Matrix Factorization Methods

Irwin King

Department of Computer Science & Engineering
The Chinese University of Hong Kong

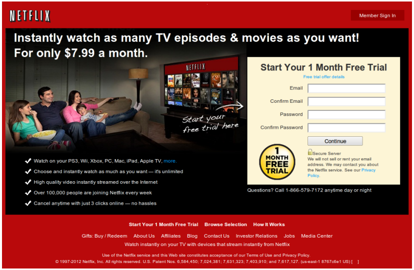
8/4/2013

Outline

Outline

The Netflix Problem

- Netflix database
  - About half a million users
  - About 18,000 movies
- People assign ratings to movies
- A sparse matrix



The Netflix Problem

- Netflix database
  - Over 480,000 users
  - About 18,000 movies
  - Over 100,000,000 ratings
- People assign ratings to movies
- A sparse matrix
  - Only 1.16% of the full matrix is observed

$$\begin{bmatrix} x & & x & & x \\ & x & & & x \\ x & & x & & \\ x & & & & x \\ & & x & x & x \end{bmatrix}$$

The Netflix Problem

- Netflix database
  - About half a million users
  - About 18,000 movies
- People assign ratings to movies
- A sparse matrix

$$\begin{bmatrix} x & & x & & x \\ & x & & & x \\ x & & x & & \\ x & & & & x \\ & & x & x & x \end{bmatrix}$$

Challenge

Complete the "Netflix Matrix"

Many such problems: collaborative filtering, partially filled out surveys ...

Matrix Completion

- Matrix $X \in \mathbb{R}^{N \times M}$
- Observe subset of entries
- Can we guess the missing entries?

$$\begin{bmatrix} x & ? & x & ? & x \\ ? & x & ? & ? & x \\ x & ? & x & ? & ? \\ x & ? & ? & ? & x \\ ? & ? & x & x & x \end{bmatrix}$$

Matrix Completion

- Matrix $X \in \mathbb{R}^{N\times M}$
- Observe subset of entries
- Can we guess the missing entries?

$$\begin{bmatrix} x & ? & x & ? & x \\ ? & x & ? & ? & x \\ x & ? & x & ? & ? \\ x & ? & ? & ? & x \\ ? & ? & x & x & x \end{bmatrix}$$

Everyone would agree this looks impossible.

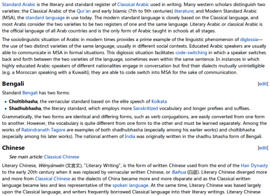
Massive High-dimensional Data

Engineering/scientific applications

Unknown matrix often has (approx.) low rank.

Images

Videos



Standard Arabic is the literary and standard register of Classical Arabic used in writing. Many western scholars distinguish two varieties: the Classical Arabic of the Qur'an and early Islamic (7th to 9th centuries) literature; and Modern Standard Arabic (MSA), the standard language in use today. The modern standard language is closely based on the Classical language, and most Arabs consider the two varieties to be two registers of one and the same language. Literary Arabic or classical Arabic is the official language of all Arab countries and is the only form of Arabic taught in schools at all stages.

The sociolinguistic situation of Arabic in modern times provides a prime example of the linguistic phenomenon of diglossia—the use of two distinct varieties of the same language, usually in different social contexts. Educated Arabic speakers are usually able to communicate in MSA in formal situations. This diglossic situation facilitates code-switching in which a speaker switches back and forth between the two varieties of the language, sometimes even within the same sentence. In instances in which highly educated Arabic-speakers of different nationalities engage in conversation but find their dialects mutually unintelligible (e.g. a Moroccan speaking with a Kuwaiti), they are able to code switch into MSA for the sake of communication.

Bengali [edit]

Standard Bengali has two forms:

- **Choltibhasha**, the vernacular standard based on the elite speech of Kolkata
- **Shadhubhasha**, the literary standard, which employs more Sanskritized vocabulary and longer prefixes and suffixes.

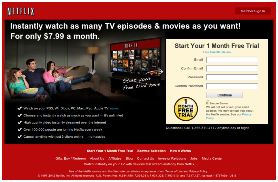
Grammatically, the two forms are identical and differing forms, such as verb conjugations, are easily converted from one form to another. However, the vocabulary is quite different from one form to the other and must be learned separately. Among the works of Rabindranath Tagore are examples of both shadhubhasha (especially among his earlier works) and choltibhasha (especially among his later works). The national anthem of India was originally written in the shadhu bhasha form of Bengali.

Chinese [edit]

See main article: Classical Chinese

Literary Chinese, *Wényánwén* (文言文), "Literary Writing", is the form of written Chinese used from the end of the Han Dynasty to the early 20th century when it was replaced by vernacular written Chinese, or *Baihua* (白話). Literary Chinese diverged more and more from Classical Chinese as the dialects of China became more and more disparate and as the Classical written language became less and less representative of the spoken language. At the same time, Literary Chinese was based largely upon the Classical language, and writers frequently borrowed Classical language into their literary writings. Literary Chinese

Text



Web data

High-dimensionality but often low-dimensional structure

Recovery Algorithm

Observation

Try to recover a lowest complexity (rank) matrix that agrees with the observation.

Recovery by minimum complexity (assuming no noise)

$$\begin{aligned} \text{minimize} \quad & \text{rank}(\hat{X}) \\ \text{subject to} \quad & \hat{X}_{ij} = X_{ij} \quad (i,j) \in \mathcal{Q}_{obs} \end{aligned}$$

Recovery Algorithm

Observation

Try to recover a lowest complexity (rank) matrix that agrees with the observation.

Recovery by minimum complexity

$$\begin{array}{rl} \text{minimize} & \text{rank}(\hat{X}) \\ \text{subject to} & \hat{X}_{ij} = X_{ij} \quad (i,j) \in \mathcal{Q}_{obs} \end{array}$$

- NP hard: not feasible for $N > 10$!
- Resort to other approaches
  - Select a low rank K, and approximate X by a rank K matrix $\hat{X}$.

Low Rank Factorization

- Assume X can be recovered by a rank K matrix $\hat{X}$
- Then $\hat{X}$ can be factorized into the product of $U \in \mathbb{R}^{K \times N}$, $V \in \mathbb{R}^{K \times M}$

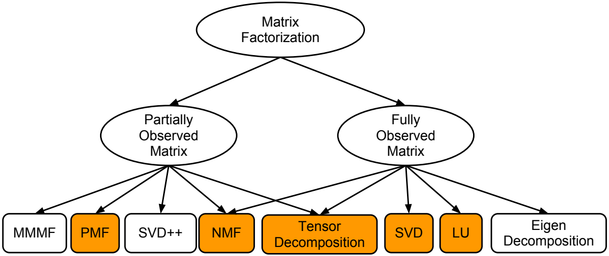
$$\hat{X} = U^T V.$$

- Define $\mathcal{E}$ to be a loss function

Recovery by rank K matrix

$$\begin{aligned} \text{minimize} \quad & \sum_{i,j \in \mathcal{Q}_{obs}} \mathcal{E}(\hat{X}_{ij} - X_{ij}) \\ \text{subject to} \quad & \hat{X} = U^T V \end{aligned}$$

Overview of Matrix Factorization Methods



- Some methods are traditional mathematical way of factorizing a matrix.
  - SVD, LU, Eigen Decomposition
- Some methods are used to factorize partially observed matrix.
  - PMF, SVD++, MMMF
- Some methods have multiple applications.
  - NMF in image processing
  - NMF in collaborative filtering

Outline

LU Decomposition

The LU Decomposition factors a matrix as the product of a lower triangular matrix (L) and an upper triangular matrix (U).

$$\underbrace{\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}}_{A} = \underbrace{\begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix}}_{L} \underbrace{\begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix}}_{U}$$

- Lower triangular matrix (L): Every entry above the main diagonal are zero
- Upper triangular matrix (U): Every entry below the main diagonal are zero

LU Decomposition

- LU Decomposition is useful when
  - Solving a system of linear equations
  - Inverting a matrix
  - Computing the determinant of a matrix
- LU Decomposition can be computed using a method similar to Gaussian Elimination

LU Decomposition

- Computing LU Decomposition of a matrix A
  - Using Gaussian elimination to compute U
  - Apply inverse operation on the corresponding entry to I to get L

$$\overset{A}{\begin{bmatrix} 1 & 2 & 3 \\ 2 & -4 & 6 \\ 3 & -9 & -3 \end{bmatrix}} \xrightarrow[\text{R3} - 3\text{R1}]{\text{R2} - 2\text{R1}} \begin{bmatrix} 1 & 2 & 3 \\ 0 & -8 & 0 \\ 0 & -15 & -12 \end{bmatrix} \xrightarrow{\text{R2 * (-1/8)}} \begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 0 \\ 0 & -15 & -12 \end{bmatrix} \xrightarrow{\text{R3 + 15R2}} \begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 0 \\ 0 & 0 & -12 \end{bmatrix} \xrightarrow{\text{R3 * (-1/12)}} \overset{U}{\begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}$$

LU Decomposition

- Computing LU Decomposition of a matrix A
  - Using Gaussian elimination to compute U
  - Apply inverse operation on the corresponding entry to I to get L

$$\underset{I}{\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix}} \xrightarrow{\text{R3 * (-12)}} \begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & -12\end{bmatrix} \xrightarrow{\text{R3 - 15R2}} \begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & -15 & -12\end{bmatrix} \xrightarrow{\text{R2 * (-8)}} \begin{bmatrix}1 & 0 & 0\\ 0 & -8 & 0\\ 0 & -15 & -12\end{bmatrix} \xrightarrow[\text{R3 + 3R1}]{\text{R2 + 2R1}} \underset{L}{\begin{bmatrix}1 & 0 & 0\\ 2 & -8 & 0\\ 3 & -15 & -12\end{bmatrix}}$$

LU Decomposition

- Computing LU Decomposition of a matrix A
  - Using Gaussian elimination to compute U
  - Apply inverse operation on the corresponding entry to I to get L

$$\underbrace{\begin{bmatrix} 1 & 2 & 3 \\ 2 & -4 & 6 \\ 3 & -9 & -3 \end{bmatrix}}_{A} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 2 & -8 & 0 \\ 3 & -15 & -12 \end{bmatrix}}_{L} \underbrace{\begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{U}$$

Outline

Singular Value Decomposition

Singular Value Decomposition

The Singular Value Decomposition (SVD) of an $N \times M$ matrix A is a factorization of the form

$$A = U\Sigma V^*$$

- V^* is the conjugate transpose of V.
- $U \in \mathbb{R}^{N\times N}$ is unitary matrix, i.e. $UU^* = I$.
- $\Sigma \in \mathbb{R}^{N\times M}$ is rectangular diagonal matrix with real entries.
- $V^* \in \mathbb{R}^{M\times M}$ is unitary matrix, i.e. $VV^* = I$.

SVD v.s. Eigen Decomposition

Singular Value Decomposition

The Singular Value Decomposition (SVD) of an $N \times M$ matrix A is a factorization of the form

$$A = U\Sigma V^*$$

- Diagonal entries of Σ are called singular values of A.
- Columns of U and V are called left singular vectors and right singular vectors of A, respectively.
- The singular values Σ_{ii}s are arranged in descending order in Σ.

SVD v.s. Eigen Decomposition

Singular Value Decomposition

The Singular Value Decomposition (SVD) of an $N \times M$ matrix A is a factorization of the form

$$A = U\Sigma V^*$$

- The left singular vectors of A are eigenvectors of AA^*, because

$$AA^* = (U\Sigma V^*)(U\Sigma V^*)^* = U\Sigma\Sigma^T U^*$$

- The right singular vectors of A are eigenvectors of A^*A, because

$$A^*A = (U\Sigma V^*)^*(U\Sigma V^*) = V\Sigma^T\Sigma V$$

- The singular values of A are the square roots of eigenvalues of both AA^* and A^*A.

SVD as Low Rank Approximation

Low Rank Approximation

$$\begin{aligned} \operatorname{argmin}_{\tilde{A}} \quad & \|A - \tilde{A}\|_{Fro} \\ \text{s.t.} \quad & \operatorname{Rank}(\tilde{A}) = r \end{aligned}$$

SVD gives the optimal solution.

Solution (Eckart-Young Theorem)

Let $A = U\Sigma V^*$ be the SVD for A, and $\tilde{\Sigma}$ is the same as Σ by keeping the largest r singular values. Then,

$$\tilde{A} = U\tilde{\Sigma}V^*$$

is the solution to the above problem.

SVD as Low Rank Approximation

Solution (Eckart-Young Theorem)

Let $A = U\Sigma V^*$ be the SVD for A, and $\tilde{\Sigma}$ is the same as Σ by keeping the largest r singular values. Then,

$$\tilde{A} = U\tilde{\Sigma}V^*$$

is the solution to the above problem.

- It works when A is fully observed.
- What if A is only partially observed?

Low Rank Approximation for Partially Observed Matrix

Low Rank Approximation for Partially Observed Matrix

$$\begin{aligned} \text{argmin}_{\tilde{A}} \quad & \sum_{i=1}^{N} \sum_{j=1}^{M} I_{ij}(A_{ij} - \tilde{A}_{ij})^2 \\ \text{s.t.} \quad & \text{Rank}(\tilde{A}) = r \end{aligned}$$

- I_{ij} is the indicator that equals 1 if A_{ij} is observed and 0 otherwise.
- We consider only the observed entries.
- A natural probabilistic extension of the above formulation is Probabilistic Matrix Factorization.

Outline

Probabilistic Matrix Factorization

- A popular collaborative filtering (CF) method
- Follow the low rank matrix factorization framework

Collaborative Filtering

Collaborative Filtering

The goal of collaborative filtering (CF) is to infer user preferences for items given a large but incomplete collection of preferences for many users.

- For example:
  - Suppose you infer from the data that most of the users who like "Star Wars" also like "Lord of the Rings" and dislike "Dune".
  - Then if a user watched and liked "Star Wars" you would recommend him/her "Lord of the Rings" but not "Dune".
- Preferences can be explicit or implicit:
  - Explicit preferences
    - Ratings assigned to items
    - Facebook "Like", Google "Plus"
  - Implicit preferences
    - Catalog browse history
    - Items rented or bought by users

Collaborative Filtering vs. Content Based Filtering

- Collaborative Filtering
  - User preferences are inferred from ratings
  - Item features are inferred from ratings
  - Cannot recommend new items
  - Very effective with sufficient data

- Content Based Filtering
  - Analyze the content of the item
  - Match the item features with users preferences
  - Item features are hard to extract
    - Music, Movies
  - Can recommend new items

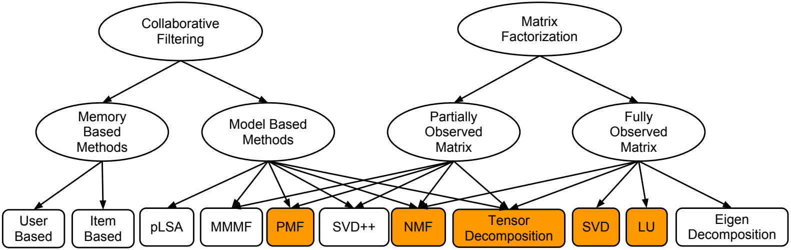
CF as Matrix Completion

- CF can be viewed as a matrix completion problem

$$\begin{array}{cc} & \text{Items} \\ \text{Users} & \begin{bmatrix} x & & x & & x \\ & x & & & x \\ x & & x & & \\ x & & & & x \\ & & x & x & x \end{bmatrix} \end{array}$$

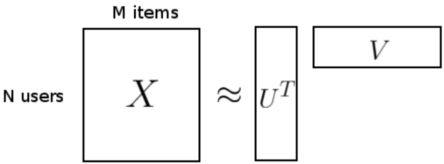
- Task: given a user/item matrix with only a small subset of entries present, fill in (some of) the missing entries.
- PMF approach: low rank matrix factorization.

Collaborative Filtering and Matrix Factorization



- Collaborative filtering can be formulated as a matrix factorization problem.
- Many matrix factorization methods can be used to solve collaborative filtering problem.
- The above is only a partial list.

Notations



- Suppose we have M items, N users and integer rating values from 1 to D.
- Let ijth entry of X, X_{ij}, be the rating of user i for item j.
- $U \in \mathbb{R}^{K \times N}$ is latent user feature matrix, U_i denote the latent feature vector for user i.
- $V \in \mathbb{R}^{K \times M}$ is latent item feature matrix, V_j denote the latent feature vector for item j.

Matrix Factorization: the Non-probabilistic View

- To predict the rating given by user i to item j,

$$\hat{R}_{ij} = U_i^T V_j = \sum_k U_{ik} V_{jk}$$

- Intuition
  - The item feature vector can be viewed as the input.
  - The user feature vector can be viewed as the weight vector.
  - The predicted rating is the output.
  - Unlike in linear regression, where inputs are fixed and weights are learned, we learn *both* the weights and the input by minimizing squared error.
  - The model is symmetric in items and users.

Probabilistic Matrix Factorization

- PMF is a simple probabilistic linear model with Gaussian observation noise.
- Given the feature vectors for the user and the item, the distribution of the corresponding rating is:

$$P(R_{ij}|U_i, V_j, \sigma^2) = \mathcal{N}(R_{ij}|U_i^T V_j, \sigma^2)$$

- The user and item feature vectors adopt zero-mean spherical Gaussian priors:

$$\begin{aligned} P(U|\sigma_U^2) &= \prod_{i=1}^{N} \mathcal{N}(U_i|\mathbf{0}, \sigma_U^2 \mathbf{I}) \\ P(V|\sigma_V^2) &= \prod_{j=1}^{M} \mathcal{N}(V_j|\mathbf{0}, \sigma_V^2 \mathbf{I}) \end{aligned}$$

Probabilistic Matrix Factorization

- Maximum A Posterior (MAP): Maximize the log-posterior over user and item features with fixed hyperparameters.
- MAP is equivalent to minimizing the following objective function:

PMF objective function

$$\mathcal{E} = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}(R_{ij} - U_i^T V_j)^2 + \frac{\lambda_U}{2}\sum_{i=1}^{N}\|U_i\|_{Fro}^2 + \frac{\lambda_V}{2}\sum_{j=1}^{M}\|V_j\|_{Fro}^2$$

Probabilistic Matrix Factorization

PMF objective function

$$\mathcal{E} = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}(R_{ij} - U_i^T V_j)^2 + \frac{\lambda_U}{2}\sum_{i=1}^{N}\|U_i\|_{Fro}^2 + \frac{\lambda_V}{2}\sum_{j=1}^{M}\|V_j\|_{Fro}^2$$

- $\lambda_U = \sigma^2/\sigma_U^2, \lambda_V = \sigma^2/\sigma_V^2$ and I_{ij} is indicator of whether user i rated item j.
- First term is the sum-of-squared-errors.
- Second and third term are quadratic regularization term to avoid over-fitting problem.

Probabilistic Matrix Factorization

PMF objective function

$$\mathcal{E} = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}(R_{ij} - U_i^T V_j)^2 + \frac{\lambda_U}{2}\sum_{i=1}^{N}\|U_i\|_{Fro}^2 + \frac{\lambda_V}{2}\sum_{j=1}^{M}\|V_j\|_{Fro}^2$$

- Non-convex problem, global minima generally not achievable
- Alternating update U and V, fix one while updating the another
- Use gradient descent

$$U_i \leftarrow U_i - \eta\frac{\partial \mathcal{E}}{\partial U_i}; \qquad \frac{\partial \mathcal{E}}{\partial U_i} = \sum_{j=1}^{M} I_{ij}(U_i^T V_j - R_{ij})V_j + \lambda_U U_i$$

$$V_j \leftarrow V_j - \eta\frac{\partial \mathcal{E}}{\partial V_j}; \qquad \frac{\partial \mathcal{E}}{\partial V_j} = \sum_{i=1}^{N} I_{ij}(U_i^T V_j - R_{ij})U_i + \lambda_V V_j$$

Probabilistic Matrix Factorization

PMF objective function

$$\mathcal{E} = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}(R_{ij} - U_i^T V_j)^2 + \frac{\lambda_U}{2}\sum_{i=1}^{N}\|U_i\|_{Fro}^2 + \frac{\lambda_V}{2}\sum_{j=1}^{M}\|V_j\|_{Fro}^2$$

- If all ratings were observed, the objective reduces to the SVD objective in the limit of prior variances going to infinity.
- PMF can be viewed as a probabilistic extension of SVD.

Probabilistic Matrix Factorization

A trick to improve stability

- Map ratings to $[0, 1]$ by $(R_{ij} - 1)/(D - 1)$
- Pass $U_i^T V_j$ through logistic function

$$g(x) = \frac{1}{1 + \exp(-x)}$$

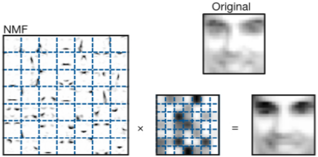
PMF objective function

$$\mathcal{E} = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}(R_{ij} - g(U_i^T V_j))^2 + \frac{\lambda_U}{2}\sum_{i=1}^{N}\|U_i\|_{Fro}^2 + \frac{\lambda_V}{2}\sum_{j=1}^{M}\|V_j\|_{Fro}^2$$

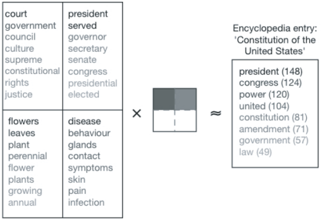
Outline

Non-negative Matrix Factorization

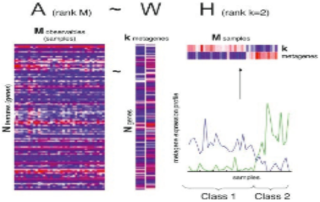
NMF is a popular method that is widely used in:



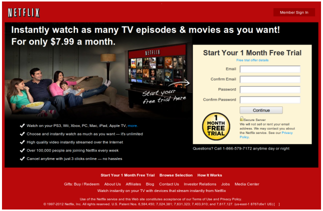
Images Mining



Text Mining



Metagenes Study



Collaborative Filtering

Non-negative Matrix Factorization

- NMF fits in the low rank matrix factorization framework with additional non-negativity constraints.
- NMF can only factorize a Non-negative matrix $A \in \mathbb{R}^{N \times M}$ into basis matrix $W \in \mathbb{R}^{N \times K}$ and weight matrix $H \in \mathbb{R}^{K \times M}$

$$\begin{aligned} & A \approx WH \\ \text{s.t.} \quad & W, H \geq \mathbf{0} \end{aligned}$$

Interpretation with NMF

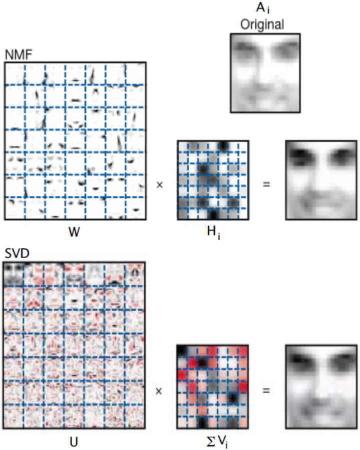
- Columns of W are the underlying basis vectors, i.e., each of the M columns of A can be built from K columns of W.
- Columns of H give the weights associated with each basis vector.

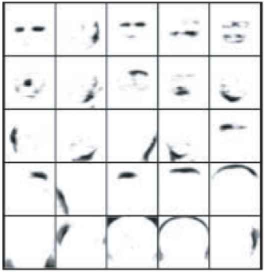
$$Ae_1 = WH_{*1} = [W_1]\, H_{11} + [W_2]\, H_{21} + \cdots + [W_K]\, H_{K1}$$

- $W, H \geq \mathbf{0}$ commands additive parts-based representation.

NMF in Image Mining

Additive parts-based representation





NMF in Image Mining

- In image processing, we often assume Poisson Noise

NMF Poisson Noise

$$\begin{aligned} \min \quad & \sum_{i,j} \left(A_{ij} \log \frac{A_{ij}}{[WH]_{ij}} - A_{ij} + [WH]_{ij}\right) \\ \text{s.t.} \quad & W, H \geq \mathbf{0} \end{aligned}$$

- Objective function can be changed to other form, the non-negative constraint is more important than the form of the objective function

NMF Gaussian Noise

$$\begin{aligned} \min \quad & \|A - WH\|_{Fro}^2 \\ \text{s.t.} \quad & W, H \geq \mathbf{0} \end{aligned}$$

Inference of NMF

NMF Gaussian Noise

$$\begin{aligned} \min \quad & \|A - WH\|_{Fro}^2 \\ \text{s.t.} \quad & W, H \geq \mathbf{0} \end{aligned}$$

- Convex in W or H, but not both.
- Global min generally not achievable.
- Many number of unknowns: NK for W and MK for H

Inference of NMF

NMF Gaussian Noise

$$\begin{aligned} \min \quad & \|A - WH\|_{Fro}^2 \\ \text{s.t.} \quad & W, H \geq \mathbf{0} \end{aligned}$$

- Alternating gradient descent can get a local minima

$$F = \|A - WH\|_{Fro}^2$$

Algorithm 1 Alternating gradient descent

$W \leftarrow \text{abs}(\text{randn}(N, K))$
$H \leftarrow \text{abs}(\text{randn}(M, K))$
for $i = 1 : MaxIteration$ **do**
  $H \leftarrow H - \eta \frac{\partial F}{\partial H}$, $H \leftarrow H. * (H \geq 0)$
  $W \leftarrow W - \eta \frac{\partial F}{\partial W}$, $H \leftarrow W. * (W \geq 0)$
end for

Alternating Gradient Descent

$W \leftarrow \text{abs}(\text{randn}(N, K))$
$H \leftarrow \text{abs}(\text{randn}(M, K))$
for $i = 1 : MaxIteration$ **do**
  $H \leftarrow H - \eta \frac{\partial F}{\partial H}$, $H \leftarrow H. * (H \geq 0)$
  $W \leftarrow W - \eta \frac{\partial F}{\partial W}$, $H \leftarrow W. * (W \geq 0)$
end for

- Pros
    - works well in practice
    - speedy convergence
    - 0 elements not locked
- Cons
    - ad hoc nonnegativity: negative elements are set to 0
    - ad hoc sparsity: negative elements are set to 0
    - no convergence theory

Inference of NMF

$W \leftarrow \text{abs}(\text{randn}(N, K))$
$H \leftarrow \text{abs}(\text{randn}(M, K))$
for $i = 1 : MaxIteration$ **do**
  $H \leftarrow H - \eta \frac{\partial F}{\partial H}$, $H \leftarrow H. * (H \geq 0)$
  $W \leftarrow W - \eta \frac{\partial F}{\partial W}$, $H \leftarrow W. * (W \geq 0)$
end for

Observation

By choosing suitable η, we can change the additive update rule to multiplicative update rule. Non-negativity of W, H is guaranteed by the initial non-negativity. Ad hoc non-negativity is no longer needed.

NMF Gaussian Noise

$$\begin{aligned} \min \quad & \|A - WH\|_{Fro}^2 \\ \text{s.t.} \quad & W, H \geq \mathbf{0} \end{aligned}$$

Algorithm 2 Multiplicative update rule

$W \leftarrow \text{abs}(\text{randn}(N, K))$
$H \leftarrow \text{abs}(\text{randn}(M, K))$
for $i = 1 : MaxIteration$ **do**
  $H \leftarrow H.*(W^T A)./(W^T WH + 10^{-9})$
  $W \leftarrow W.*(AH^T)./(WHH^T + 10^{-9})$
end for

- Non-negativity is guaranteed.

Inference of NMF

NMF Poisson Noise

$$\begin{aligned} \min \quad & \sum_{i,j} \left(A_{ij} \log \frac{A_{ij}}{[WH]_{ij}} - A_{ij} + [WH]_{ij}\right) \\ \text{s.t.} \quad & W, H \geq \mathbf{0} \end{aligned}$$

Algorithm 3 Multiplicative update rule

$W \leftarrow \text{abs}(\text{randn}(N, K))$
$H \leftarrow \text{abs}(\text{randn}(M, K))$
for $i = 1 : MaxIteration$ **do**
  $H \leftarrow H. * (W^T(A./(WH + 10^{-9}))).\!/W^T ee^T$
  $W \leftarrow W. * ((A./(WH + 10^{-9}))H^T).\!/ee^T H^T$
end for

Multiplicative Update Rule

- Pros
  - Convergence theory: guaranteed to converge to fixed point
  - Good initialization of W, H speeds convergence and gets to better fixed point
- Cons
  - Fixed point may be local min or saddle point
  - Slow: many matrix multiplications at each iteration
  - 0 elements locked

Properties of NMF

- Basis vectors W_i are not orthogonal
- $W_k, H_k \geq 0$ have immediate interpretation
  - EX: large w_{ij}'s $\Rightarrow$ basis vector W_i is mostly about terms j
  - EX: h_{i1} denotes how much sample i is pointing in the "direction" of topic vector W_1

$$Ae_1 = WH_{*1} = [W_1]\, H_{11} + [W_2]\, H_{21} + \cdots + [W_K]\, H_{K1}$$

- NMF is algorithm-dependent: W, H not unique

Outline

- A *tensor* is a multidimensional array.
- *Tensors* are generalizations of vectors (first-order tensors) and matrices (second-order tensors) to arrays of higher orders ($N > 2$).

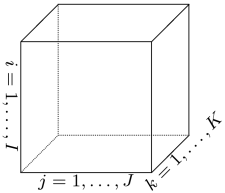


Fig.: A third-order tensor: $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$

Tensor Decomposition

Tensor Decomposition decomposes a *tensor* into a set of low-order matrices.

Two most widely used tensor decomposition methods:

1. Tucker decomposition
2. CANDECOMP/PARAFAC (CP) decomposition

Tucker decomposition

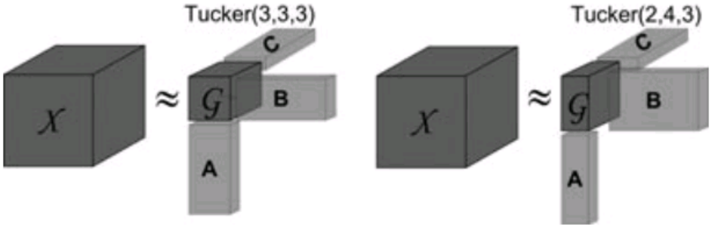
Tucker decomposition

Tucker decomposition decomposes a tensor into a set of matrices and one small core tensor.

Given a third-order tensor $\mathcal{X}^{I \times J \times K}$:

$$\mathcal{X}^{I \times J \times K} \approx \mathcal{G}^{L \times M \times N} \times_1 A^{I \times L} \times_2 B^{J \times M} \times_3 C^{K \times N},$$

where $\times_n$ means n-mode product.



CP Decomposition

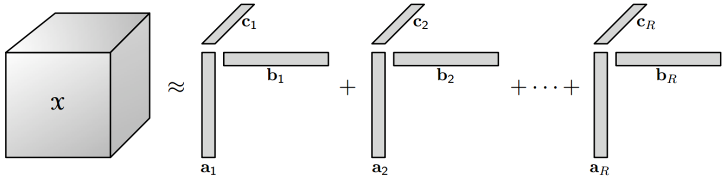
CP Decomposition

The CP decomposition factorizes a tensor into a sum of component rank-one tensors.

Given a third-order tensor $\mathcal{X}^{I\times J\times K}$:

$$\mathcal{X}^{I\times J\times K} \approx \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{b}_r,$$

where R is a positive integer, and $\mathbf{a}_r \in \mathbb{R}^I$, $\mathbf{b}_r \in \mathbb{R}^J$, $\mathbf{c}_r \in \mathbb{R}^K$, for $r = 1, ..., R$.



Outline

PMF Demonstration

- Application of PMF in Collaborative Filtering is used.
- Required Packages:
  - Python version 2.7
  - NumPy
  - SciPy
  - Matplotlib
- Script provided: pmf.py
  - Code credit: Danny Tarlow
  - Available at http://blog.smellthedata.com/2009/06/netflix-prize-tribute-recommendation.html

Required Packages

NumPy
    http://numpy.scipy.org/
SciPy
    http://www.scipy.org/
Matplotlib
    http://matplotlib.sourceforge.net/users/installing.html

PMF Demonstration

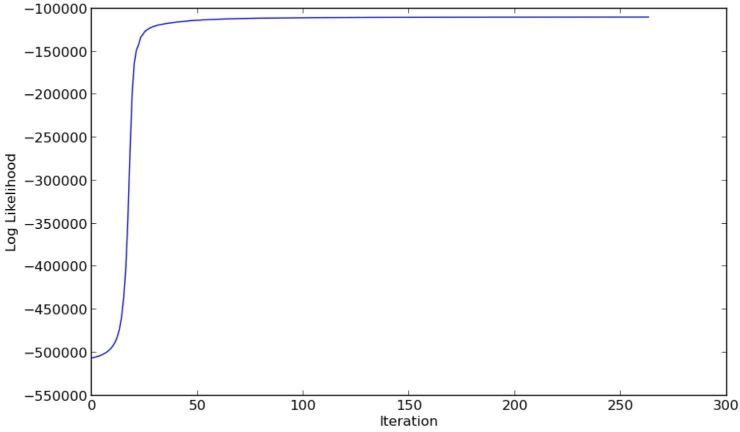
- Install all the required packages
- Run the script "python pmf.py"

What the script does?

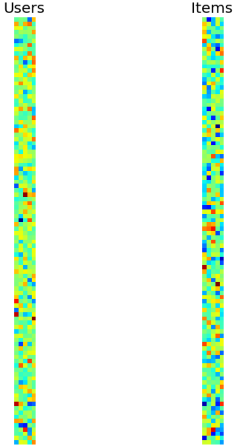
100 users' partial ratings on 100 items is simulated. 30% of the rating matrix is observed. Then PMF algorithm is performed on the generated dataset using a factorization dimension 5. When the learning is done, the convergency of the log-likelihood, user features, item features and predicted ratings are plotted.

PMF Demonstration

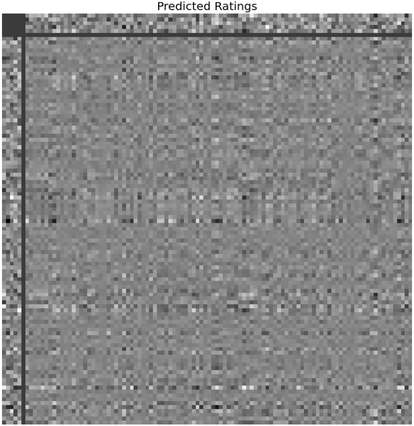
Figure: Convergency of the loglikelihood



PMF Demonstration



User and Item features



Predicted ratings

NMF Demonstration

- Application of NMF in image processing is used.
- Required Packages:
  - Python version 2.7
  - Python Image Library (PIL)
  - Python Matrix Factorization Module (PyMF)
  - NumPy
  - SciPy
- NMF toolbox in R:
  - `http://cran.r-project.org/web/packages/NMF/index.html`
  - `http://nmf.r-forge.r-project.org`

Required Packages

Python Image Library (PIL)

    http://www.pythonware.com/products/pil/index.htm

Python Matrix Factorization Module (PyMF)

    http://code.google.com/p/pymf/

NumPy

    http://numpy.scipy.org/

SciPy

    http://www.scipy.org/

NMF Demonstration

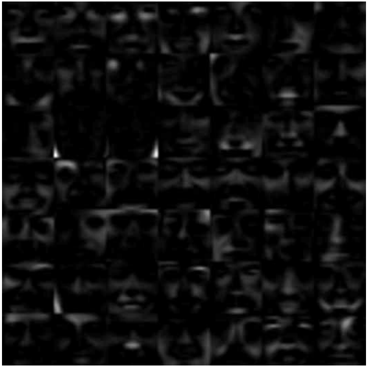
- Install all the required packages
- Run the script "python nmfdemo.py"

What the script does?

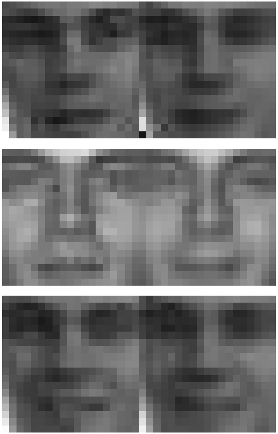
2429 19×19 face image is loaded into a matrix "data", one column per image. NMF is then performed on "data". The original image and the recovered image placed side by side is saved in folder "recover".

NMF Demonstration

Figure: 49 Basis Images (normalized)



Original Recovered



QA

Thanks for your attention!

References:

- Agarwal et al. Regression-based Latent Factor Models. KDD 2009.
- Chen et al. User Reputation in a Common Rating Environment. KDD 2011.
- Kolda et al. Tensor Decompositions and Applications. SIAM Review 2009.
- Koren et al. Factorization meets the neighborhood: a multifaceted collaborative filtering model. KDD 2008.
- Koren et al. Matrix Factorization Techniques for Recommender Systems. Computer 2009.
- Morup et al. Applications of tensor (multiway array) factorizations and decompositions in data mining. WIREs 2011.
- Rendle et al. Pairwise interaction tensor factorization for personalized tag recommendation. WSDM 2010.

Some of the slides are modified from materials:

- `http://videolectures.net/site/normal_dl/tag=623106/mlss2011_candes_lowrank_01.pdf`
- `http://www.cs.toronto.edu/~hinton/csc2515/notes/pmf_tutorial.pdf`
- `http://langvillea.people.cofc.edu/NISS-NMF.pdf`