APAN PS5430
Applied Text & Natural Language Analytics
Week 8: Topic Modeling I

Javid Huseynov, Ph.D.
Thursday, March 12, 2020

COLUMBIA UNIVERSITY

Week 8 Agenda

- Association Rule Mining
  - Definition
  - Approaches
  - Apriori Algorithm
- Topic Modeling
- Lexical Semantics Challenges
- Latent Semantic Analysis
- Latent Dirichlet Allocation
- Class Exercises

Association Rule Mining (ARM)

- Task of finding frequent patterns, correlations, associations, or causal structures found in databases
- Based on "market-basket" analysis
- Given a set of transactions, ARM finds the rules which predict the occurrence of some items in the transaction based on the occurrence of other items:

$\{bread\} \Rightarrow \{milk\}$
$\{soda\} \Rightarrow \{chips\}$
$\{bread\} \Rightarrow \{jam\}$

TID	Items
1	Bread, Peanuts, Milk, Fruit, Jam
2	Bread, Jam, Soda, Chips, Milk, Fruit
3	Steak, Jam, Soda, Chips, Bread
4	Jam, Soda, Peanuts, Milk, Fruit
5	Jam, Soda, Chips, Milk, Bread
6	Fruit, Soda, Chips, Milk
7	Fruit, Soda, Peanuts, Milk
8	Fruit, Peanuts, Cheese, Yogurt

ARM Definitions

Itemset

- Collection of items (e.g. {milk, bread})
- *k*-itemset contains k items

Support count (σ)

- Frequency of occurrence of an itemset
- $\sigma(\{\text{Milk, Bread}\}) = 3$

Support (s)

- Fraction of transactions that contain both X and Y,

  e.g. $s = \dfrac{\sigma(\{\text{Milk, Bread}\})}{\#\, of\ transactions} = 0.38$

Confidence (c)

- How often items in Y appear in transactions with X,

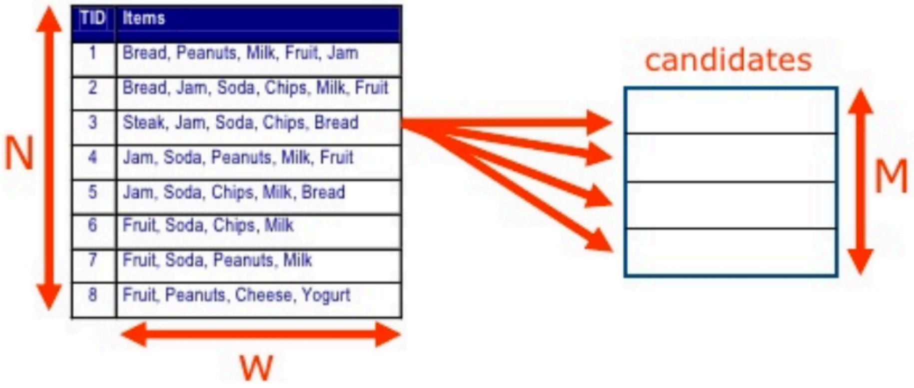
  e.g. $c = \dfrac{\sigma(\{\text{Milk, Bread}\})}{\sigma(\{\text{Bread}\})} = 0.75$

TID	Items
1	Bread, Peanuts, Milk, Fruit, Jam
2	Bread, Jam, Soda, Chips, Milk, Fruit
3	Steak, Jam, Soda, Chips, Bread
4	Jam, Soda, Peanuts, Milk, Fruit
5	Jam, Soda, Chips, Milk, Bread
6	Fruit, Soda, Chips, Milk
7	Fruit, Soda, Peanuts, Milk
8	Fruit, Peanuts, Cheese, Yogurt

ARM Approaches

Brute-force approach:

- Each itemset in the lattice is a candidate frequent itemset
- Count the support of each candidate by scanning the database



TID	Items
1	Bread, Peanuts, Milk, Fruit, Jam
2	Bread, Jam, Soda, Chips, Milk, Fruit
3	Steak, Jam, Soda, Chips, Bread
4	Jam, Soda, Peanuts, Milk, Fruit
5	Jam, Soda, Chips, Milk, Bread
6	Fruit, Soda, Chips, Milk
7	Fruit, Soda, Peanuts, Milk
8	Fruit, Peanuts, Cheese, Yogurt

- Match each transaction against every candidate
- Complexity ~ O(NMw) => Expensive since $M = 2^d$

Given a set of transactions *T*, find all rules having:

- Support >= minimum support
- Confidence >= minimum confidence

Apriori Principle

- If an itemset is frequent, then its subsets must also be frequent
- Support of an itemset never exceeds the support of its subsets

$$\forall X, Y : (X \subseteq Y) \Rightarrow s(X) \geq s(Y)$$

ARM: Apriori Algorithm

- Let k=1
- Generate frequent itemsets of length 1
- Repeat until no new frequent itemsets are identified
  - Generate length (k+1) candidate itemsets from length k frequent itemsets
  - Prune candidate itemsets containing subsets of length k that are infrequent
  - Count the support of each candidate by scanning the DB
  - Eliminate candidates that are infrequent, leaving only those that are frequent

C_k: Candidate itemset of size k

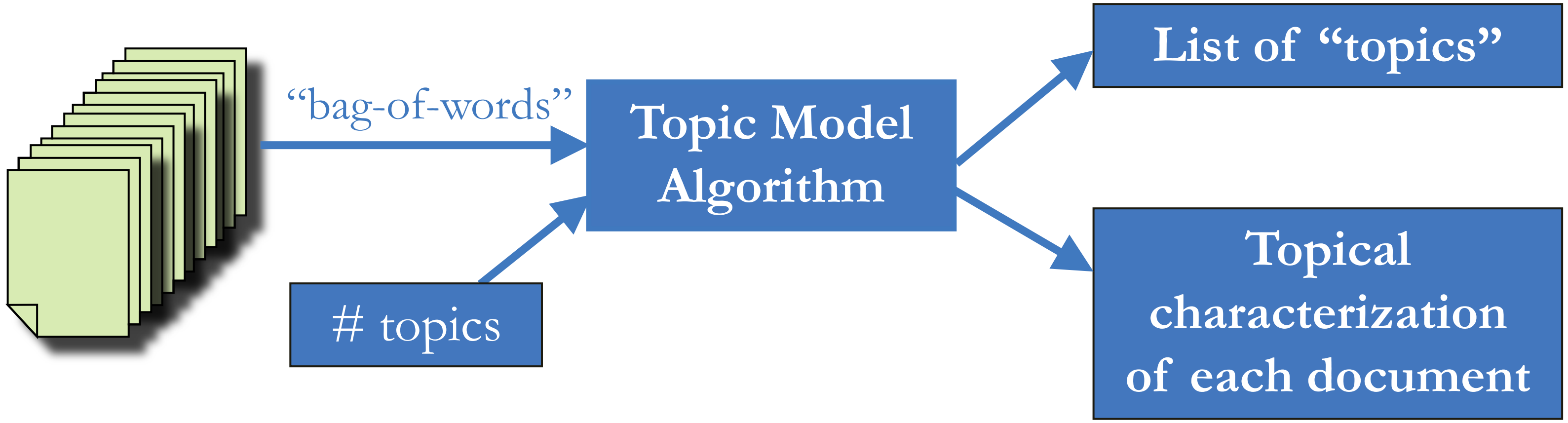
L_k : frequent itemset of size k

```
L_1 = {frequent items};
for (k = 1; L_k != ∅; k++) do begin
    C_{k+1} = candidates generated from L_k;
    for each transaction t in database do
        increment the count of all candidates in C_{k+1}
        that are contained in t
    L_{k+1} = candidates in C_{k+1} with min_support
    end
return ∪_k L_k;
```

Topic Modeling

Unsupervised method to organize, understand, summarize a set of documents

- Discovering hidden topical patterns that are present across the collection
- Annotating documents according to these topics
- Using these annotations to organize, search and summarize texts



Useful applications:

- Search
- Automated tagging
- Summarization

Problems in lexical semantics

- **Polysemy (Ambiguity)**
- *Same word* may have *different meanings* when it appears in different contexts
- Vector-space model is unable to discriminate between these different meanings

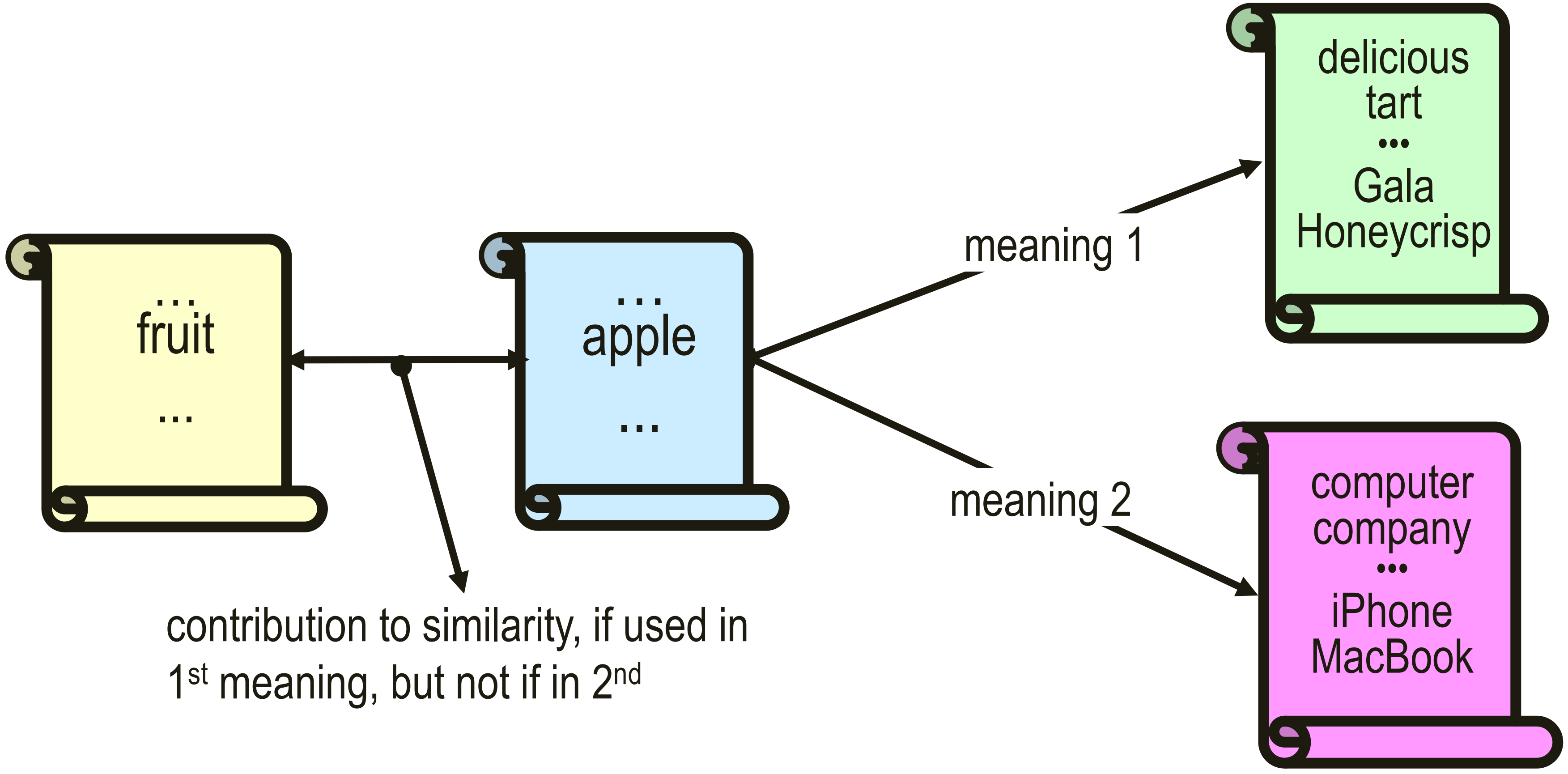
$$\mathrm{sim}_{\mathrm{true}}(d,q) < \cos(\angle(\vec{d},\vec{q}))$$

- **Synonymy (Association)**
- *Different words* may have *same meaning* when they appear in a similar context
- No associations between words are captured in the vector-space representations

$$\mathrm{sim}_{\mathrm{true}}(d,q) > \cos(\angle(\vec{d},\vec{q}))$$

Problems in lexical semantics

- Document similarity on a single word level: polysemy and context

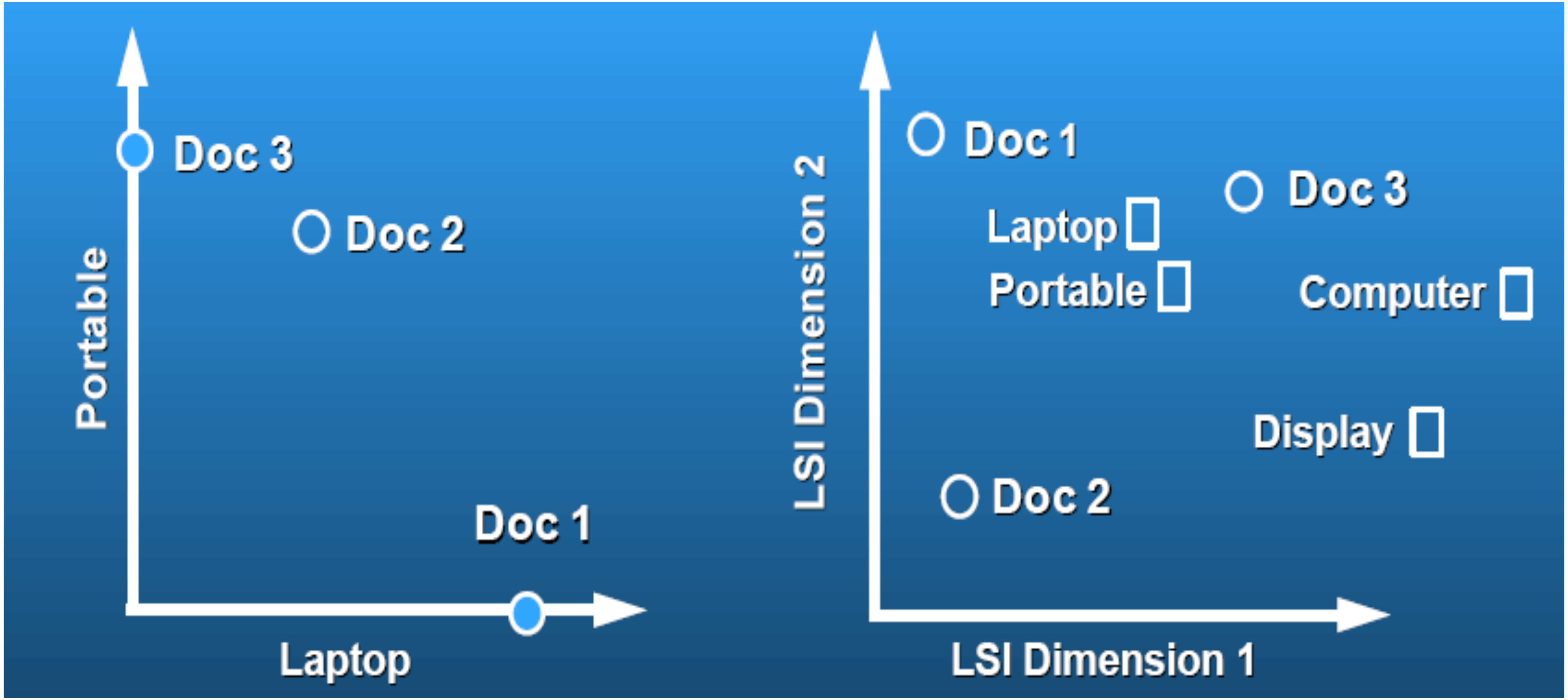


Latent semantic indexing (LSI), aka Latest Semantic Analysis (LSA)

- Term-document (TFIDF) matrices can be large (sparse), while topic space is small
- LSI takes documents that are semantically similar (talk about the same topics), but are not similar in the vector space (because they use different words) and re-represents them in a reduced vector space in which they have higher similarity.

- Perform a **low-rank approximation** of term-document matrix
  - Map documents and terms to a low dimensional representation (sparsity reduction) using **Singular Value Decomposition** (SVD)
  - Design a mapping such that the low-dimensional or **latent semantic space** reflects semantic associations
  - Compute document similarity based on the inner product in this latent semantic space

LSA: ABSTRACT Illustrative example

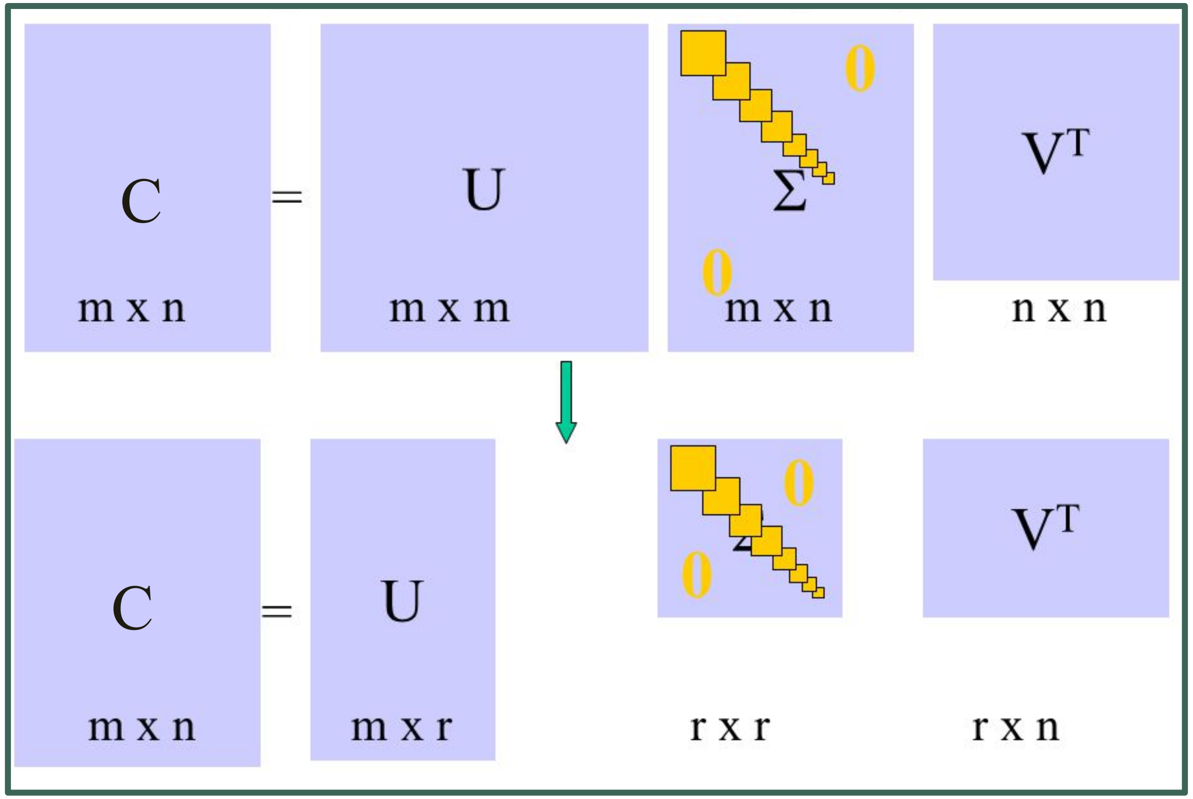


Singular Value Decomposition (SVD)

- Any real ***m x n*** matrix ***C*** can be decomposed into:

$$\mathbf{C = U\Sigma V^T}$$

- ***U*** is ***m x n,*** column orthonormal ($\mathbf{U^TU = I}$)
- $\mathbf{\Sigma}$ is ***n x n*** and diagonal:
  - $\boldsymbol{\Sigma = diag(\sigma_1, \sigma_2, \ldots, \sigma_n)}$
  - $\boldsymbol{\sigma_1}$ are called *singular* values of ***C***
  - $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n \geq 0$
- *V* is ***n x n*** and orthonormal ($VV^T = V^TV = I$)



SVD Example (LSA)

- Term-Document Matrix $C = U\Sigma V^T$

C	d_1	d_2	d_3	d_4	d_5	d_6
ship	1	0	1	0	0	0
boat	0	1	0	0	0	0
ocean	1	1	0	0	0	0
wood	1	0	0	1	1	0
tree	0	0	0	1	0	1

U	1	2	3	4	5
ship	−0.44	−0.30	0.57	0.58	0.25
boat	−0.13	−0.33	−0.59	0.00	0.73
ocean	−0.48	−0.51	−0.37	0.00	−0.61
wood	−0.70	0.35	0.15	−0.58	0.16
tree	−0.26	0.65	−0.41	0.58	−0.09

Σ	1	2	3	4	5
1	2.16	0.00	0.00	0.00	0.00
2	0.00	1.59	0.00	0.00	0.00
3	0.00	0.00	1.28	0.00	0.00
4	0.00	0.00	0.00	1.00	0.00
5	0.00	0.00	0.00	0.00	0.39

V^T	d_1	d_2	d_3	d_4	d_5	d_6
1	−0.75	−0.28	−0.20	−0.45	−0.33	−0.12
2	−0.29	−0.53	−0.19	0.63	0.22	0.41
3	0.28	−0.75	0.45	−0.20	0.12	−0.33
4	0.00	0.00	0.58	0.00	−0.58	0.58
5	−0.53	0.29	0.63	0.19	0.41	−0.22

SVD Example (LSA)

- Dimensionality reduction $C_2 = U\Sigma_2 V^T$

U	1	2	3	4	5
ship	−0.44	−0.30	0.00	0.00	0.00
boat	−0.13	−0.33	0.00	0.00	0.00
ocean	−0.48	−0.51	0.00	0.00	0.00
wood	−0.70	0.35	0.00	0.00	0.00
tree	−0.26	0.65	0.00	0.00	0.00

Σ_2	1	2	3	4	5
1	2.16	0.00	0.00	0.00	0.00
2	0.00	1.59	0.00	0.00	0.00
3	0.00	0.00	0.00	0.00	0.00
4	0.00	0.00	0.00	0.00	0.00
5	0.00	0.00	0.00	0.00	0.00

V^T	d_1	d_2	d_3	d_4	d_5	d_6
1	−0.75	−0.28	−0.20	−0.45	−0.33	−0.12
2	−0.29	−0.53	−0.19	0.63	0.22	0.41
3	0.00	0.00	0.00	0.00	0.00	0.00
4	0.00	0.00	0.00	0.00	0.00	0.00
5	0.00	0.00	0.00	0.00	0.00	0.00

C_2	d_1	d_2	d_3	d_4	d_5	d_6
ship	0.85	0.52	0.28	0.13	0.21	−0.08
boat	0.36	0.36	0.16	−0.20	−0.02	−0.18
ocean	1.01	0.72	0.36	−0.04	0.16	−0.21
wood	0.97	0.12	0.20	1.03	0.62	0.41
tree	0.12	−0.39	−0.08	0.90	0.41	0.49

SVD Example (LSA)

- Similarity of d_2 and d_3 in the original space: 0.
- Similarity of d_2 and d_3 in the reduced space: $0.52 * 0.28 + 0.36 * 0.16 + 0.72 * 0.36 + 0.12 * 0.20 + -0.39 * -0.08 \approx \mathbf{0.52}$

C	d_1	d_2	d_3	d_4	d_5	d_6
ship	1	0	1	0	0	0
boat	0	1	0	0	0	0
ocean	1	1	0	0	0	0
wood	1	0	0	1	1	0
tree	0	0	0	1	0	1

C_2	d_1	d_2	d_3	d_4	d_5	d_6
ship	0.85	0.52	0.28	0.13	0.21	−0.08
boat	0.36	0.36	0.16	−0.20	−0.02	−0.18
ocean	1.01	0.72	0.36	−0.04	0.16	−0.21
wood	0.97	0.12	0.20	1.03	0.62	0.41
tree	0.12	−0.39	−0.08	0.90	0.41	0.49

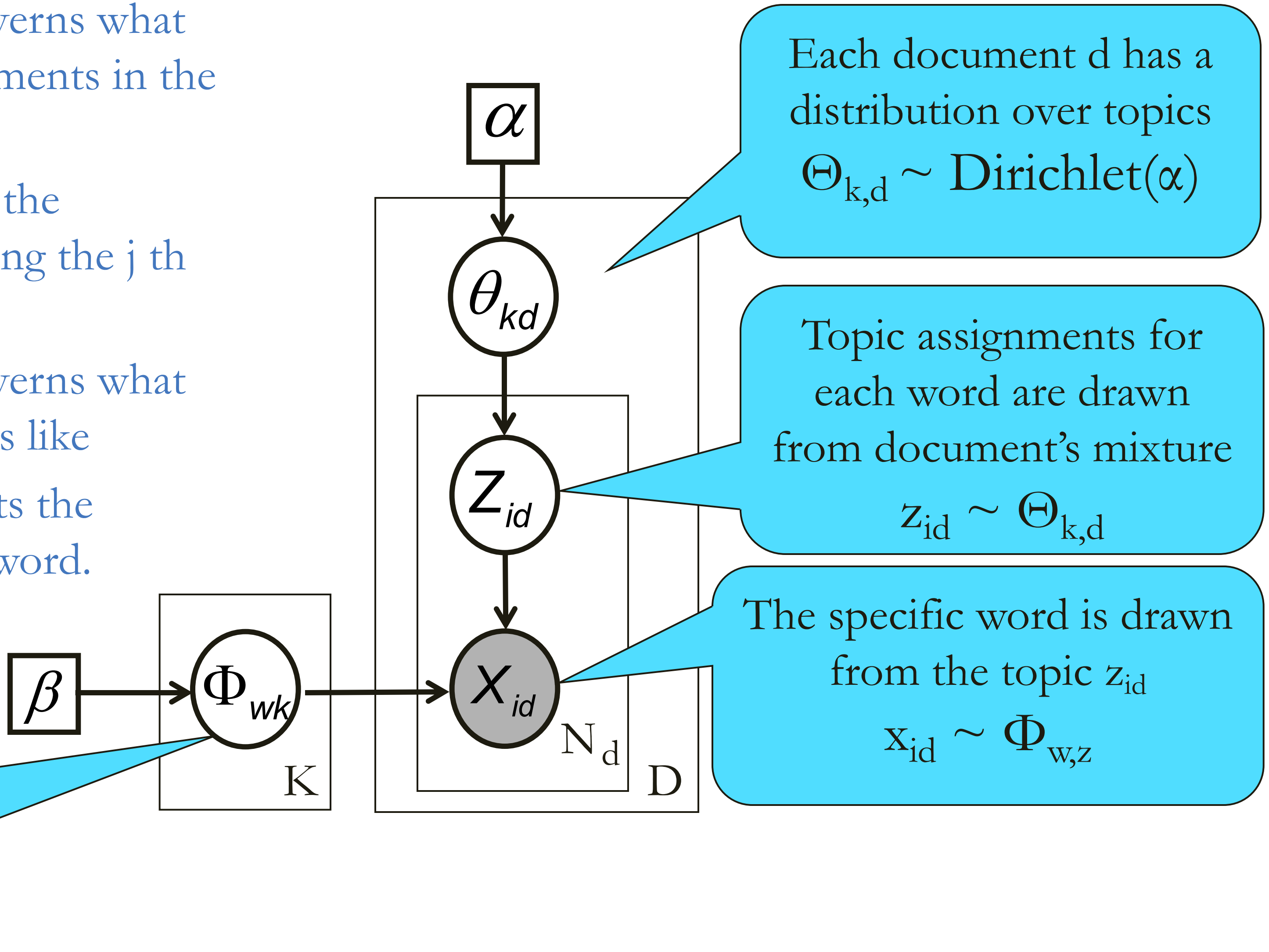
Latent Dirichlet Allocation (LDA)

- History:
  - 1988: *Latent Semantic Analysis* (LSA)
    - Singular Value Decomposition (SVD) of word-document count matrix
  - 1999: *Probabilistic Latent Semantic Analysis* (PLSA)
    - Non-negative matrix factorization (NMF)
  - 2003: *Latent Dirichlet Allocation* (LDA)
    - Bayesian version of PLSA
- Intuition:
  - Each document can be described by a distribution of topics and each topic can be described by a distribution of words

$$\underset{W \times D}{\boxed{\text{P (word | doc)}}} \approx \underset{W \times K}{\boxed{\text{P (word | topic)}}} * \underset{K \times D}{\boxed{\text{P (topic | doc)}}}$$

LDA Graphical Model

- α — Distribution related parameter that governs what the distribution of topics is for all the documents in the corpus looks like
- θ — Random matrix where $\theta(i,j)$ represents the probability of the i th document to containing the j th topic
- η — Distribution related parameter that governs what the distribution of words in each topic looks like
- β — A random matrix where $\beta(i,j)$ represents the probability of i th topic containing the j th word.



LDA: Simpler Explanation

- Given a set of ***M*** documents, each having ***N*** words, where each word is generated by a single topic from a set of ***K*** topics:
- Find the joint posterior probability of:
  - θ — A distribution of topics, one for each document
  - **z** — N Topics for each document,
  - β — A distribution of words, one for each topic

$$P(\theta_{1:M}, \mathbf{z}_{1:M}, \beta_{1:k} | \mathcal{D}; \alpha_{1:M}, \eta_{1:k})$$

- Assuming:
  - D – the corpus of documents
  - α – A parameter vector for each document (document — Topic distribution)
  - η – A parameter vector for each topic (topic — word distribution)

LDA: Another Explanation

- Go through each document and randomly assign each word to one of ***K*** topics
  [This random assignment gives topic representations of all documents and word distributions of all the topics, albeit not very good ones]

- For each document ***d***, go through each word ***w*** and compute:
  - p(topic ***t*** | document ***d***): proportion of words in document ***d*** assigned to topic ***t***
  - p(word ***w***| topic ***t***): proportion of assignments to topic ***t***, over all documents ***d*** that come from word ***w***

- Reassign word ***w*** a new topic ***t'***, where we choose topic ***t'*** with probability

  p(topic ***t'*** | document ***d***) * ***p***(word ***w*** | topic ***t'***)

- This generative model predicts the probability that topic ***t'*** generated word ***w***