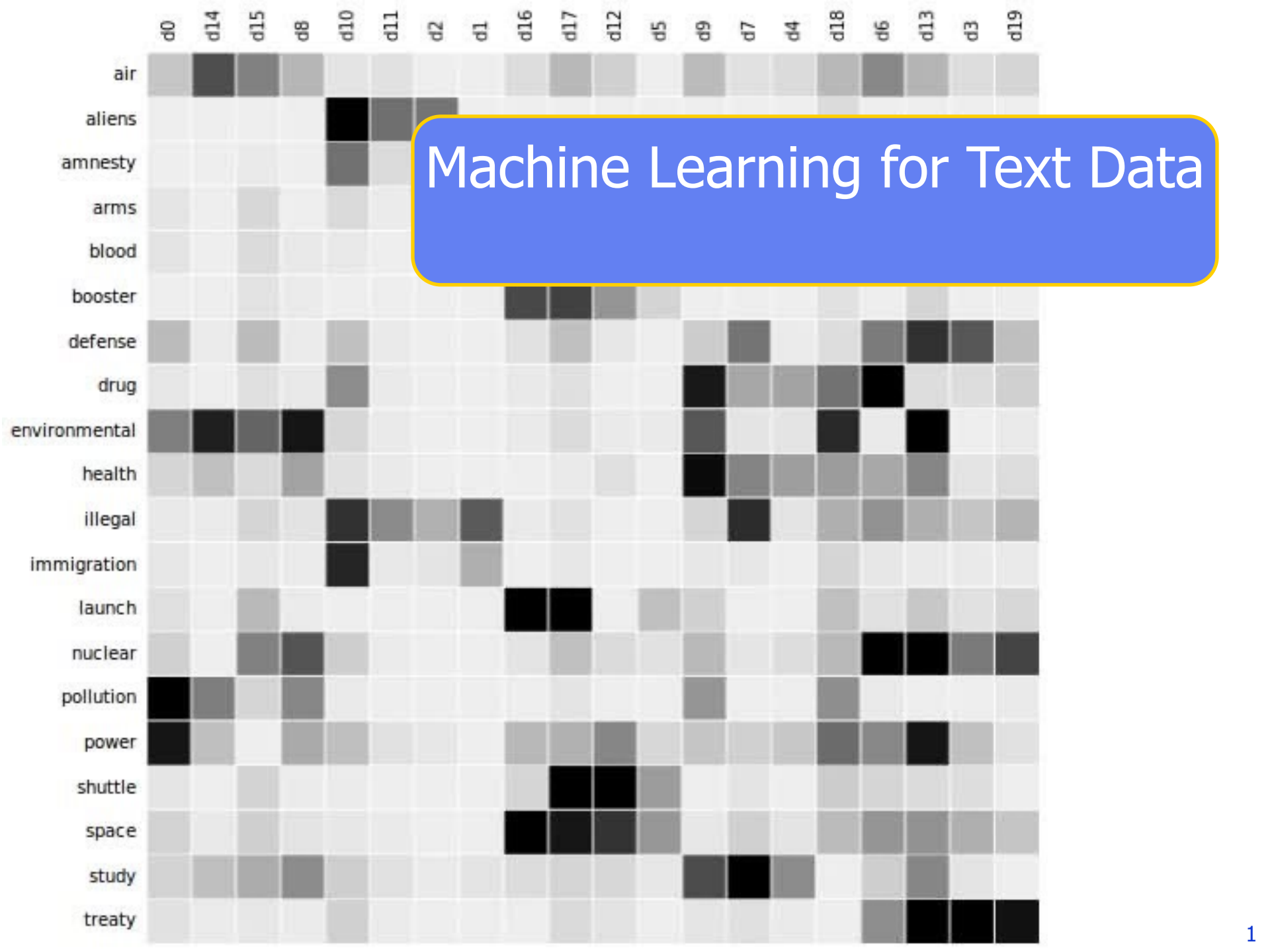

d0
d14
d15
d8
d10
d11
d2
d1
d16
d17
d12
d5
d9
d7
d4
d18
d6
d13
d3
d19
air
aliens
amnesty
arms
blood
booster
defense
drug
environmental
health
illegal
immigration
launch
nuclear
pollution
power
shuttle
space
study
treaty
Machine Learning for Text Data

Agenda

- Text Representation
- Text Similarity
- Term Frequency and BoW Model
- Vector-Space Model
- Applications
- Latent Semantics Analysis (LSA)
- Summary

Text Representation

- Each document becomes a 'term' vector,
  - each term is a component (attribute) of the vector,
  - the value of each component is the number of times the corresponding term occurs in the document.

Document 1

The quick brown fox jumped over the lazy dog's back.

Document 2

Now is the time for all good men to come to the aid of their party.

Term	Document 1	Document 2
aid	0	1
all	0	1
back	1	0
brown	1	0
come	0	1
dog	1	0
fox	1	0
good	0	1
jump	1	0
lazy	1	0
men	0	1
now	0	1
over	1	0
party	0	1
quick	1	0
their	0	1
time	0	1

Stopword List

for
is
of
the
to

Text Similarity

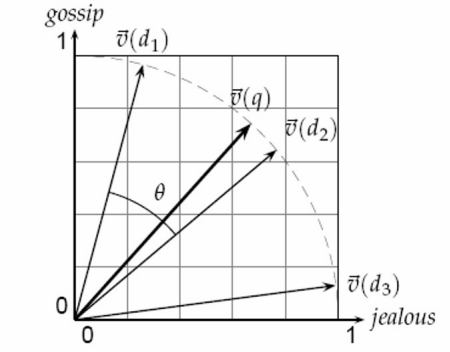
- Finds similar documents based on a set of common keywords
- Answer should be based on the degree of relevance based on the *nearness of the keywords*, *relative frequency of the keywords*, etc.
- Basic techniques
  - Stop list
    - Set of words that are deemed "irrelevant", even though they may appear frequently (30 most common words account for 30% of the tokens in written text)
    - E.g., *a, the, of, for, with*, etc.
    - Stop lists may vary when document set varies

Text Similarity (cont.)

- Basic techniques (cont.)
  - Word stem
    - Several words are small syntactic variants of each other since they share a common word stem
    - E.g., *drug, drugs, drugged*
  - A term frequency table
    - Each entry *frequency_table(i, j)* = # of occurrences of the word t_i in document d_j
    - Usually, the *ratio* instead of the absolute number of occurrences is used
  - Similarity metrics: measure the closeness of a document to a query (a set of keywords)
    - Relative term occurrences
    - Cosine distance:

$$S(v_1, v_2) = \frac{v_1 \cdot v_2}{\|v_1\| \times \|v_2\|}$$

Cosine Similarity



Cosine similarity illustrated. $\text{sim}(d_1, d_2) = \cos\theta$.

- If d_1 and d_2 are two document vectors, then

$$\cos(d_1, d_2) = \frac{d_1 \cdot d_2}{\|d_1\| \times \|d_2\|}$$

where • indicates vector dot product and $||d||$ denotes the length of vector d.

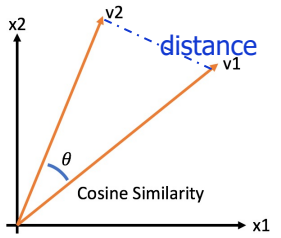
- Example:

d_1 = **3 2 0 5 0 0 0 2 0 0**
d_2 = **1 0 0 0 0 0 0 1 0 2**

$d_1 \bullet d_2$= 3*1 + 2*0 + 0*0 + 5*0 + 0*0 + 0*0 + 0*0 + 2*1 + 0*0 + 0*2 = 5
$||d_1||$ = $(3*3+2*2+0*0+5*5+0*0+0*0+0*0+2*2+0*0+0*0)^{0.5}$ = $(42)^{0.5}$ = 6.481
$||d_2||$ = $(1*1+0*0+0*0+0*0+0*0+0*0+0*0+1*1+0*0+2*2)^{0.5}$ = $(6)^{0.5}$ = 2.245

$\cos(d_1, d_2) = 0.3150$ vs
$\text{dist}(d_1, d_2) = \sqrt{(3-1)^2+(2-0)^2+\cdots+(0-2)^2}$

Cosine similarity does not take into consideration of absolute term frequencies!



Term frequency and weighting

- A word that appears often in a document is probably very descriptive of what the document is about
- Assign to each term in a document *a weight* for that term, that depends on the number of occurrences of that term in the document
- Term frequency (tf)
  - Assign the weight to be equal to the number of occurrences of term t in document d

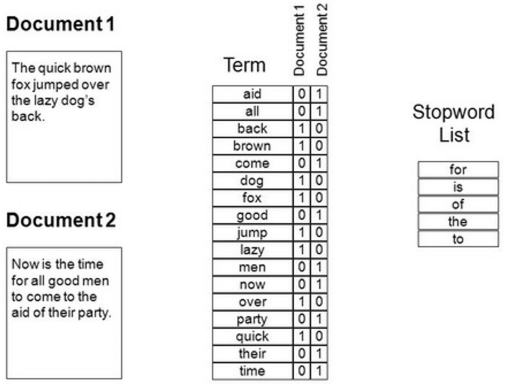
Bag of Words (BoW) model

- A document can now be viewed as the collection of terms in it and their associated weight, e.g.
  - Mary is smarter than John
  - John is smarter than Mary

  which are equivalent in the BoW model
- So, BoW is a degenerated model. Some variants would like to recover the positional information. You can learn more of it in the NLP (Natural Language Processing) subject.

Problems with term frequency

- Stop words
  - Semantically vacuous (meaningless!)



- Synonym or Polysemy
  - "Auto" or "car" would not be at all indicative about what a document/sentence is about

- Need a mechanism for attenuating the effect of terms that occur too often in the collection to be meaningful for relevance/meaning determination

Collection or Document frequency

- (i) Scale down the term weight of terms with high **collection frequency (cf)**
  - Reduce the **tf** weight of a term by a factor that grows with the collection frequency
- (ii) More common for this purpose is **document frequency (df)** (how many documents in the collection contain the term)

Word	cf	df
try	10422	8760
insurance	10440	3997

cf: *number of times the word appears in the document collection (all documents)*

df: *number of documents the word appears in*

Inverse document frequency (idf)

$$\text{idf}_t = \log \frac{N}{\text{df}_t}.$$

N number of documents in the collection; log is typically referred to natural log (ln)

- N=1000; df[the]=1000; idf[the] = 0
- N=1000; df[some]=100; idf[some] = 2.3
- N=1000; df[car]=10; idf[car] = 4.6
- N=1000; df[merger]=1; idf[merger] = 6.9

tf-idf weighting

$$\text{tf-idf}_{t,d} = \text{tf}_{t,d} \times \text{idf}_t.$$

- Highest when term t occurs many times within a small number of documents
  - Thus lending high discriminating power to those documents
- Lower when the term occurs fewer times in a document, or occurs in many documents
  - Thus offering a less pronounced relevance signal
- Lowest when the term occurs in virtually all documents
  - Like stop words: *a, the, of, for, with*

Document Representation:
Vector-Space Model

- Each document is viewed as a vector with one component corresponding to each term in the dictionary
- The value of each component is the **tf-idf** score for that word
- For dictionary terms that do not occur in the document, the weights are 0

Applications: Text Classification

- In a Kaggle competition (https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge) about text classification with six classes, one may need to analyse such table.

Feature representation

Class labeling

Doc	tfidf1	tfidf2		tfidfX	toxic	severe_ toxic	obscene	threat	insult	identity_ hate
1										
2										
…										
N										

- If k-nearest neighbour is used, a test doc's tfidf vector will be used to find kNN based on cosine distances. The neighbors' labels on the six toxicity levels can then be used to predict the test doc's ones.

Applications: Text Clustering

- For each of the terms (words or phrases) we determine to be important across all documents, we will have a separate feature. If there are 10,000 terms, then each document will have 10,000 new features. Each value will be the tf-idf weight of that term for that document, i.e., each document will be in a 10000-D space for ML!
- Corresponding term-document set:

Doc	tfidf1	tfidf2					tfidf10000
1							
2							
…							
N							

i.e. a data matrix in our clustering slides, a Nx10000 matrix!

- Thus, we can apply a clustering algorithm say k-means on it!

What's wrong with such document representation?

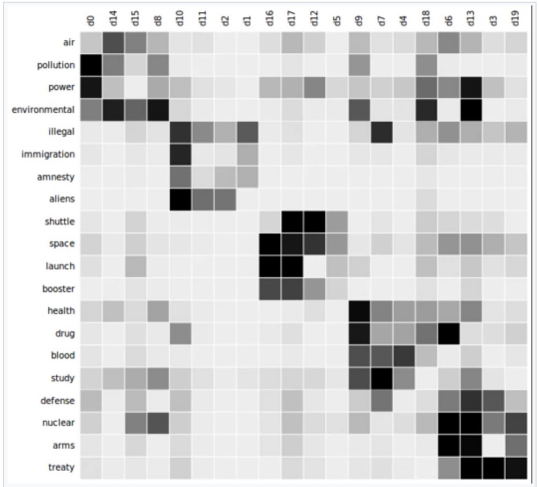
- The term-doc matrix may have large M=50000 terms, N=10M docs (and rank close to 50000)
- Practically, we need to find an approximation with lower rank.
- Effectively, dimensionality reduction is applied.

More practical representation:

Latent Semantic Analysis (LSA)

- Latent semantic analysis (LSA) is a technique in natural language processing (NLP) of analyzing relationships between a set of documents and the terms they contain by producing a set of concepts related to the documents and terms.
- LSA assumes a **term-document matrix** which describes the occurrences of terms in documents; it is a sparse matrix whose rows correspond to terms and whose columns correspond to documents.
- In LSA, SVD (singular value decomposition) is used to reduce the number of rows while preserving the similarity structure among columns.

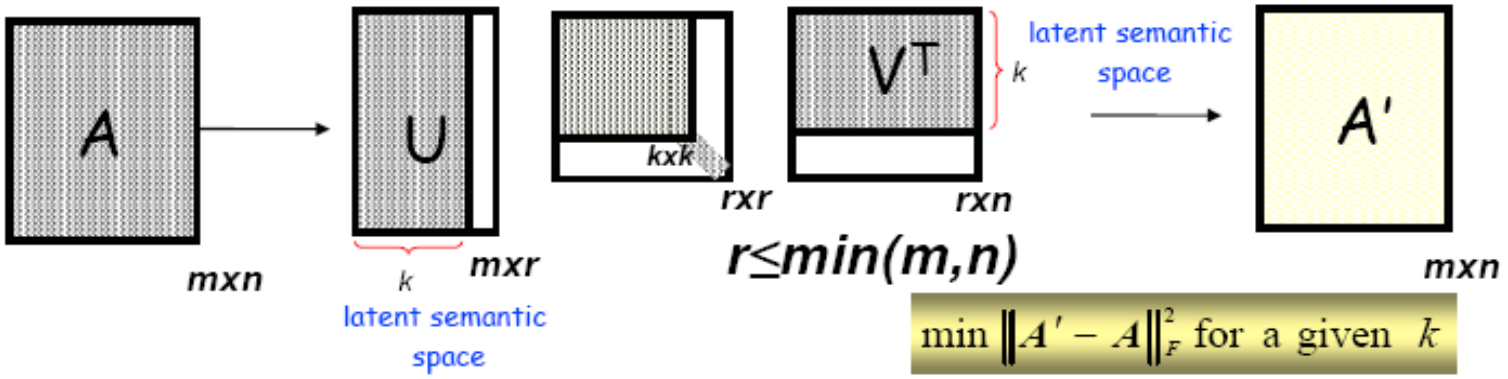
Wiki link: LSA



Animation of the topic detection process in a document-word matrix. Every column corresponds to a document, every row to a word. A cell stores the weighting of a word in a document (e.g. by tf-idf), dark cells indicate high weights. LSA groups both documents that contain similar words, as well as words that occur in a similar set of documents. The resulting patterns are used to detect latent components.[3]

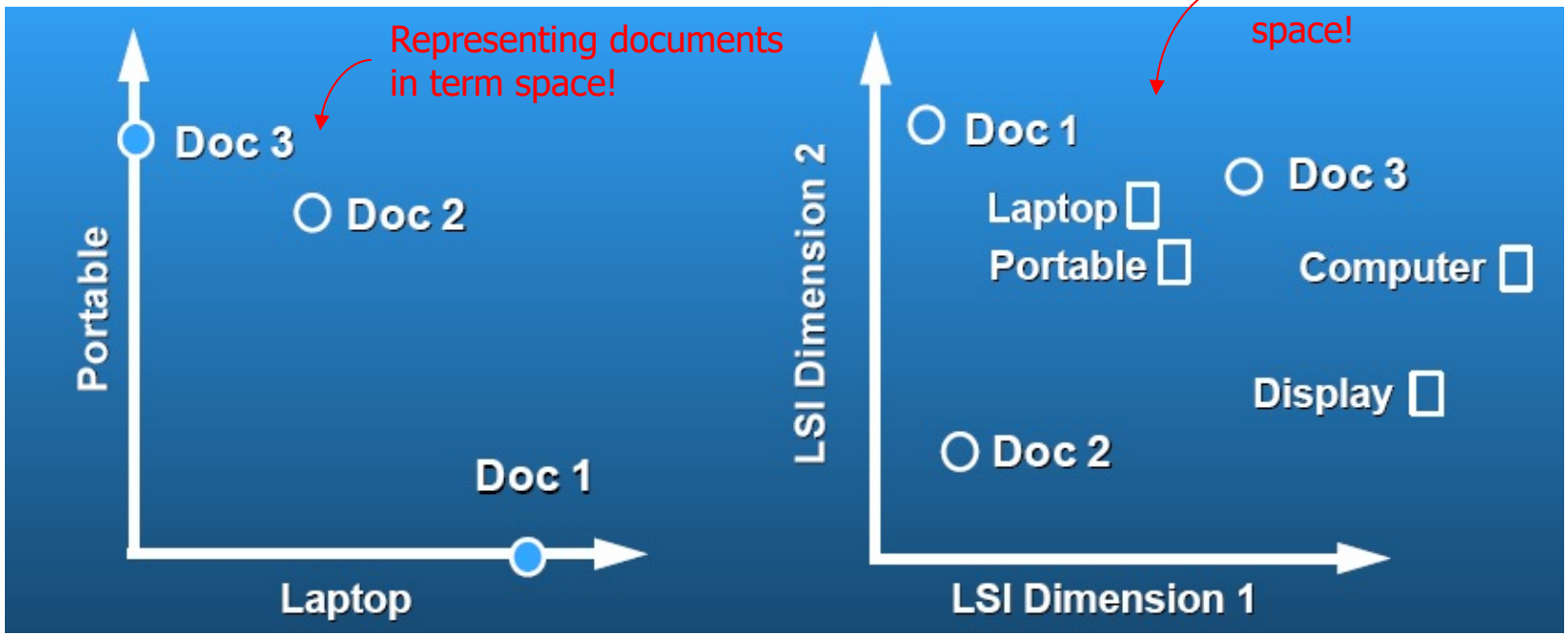
Latent Semantic Analysis

- SVD in LSA: $A = U\Sigma V^T$



Latent Semantic Analysis/Indexing (LSA/LSI)

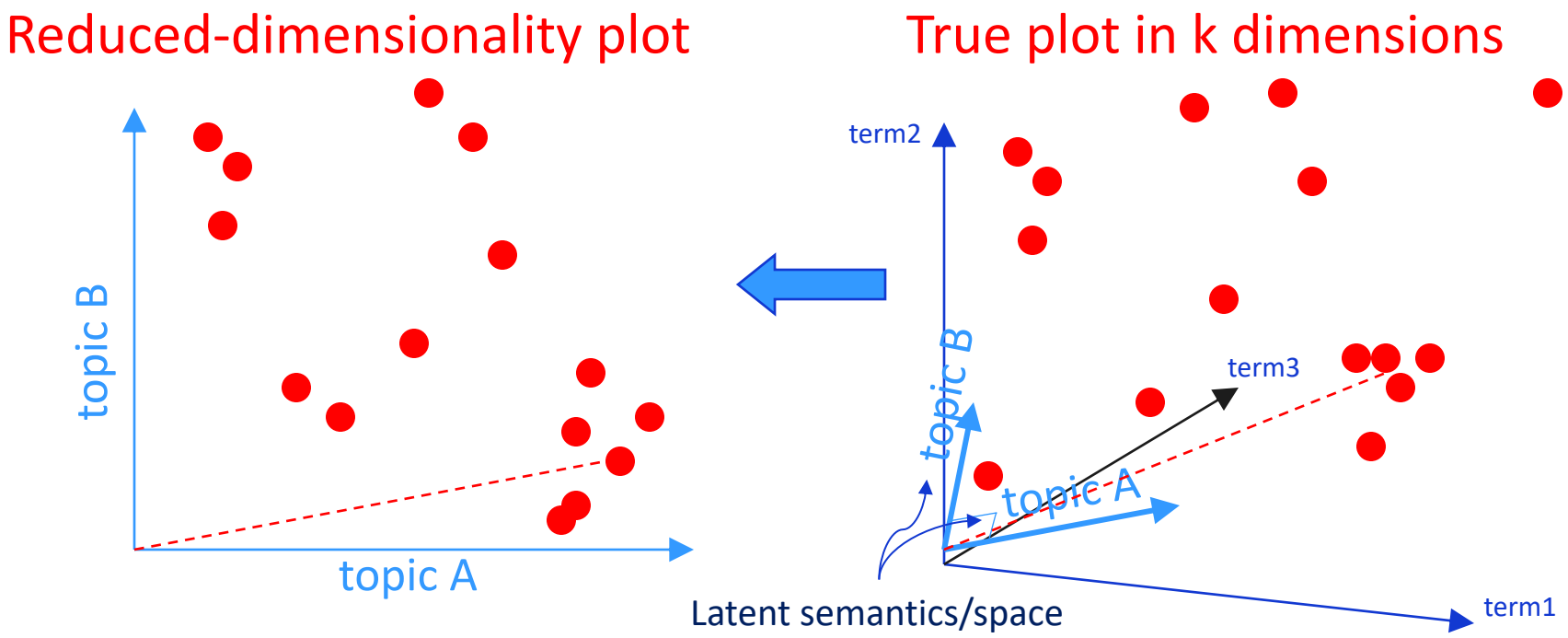
- **Latent semantic space**: illustrating example



- Based on a **low-rank approximation** of **document-term matrix** (typical rank **100-300**), we can compute document similarity based on the **inner product** in this **latent semantic space!** Also, any ML tasks can be formulated!

Latent Semantic Analysis

- So, what is the latent space?
- SVD finds a small number of topic vectors
- Approximates each doc as linear combination of topics
- Coordinates in reduced plot = linear coefficients
  - How much of topic A in this document? How much of topic B?
  - Each topic is a collection of words that tend to appear together



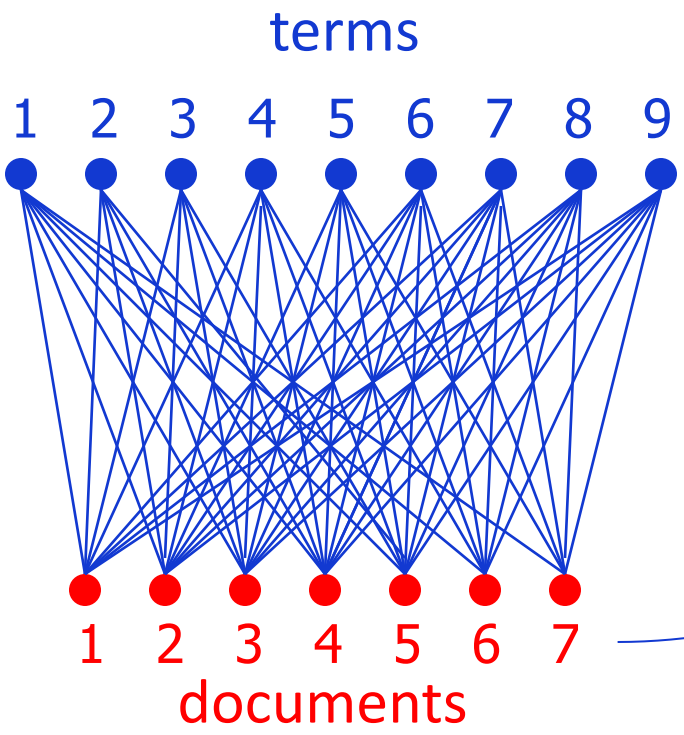
Latent Semantic Analysis

- Topics extracted for text analysis might help sense disambiguation

- Each word is like a tiny document: (0,0,0,1,0,0,…)
- Express word as a linear combination of topics
- Each topic corresponds to a "sense" of text data?
  - E.g., "Jordan" has Mideast and Sports topics
  - Word's sense in a document: which of its topics are strongest in the document?
- Groups sensing as well as splitting them
  - One word has several topics and many words have same topic

Latent Semantic Analysis

- A perspective on Principal Components Analysis (PCA)
- Imagine an electrical circuit that connects terms to docs …

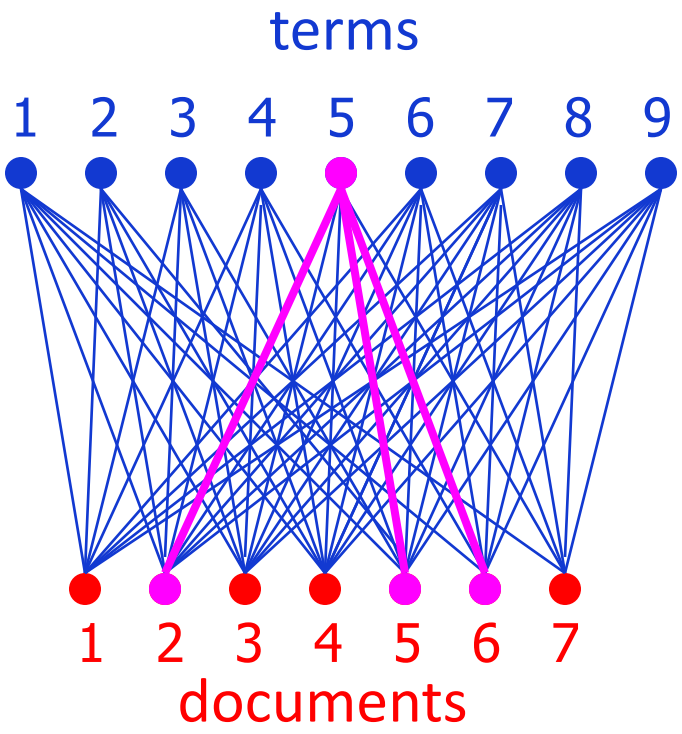


matrix of strengths
(how strong is each
term in each document?)

Each connection has a
weight given by the matrix.

Latent Semantic Analysis

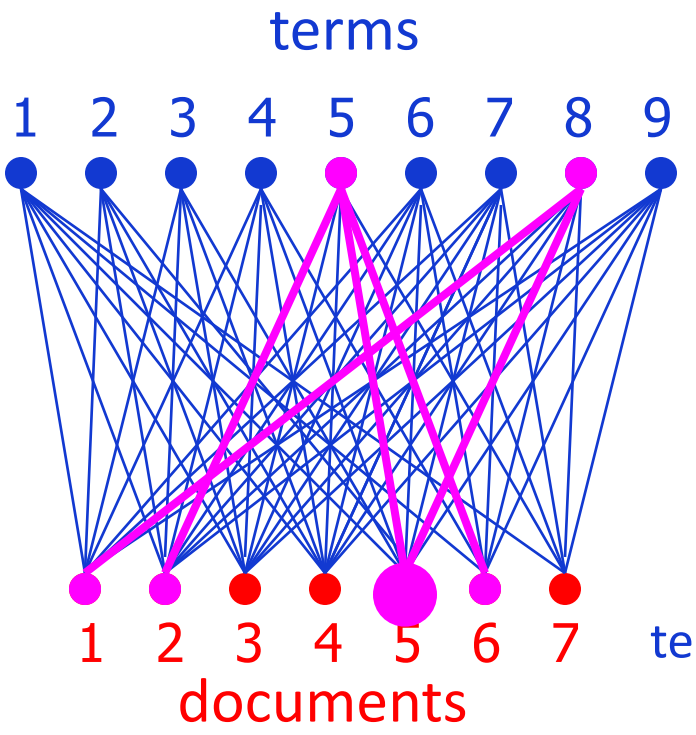
- Which documents is term 5 strong in?



docs 2, 5, 6
light up strongest.

Latent Semantic Analysis

- Which documents are terms 5 and 8 strong in?



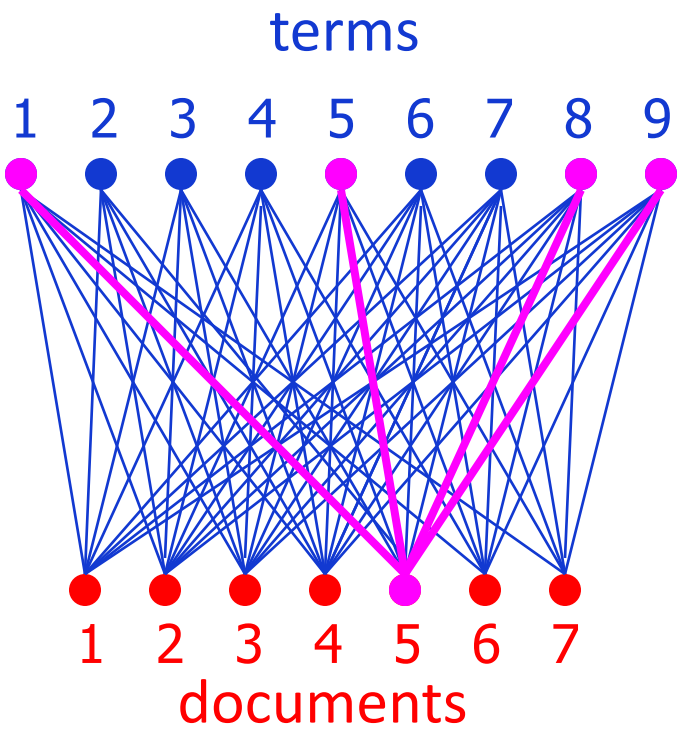
This answers a query
consisting of terms 5 and 8!

really just matrix multiplication:
term vector (query) x strength matrix = doc vector

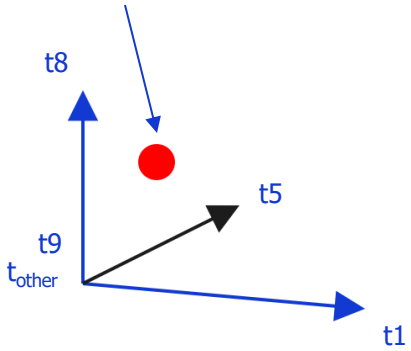
1x9 vector 9x7 matrix 1x7 vector

Latent Semantic Analysis

- Conversely, what terms are strong in document 5?

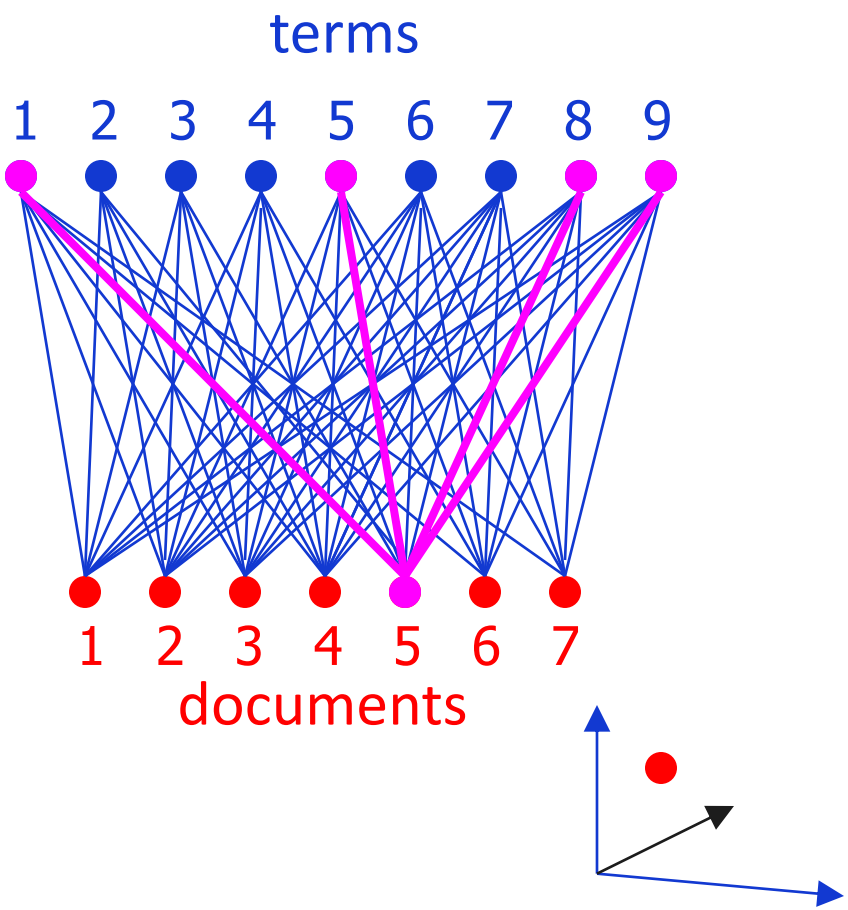


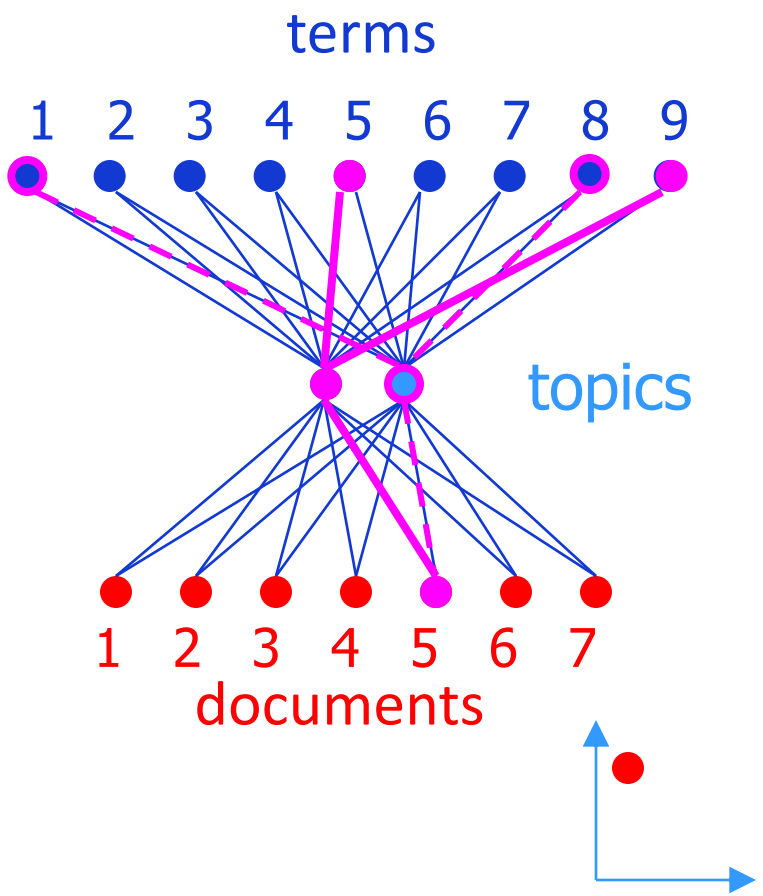
gives doc 5's coordinates!



Latent Semantic Analysis

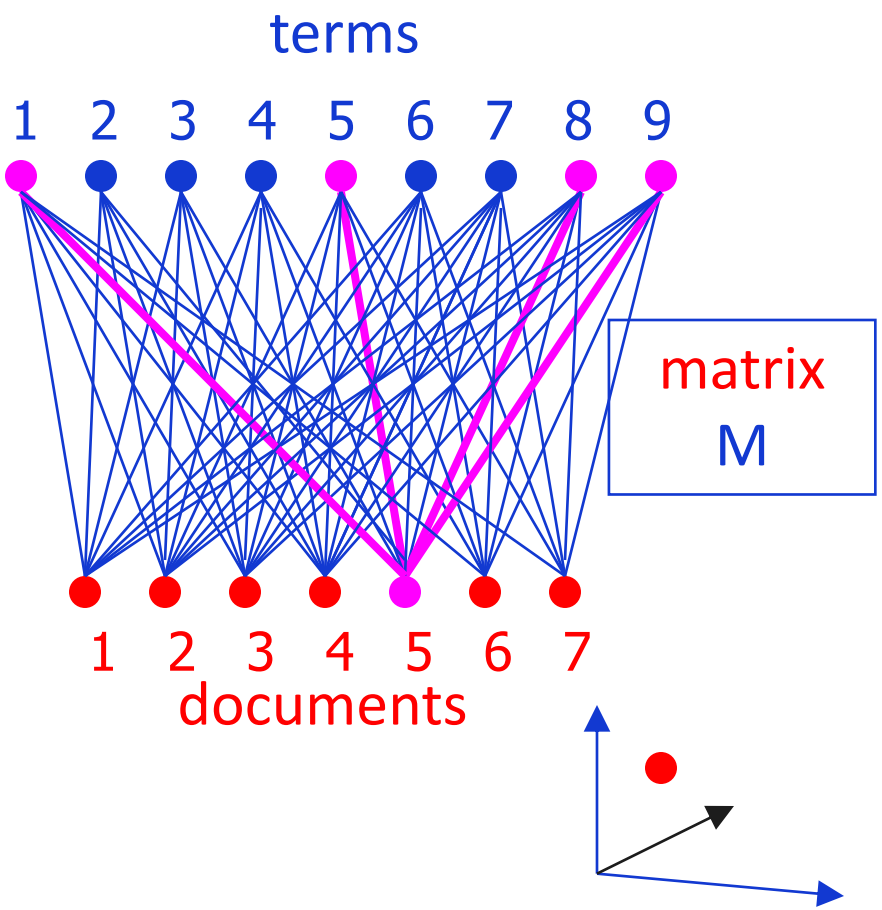
- SVD approximates by smaller 3-layer network
  - Forces sparse data through a bottleneck, smoothing it

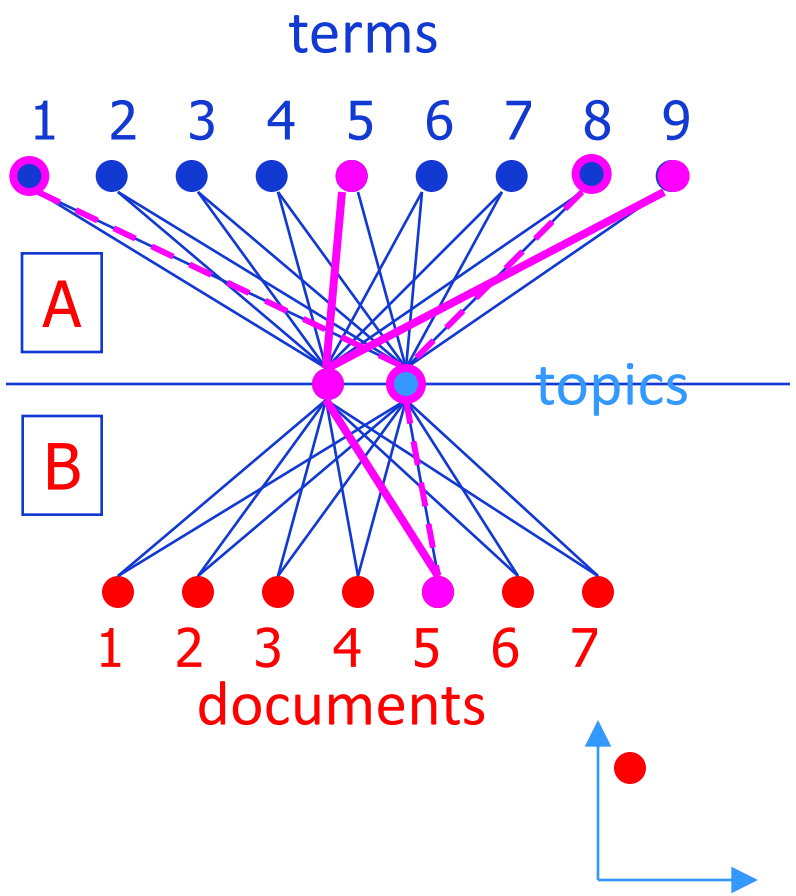




Latent Semantic Analysis

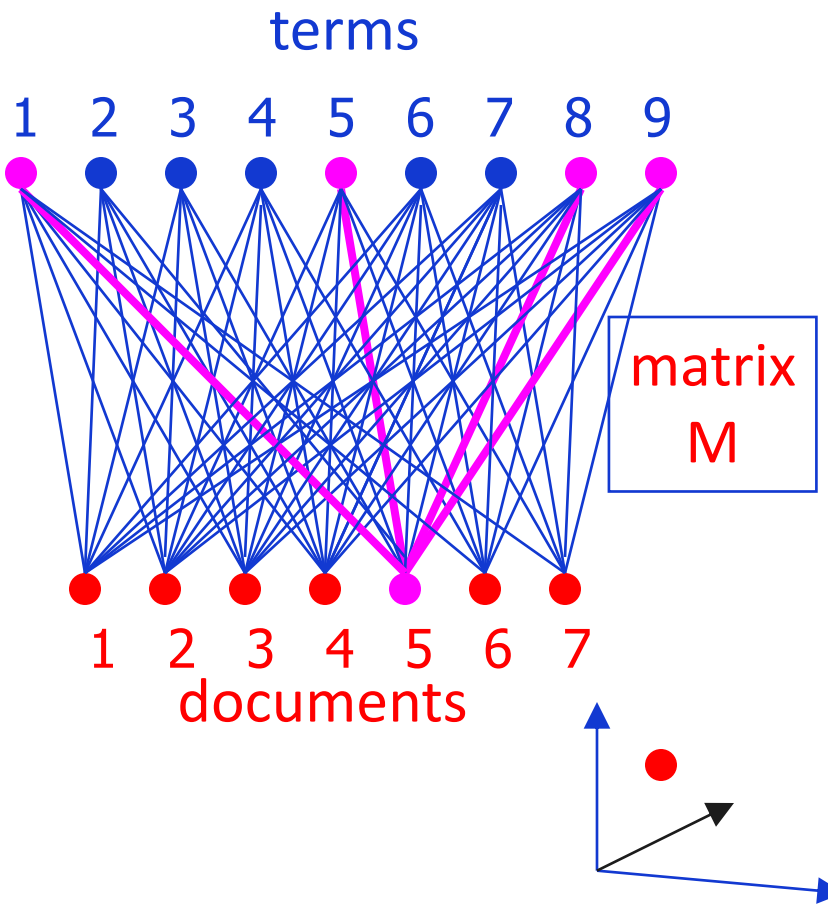
- smooth sparse data by matrix approximation: **M ≈ A B**
  - **A** encodes camera angle, **B** gives each doc's new coordinates

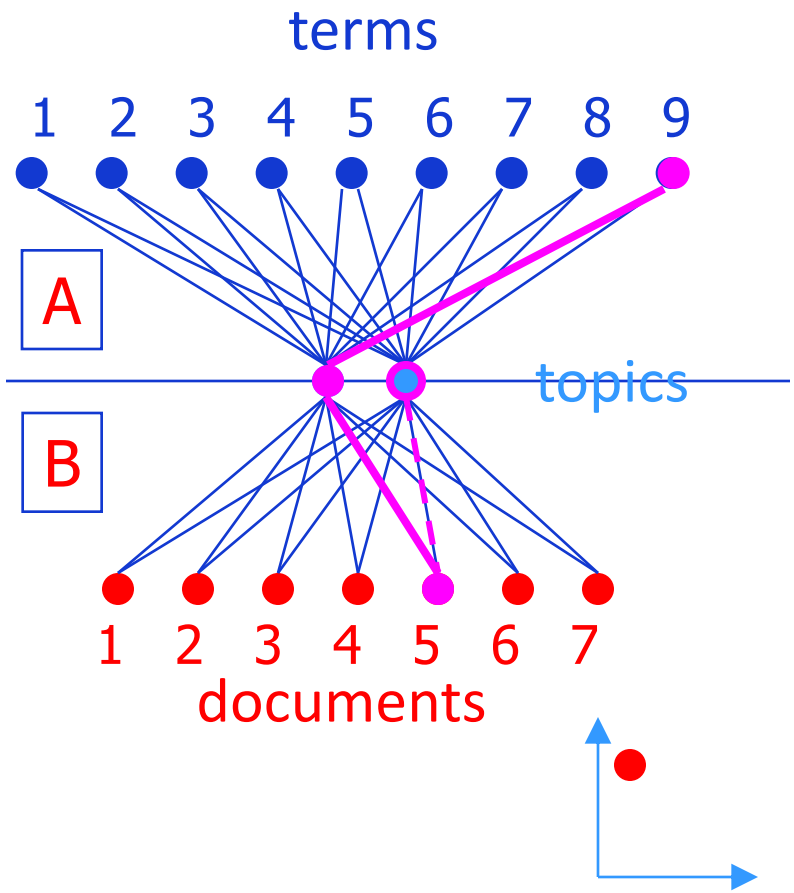




Latent Semantic Analysis

- Completely symmetric! Regard A, B as projecting terms and docs into a low-dimensional "topic space" where their similarity can be judged.



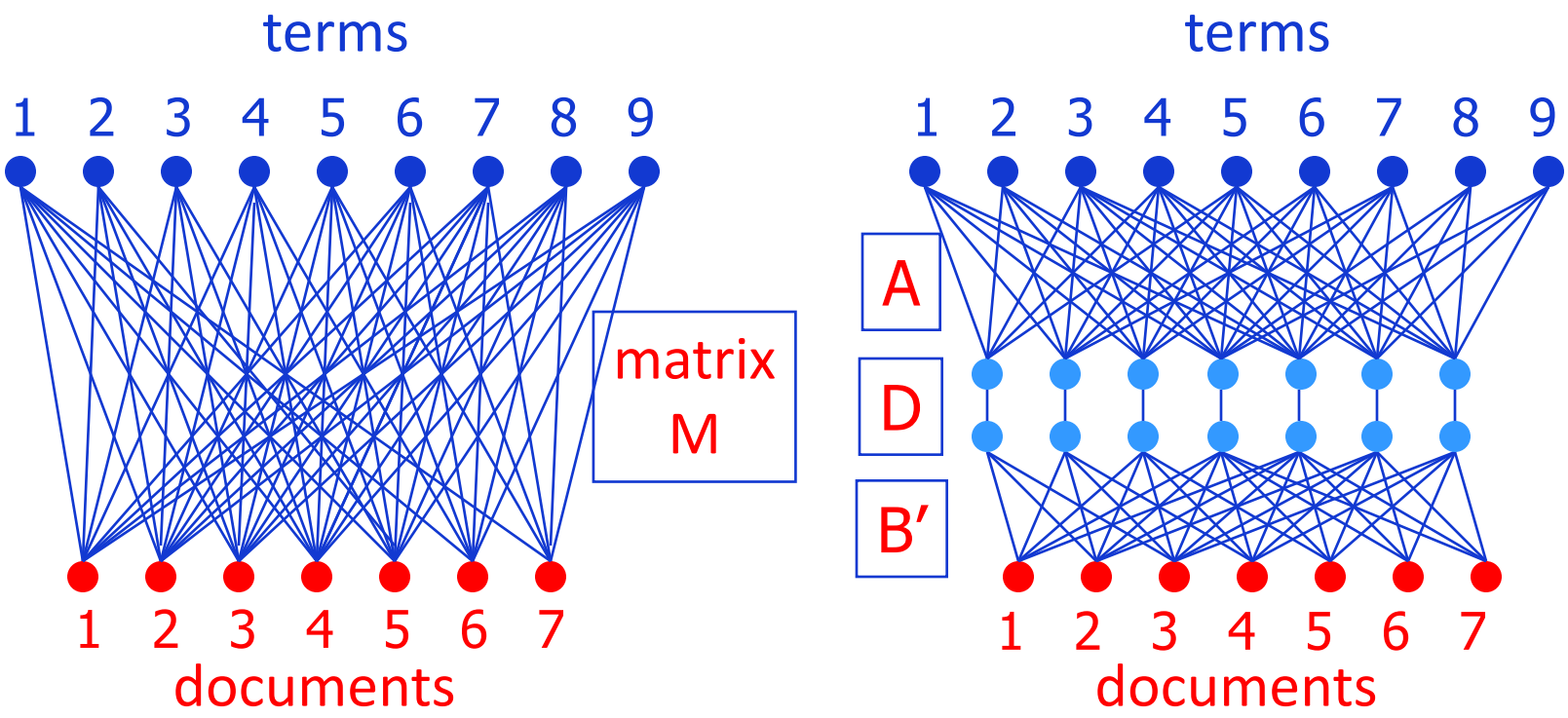


Latent Semantic Analysis

- Completely symmetric.
  - Regard A, B as projecting terms and docs into a low-dimensional "topic space" where their similarity can be judged.
- Cluster documents (helps sparsity problem!)
- Cluster words
- Compare a word with a doc
- Identify a word's topics with its senses
  - sense disambiguation by looking at document's senses
- Identify a document's topics with its topics
  - topic categorization

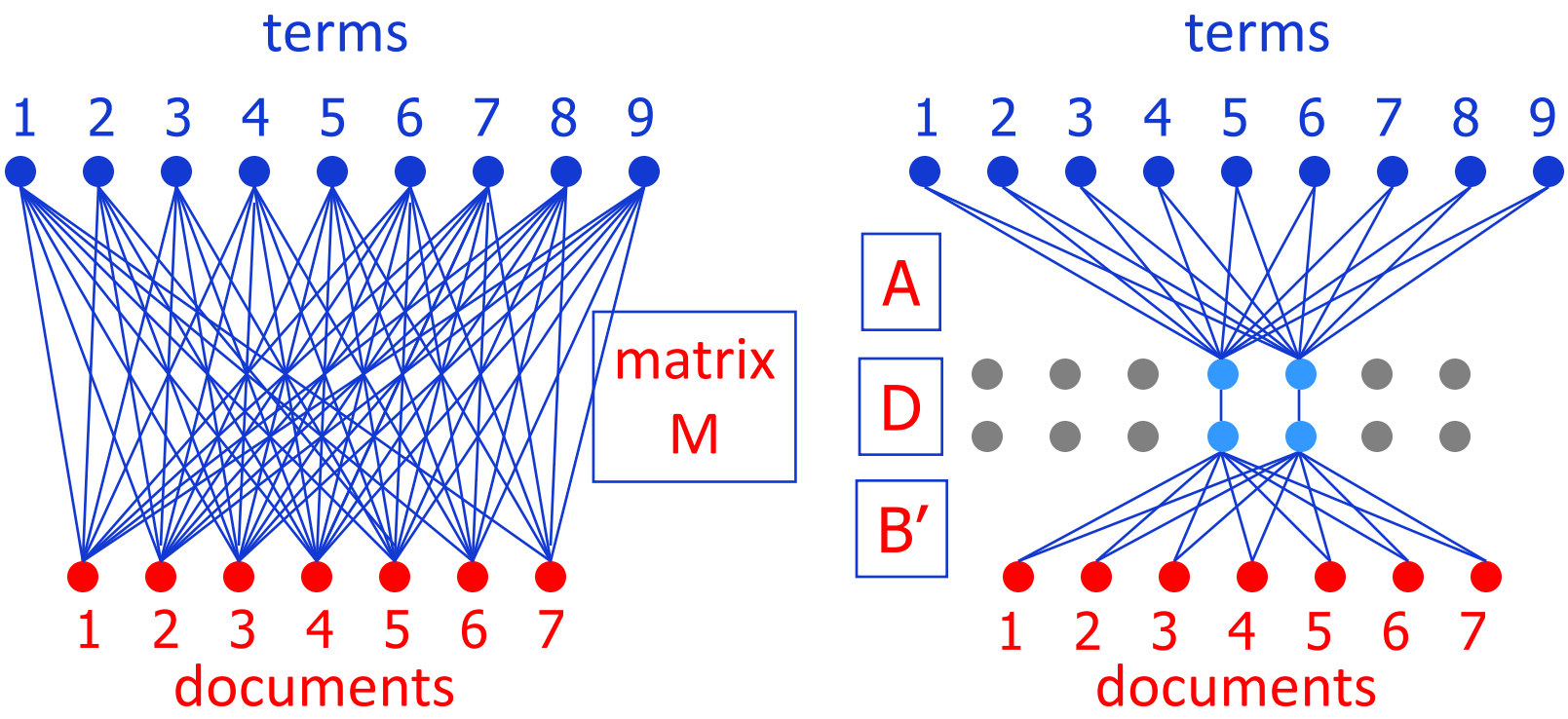
If you've seen SVD before …

- SVD actually decomposes **M = A D B' exactly**
  - A = camera angle (orthonormal); D diagonal; B' orthonormal



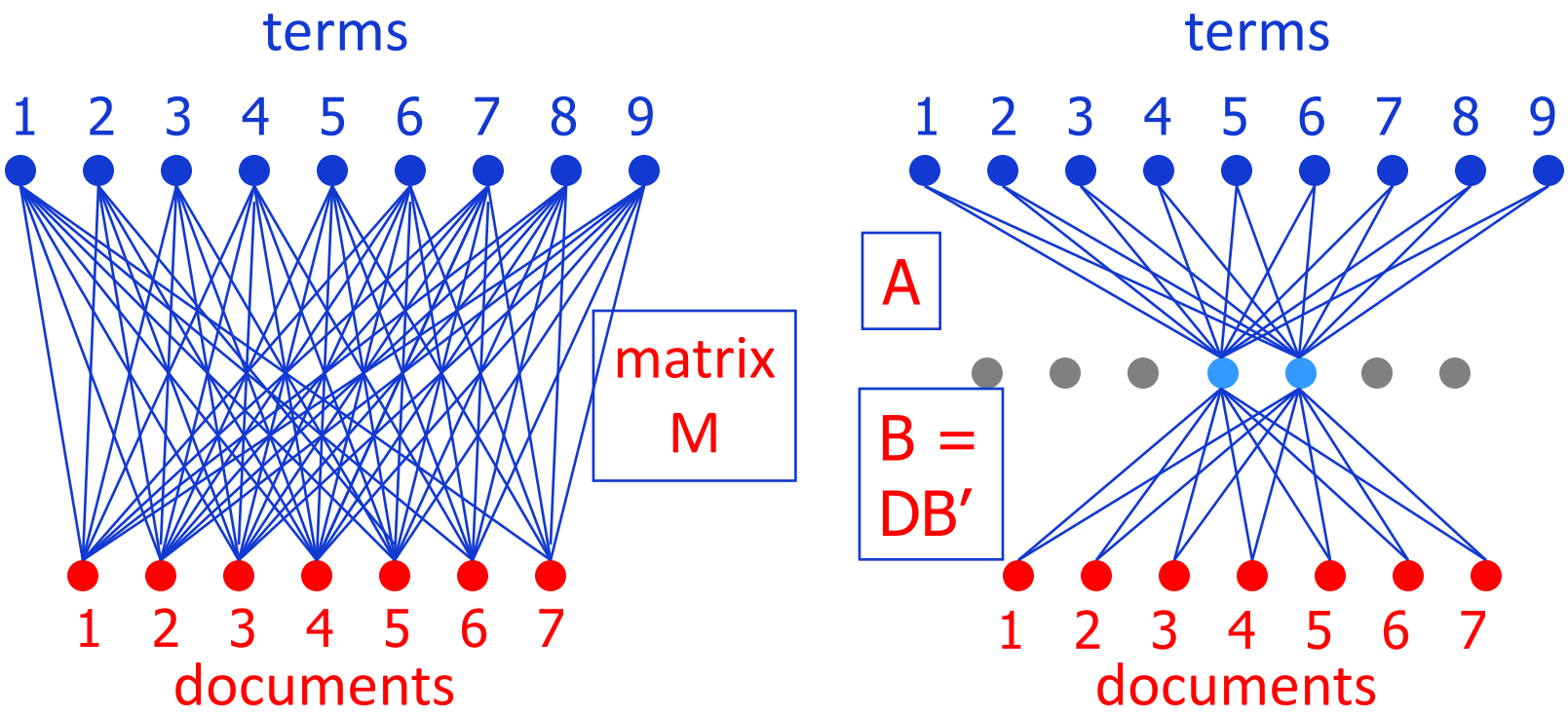
If you've seen SVD before …

- Keep only the largest j < k diagonal elements of D
  - This gives best possible approximation to M using only j light blue units



If you've seen SVD before …

- To simplify picture, can write **M** ≈ A (DB') = AB



- How should you pick j (number of blue units)?
- Just like picking number of clusters:
  - How well does system work with each j (on held-out data)?

Summary

- Text data also uses feature engineering!
- Fundamental problem is text data representation.
- Text similarity should be independent of document length.
- Document vector-space model is intuitive but could be big (in size).
- LSA is an effective document processing technique. It serves as an introductory discussion of dimensionality reduction. More importantly, it helps to determine the theme (topics) within text.
- LSA also serves as a bridge to deep learning for text data.

Acknowledgement

- Slides from Jiawei Han et al.'s Data Mining: Concepts and Techniques
- Slides from following sources:
  - Ani Nenkova, U of Pennsylvania
  - J.Y. Nie, U of Montreal
  - Richard Wicentowski, Swarthmore College