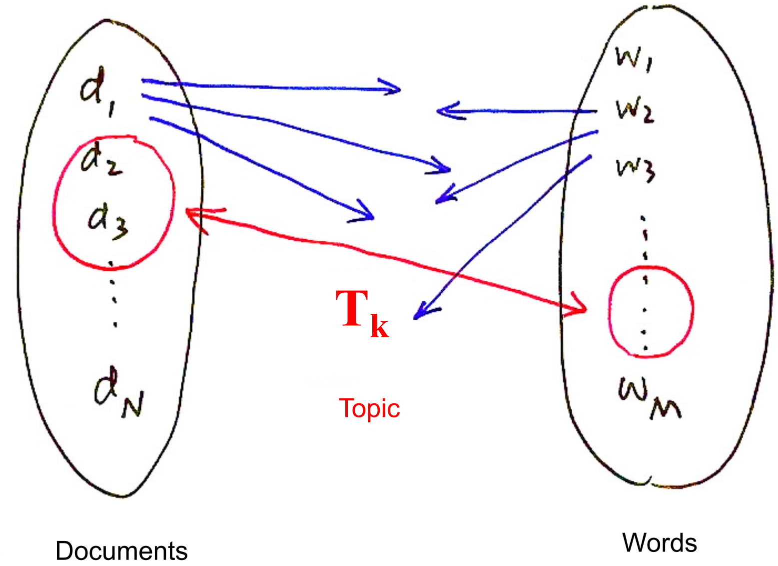
14.0 Linguistic Processing and Latent Topic Analysis

Latent Semantic Analysis (LSA)



Latent Semantic Analysis (LSA) - Word-Document Matrix Representation

- **Vocabulary V of size M and Corpus T of size N**
  - $V=\{w_1,w_2,\ldots w_i,\ldots w_M\}$, w_i: the i-th word ,e.g. $M=2\times 10^4$
    $T=\{d_1,d_2,\ldots d_j,\ldots d_N\}$, d_j: the j-th document ,e.g. $N=10^5$
  - c_{ij}: number of times w_i occurs in d_j
    n_j: total number of words present in d_j
    $t_i = \Sigma_j c_{ij}$: total number of times w_i occurs in T

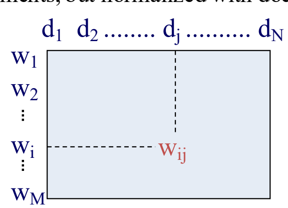
$$\Rightarrow \varepsilon_i = -\frac{1}{\log N}\sum_{j=1}^{N}\left(\frac{c_{ij}}{t_i}\right)\log\left(\frac{c_{ij}}{t_i}\right),\quad \text{normalized entropy (indexing power) of } w_i \text{ in T}$$

$$0\le\varepsilon_i\le 1,\quad \varepsilon_i=0 \ \text{ if } c_{ij}=t_i \text{ for some j and } c_{ij}=0 \text{ for other j}$$
$$\varepsilon_i=1 \ \text{ if } c_{ij}=t_i/N \text{ for all j}$$

  - $w_{ij}=(1-\varepsilon_i)\dfrac{c_{ij}}{n_j}$, word frequencies in doucments, but normalized with document length and word entropy

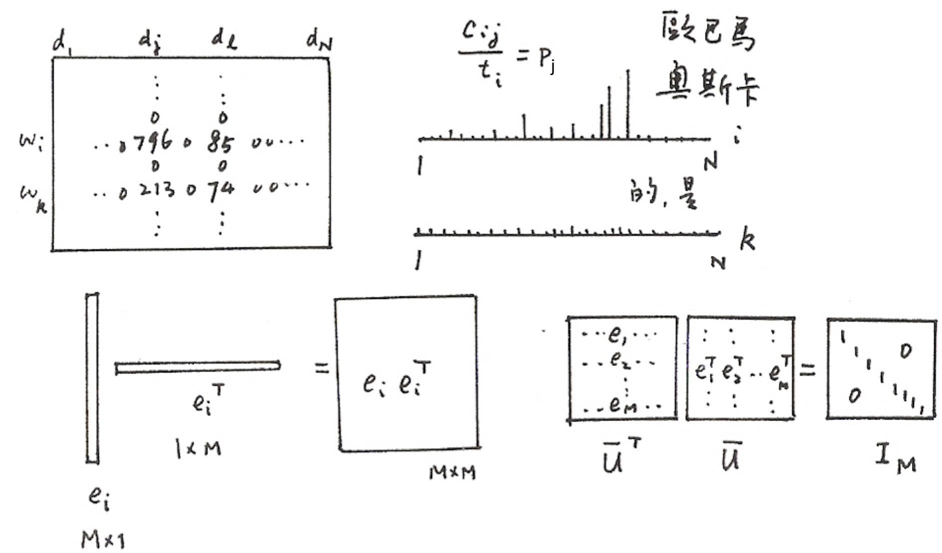
- **Word-Document Matrix W**

  $W=[w_{ij}]$



  - each row of W is a N-dim "feature vector" for a word w_i with respect to all documents d_j
    each column of W is a M-dim "feature vector" for a document d_j with respect to all words w_i

Latent Semantic Analysis (LSA)

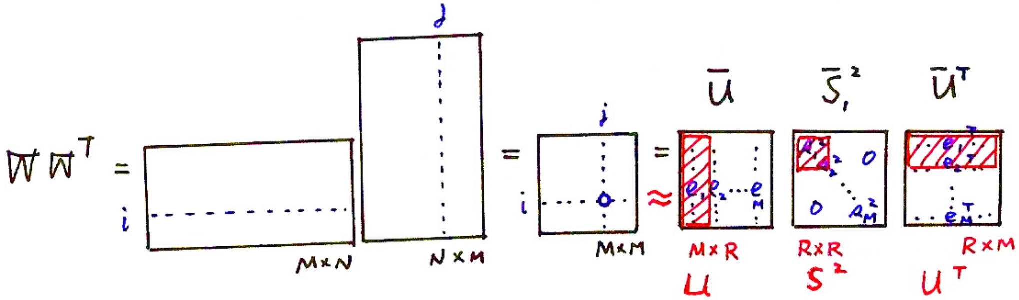


Dimensionality Reduction (1/2)

- $WW^T = \overline{U}\overline{S}_1^2\,\overline{U}^T$
  - (i, j) element of WW^T : inner product of i-th and j-th rows of W
    "similarity" between w_i and w_j
    $\overline{U} = [e_1, e_2, \ldots e_M]$, $\overline{S}_1^2 = [s_i^2]_{M\times M}$, s_i^2 : eigenvalues of WW^T, $s_i^2 \geq s_{i+1}^2$
    $WW^T = \sum_i s_i^2 e_i e_i^T$, e_i : orthonormal eigenvectors, $\overline{U}^T\overline{U} = I_M$
    s_i^2 : weights (significance of the "component matrices" $e_i e_i^T$)
  - dimensionality reduction: selection of R largest eigenvalues (R=800 for example)

$$W_{M\times N}W^T_{N\times M} \approx U_{M\times R}S^2_{R\times R}U^T_{R\times M},\ U_{M\times R} = [e_1, e_2, \ldots e_R]$$

R "concepts" or "latent semantic concepts"

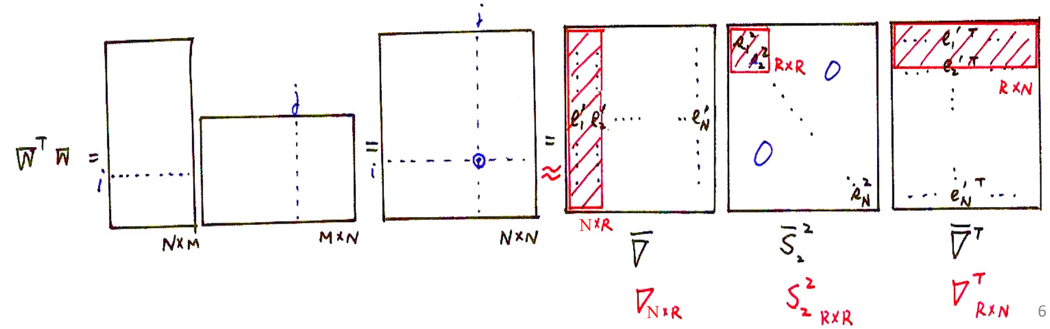


Dimensionality Reduction (2/2)

- $W^T W = \overline{V}\overline{S}_2^2\,\overline{V}^T$
  - (i, j) element of $W^T W$: inner product of i-th and j-th columns of W
    "similarity" between d_i and d_j
    $\overline{V} = [e'_1, e'_2, \ldots e'_N]$, $\overline{S}_2^2 = [s_i^2]_{N\times N}$, s_i^2 : eigenvalues of $W^T W$, $s_i^2 \geq s_{i+1}^2$, $s_i^2 = 0$ for $i > \min(M,N)$
    $W^T W = \sum_i s_i^2 e'_i e'^T_i$, e'_i : orthonormal eigenvectors, $\overline{V}^T\overline{V} = I_N$
    s_i^2 : weights (significance of the "component matrices" $e'_i e'^T_i$)
  - dimensionality reduction: selection of R largest eigenvalues

$$W^T_{N\times M}W_{M\times N} \approx V_{N\times R}S^2_{R\times R}V^T_{R\times N},\quad V_{N\times R} = [e'_1, e'_2 \ldots e'_R]$$

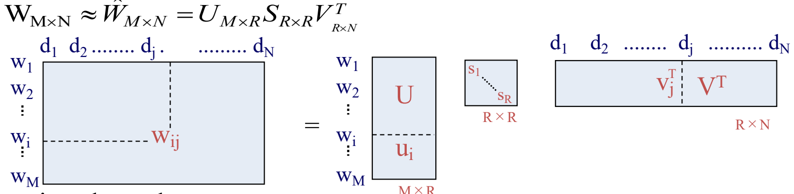
R "concepts" or "latent semantic concepts"



Singular Value Decomposition (SVD)

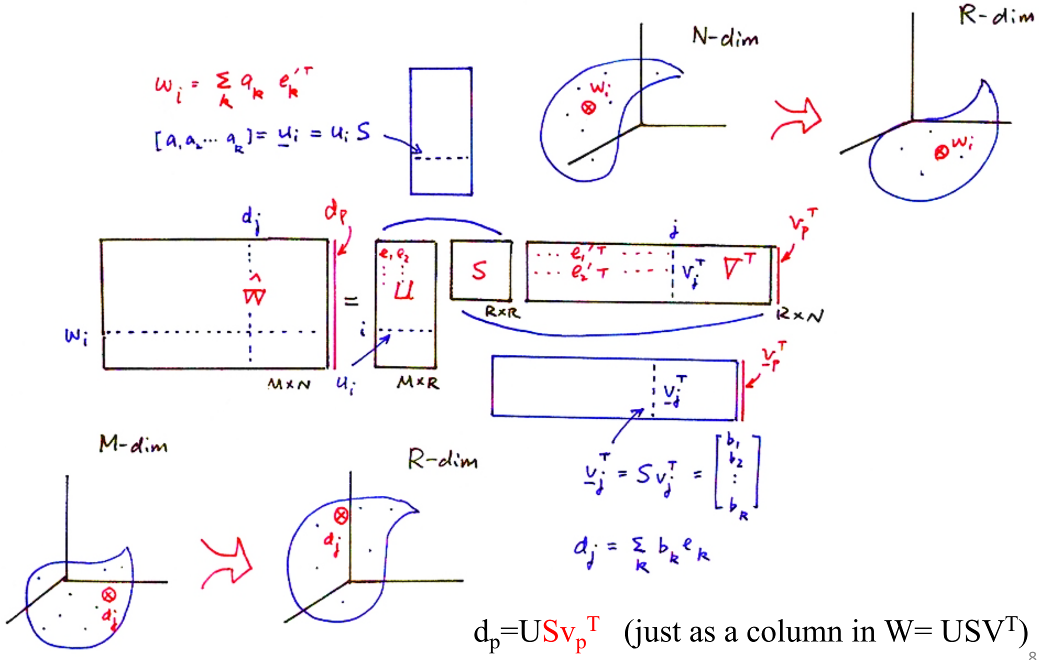
- **Singular Value Decomposition (SVD)**

$$W_{M\times N} \approx \hat{W}_{M\times N} = U_{M\times R}S_{R\times R}V^T_{R\times N}$$



  - s_i: singular values, $s_1 \geq s_2 \ldots \geq s_R$
    U: left singular matrix, V: right singular matrix
- **Vectors for word w_i: $u_iS = \underline{u}_i$ (a row)**
  - a vector with dimensionality N reduced to a vector $u_iS = \underline{u}_i$ with dimensionality R
  - N-dimensional space defined by N documents reduced to R-dimensional space defined by R "concepts"
  - the R row vectors of V^T, or column vectors of V, or eigenvectors $\{e'_1, \ldots e'_R\}$, are the R orthonormal basis for the "latent semantic space" with dimensionality R, with which $u_iS = \underline{u}_i$ is represented
  - words with similar "semantic concepts" have "closer" location in the "latent semantic space"
    - they tend to appear in similar "types" of documents, although not necessarily in exactly the same documents

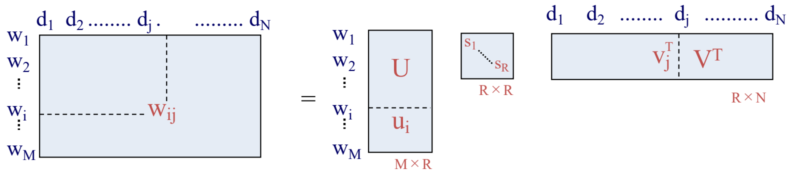
Singular Value Decomposition (SVD)



$d_p = USv_p^T$ (just as a column in $W = USV^T$)

Singular Value Decomposition (SVD)

- **Singular Value Decomposition (SVD)**



- **Vectors for document d_j: $v_jS=\underline{v}_j$ (a row, or $\underline{v}_j^T = S\, v_j^T$ for a column)**
  - a vector with dimensionality M reduced to a vector $v_jS=\underline{v}_j$ with dimensionality R
  - M-dimensional space defined by M words reduced to R-dimensional space defined by R "concepts"
  - the R columns of U, or eigenvectors $\{e_1,...e_R\}$, are the R orthonormal basis for the "latent semantic space" with dimensionality R, with which $v_jS=\underline{v}_j$ is represented
  - documents with similar "semantic concepts" have "closer" location in the "latent semantic space"
    - they tend to include similar "types" of words, although not necessarily exactly the same words
- **The Association Structure between words w_i and documents d_j is preserved with noisy information deleted, while the dimensionality is reduced to a common set of R "concepts"**

Example Applications in Linguistic Processing

- **Word Clustering**
  - example applications: class-based language modeling, information retrieval ,etc.
  - words with similar "semantic concepts" have "closer" location in the "latent semantic space"
    - they tend to appear in similar "types" of documents, although not necessarily in exactly the same documents
  - each component in the reduced word vector $u_jS=\underline{u}_j$ is the "association" of the word with the corresponding "concept"
  - example similarity measure between two words:

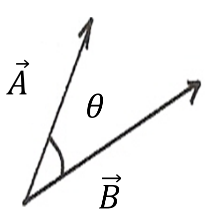
$$sim(w_i, w_j) = \frac{\underline{u}_i \cdot \underline{u}_j}{|\underline{u}_i| \cdot |\underline{u}_j|} = \frac{u_i S^2 u_j^T}{|u_i S| \cdot |u_j S|}$$

- **Document Clustering**
  - example applications: clustered language modeling, language model adaptation, information retrieval, etc.
  - documents with similar "semantic concepts" have "closer" location in the "latent semantic space"
    - they tend to include similar "types" of words, although not necessarily exactly the same words
  - each component on the reduced document vector $v_jS=\underline{v}_j$ is the "association" of the document with the corresponding "concept"
  - example "similarity" measure between two documents:

$$sim(d_i, d_j) = \frac{\underline{v}_i \cdot \underline{v}_j}{|\underline{v}_i| \cdot |\underline{v}_j|} = \frac{v_i S^2 v_j}{|v_i S| \cdot |v_j S|}$$

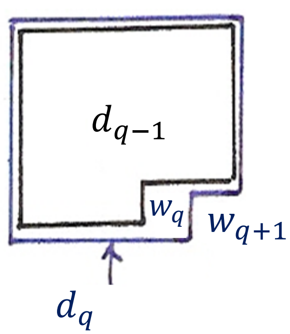
LSA for Linguistic Processing

Cosine Similarity



$$-1 \le \cos\theta = \frac{\vec{A}\cdot\vec{B}}{|\vec{A}||\vec{B}|} \le 1$$

$$= 0 \text{ if } \vec{A} \perp \vec{B}$$



$$\vec{A}\cdot\vec{B} = \underbrace{|\vec{A}||\vec{B}|}_{\text{magnitude}}\ \underbrace{\cos\theta}_{\text{Similarity}}$$

Example Applications in Linguistic Processing

- **Information Retrieval**
  - "concept matching" vs "lexical matching" : relevant documents are associated with similar "concepts", but may not include exactly the same words
  - example approach: treating the query as a new document (by "folding-in"), and evaluating its "similarity" with all possible documents
- **Fold-in**
  - consider a new document outside of the training corpus T, but with similar language patterns or "concepts"
  - construct a new column d_p ,p>N, with respect to the M words
  - assuming U and S remain unchanged

    $d_p=USv_p^T$ (just as a column in $W= USV^T$)

    $\underline{v}_p = v_pS = d_p^TU$ as an R-dim representation of the new document

    (i.e. obtaining the projection of d_p on the basis e_i of U by inner product)

Integration with N-gram Language Models

- **Language Modeling for Speech Recognition**
  - $\text{Prob}(w_q|d_{q-1})$
    w_q: the q-th word in the current document to be recognized (q: sequence index)
    d_{q-1}: the recognized history in the current document
    $\underline{v}_{q-1}=d_{q-1}^T U$: representation of d_{q-1} by v_{q-1} (folded-in)
  - $\text{Prob}(w_q|d_{q-1})$ can be estimated by $\underline{u}_q$ and $\underline{v}_{q-1}$ in the R-dim space
  - integration with N-gram
    $\text{Prob}(w_q|H_{q-1}) = \text{Prob}(w_q|h^{(n)}_{q-1}, d_{q-1})$
    H_{q-1}: history up to w_{q-1}
    $h^{(n)}_{q-1}$: $<w_{q-n+1}, w_{q-n+2}, \ldots w_{q-1}>$
  - N-gram gives local relationships, while d_{q-1} gives semantic concepts
  - d_{q-1} emphasizes more the key content words, while N-gram counts all words similarly including function words
- **$\underline{v}_{q-1}$ for d_{q-1} can be estimated iteratively**
  - assuming the q-th word in the current document is w_i

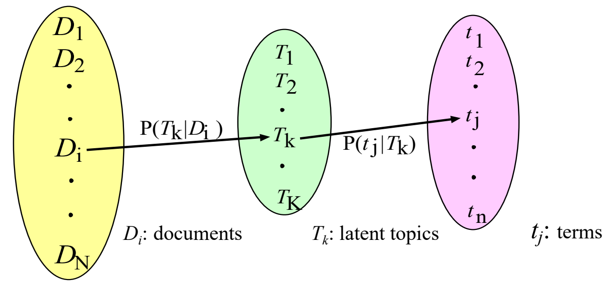
$$d_q = \left(\frac{q-1}{q}\right)d_{q-1} + \left(\frac{1-\varepsilon_i}{q}\right)[00\ldots 0\underset{\uparrow}{1}00\ldots 0]^T$$

i-th dimentionality out of M

$$\underline{v}_q = d_q^T U = \left(\frac{q-1}{q}\right)\underline{v}_{q-1} + \left(\frac{1-\varepsilon_i}{q}\right)u_i \quad \text{,updated word-by-word}$$

$\underline{v}_q$ moves in the R-dim space initially, eventually settle down somewhere

Probabilistic Latent Semantic Analysis (PLSA)



- Exactly the same as LSA, using a set of latent topics $\{T_1, T_2, \cdots T_K\}$ to construct a new relationship between the documents and terms, but with a probabilistic framework

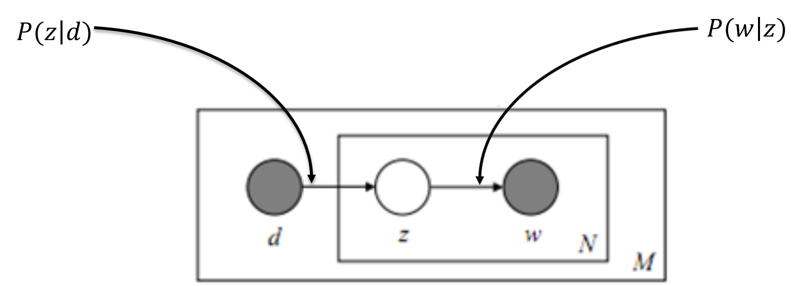
$$P(t_j \mid D_i) = \sum_{k=1}^{K} P(t_j \mid T_k) P(T_k \mid D_i)$$

- Trained with EM by maximizing the total likelihood

$$L_T = \sum_{i=1}^{N}\sum_{j=1}^{n} c(t_j, D_i) \log P(t_j \mid D_i)$$

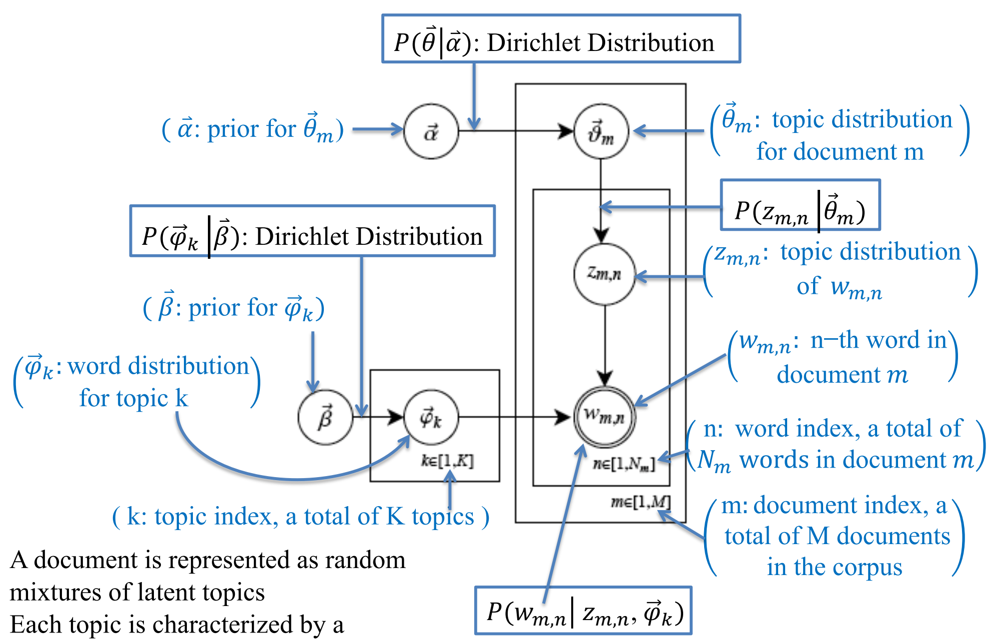
$c(t_j, D_i)$: frequency count of term t_j in the document D_i

Probabilistic Latent Semantic Analysis (PLSA)



w: word
z: topic
d: document
N: words in document d
M: documents in corpus

Latent Dirichlet Allocation(LDA)



- A document is represented as random mixtures of latent topics
- Each topic is characterized by a distribution over words

Gibbs Sampling in general

- **To obtain a distribution of a given form with unknown parameters $\{z_i: i = 1, \cdots, M\}$**

1. Initialize $\left\{z_i^{(0)}: i = 1, \cdots, M\right\}$
2. For $\tau = 0, \cdots, T$:
   - Sample $z_1^{(\tau+1)} \sim p\left(z_1 \middle| z_2^{(\tau)}, z_3^{(\tau)}, \cdots, z_M^{(\tau)}\right)$

   Take a sample of z_1 base on the distribution $p\left(z_1 \middle| z_2^{(\tau)}, z_3^{(\tau)}, \cdots, z_M^{(\tau)}\right)$
   - Sample $z_2^{(\tau+1)} \sim p\left(z_2 \middle| z_1^{(\tau+1)}, z_3^{(\tau)}, \cdots, z_M^{(\tau)}\right)$

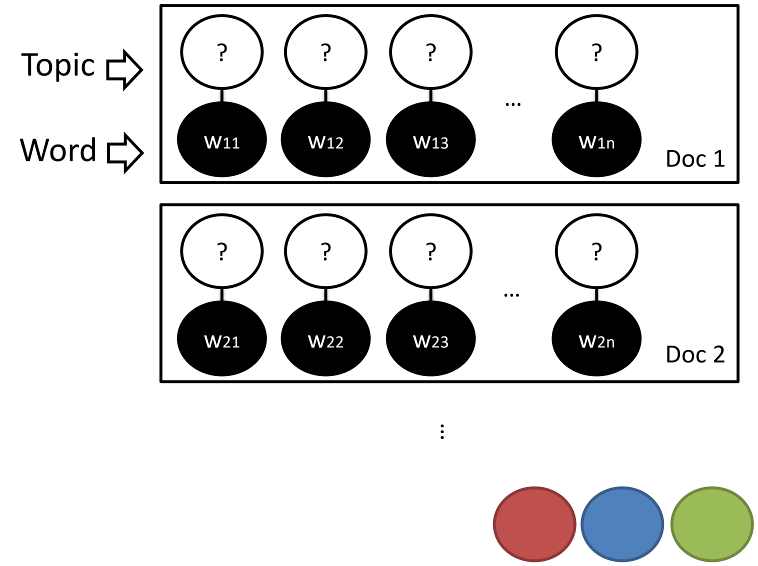
   ⋮
   - Sample $z_j^{(\tau+1)} \sim p\left(z_j \middle| z_1^{(\tau+1)}, \cdots, z_{j-1}^{(\tau+1)}, z_{j+1}^{(\tau)}, \cdots, z_M^{(\tau)}\right)$

   ⋮
   - Sample $z_M^{(\tau+1)} \sim p\left(z_M \middle| z_1^{(\tau+1)}, z_2^{(\tau+1)}, \cdots, z_{M-1}^{(\tau+1)}\right)$

- **Apply Markov Chain Monte Carlo and sample each variable sequentially conditioned on the other variables until the distribution converges, then estimate the parameters based on the converged distribution**

Gibbs Sampling applied on LDA

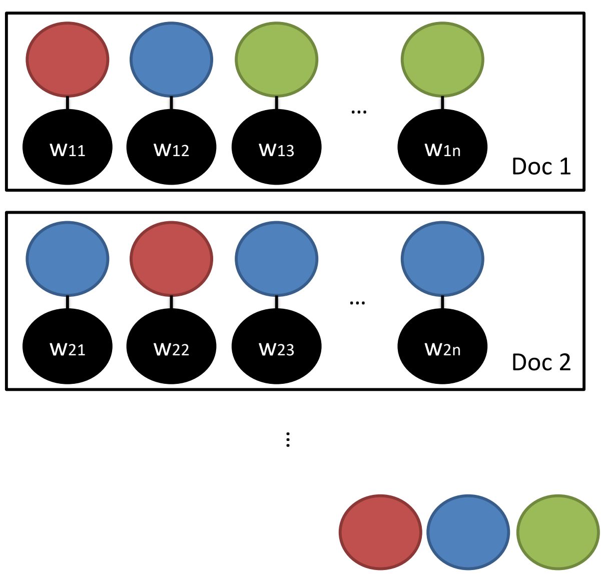
Sample P(Z,W) :



Gibbs Sampling applied on LDA

Sample P(Z,W) :

1. Random Initialization

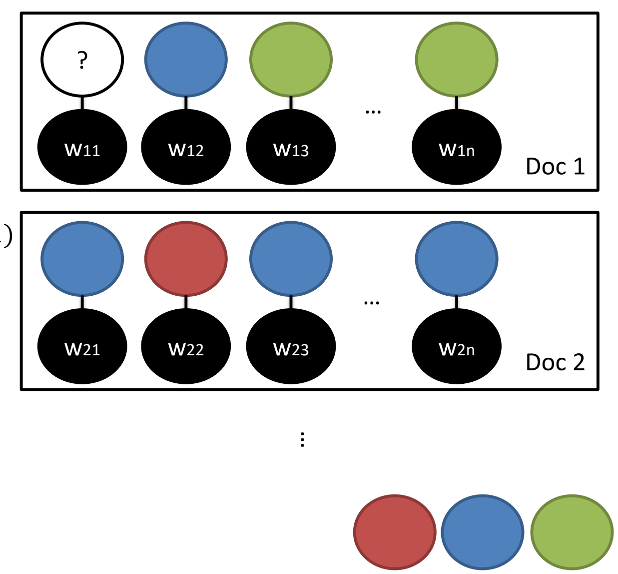


Gibbs Sampling applied on LDA

Sample P(Z,W) :

1. Random Initialization
2. Erase Z_{11}, and draw a new Z_{11} ~

$P(z_{11} | z_{12} \cdots z_{M,N_M}, w_{11}, w_{12}, \cdots, w_{M,N_M})$



Gibbs Sampling applied on LDA

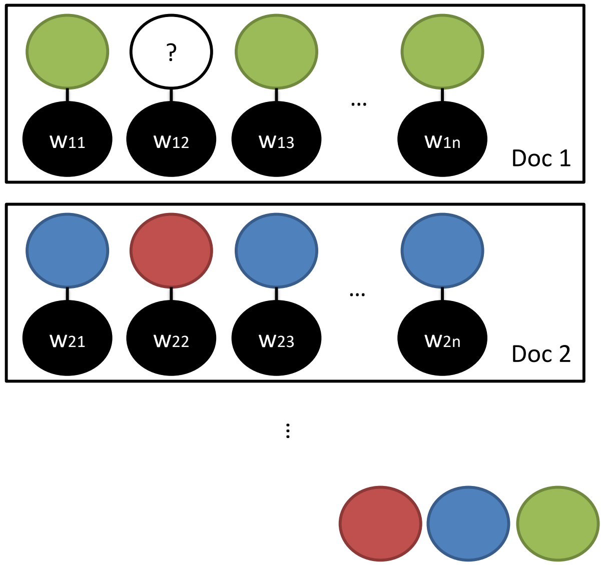
Sample P(Z,W) :

1. Random Initialization
2. Erase Z_{11}, and draw a new Z_{11} ~

$P(z_{11}|z_{12}\cdots z_{M,N_M}, w_{11}, w_{12}, \cdots, w_{M,N_M})$

3. Erase Z_{12}, and draw a new Z_{12} ~

$P(z_{12}|z_{11}, z_{13}\cdots z_{M,N_M}, w_{11}, w_{12}, \cdots, w_{M,N_M})$



Gibbs Sampling applied on LDA

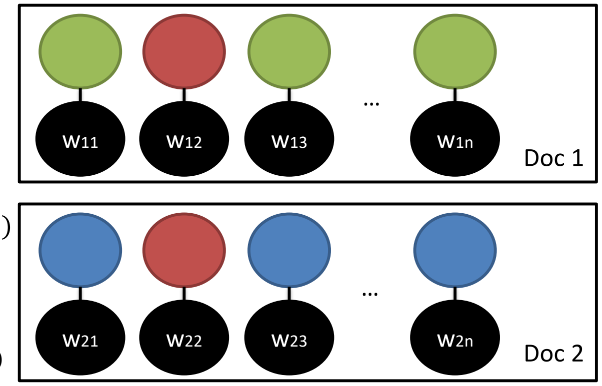
Sample P(Z,W) :

1. Random Initialization
2. Erase Z_{11}, and draw a new Z_{11} ~

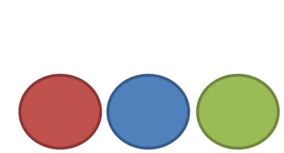
$P(z_{11}|z_{12}\cdots z_{M,N_M}, w_{11}, w_{12}, \cdots, w_{M,N_M})$

3. Erase Z_{12}, and draw a new Z_{12} ~

$P(z_{12}|z_{11}, z_{13}\cdots z_{M,N_M}, w_{11}, w_{12}, \cdots, w_{M,N_M})$



4. Iteratively update topic assignment for each word until converge
5. Compute θ, ϕ according to the final setting



Matrix Factorization (MF) for Recommendation systems

	Movie 1	Movie 2	Movie 3	Movie 4	Movie 5	Movie 6	Movie 7	Movie 8	Movie 9
User A	3.7				4.0				
User B	4.0					4.3			
User C			4.1						
User D		2.3							2.5
User E								3.3	
User F				2.9					
User G		2.6					2.7		

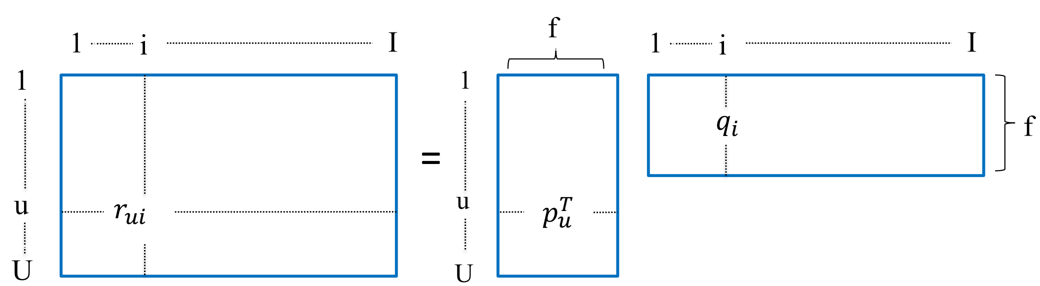
$R = [r_{ui}]$: rating
u: user
i: item

Matrix Factorization (MF)

- **Mapping both users and items to a joint latent factor space of dimensionality f**

$$q_i \in \mathbb{R}^f$$
$$p_u \in \mathbb{R}^f$$
$$\hat{r}_{ui} = q_i^T p_u.$$

latent factor: towards male, seriousness, etc.



Matrix Factorization (MF)

- **Objective function**

$$\min_{q,p} \sum_{(u,i)} (r_{ui} - q_i^T p_u)^2 + \lambda(\|q_i\|^2 + \|p_u\|^2)$$

- **Training**
  - gradient descent (GD)

$$e_{ui} \overset{def}{=} r_{ui} - q_i^T p_u.$$

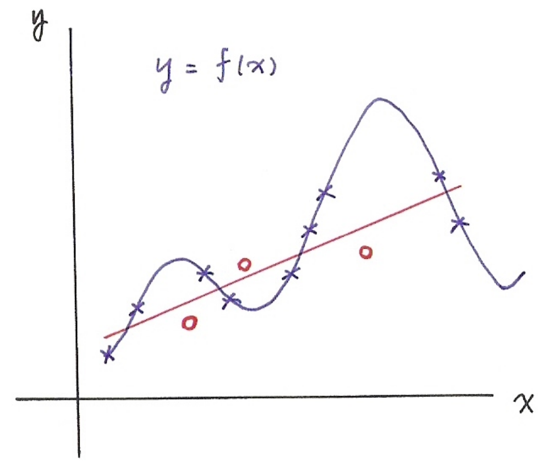
$$q_i \leftarrow q_i + \gamma \cdot (e_{ui} \cdot p_u - \lambda \cdot q_i)$$

$$p_u \leftarrow p_u + \gamma \cdot (e_{ui} \cdot q_i - \lambda \cdot p_u)$$

  - Alternating least square (ALS): alternatively fix p_u 's or q_i 's and compute the other as a least square problem
- **Different from SVD (LSA)**
  - SVD assumes missing entries to be zero (a poor assumption)

Overfitting Problem

- **A good model is not just to fit all the training data**
  - needs to cover unseen data well which may have distributions slightly different from that of training data
  - too complicated models with too many parameters usually leads to overfitting



Extensions of Matrix Factorization (MF)

- **Biased MF**
  - add global bias μ (usually = average rating) , user bias b_u , and item bias b_i as parameters

$$\hat{r}_{ui} = \mu + b_i + b_u + q_i^T p_u$$

- **Non-negative Matrix Factorization**
  - restrict the value in each component of p_u and q_i to be non-negative

References

- **LSA and PLSA**
  - “Exploiting Latent Semantic Information in Statistical Language Modeling”, Proceedings of the IEEE, Aug 2000
  - “Latent Semantic Mapping”, IEEE Signal Processing Magazine, Sept. 2005, Special Issue on Speech Technology in Human-Machine Communication
  - “Probabilistic Latent Semantic Indexing”, ACM Special Interest Group on Information Retrieval (ACM SIGIR), 1999
  - “Probabilistic Latent Semantic Indexing”, Proc. of Uncertainty in Artificial Intelligence, 1999
  - “Spoken Document Understanding and Organization”, IEEE Signal Processing Magazine, Sept. 2005, Special Issue on Speech Technology in Human-Machine Communication
- **LDA and Gibbs Sampling**
  - Christopher M. Bishop, Pattern Recognition and Machine Learning, Springer 2006
  - Blei, David M.; Andrew Y. Ng, Michael I. Jordan. "Latent Dirichlet Allocation”, Journal of Machine Learning Research 2003
  - Gregor Heinrich, ”Parameter estimation for text analysis”, 2005

References

- **Matrix Factorization**

  - A Linear Ensemble of Individual and Blended Models for Music Rating Prediction. In JMLR W&CP, volume 18, 2011.
  - Y. Koren, R. M. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. Computer, 42(8):30-37, 2009.
  - Introduction to Matrix Factorization Methods Collaborative Filtering (http://www.intelligentmining.com/knowledge/slides/Collaborative.Filtering.Factorization.pdf)
  - GraphLab API: Collaborative Filtering (http://docs.graphlab.org/collaborative_filtering.html)
  - J Mairal, F Bach, J Ponce, G Sapiro, Online learning for matrix factorization and sparse coding, The Journal of Machine Learning, 2010