1.4 EMBEDDING TECHNIQUES

1.4.1 Latent Semantic Indexing

Vector space retrieval is vague and noisy

- Based on index terms
- Unrelated documents might be included in the answer set
  - apple (company) vs. apple (fruit)
- Relevant documents that do not contain at least one index term are not retrieved
  - car vs. automobile

Observation

- The user information need is more related to concepts and ideas than to index terms

The Problem

Vector Space Retrieval handles poorly the following two situations

1. *Synonymy*: different terms refer to the same concept, e.g. car and automobile
   – Result: poor recall
2. *Homonymy*: the same term may have different meanings, e.g. apple, model, bank
   – Result: poor precision

Example: 3 documents

doc_1	doc_2	doc_3
apple	apple	iOS
blackberry	blackberry	iPad
orange	smartphone	RIM
vitamine	orange	mobile
fruit	carrier	handy
health	tablet	telcom
tablet	swisscom	provider

High similarity No similarity

Key Idea

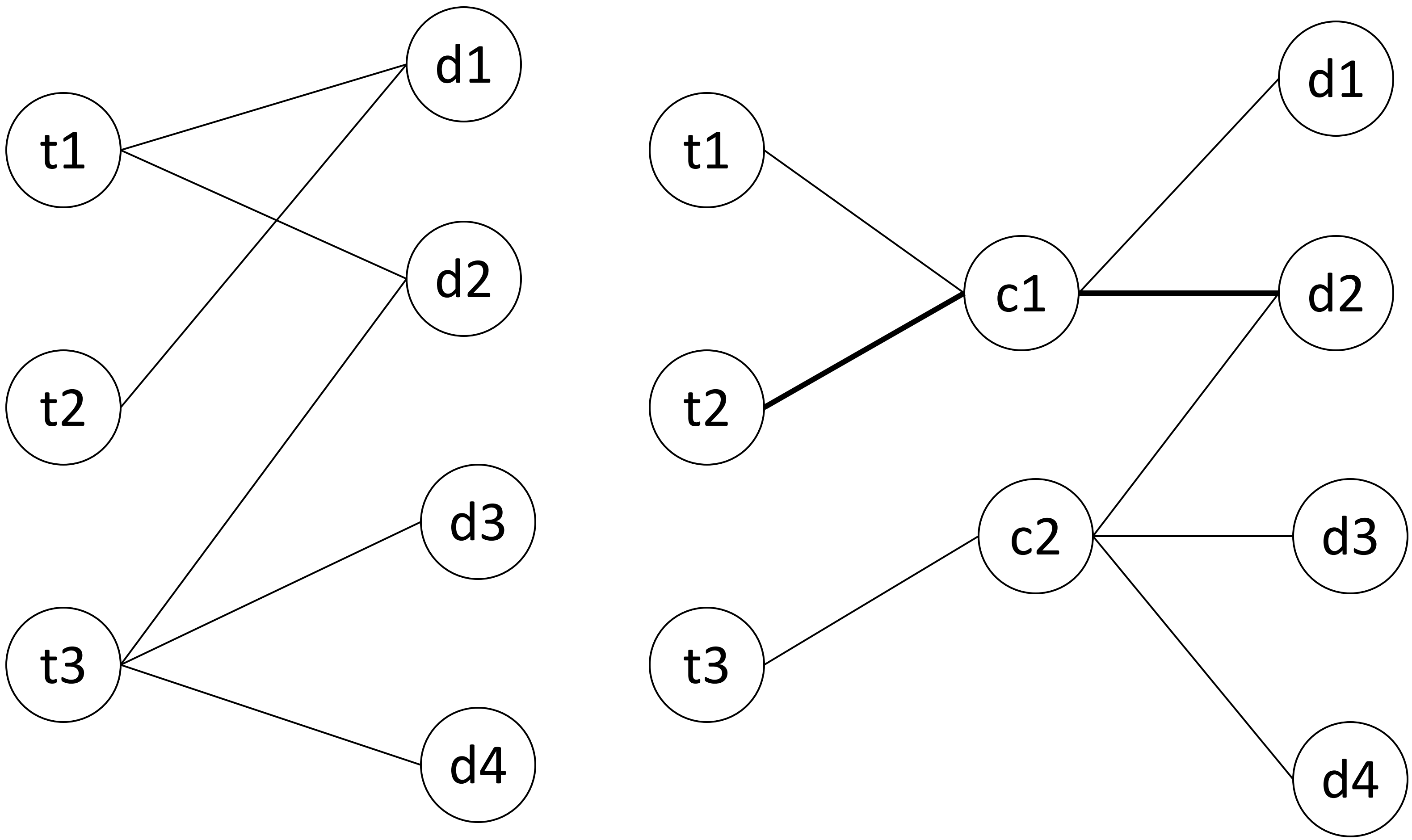
Map documents and queries into a lower-dimensional space composed of higher-level concepts

- Each concept represented by a combination of terms
- Fewer concepts than terms
- e.g. vehicle = {car, automobile, wheels, auto, sportscar}

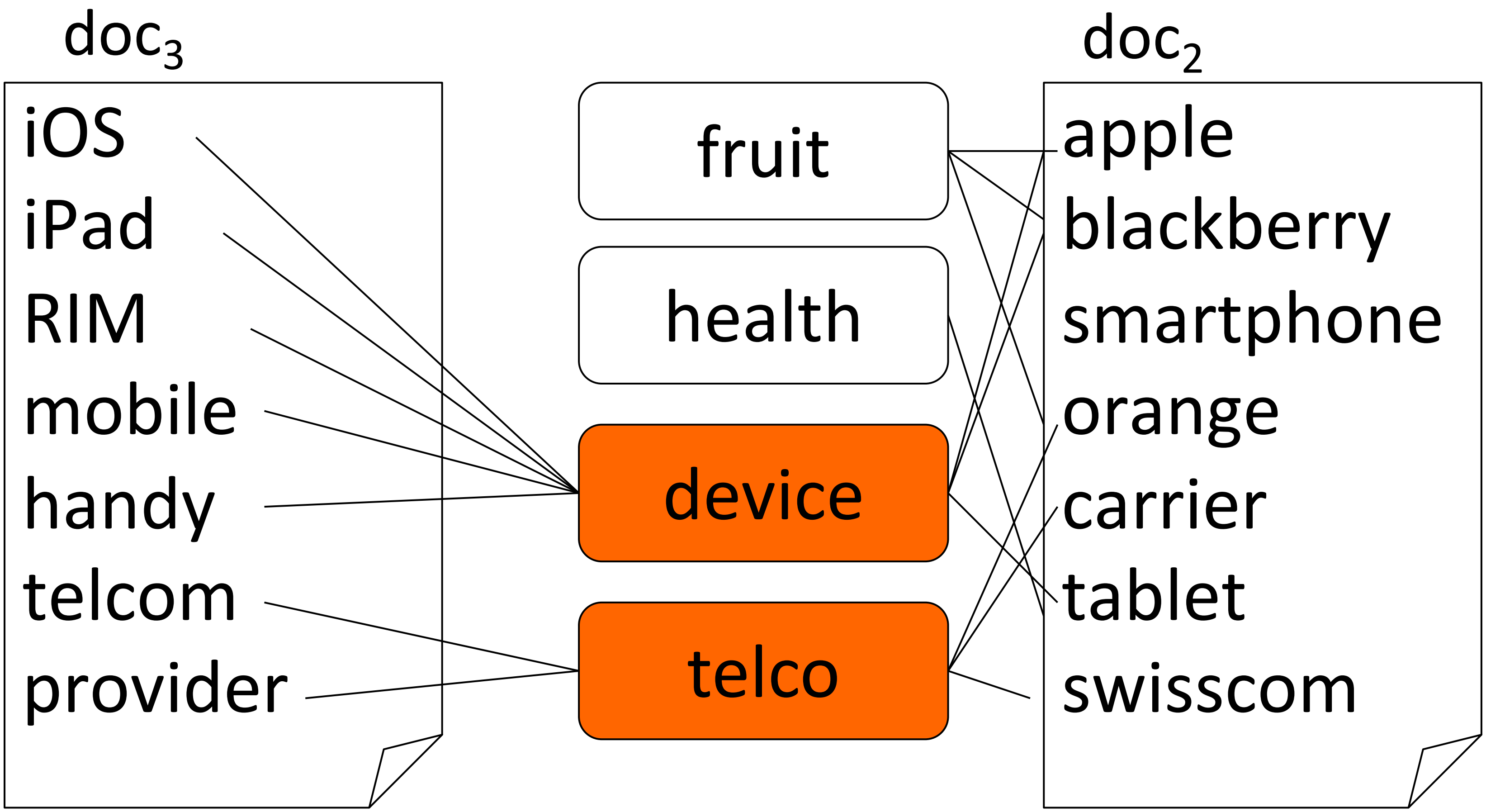
Dimensionality reduction

- Retrieval (and clustering) in a reduced concept space might be superior to retrieval in the high-dimensional space of index terms

Using Concepts for Retrieval



Example: Concept Space



Similarity Computation in Concept Space

Concept represented by terms, e.g.

device = {iOS, iPad, RIM, mobile, handy, tablet, apple, blackberry}

Document represented by concept vector, counting number of concept terms, e.g.

$doc_1 = (4, 3, 3, 1)$
$doc_3 = (0, 0, 5, 2)$

Similarity computed by scalar product of normalized concept vectors

Result

Concept vector (fruit, health, device, telco)

$doc_1 = (4,3,3,1)$	$doc_2=(3,1,3,3)$	$doc_3=(0,0,5,2)$
apple	apple	iOS
blackberry	blackberry	iPad
orange	smartphone	RIM
vitamine	orange	mobile
fruit	carrier	handy
health	tablet	telcom
tablet	swisscom	provider

Similarity(doc_1, doc_2) = 0.245

Similarity(doc_2, doc_3) = 0.3

Similarity(doc_1, doc_3) = 0.22

Basic Definitions

Problem: how to identify and compute “concepts” ?

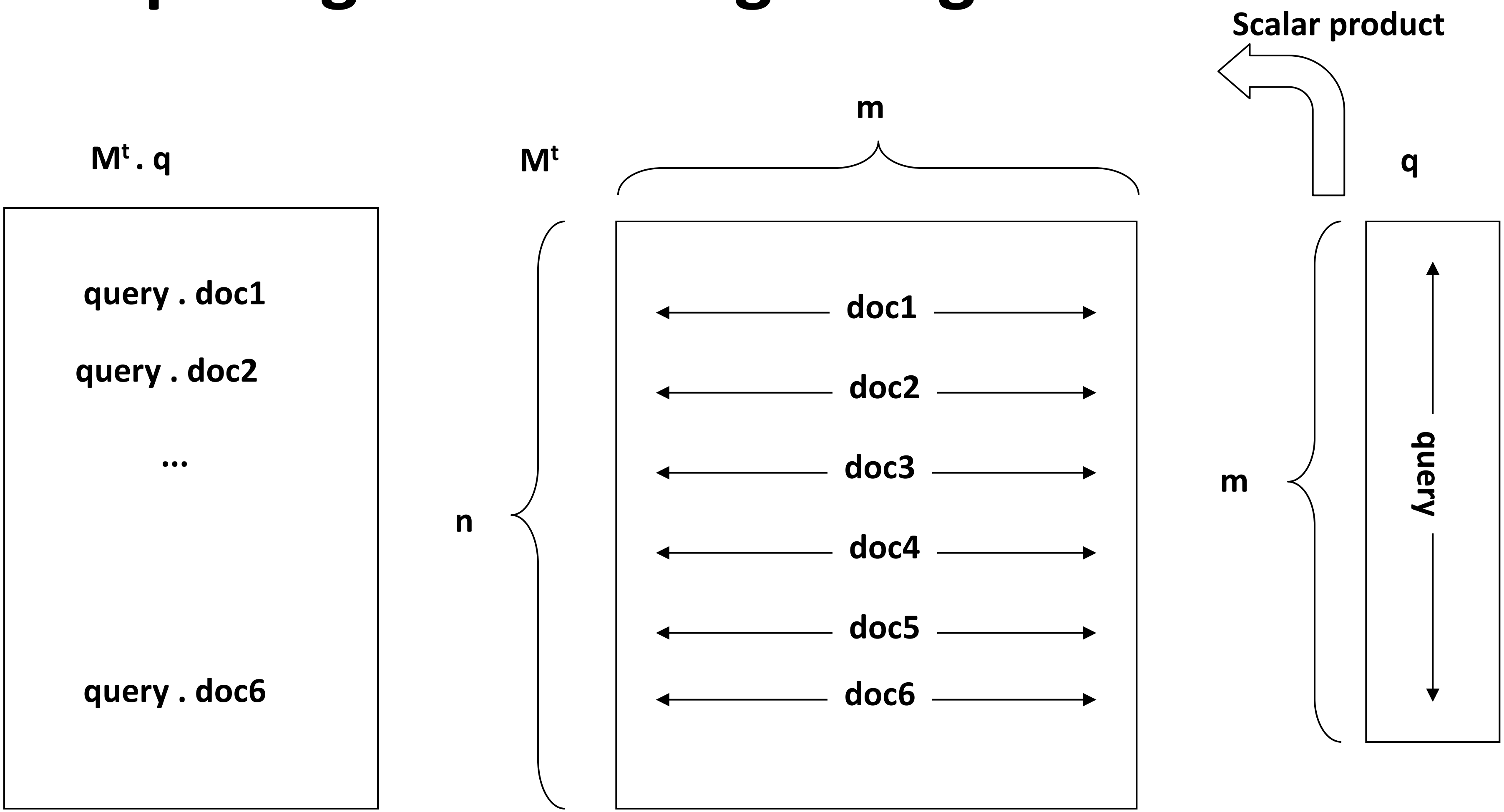
Consider the term-document matrix

– Let M_{ij} be a term-document matrix with m rows (terms) and n columns (documents)

– To each element of this matrix is assigned a weight w_{ij} associated with t_i and d_j

– The weight w_{ij} can be based on a tf-idf weighting scheme

Computing the Ranking Using M



In vector space retrieval each row of the matrix M corresponds to

A. A document

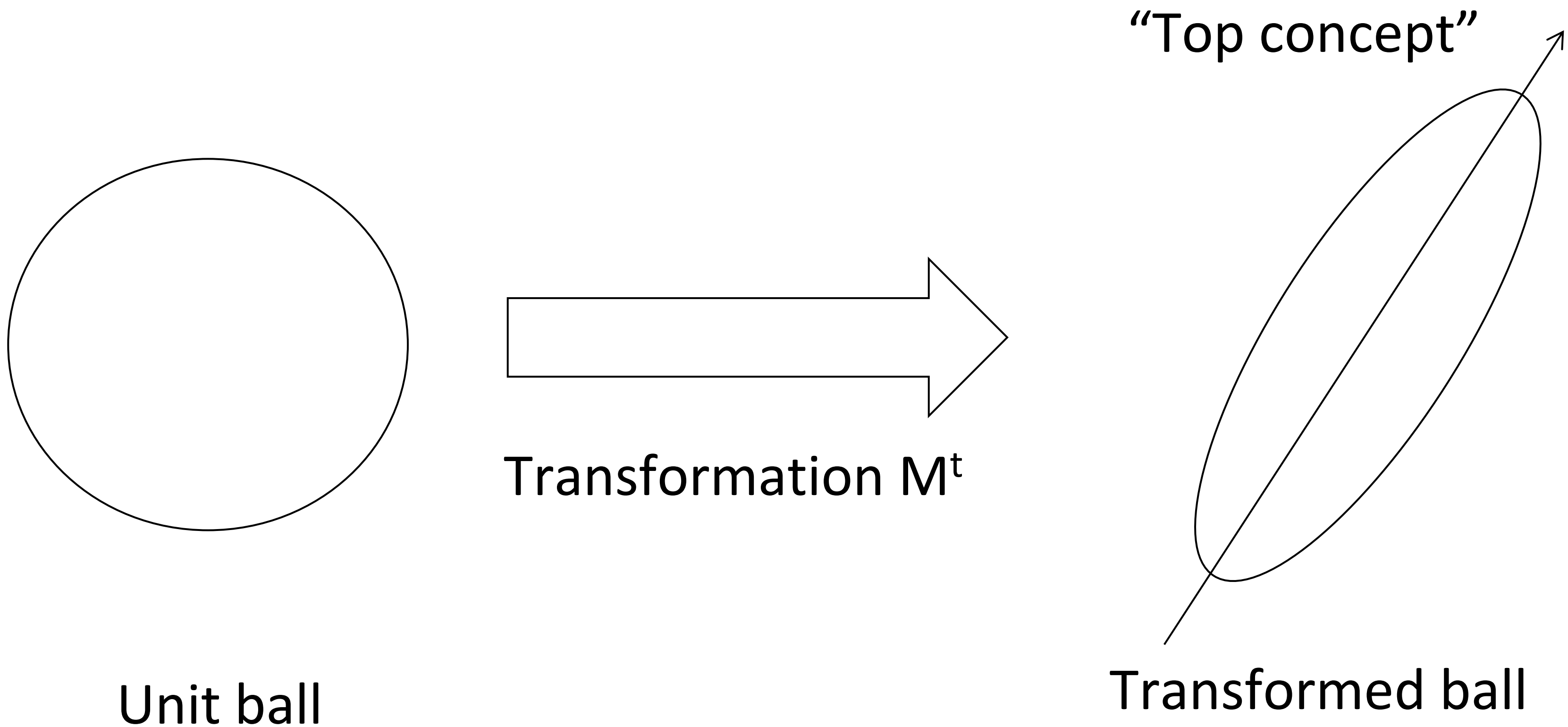
B. A concept

C. A query

D. A term

Identifying Top Concepts

Key Idea: extract the essential features of M^t and approximate it by the most important ones



Singular Value Decomposition (SVD)

Represent Matrix M as $M = K.S.D^t$

– K and D are matrices with orthonormal columns

$$K.K^t = I = D.D^t$$

– S is an r x r diagonal matrix of the singular values sorted in decreasing order where r = min(m, n), i.e. the rank of M

– Such a decomposition always exists and is unique (up to sign)

Construction of SVD

K is the matrix of eigenvectors derived from $M.M^t$

D is the matrix of eigenvectors derived from $M^t.M$

Algorithms for constructing the SVD of a m x n matrix have complexity $O(n^3)$ if $m \leq n$

Interpretation of SVD

We can write M = K.S.D^t as sum of outer vector products

$$M = \sum_{i=1}^{r} s_i k_i \otimes d_i^t$$

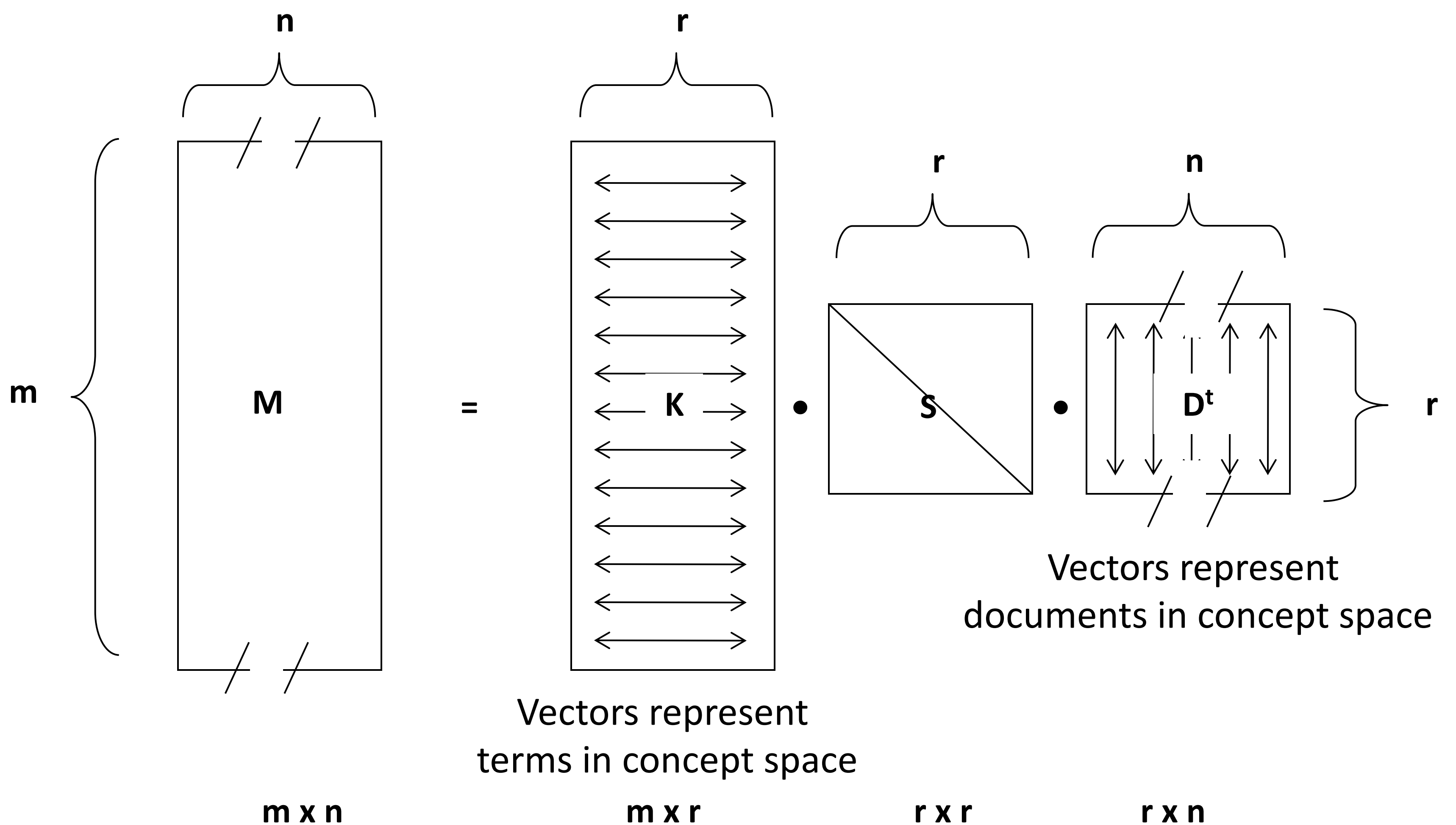
The s_i are ordered in decreasing size

By taking only the largest ones we obtain a «good» approximation of M (least square approximation)

The singular values s_i are the lengths of the semi-axes of the hyperellipsoid E defined by

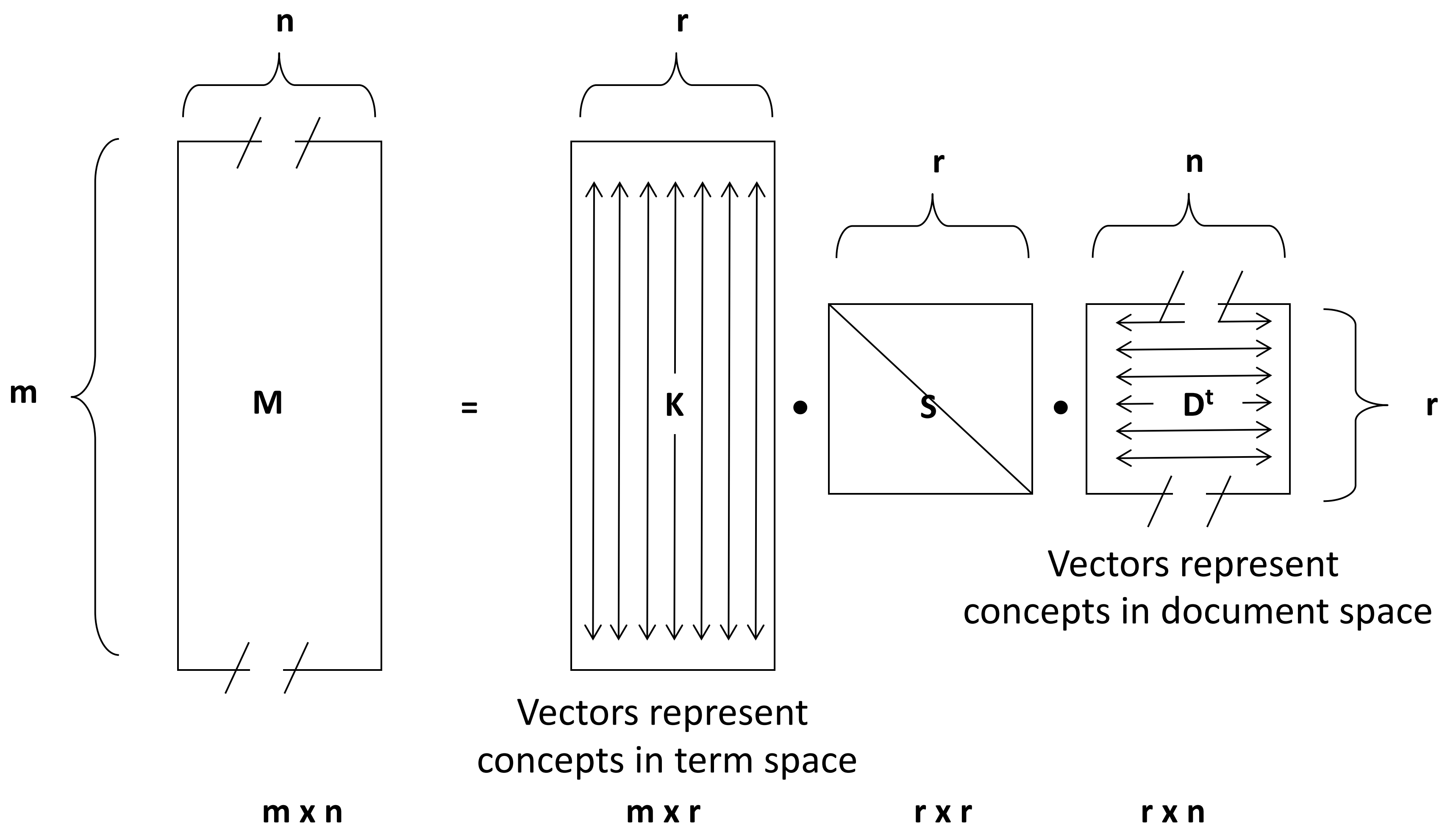
$$E = \left\{ Mx \mid \|x\|_2 = 1 \right\}$$

Illustration of SVD



Assuming m ≤ n

Illustration of SVD – Another Perspective



Assuming m ≤ n

Latent Semantic Indexing (LSI)

In the matrix S, select only the s largest singular values

– Keep the corresponding columns in K and D

The resultant matrix is called M_s and is given by

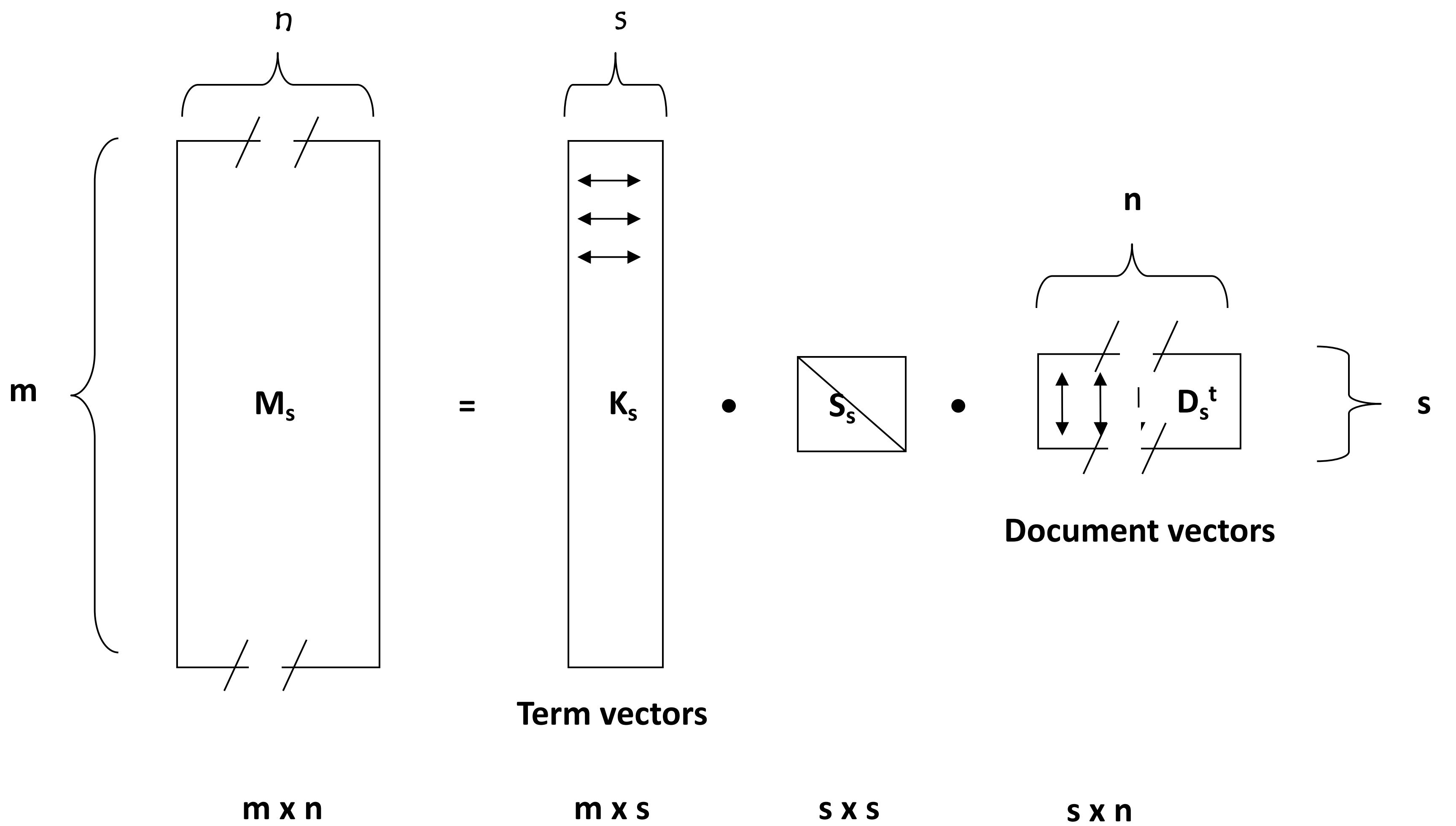
– $M_s = K_s.S_s.D_s^t$ where s, $s < r$, is the dimensionality of the concept space

The parameter s should be

– large enough to allow fitting the characteristics of the data

– small enough to filter out the non-relevant representational details

Illustration of Latent Semantic Indexing



Answering Queries

Documents can be compared by computing cosine similarity in the concept space, i.e., comparing their columns $(D_s^t)_i$ and $(D_s^t)_j$ in matrix D_s^t

A query q is treated like one further document

- it is added as an additional column to matrix M
- the same transformation is applied to this column as for mapping M to D

Mapping Queries

Mapping of M to D

$M = K.S.D^t$

$S^{-1}.K^t.M = D^t$ (since $K.K^t = 1$)

$D = M^t.K.S^{-1}$

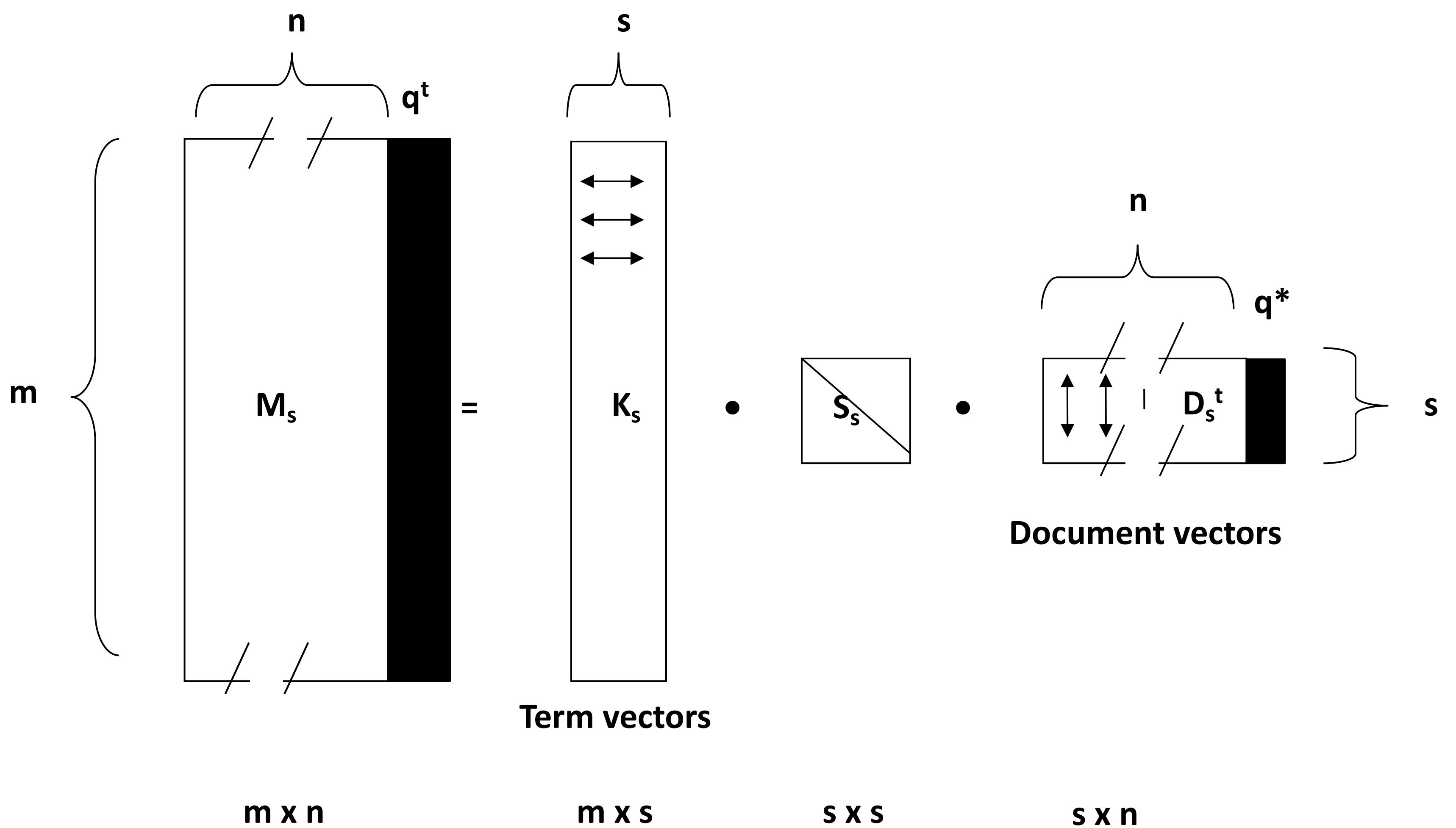
Apply same transformation to q:

$q^* = q^t.K_s.S_s^{-1}$

Then compare transformed vector by using the standard cosine measure

$$sim(q^*, d_i) = \frac{q^* \bullet (D_s^t)_i}{|q^*| |(D_s^t)_i|}$$

Illustration of LSI Querying



Example: Documents

B1 A Course on Integral Equations
B2 Attractors for Semigroups and Evolution Equations
B3 Automatic Differentiation of Algorithms: Theory, Implementation, and Application
B4 Geometrical Aspects of Partial Differential Equations
B5 Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra
B6 Introduction to Hamiltonian Dynamical Systems and the N-Body Problem
B7 Knapsack Problems: Algorithms and Computer Implementations
B8 Methods of Solving Singular Systems of Ordinary Differential Equations
B9 Nonlinear Systems
B10 Ordinary Differential Equations
B11 Oscillation Theory for Neutral Differential Equations with Delay
B12 Oscillation Theory of Delay Differential Equations
B13 Pseudodifferential Operators and Nonlinear Partial Differential Equations
B14 Sinc Methods for Quadrature and Differential Equations
B15 Stability of Stochastic Differential Equations with Respect to Semi-Martingales
B16 The Boundary Integral Approach to Static and Dynamic Contact Problems
B17 The Double Mellin-Barnes Type Integrals and Their Applications to Convolution Theory

Implementation in Python

```
from sklearn.feature_extraction.text import TfidfVectorizer

tf = TfidfVectorizer(analyzer='word', ngram_range=(1,1), min_df = 2, stop_words = 'english')
features = tf.fit_transform(titles)
M = np.transpose(np.array(features.todense()))
```

```
# compute SVD
K, S, Dt = np.linalg.svd(M, full_matrices=False)

# LSI select dimensions
K_sel = K[:,0:2]
S_sel = np.diag(S)[0:2,0:2]
Dt_sel = Dt[0:2,:]
```

Results (s=2)

```
K_sel
```

```
array([[ 0.01781272, -0.4729881 ],
       [ 0.03264057, -0.43230378],
       [ 0.15088442, -0.17568951],
       [ 0.55589867,  0.07082109],
       [ 0.6843092 ,  0.1075997 ],
       [ 0.01570413, -0.37133288],
       [ 0.09073864, -0.07173948],
       [ 0.01775573, -0.20943739],
       [ 0.19758761,  0.08201858],
       [ 0.11060875,  0.05205271],
       [ 0.19758761,  0.08201858],
       [ 0.15088442, -0.17568951],
       [ 0.20802226,  0.09313466],
       [ 0.01555703, -0.22913745],
       [ 0.10330872, -0.00853892],
       [ 0.14994428, -0.49674497]])
```

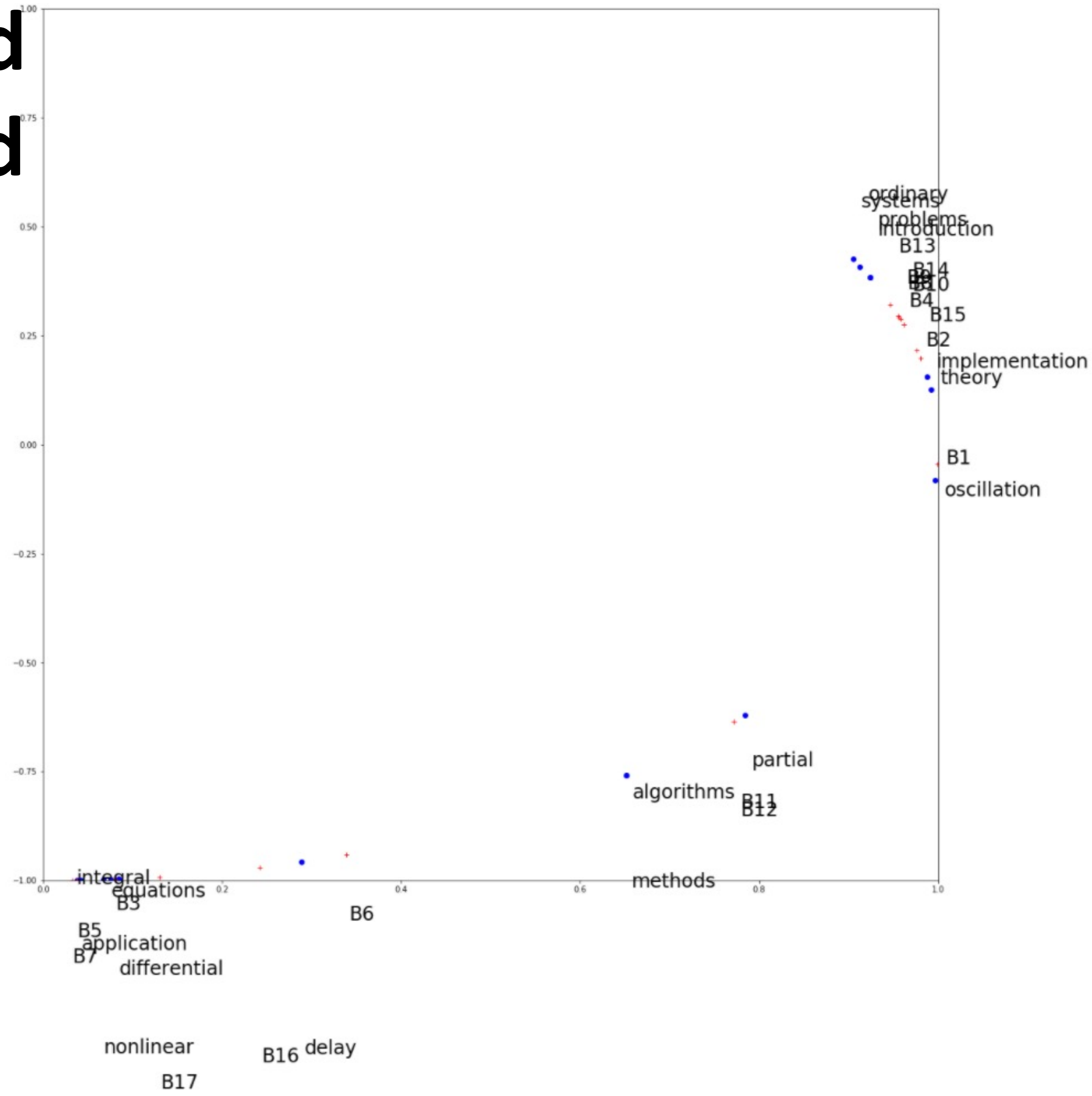
```
S_sel
```

```
array([[2.12109044, 0.        ],
       [0.        , 1.50037763]])
```

```
np.transpose(Dt_sel)
```

```
array([[ 0.18982901, -0.0083562 ],
       [ 0.32262142,  0.07171508],
       [ 0.04727092, -0.58437076],
       [ 0.33399497,  0.1003478 ],
       [ 0.01183895, -0.31440669],
       [ 0.03875274, -0.10771362],
       [ 0.01329859, -0.40782546],
       [ 0.3018997 ,  0.09205811],
       [ 0.07134057,  0.02202618],
       [ 0.33016936,  0.09458633],
       [ 0.29162009, -0.24019296],
       [ 0.29162009, -0.24019296],
       [ 0.29566823,  0.10051203],
       [ 0.33016936,  0.09458633],
       [ 0.40993831,  0.08283115],
       [ 0.03543573, -0.14179904],
       [ 0.05664377, -0.43260133]])
```

Plot of Terms and Documents in 2-d Concept Space



Applying SVD to a term-document matrix M. Each concept is represented in K

A. as a singular value
B. as a linear combination of terms of the vocabulary
C. as a linear combination of documents in the document collection
D. as a least squares approximation of the matrix M

The number of term vectors in the matrix K_s used for LSI

A. Is smaller than the number of rows in the matrix M

B. Is the same as the number of rows in the matrix M

C. Is larger than the number of rows in the matrix M

A query transformed into the concept space for LSI has …

A. s components (number of singular values)
B. m components (size of vocabulary)
C. n components (number of documents)

Discussion of Latent Semantic Indexing

Latent semantic indexing provides an interesting conceptualization of the IR problem

Advantages

- It allows reducing the complexity of the underlying concept representation
- Facilitates interfacing with the user

Disadvantages

- Computationally expensive
- Poor statistical explanation

Alternative Techniques

Probabilistic Latent Semantic Analysis

- Based on Bayesian Networks

Latent Dirichlet Allocation

- Based on Dirichlet Distribution
- State-of-the-art method for concept extraction

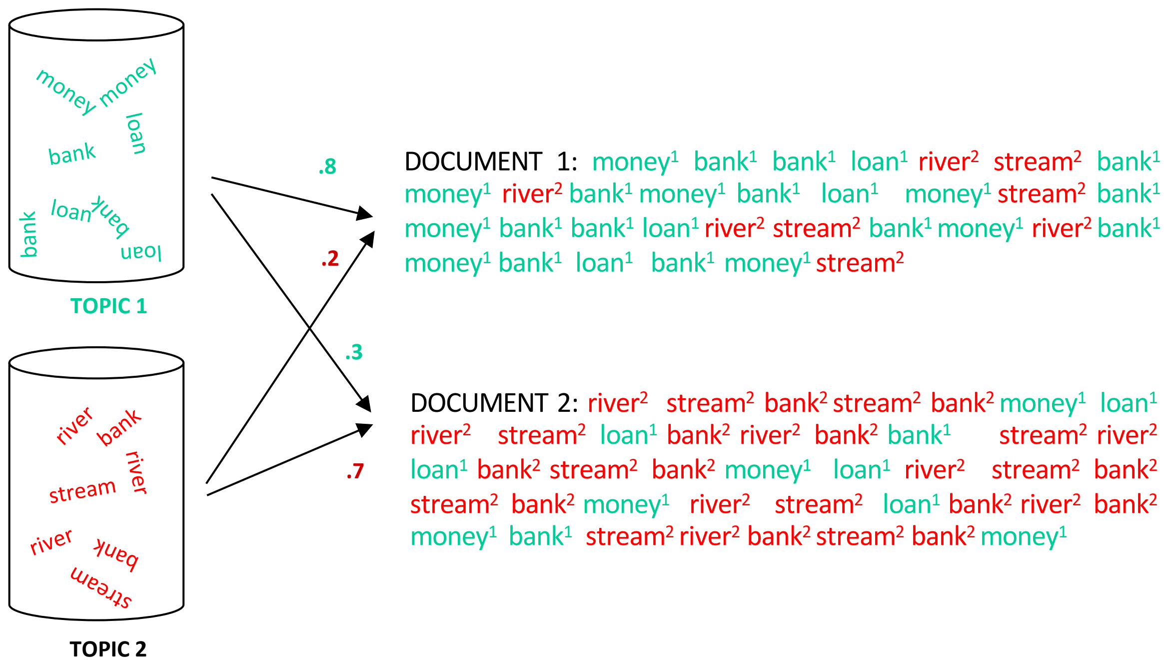
Same objective of creating a lower-dimensional concept space based on the term-document matrix

- Better explained mathematical foundation
- Better experimental results

1.4.2 Latent Dirichlet Allocation (LDA)

Idea: assume a document collection is (randomly) generated from a known set of topics (probabilistic generative model)

Topic is a mixture of words



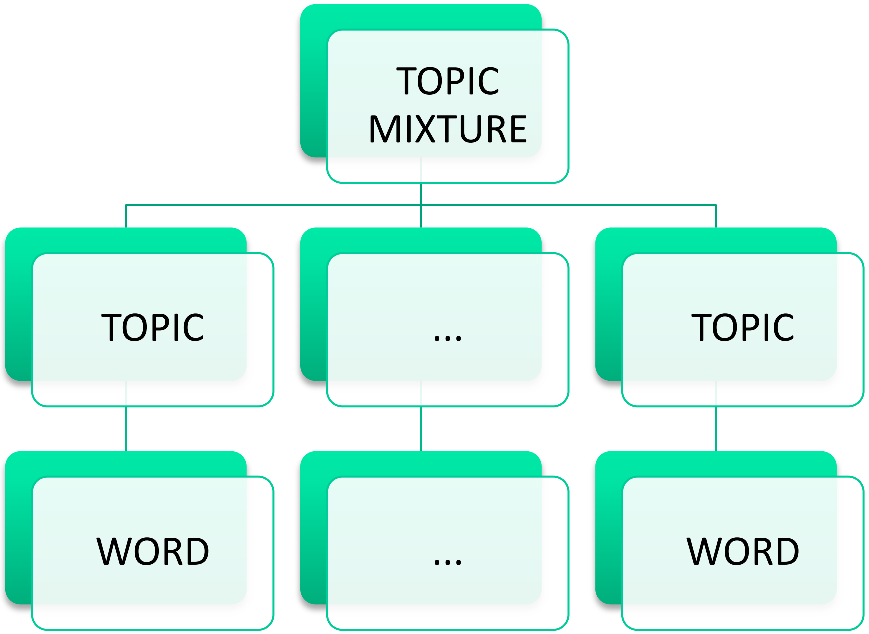
Document is a mixture of topics

Document Generation using a Probabilistic Process

For each document, choose a mixture of topics

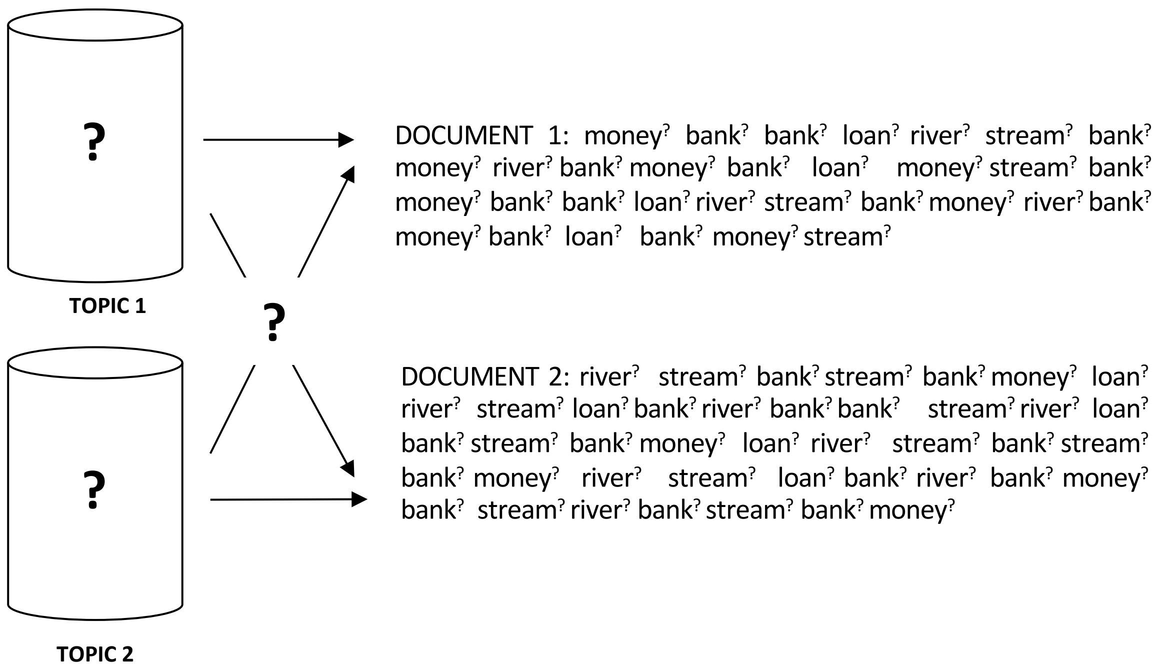
For every word position, sample a topic from the topic mixture

For every word position, sample a word from the chosen topic



LDA: Topic Identification

Approach: Inverting the process: given a document collection, reconstruct the topic model



Latent Dirichlet Allocation

Topics are **interpretable** unlike the arbitrary dimensions of LSI

- Distributions follow a Dirichlet distribution
- Construction of topic model is mathematically involved, but computationally feasible
- Considered as the state-of-the art method for topic identification

Use of Topic Models

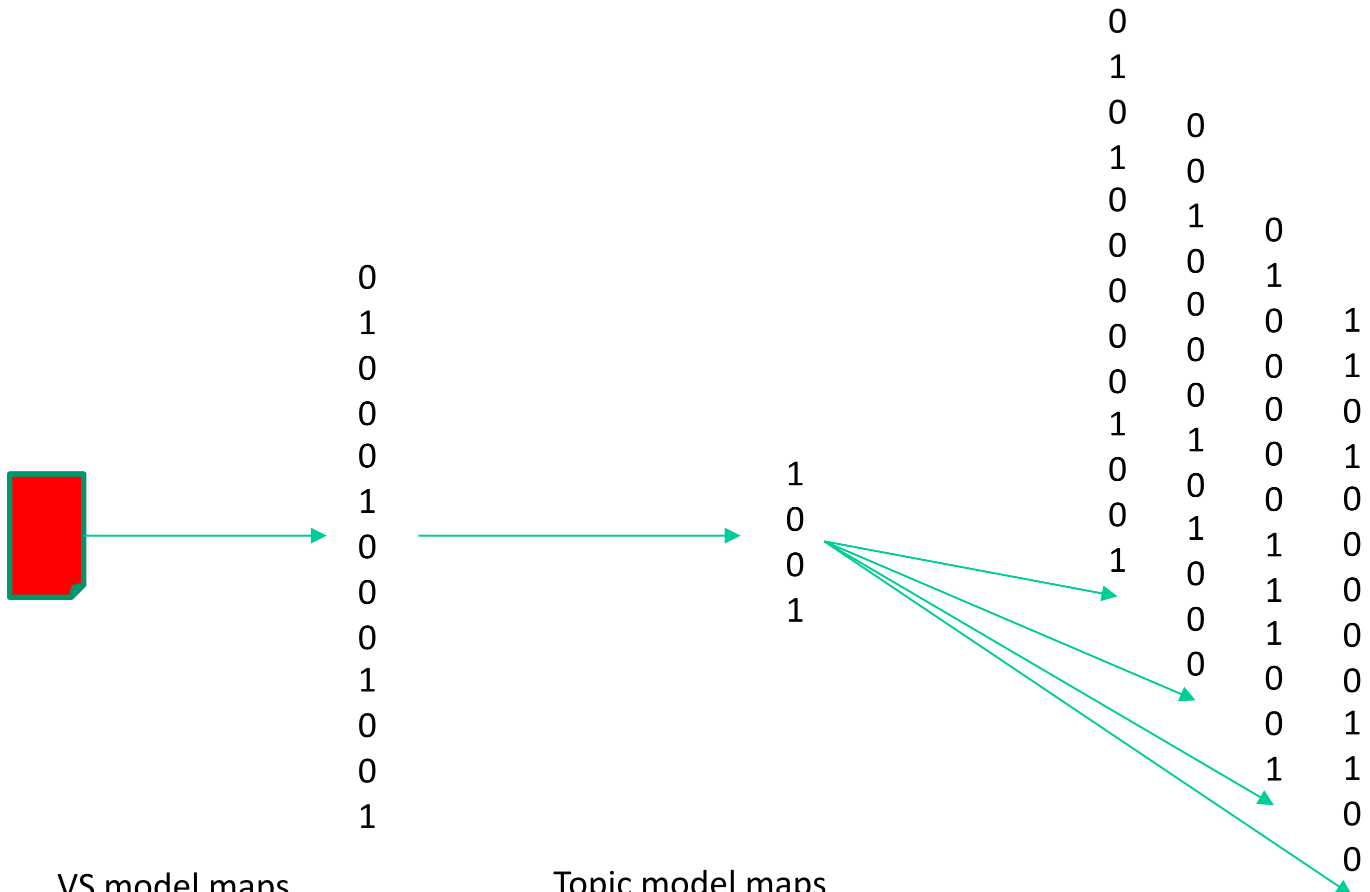
Unsupervised Learning of topics

- Understanding main topics of a topic collection
- Organizing the document collection

Use for document retrieval: use topic vectors instead of term vectors to represent documents and queries

Document classification (Supervised Learning): use topics as features

Summary



VS model maps documents to sparse term vectors (dimension = term in vocabulary)

Topic model maps documents to dense vectors (dimension = topic)

Each topic corresponds to a distribution over terms