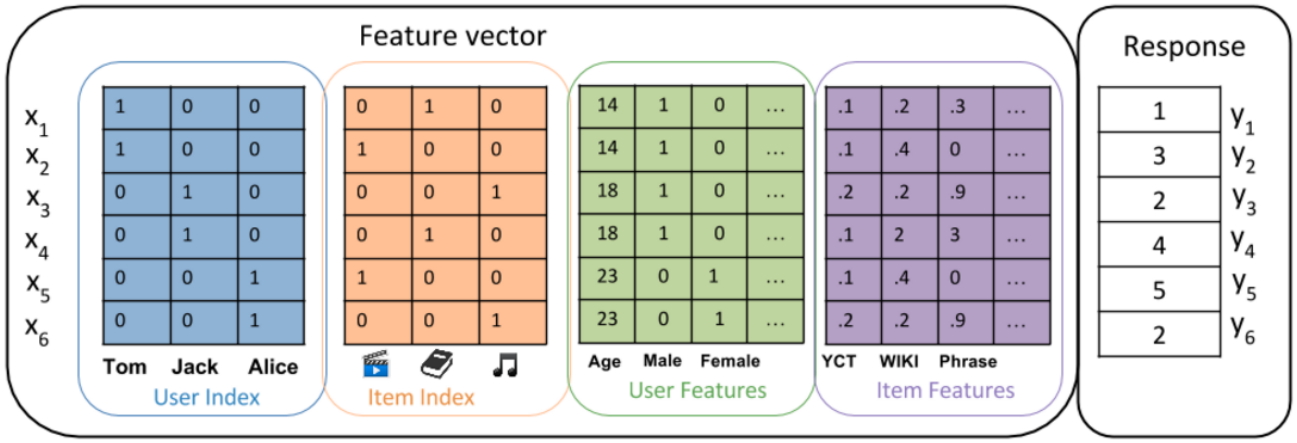
$$(i, j, A_{ij}) \Rightarrow (\boldsymbol{x}^{(i,j)}, A_{ij}), \text{ where}$$

$\boldsymbol{x}^{(i,j)}$ is the feature extracted for user i and item j

- Example: $\boldsymbol{x} = [\boldsymbol{u}_i^T \quad \boldsymbol{v}_j^T]$, where $\boldsymbol{u}_i$ is the feature for user i and $\boldsymbol{v}_j$ is the feature for user j
- Now we have a classification or regression problem with training data $\{(\boldsymbol{x}^{(i,j)}, A_{ij})\}_{(i,j)\in\Omega}$

Factorization Machine

- Many different ways to set the feature vector.



(Figure from Yahoo Research)

Factorization Machine

- What model should we use for this classification/regression problem?
- Choice I: Linear model (usually too simple and tend to underfit)

$$\boldsymbol{w}^T \boldsymbol{x}$$

Factorization Machine

- What model should we use for this classification/regression problem?
- Choice I: Linear model (usually too simple and tend to underfit)

$$\boldsymbol{w}^T \boldsymbol{x}$$

- Choice II: degree-2 polynomial (not work well for sparse data)

$$\boldsymbol{x}^T M \boldsymbol{x} = \sum_{i,j=1}^{d} x_i x_j M_{ij}$$

In sparse data, many (i, j) pair only occurs once in training data, so not enough information to learn M_{ij}. $O(d^2)$ parameters.

Factorization Machine

- What model should we use for this classification/regression problem?
- Choice I: Linear model (usually too simple and tend to underfit)

$$\boldsymbol{w}^T \boldsymbol{x}$$

- Choice II: degree-2 polynomial (not work well for sparse data)

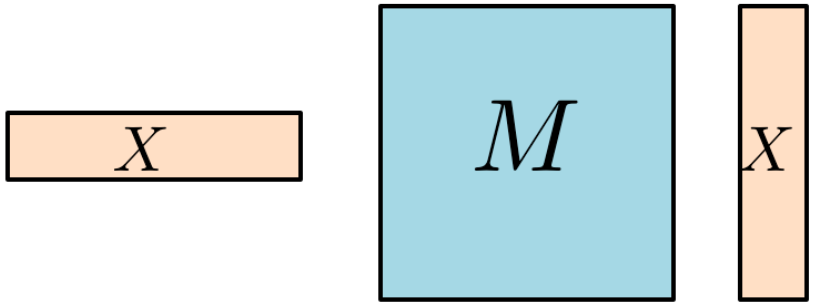
$$\boldsymbol{x}^T M \boldsymbol{x} = \sum_{i,j=1}^{d} x_i x_j M_{ij}$$

  In sparse data, many (i, j) pair only occurs once in training data, so not enough information to learn M_{ij}. $O(d^2)$ parameters.
- Choice III: Low-rank + degree 2 = Factorization Machine

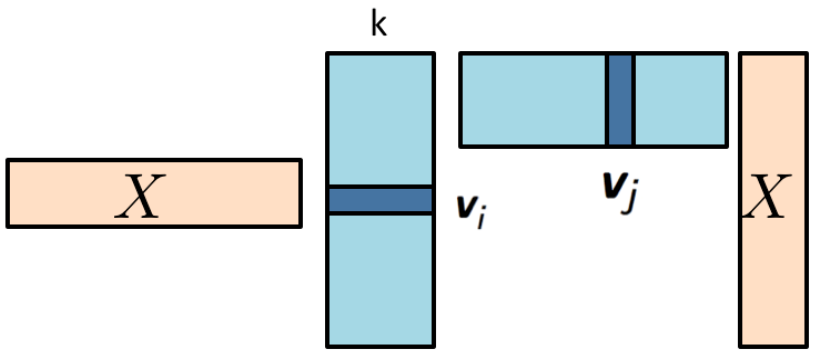
$$\boldsymbol{x}^T V^T V \boldsymbol{x} = \sum_{i,j} \boldsymbol{v}_i^T \boldsymbol{v}_j x_i x_j$$

  where $V \in \mathbb{R}^{k \times n}$ and $\boldsymbol{v}_i$ is the i-th column of V.
  $O(dk)$ parameters

Factorization Machine



Degree-2 Polynomial



Factorization Machine

Factorization Machine: Training

- Ω: observed entries, $\boldsymbol{x}^{(i,j)}$: feature generated by i-th user and j-th item (e.g, $[\boldsymbol{u}_i^T \boldsymbol{v}_j^T]$, where $\boldsymbol{u}_i, \boldsymbol{v}_j$ are user/item features)
- Solve the following minimization problem:

$$\min_{V \in \mathbb{R}^{d \times k}} \sum_{(i,j) \in \Omega} \text{loss}((\boldsymbol{x}^{(i,j)})^T V V^T \boldsymbol{x}^{(i,j)}, A_{ij}) + \lambda \| V \|_F^2$$

- LIBFM (by Steffen Rendle):
  - Alternating Minimization
  - SGD
  - Sampling
- Is inductive matrix completion a special case of factorization machine?

Graph-based Regularization

The similarity between users

- Assume we have m users with feature vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$
- Similarity of rows: Define the similarity matrix $S \in \mathbb{R}^{m \times m}$

$$S_{ij} = \text{similarity}(\boldsymbol{x}_i, \boldsymbol{x}_j),$$

  where $\boldsymbol{x}_i, \boldsymbol{x}_j$ are features for the $i(j)$-th rows.
- The similarity function can be defined by many ways, for example,

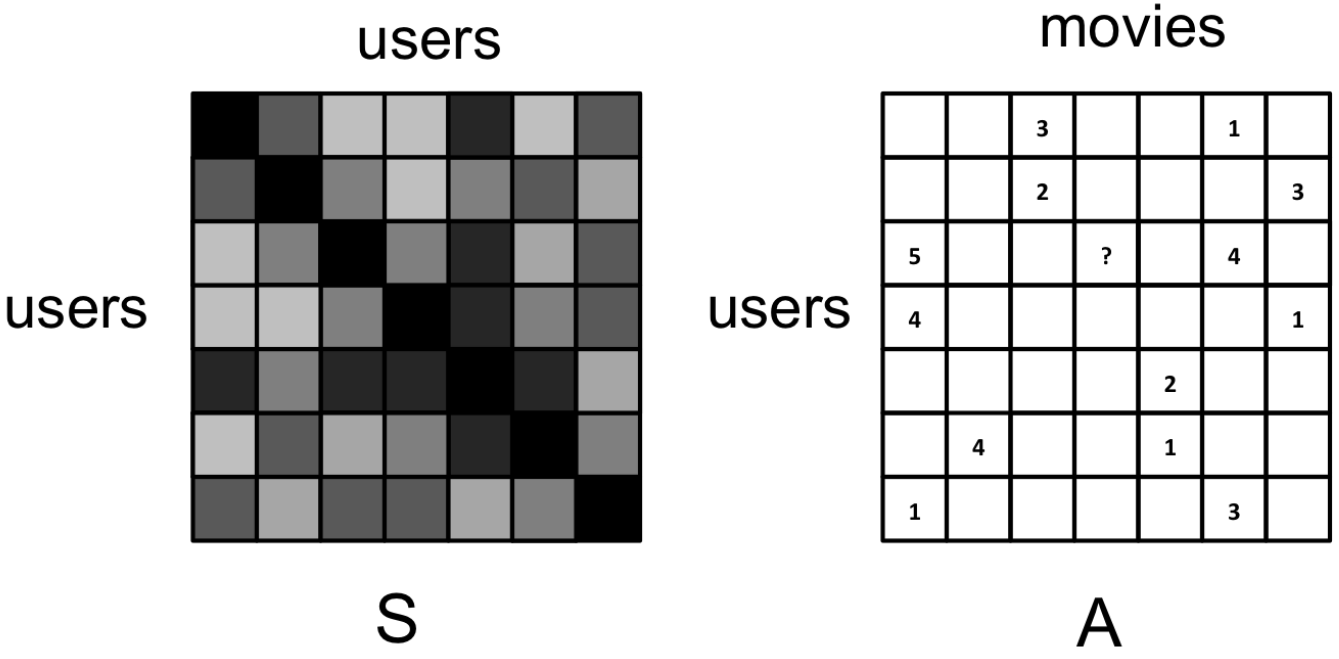
$$\text{similarity}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\gamma \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2)$$

The similarity graph

- S is a dense $m \times m$ matrix $\Rightarrow$ High computational cost
- Usually, a sparse similarity graph is preferred
- Several approaches:
  - K-nearest neighbor graph
  - Thresholding S matrix

Matrix Completion with Graph Regularization

- Given, a partially observed rating matrix $A \in \mathbb{R}^{m \times n}$ and a similarity graph between users $S \in \mathbb{R}^{m \times m}$
- The similarity graph can be
  - Constructed from features
  - Obtained by social activities
- Goal: predict unknown ratings



Matrix Completion with Graph Regularization

- Two approaches:
  - Average-based Regularization
  - Individual-based Regularization
- Discussed in
  - Ma et al., "Recommender Systems with Social Regularization". In WSDM 2011
  - Rao et al., "Collaborative Filtering with Graph Information: Consistency and Scalable Methods". In NIPS 2015

Average-based Regularization

- Compute matrices $U \in \mathbb{R}^{m \times k}$ and $V \in \mathbb{R}^{n \times k}$ by solving

$$\underset{U \in \mathbb{R}^{m \times k}, V \in \mathbb{R}^{n \times k}}{\operatorname{argmin}} \sum_{i,j \in \Omega} (A_{ij} - (UV^T)_{ij})^2 + \frac{\alpha}{2} \sum_{i=1}^{m} \| \boldsymbol{u}_i - \frac{1}{\sum_k S_{ik}} \sum_{j=1}^{m} S_{ij} \boldsymbol{u}_j \|^2 + \lambda \| U \|_F^2 + \lambda \| V \|_F^2$$

where $\boldsymbol{u}_i$ is the i-th row of U.

Individual-based Regularization

- Compute matrices $U \in \mathbb{R}^{m \times k}$ and $V \in \mathbb{R}^{n \times k}$ by solving

$$\underset{U \in \mathbb{R}^{m \times k}, V \in \mathbb{R}^{n \times k}}{\operatorname{argmin}} \sum_{i,j \in \Omega} (A_{ij} - (UV^T)_{ij})^2 + \sum_{i,j} \frac{S_{ij}}{2} \|\boldsymbol{u}_i - \boldsymbol{u}_j\|^2 + \lambda \|U\|_F^2 + \lambda \|V\|_F^2$$

  where $\boldsymbol{u}_i$ is the i-th row of U.

- Can be written as

$$\underset{U \in \mathbb{R}^{m \times k}, V \in \mathbb{R}^{n \times k}}{\operatorname{argmin}} \sum_{i,j \in \Omega} (A_{ij} - (UV^T)_{ij})^2 + \operatorname{trace}(U^T(D - S)U) + \lambda \|U\|_F^2 + \lambda \|V\|_F^2$$

  where D is a diagonal matrix with $D_{ii} = \sum_j S_{ij}$.

- The matrix $L = D - S$ is called the Laplacian matrix

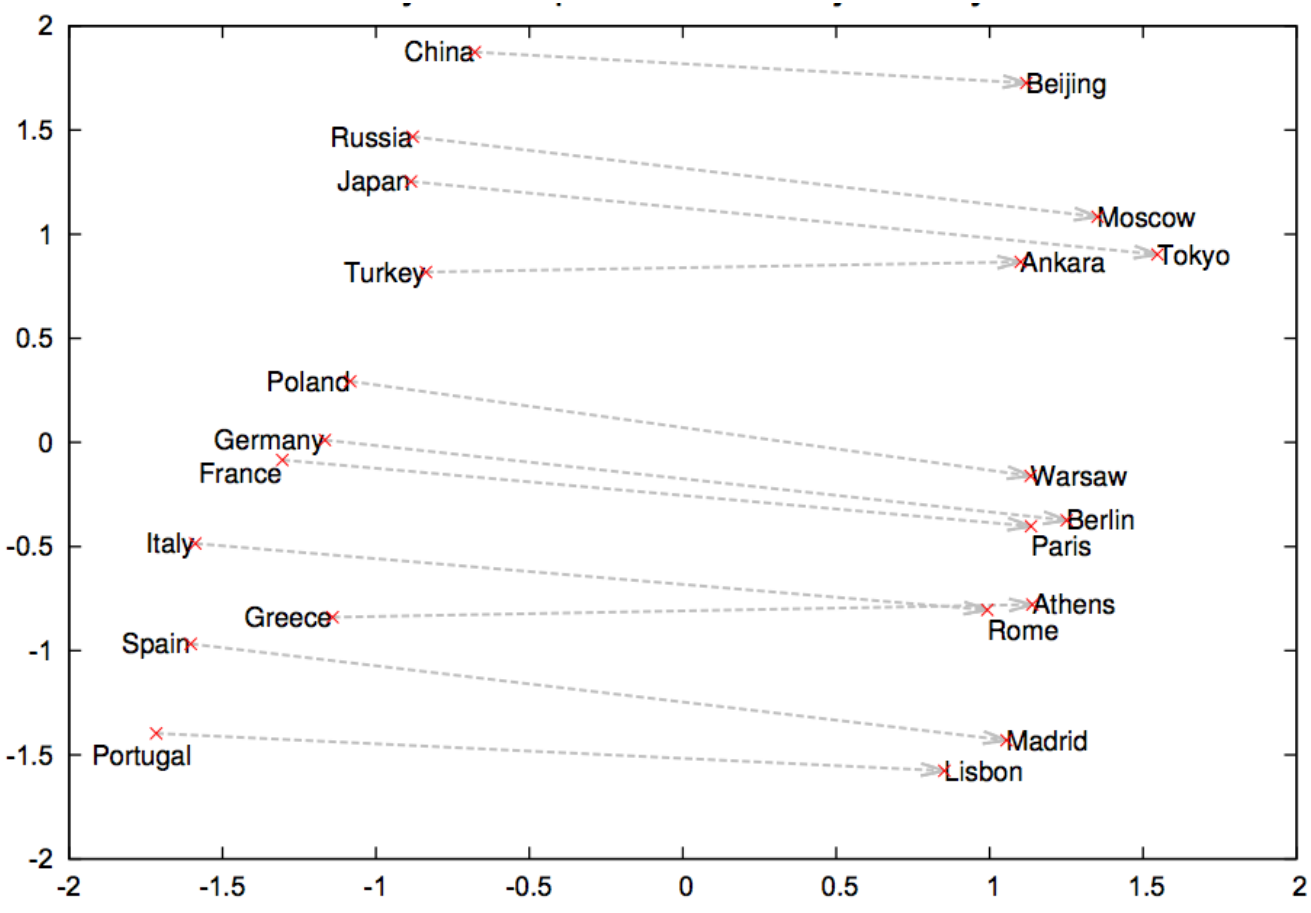
Word Embedding

Word Embedding

- For each word i, learn the low dimensional embedding $\boldsymbol{w}_i \in \mathbb{R}^k$

  (Mikolov et al., "Distributed Representations of Words and Phrases and their Compositionality". In NIPS, 2013)

Google word2vec (2013)



A Matrix Factorization View

- Skip-gram: two words i, j co-occurent if i, j occur in the same window (typically 5)

  to learn the representations that are good at predicting nearby words

- Counting the co-occurent probability:

$$A_{ij} = \text{co-occurrence count of words } i, j$$

- Original objective:

$$\max_W \sum_{i,j} A_{ij} \log(P(i \mid j)),$$

  where $P(i \mid j) = \frac{\exp(\boldsymbol{w}_i^T \boldsymbol{w}_j)}{\sum_{p=1}^n \exp(\boldsymbol{w}_p^T \boldsymbol{w}_j)}$.

- Very expensive to solve:

  Even though A is sparse, each loss requires computing $\sum_{p=1}^n \exp(\boldsymbol{w}_p^T \boldsymbol{w}_j)$

Approximate as matrix factorization

- Discussed in (Levy and Goldberg, "Neural Word Embedding as Implicit Matrix Factorization", NIPS 2014)
- Define the probability

$$P(co-occurent \mid i,j) = 1/(1+e^{-\boldsymbol{w}_i^T \boldsymbol{w}_j})$$

- Minimize

$$\min_W \sum_{i,j} A_{ij} \big(\log(\frac{1}{1+e^{-\boldsymbol{w}_i^T \boldsymbol{w}_j}}) + kE_{c\sim P(c)}[\log(\frac{1}{1+e^{\boldsymbol{w}_c^T \boldsymbol{w}_j}})] \big)$$

  where P can be any word distribution (uniform or empirical uni-gram distribution, or uigram$^{3/4}$)
- Solved by SGD.

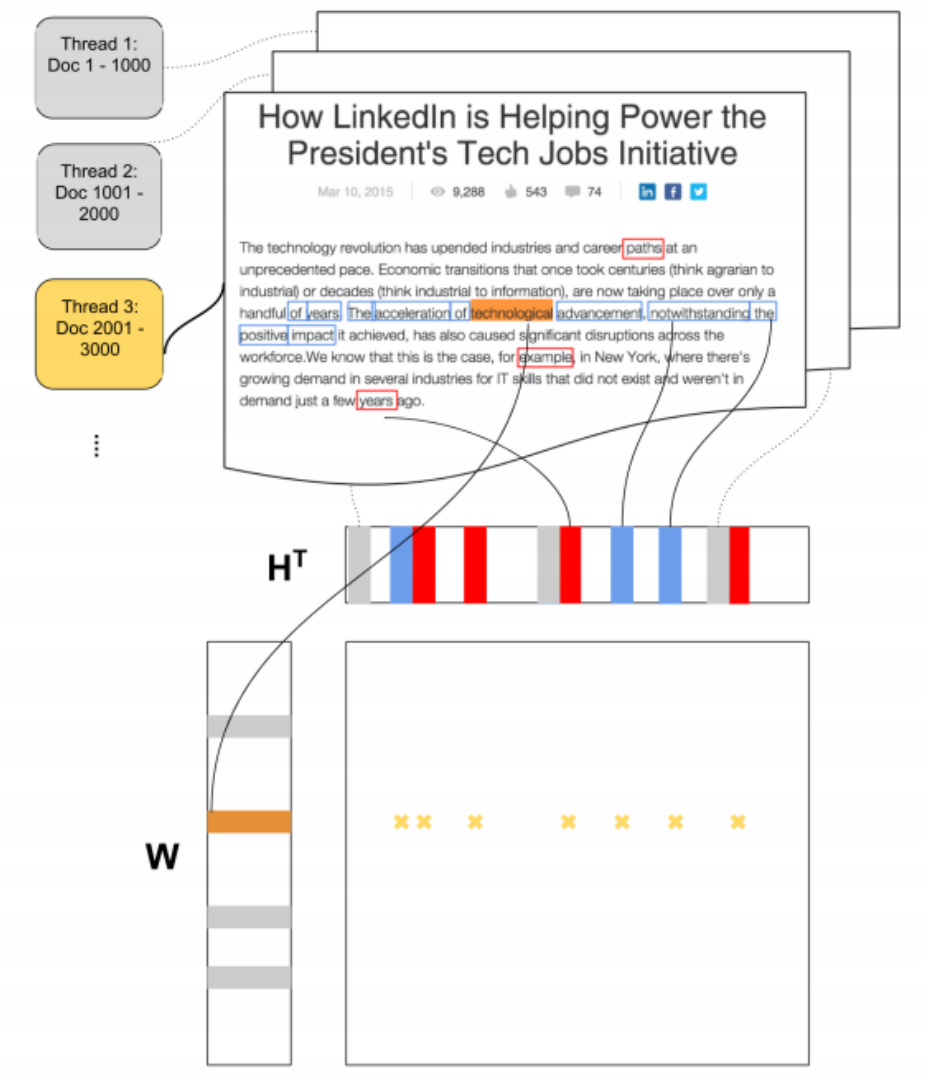
Approximate as matrix factorization

- Equivalent to the following minimization problem (?)

$$\min_{W} \sum_{i,j:A_{ij}>0} A_{ij} \log\left(\frac{1}{1+e^{-\boldsymbol{w}_i^T \boldsymbol{w}_j}}\right) + \sum_{i,j} C_j \log\left(\frac{1}{1+e^{\boldsymbol{w}_i^T \boldsymbol{w}_j}}\right)$$

where $C_j = \frac{k(\sum_i A_{ij})}{n}$ (assume uniform sampling distribution).

Google Word2vec Training



Coming up

Questions?