Probabilistic Latent Semantic Analysis

Seungjin Choi

Department of Computer Science and Engineering
Pohang University of Science and Technology
77 Cheongam-ro, Nam-gu, Pohang 37673, Korea
seungjin@postech.ac.kr

Outline

- Singular value decomposition
- Latent semantic analysis (a.k.a. latent semantic indexing)
- Probabilistic latent semantic analysis (a.k.a. probabilistic latent semantic indexing)

Range Space and Null Space

Definition

For any $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, its range space $\mathcal{R}(\boldsymbol{A})$ and null space $\mathcal{N}(\boldsymbol{A})$ are defined as follows:

$$\begin{aligned} \mathcal{R}(\boldsymbol{A}) &= \{\boldsymbol{y} \in \mathbb{R}^m \mid \boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} \text{ for some } \boldsymbol{x} \in \mathbb{R}^n\} \\ \mathcal{N}(\boldsymbol{A}) &= \{\boldsymbol{x} \in \mathbb{R}^n \mid \boldsymbol{A}\boldsymbol{x} = 0\}. \end{aligned}$$

Examples: What are range space and null space of $\boldsymbol{A}$ and $\boldsymbol{B}$?

$$\boldsymbol{A} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad \boldsymbol{B} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

Linear Algebraic Equations

A linear algebraic equation has the form

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \tag{1}$$

where $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, $\boldsymbol{x} \in \mathbb{R}^{n}$, and $\boldsymbol{b} \in \mathbb{R}^{m}$.

Theorem

A solution of $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ exists if and only if $\boldsymbol{b} \in \mathcal{R}(\boldsymbol{A})$.

Theorem

Let $\boldsymbol{x}^{\circ}$ be a particular solution to (1), then
$\boldsymbol{x} = \boldsymbol{x}^{\circ} + \mathcal{N}(\boldsymbol{A}) = \{\boldsymbol{x}^{\circ} + \boldsymbol{w} \mid \boldsymbol{w} \in \mathcal{N}(\boldsymbol{A})\}$ is the general solution.

Remark: Note that $\mathcal{N}(\boldsymbol{A})$ is the set of homogeneous solutions to (1). Therefore the set of all solutions is the sum of a particular solution and the set of homogeneous solutions.

Singular Value Decomposition

Theorem (SVD)

Let $\boldsymbol{A} \in \mathbb{R}^{m \times n}$. Then there exist orthogonal matrices

$$\begin{aligned} \boldsymbol{U} &= [\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_m] \in \mathbb{R}^{m \times m} \\ \boldsymbol{V} &= [\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n] \in \mathbb{R}^{n \times n} \end{aligned}$$

such that

$$\boldsymbol{U}^T \boldsymbol{A} \boldsymbol{V} = diag(\sigma_1, \sigma_2, \ldots, \sigma_p) \in \mathbb{R}^{m \times n} \tag{2}$$

where $p = \min\{m, n\}$ and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p \geq 0$. The scalars $\{\sigma_i \in \mathbb{R}\}$ are called singular values.

$$\underline{\text{SVD } \boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top}$$

Low-Rank Approximation

Theorem (Eckart and Young, 1936)

Suppose that $\boldsymbol{A}$ *is an* $m \times n$ *matrix of rank* r*, with singular values* $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r$*. and SVD* $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$*. Then the best rank-*$k$ *approximation of* $\boldsymbol{A}$ *is* $\boldsymbol{A}_k = \boldsymbol{U}_1\boldsymbol{\Sigma}_1\boldsymbol{V}_1^\top$*, where*

$$\begin{aligned} \boldsymbol{U} &= [\boldsymbol{U}_1, \boldsymbol{U}_2], \\ \boldsymbol{V} &= [\boldsymbol{V}_1, \boldsymbol{V}_2], \\ \boldsymbol{\Sigma}_1 &= \mathit{diag}(\sigma_1, \ldots, \sigma_k). \end{aligned}$$

That is,

$$\|\boldsymbol{A} - \boldsymbol{A}_k\|_F = \min \left\{ \|\boldsymbol{A} - \widehat{\boldsymbol{A}}\|_F \,\middle|\, \widehat{\boldsymbol{A}} \in \mathbb{R}^{m \times n},\ \mathit{rank}(\widehat{\boldsymbol{A}}) = k \right\}.$$

Proof of Eckart-Young Theorem

Note that

$$\begin{aligned} \|\boldsymbol{A} - \widehat{\boldsymbol{A}}\|_F &= \|\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top - \widehat{\boldsymbol{A}}\|_F \\ &= \|\boldsymbol{\Sigma} - \underbrace{\boldsymbol{U}^\top \widehat{\boldsymbol{A}} \boldsymbol{V}}_{\boldsymbol{N}} \|_F. \end{aligned}$$

Direct calculations give

$$\begin{aligned} \|\boldsymbol{\Sigma} - \boldsymbol{N}\|_F^2 &= \sum_{i,j} |\Sigma_{i,j} - N_{i,j}|^2 \\ &= \sum_{i=1}^{r} |\sigma_i - N_{i,i}|^2 + \sum_{i>r} |N_{i,i}|^2 + \sum_{i \neq j} |N_{i,j}|^2, \end{aligned}$$

which is minimal when all the non-diagonal entries of N are equal to zero and so are all $N_{i,i}$ for $i > r$. The minimum of $\sum_{i=1}^{r} |\sigma_i - N_{i,i}|^2$ is attained when $\sigma_i = N_{i,i}$ for $i = 1, \ldots, k$ and all other $N_{i,i}$ are zero.

Least Squares Problem

Consider a matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ with $\text{Rank}(\boldsymbol{A}) = r$. Then

$$\begin{aligned} \mathcal{R}(\boldsymbol{A}) &= \mathcal{R}(\boldsymbol{U}_1) = \text{sp}\left[\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_r\right] \\ \mathcal{N}(\boldsymbol{A}) &= \text{sp}\left[\boldsymbol{v}_{r+1}, \boldsymbol{v}_{r+2}, \ldots, \boldsymbol{v}_n\right] = \mathcal{R}(\boldsymbol{V}_2). \end{aligned}$$

The least squares (LS) problem is as follows:

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} \left\| \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x} \right\|^2. \tag{3}$$

Setting $\nabla_x \left\| \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x} \right\|^2 = 0$ leads to

$$\boldsymbol{A}^T \boldsymbol{A}\boldsymbol{x} = \boldsymbol{A}^T \boldsymbol{b}. \tag{4}$$

Remarks

- From SVD, one can see that the particular solution to (4) is $\boldsymbol{x}^\circ = \boldsymbol{V}_1 \boldsymbol{\Sigma}^{-1} \boldsymbol{U}_1^T \boldsymbol{b}$.
- The complete solution is $\boldsymbol{x} = \boldsymbol{x}^\circ + \mathcal{N}(\boldsymbol{A}^T \boldsymbol{A})$. Note that $\mathcal{N}(\boldsymbol{A}^T \boldsymbol{A}) = \mathcal{N}(\boldsymbol{A}) = \text{sp}[\boldsymbol{V}_2]$. Hence $\boldsymbol{x} = \boldsymbol{V}_1 \boldsymbol{\Sigma}^{-1} \boldsymbol{U}_1^T \boldsymbol{b} + \boldsymbol{V}_2 \boldsymbol{w}$ for some $\boldsymbol{w} \in \mathbb{R}^r$.
- Since $\boldsymbol{V}_1 \boldsymbol{\Sigma}^{-1} \boldsymbol{U}_1^T \boldsymbol{b} \perp \boldsymbol{V}_2 \boldsymbol{w}$ for all $\boldsymbol{w}$, $\boldsymbol{x}^\circ = \boldsymbol{V}_1 \boldsymbol{\Sigma}^{-1} \boldsymbol{U}_1^T \boldsymbol{b}$ is the MMSE solution for which $\|\boldsymbol{x}\|$ is the smallest among all solutions.
- The pseudo-inverse of $\boldsymbol{A}$ is

$$\boldsymbol{A}^\dagger = \boldsymbol{V}_1 \boldsymbol{\Sigma}^{-1} \boldsymbol{U}_1^T. \tag{5}$$

- The condition number of $\boldsymbol{A}$ is given by $\frac{\sigma_1}{\sigma_r}$.

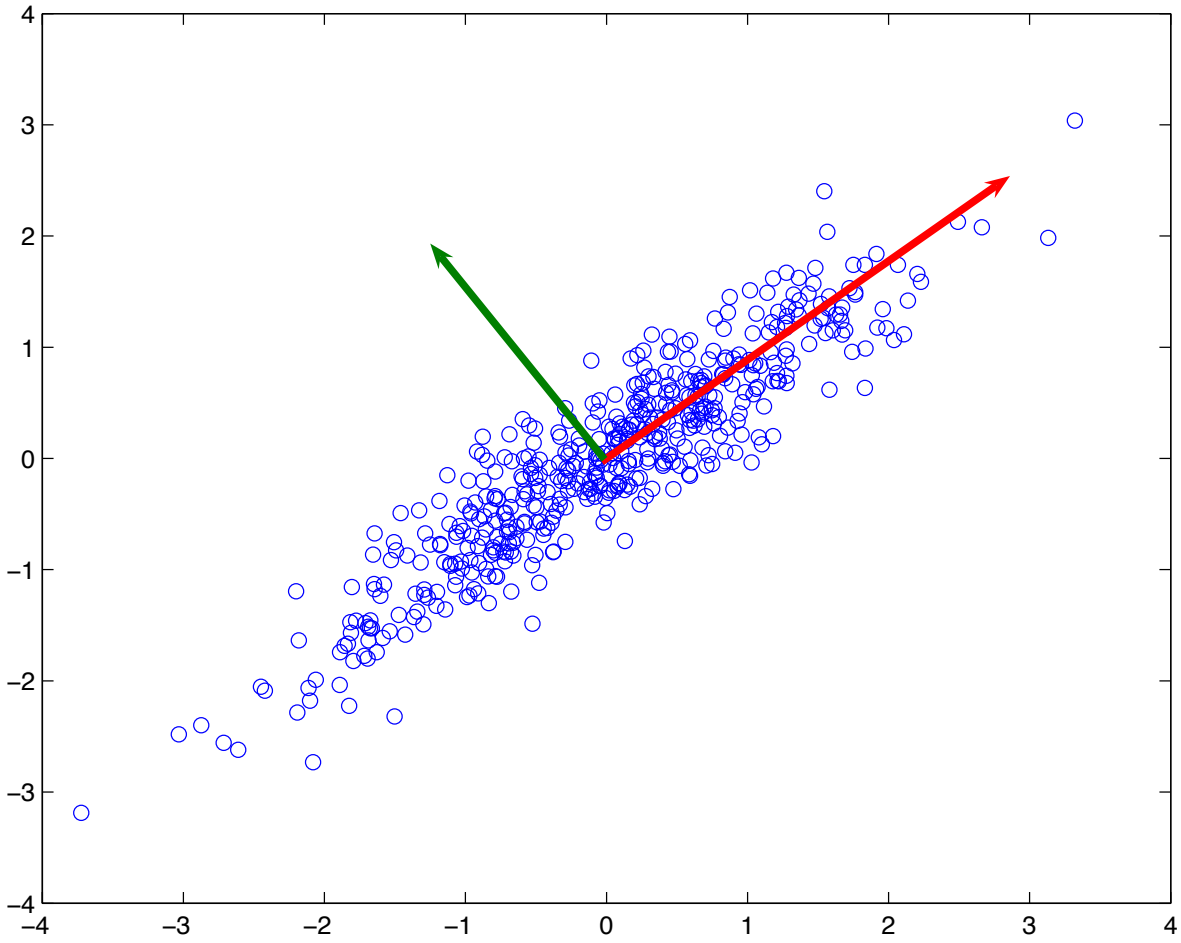
PCA

- Principal component analysis (PCA) is a well-established technique for dimension reduction. Its applications include data compression, image processing, data visualization, exploratory data analysis, pattern recognition, and time series prediction.
- The most common derivation of PCA is in terms of a orthogonal projection which maximizes the variance in the projected space. Given a set of m-dimensional observation vector, $\{\boldsymbol{x}_t\}$, the PCA aims at finding a orthogonal linear projection $\boldsymbol{y} = \boldsymbol{W}\boldsymbol{x}$ such that the variance of $\boldsymbol{y} \in \mathbb{R}^q$ $(q < d)$ is maximized. The ith element of $\boldsymbol{y}$ is called the *ith principal component*.
- Alternatively, the PCA provides an orthogonal linear projection which minimize the squared reconstruction error $\sum_t \|\boldsymbol{x}_t - \hat{\boldsymbol{x}}_t\|^2$. Thus the PCA is an optimal linear encoding in MS sense.

PCA and SVD

- It was shown that the ith row vector of $\boldsymbol{W}$ denoted by $\boldsymbol{w}_i^T$ corresponds to the normalized eigenvector associated with the ith largest eigenvalue of the covariance matrix $\boldsymbol{R}_x = E\{\boldsymbol{x}\boldsymbol{x}^T\}$
- Principal components can be found by SVD, linear neural networks, or probabilistic methods.
- The SVD of $\boldsymbol{R}_x$ has the form $\boldsymbol{R}_x = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$. Then we select n column vectors to contruct $\boldsymbol{U}_1 = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n]$. The the PCA transform leads to $\boldsymbol{y} = \boldsymbol{U}_1^T \boldsymbol{x}$.

PCA: An Example



Spectral Decomposition (Eigen-Decomposition)

Given a symmetric matrix $\boldsymbol{C} \in \mathbb{R}^{m \times m}$, its spectral decomposition is given by

$$\boldsymbol{C} = \lambda_1 \boldsymbol{u}_1 \boldsymbol{u}_1^T + \lambda_2 \boldsymbol{u}_2 \boldsymbol{u}_2^T + \cdots + \lambda_m \boldsymbol{u}_m \boldsymbol{u}_m^T,$$

where $|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_m|$ are eigenvalues of $\boldsymbol{C}$ and $\boldsymbol{u}_i$ are associated eigenvectors.

Power Iteration

- The power iteration is a classical method which finds the largest eigenvector (associated with the largest eigenvalue) of a matrix $\boldsymbol{C} \in \mathbb{R}^{m \times m}$.
- Given a symmetric matrix $\boldsymbol{C} \in \mathbb{R}^{m \times m}$ (hence its eigenvalues are real), the power iteration starts from a nonzero vector $\boldsymbol{w}(0)$ and iteratively updates $\boldsymbol{w}(t)$ by

$$\widetilde{\boldsymbol{w}}(t+1) = \boldsymbol{C}\boldsymbol{w}(t), \tag{6}$$

$$\boldsymbol{w}(t+1) = \frac{\widetilde{\boldsymbol{w}}(t+1)}{\|\widetilde{\boldsymbol{w}}(t+1)\|_2}, \tag{7}$$

  where $\|\cdot\|_2$ represents Euclidean norm.
- Combining (6) and (7) leads to the updating rule which has the form

$$\boldsymbol{w}(t+1) = \boldsymbol{C}\boldsymbol{w}(t)\left[\boldsymbol{w}^T(t)\boldsymbol{C}^2\boldsymbol{w}(t)\right]^{-\frac{1}{2}}. \tag{8}$$

- Assume that $\boldsymbol{C}$ has an unique eigenvalue of maximum modulus λ_1 associated with the leading eigenvector $\boldsymbol{u}_1$. Then the power iteration (8) leads $\boldsymbol{w}(t)$ to converge to $\boldsymbol{u}_1$.

Deflation

- Suppose that we are interested in computing eigenvectors of the data covariance matrix $\boldsymbol{C} = \langle \boldsymbol{x}\boldsymbol{x}^T \rangle$.
- The power iteration is applied to $\boldsymbol{C}$ for extracting its first eigenvector. Our question arises, "How can we compute the second eigenvector of $\boldsymbol{C}$ using the power iteration?".
- The deflation method is a common numerical technique for computing several eigenvalues and eigenvectors of $\boldsymbol{C}$.
- Assume that the first eigenvector is already computed. Then the output value can be deflated by the following transformation:

$$\widetilde{\boldsymbol{x}} = \left(\boldsymbol{I} - \boldsymbol{u}_1 \boldsymbol{u}_1^T\right) \boldsymbol{x}. \tag{9}$$

- One can easily see that $\langle \widetilde{\boldsymbol{x}}\widetilde{\boldsymbol{x}}^T \rangle = \sum_{i=2}^{m} \lambda_i \boldsymbol{u}_i \boldsymbol{u}_i^T$.
- The power iteration is applied to the deflated data, in order to extract the second eigenvector of $\boldsymbol{C}$.

Term-Document Matrix

A term-document matrix $\boldsymbol{X} \in \mathbb{R}^{D \times N}$ is a collection of vector space representations of documents, where rows are terms (words) and columns are documents

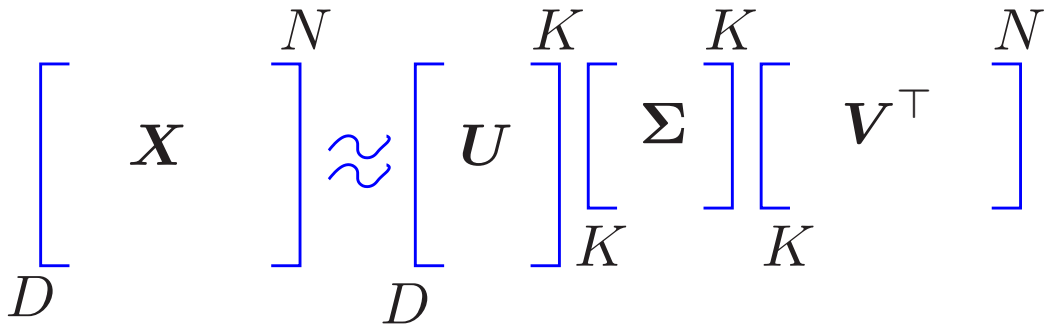
$$X_{ij} = t_{ij} \log \left(\frac{N}{idf_i} \right),$$

where t_{ij} is the term frequency of word i in document j and idf_i is the number of documents containing word i.
We write

$$\begin{aligned} \boldsymbol{X} &= [\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_N] \quad \text{(document vectors)} \\ &= \begin{bmatrix} \boldsymbol{t}_1^\top \\ \boldsymbol{t}_2^\top \\ \vdots \\ \boldsymbol{t}_D^\top \end{bmatrix} \quad \text{(term vectors)}. \end{aligned}$$

Latent Semantic Analysis

- A method for automatic indexing and retrieval, uncovering latent semantic structure of a term-document matrix.
- SVD of $\boldsymbol{X} \in \mathbb{R}^{D \times N}$ is given by $\boldsymbol{X} \approx \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$ where $\boldsymbol{U} \in \mathbb{R}^{D \times K}$, $\boldsymbol{\Sigma} \in \mathbb{R}^{K \times K}$, and $\boldsymbol{V} \in \mathbb{R}^{N \times K}$.



- Comparing documents:

$$\boldsymbol{\Sigma}^{-1}\boldsymbol{U}^\top\boldsymbol{X} = \boldsymbol{\Sigma}^{-1}\boldsymbol{U}^\top\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top = \boldsymbol{V}^\top = [\widehat{\boldsymbol{d}}_1, \ldots, \widehat{\boldsymbol{d}}_N].$$

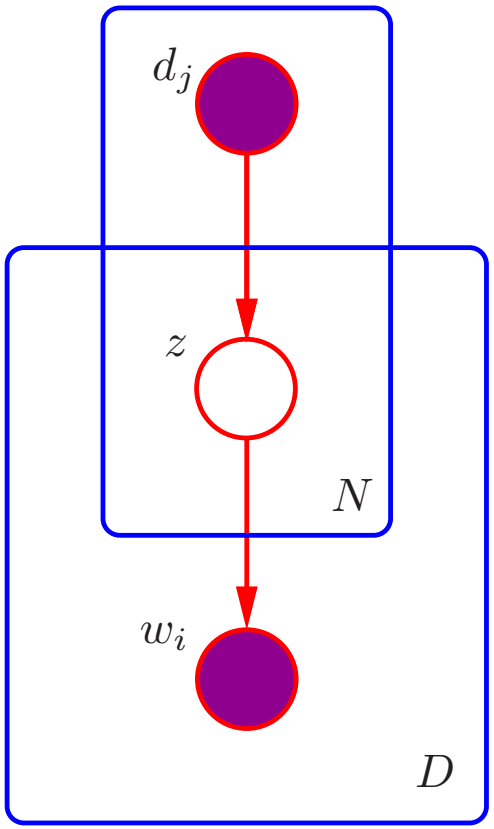
- Comparing terms:

$$\boldsymbol{X}\boldsymbol{V}\boldsymbol{\Sigma}^{-1} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top\boldsymbol{V}\boldsymbol{\Sigma}^{-1} = \boldsymbol{U} = [\widehat{\boldsymbol{t}}_1, \ldots, \widehat{\boldsymbol{t}}_D]^\top.$$

Probabilistic Latent Semantic Analysis

- The key idea in latent semantic analysis is to map high-dimensional count vectors (co-occurrence data, dyadic data) to a lower-dimensional representation in a so-called latent semantic space.
- The goal of LSA is to find a data mapping which reveals semantical relations between the entries of interest.
- Probabilistic latent semantic analysis (PLSA) is a probabilistic variant of LSA:
  - Has a sound statistical foundation;
  - Defines a proper generative model of the data.
- Aspect model: A latent variable model for co-occurrence data which associates an unobserved class $z \in \{z_1, \ldots, z_K\}$ with each occurrence of a word $w \in \{w_1, \ldots, w_D\}$ in a document $d \in \{d_1, \ldots, d_N\}$.

PLSA: Graphical Representation

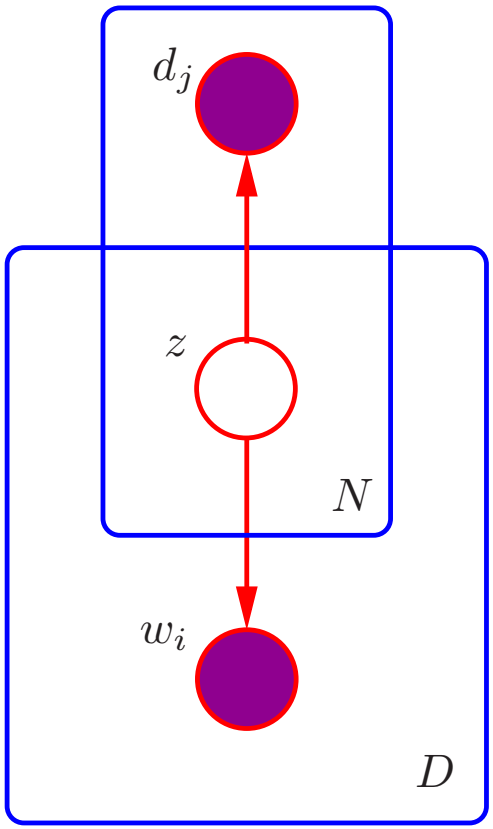


- A document d_j and a term (word) w_i are conditionally independent given an unobserved topic z:

$$p(w_i, d_j) = p(d_j) \left[\sum_z p(w_i|z) p(z|d_j) \right].$$

- Generation process:
  - Select a document d_j with probability $p(d_j)$.
  - Pick a latent class (topic) with probability $p(z|d_j)$.
  - Generate a word w_i with probability $p(w_i|z)$.

PLSA: Symmetric Parameterization



$$
\begin{aligned}
p(w_i, d_j) &= \sum_z p(w_i, d_j, z) \\
&= \sum_z p(w_i, d_j|z)p(z) \\
&= \sum_z p(w_i|z)p(d_j|z)p(z).
\end{aligned}
$$

Model Fitting: EM Algorithm

- Dyadic data $\boldsymbol{X}$: Entries X_{ij} are made for dyads (w_i, d_j) which refer to a domain with two sets of objects, $\mathcal{W} = \{w_1, \ldots, w_D\}$ and $\mathcal{D} = \{d_1, \ldots, d_N\}$.
- Complete-data likelihood:

$$p(\boldsymbol{X}, z) = \prod_i \prod_j p(w_i, d_j, z)^{C_{ij}} = \left[p(w_i|z)p(d_j|z)p(z)\right]^{C_{ij}},$$

where C_{ij} are the empirical counts for dyads (w_i, d_j) and $X_{ij} = C_{ij} / \sum_i \sum_j C_{ij}$.

- EM optimization
  - E-step: Compute the expected complete-data log-likelihood:

$$\langle \mathcal{L}_c \rangle = \sum_i \sum_j \sum_k p(z_k|w_i, d_j) C_{ij} \log \left[p(w_i|z_k)p(d_j|z_k)p(z_k)\right].$$

  - M-step: Re-estimate parameters $p(w_i|z_k), p(d_j|z_k), p(z_k)$ which maximizes $\langle \mathcal{L}_c \rangle$.

E-Step: Compute $p(z|w_i, d_j)$

Compute the posterior distribution over latent variables:

$$\begin{aligned} p(z_k|w_i, d_j) &= \frac{p(w_i, d_j|z_k)p(z_k)}{p(w_i, d_j)} \\ &= \frac{p(w_i|z_k)p(d_j|z_k)p(z_k)}{\sum_l p(w_i|z_l)p(d_j|z_l)p(z_l)}, \end{aligned}$$

where $p(w_i|z_k), p(d_j|z_k), p(z_k)$ are estimated in the M-step.

M-Step: Re-estimate Parameters

Re-estimate parameters:

$$
\begin{aligned}
p(w_i|z_k) &= \frac{\sum_j C_{ij} p(z_k|w_i, d_j)}{\sum_i \sum_j C_{ij} p(z_k|w_i, d_j)}, \\
p(d_j|z_k) &= \frac{\sum_i C_{ij} p(z_k|w_i, d_j)}{\sum_i \sum_j C_{ij} p(z_k|w_i, d_j)}, \\
p(z_k) &= \frac{\sum_i \sum_j C_{ij} p(z_k|w_i, d_j)}{\sum_i \sum_j C_{ij}}.
\end{aligned}
$$

Updating equations in the M-step are determined by solving

$$\begin{aligned}
\frac{\partial}{\partial p(w_i|z_k)}\left[\langle \mathcal{L}_c \rangle + \lambda(1 - \sum_i p(w_i|z_k))\right] &= 0, \\
\frac{\partial}{\partial p(d_j|z_k)}\left[\langle \mathcal{L}_c \rangle + \lambda(1 - \sum_j p(d_j|z_k))\right] &= 0, \\
\frac{\partial}{\partial p(z_k)}\left[\langle \mathcal{L}_c \rangle + \lambda(1 - \sum_l p(z_l))\right] &= 0,
\end{aligned}$$

for $p(w_i|z_k), p(d_j|z_k), p(z_k)$, respectively.

Document Clustering by PLSA

- You are given parameters $p(w_i|z_k), p(d_j|z_k), p(z_k)$ estimated by EM optimization.
- Compute

$$p(z_k|d_j) \propto p(d_j|z_k)p(z_k).$$

- Assign document d_j to cluster k^* if

$$k^* = \arg\max_k p(z_k|d_j).$$

PLSA: Revisited

- The PLSA models each word in a document as a sample from a mixture model where the mixture components are multinomial random variables that can be viewed as representations of topics,

$$p(w, d) = \sum_z p(z)p(w|z)p(d|z) = p(d) \sum_z p(w|z)p(z|d).$$

- Each document is represented as a list of mixing proportions for mixture components and reduced to a probability distribution on a fixed set of topics. This distribution is the reduced description associated with the document.
- In PLSA, each document is represented as a list of numbers (the mixing proportions for topics) and there is no generative probabilistic model for these numbers. This leads to the following problems:
  - The number of parameters in the model grows linearly with the size of corpus, which leads to problems with overfitting.
  - It is not clear how to assign probability to a document outside of the training set.

References

- S. Deerwester, S. T. Dumais, and R. Harshman, "Indexing by latent semantic analysis," Journal of the Americal Society of Information Science, vol. 41, no. 6, pp. 391-407, 1990.
- T. Hofmann, "Probabilistic latent semantic indexing," in Proc. SIGIR-1999.
- T. Hofmann, "Probabilistic latent semantic analysis," in Proc. UAI-1999.