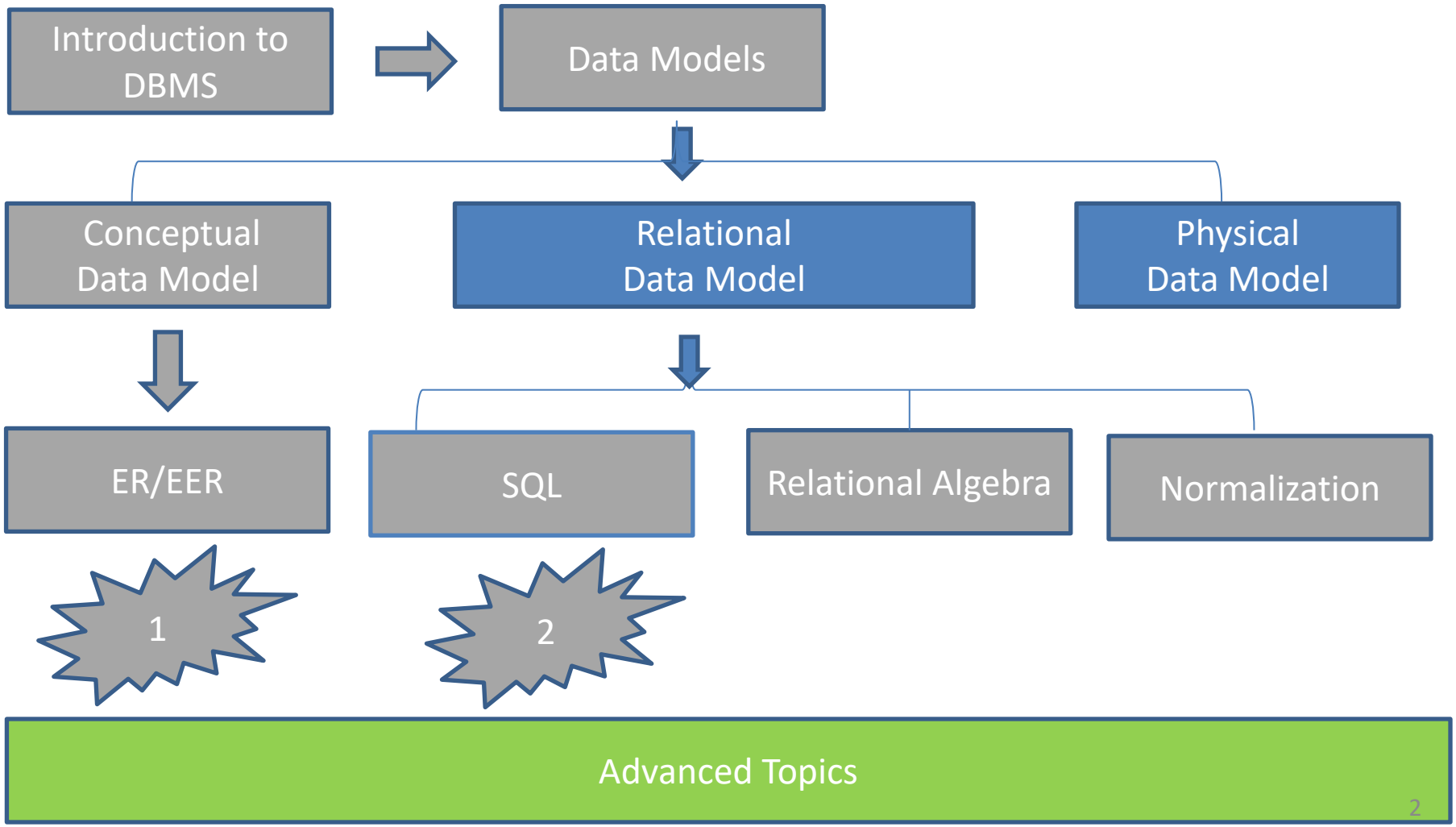
An Introduction to the Database Management Systems

By
Hossein Rahmani

Slides originally by Book(s) Resources

Road Map

(Might change!)



What is Text Mining?

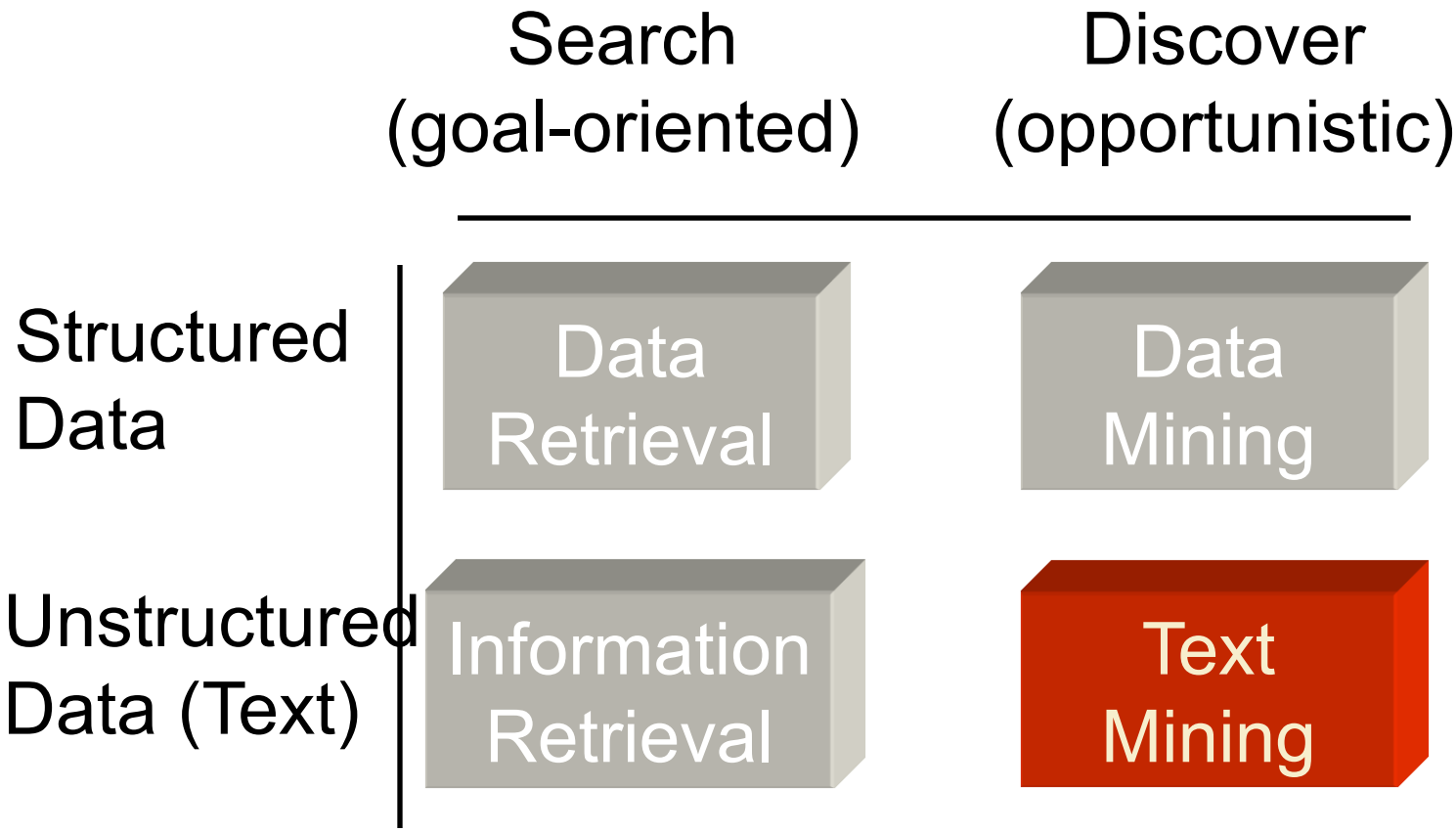
- There are many examples of text-based documents (all in 'electronic' format…)
  - e-mails, corporate Web pages, customer surveys, résumés, medical records, technical papers, incident reports, news stories and more…
- Not enough time or patience to read
  - Can we extract the most vital kernels of information…
- So, we wish to find a way to gain knowledge (in summarised form) from all that text, without reading or examining them fully first…!

What is Text Mining?

- Traditional data mining uses 'structured data' (n x p matrix)

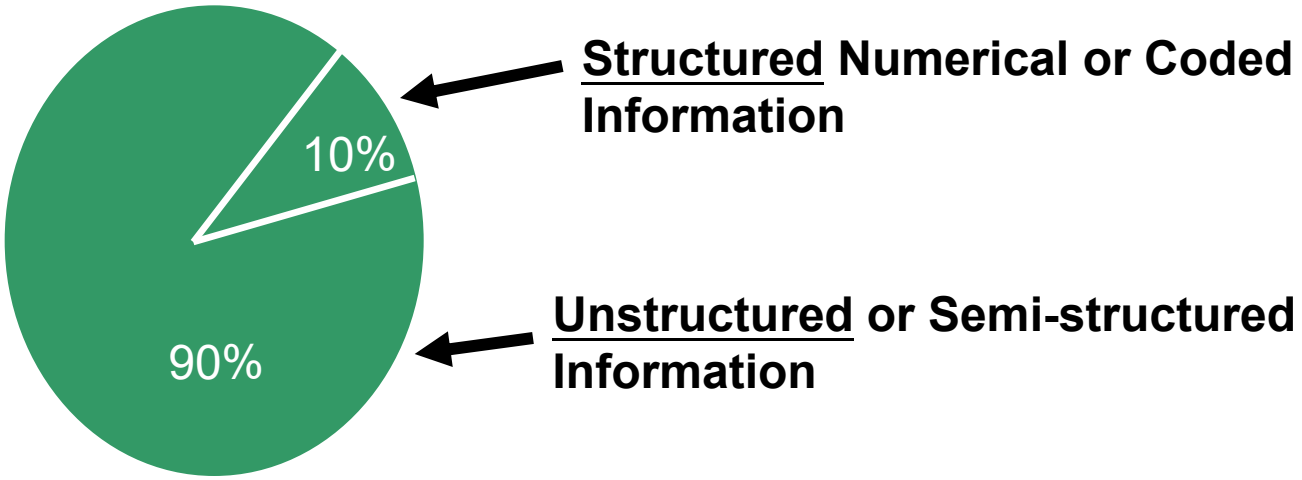
- The analysis of 'free-form text' is also referred to as 'unstructured data',
  - successful categorisation of such data can be a difficult and time-consuming task…

“Search” versus “Discover”



Motivation for Text Mining

- Approximately **90%** of the world's data is held in unstructured formats (source: Oracle Corporation)
- Information intensive business processes demand that we transcend from simple document retrieval to "knowledge" discovery.



Text Mining: Examples

- Text mining is an exercise to gain knowledge from stores of language text.
- Text:
  - Web pages
  - Medical records
  - Customer surveys
  - Email filtering (spam)
  - Incident reports
  - Drug interaction reports
  - News stories (e.g. predict stock movement)

Text Mining

- Typically falls into one of two categories
  - Analysis of text: I have a bunch of text I am interested in, tell me something about it
    - E.g. sentiment analysis, “buzz” searches
  - Retrieval: There is a large corpus of text documents, and I want the one closest to a specified query
    - E.g. web search, library catalogs, legal and medical precedent studies

Text Mining: Analysis

- Which words are most present
- Which words are most surprising
- Which words help *define* the document
- What are the interesting text phrases?

Text Mining: Retrieval

- Find k objects in the corpus of documents which are most similar to my query.
- Can be viewed as "interactive" data mining - query not specified a priori.
- Main problems of text retrieval:
  - What does "similar" mean?
  - How do I know if I have the right documents?
  - How can I incorporate user feedback?

Text Retrieval: Challenges

- Calculating similarity is not obvious - what is the distance between two sentences or queries?
- Evaluating retrieval is hard: what is the "right" answer ? (no ground truth)
- User can query things you have not seen before e.g. misspelled, foreign, new terms.
- Goal (score function) is different than in classification/regression: not looking to model all of the data, just get best results for a given user.
- Words can hide semantic content
  - Synonymy: A keyword *T* does not appear anywhere in the document, even though the document is closely related to *T*, e.g., data mining
  - Polysemy: The same keyword may mean different things in different contexts, e.g., mining

Term / document matrix

- Most common form of representation in text mining is the *term - document* matrix
  - Term: typically a single word, but could be a word phrase like “data mining”
  - Document: a generic term meaning a collection of text to be retrieved
  - Can be large - terms are often 50k or larger, documents can be in the billions (www).

Term document matrix

Example: 10 documents: 6 terms

	Database	SQL	Index	Regression	Likelihood	linear
D1	24	21	9	0	0	3
D2	32	10	5	0	3	0
D3	12	16	5	0	0	0
D4	6	7	2	0	0	0
D5	43	31	20	0	3	0
D6	2	0	0	18	7	6
D7	0	0	1	32	12	0
D8	3	0	0	22	4	4
D9	1	0	0	34	27	25
D10	6	0	0	17	4	23

$$D_1 = (d_{i1}, d_{i2}, \ldots, d_{it})$$

- Each document now is just a vector of terms, sometimes boolean

Term document matrix

- We have lost all semantic content

- Be careful constructing your term list!
  - Not all words are created equal!
  - Words that are the same should be treated the same!

- Stop Words
- Stemming

Stop words

- Many of the most frequently used words in English are worthless in retrieval and text mining – these words are called *stop words*.
  - the, of, and, to, ….
  - Typically about 400 to 500 such words
  - For an application, an additional domain specific stop words list may be constructed
- Why do we need to remove stop words?
  - Reduce indexing (or data) file size
    - stopwords accounts 20-30% of total word counts.
  - Improve efficiency
    - stop words are not useful for searching or text mining
    - stop words always have a large number of hits

Stemming

- Techniques used to find out the root/stem of a word:
  - E.g.,
    - user
    - users
    - used
    - using

    engineering
    engineered
    engineer
- stem: use engineer

Usefulness

- improving effectiveness of retrieval and text mining
  - matching similar words
- reducing indexing size
  - combing words with same roots may reduce indexing size as much as 40-50%.

Basic stemming methods

- remove ending
  - if a word ends with a consonant other than s,
    followed by an s, then delete s.
  - if a word ends in es, drop the s.
  - if a word ends in ing, delete the ing unless the remaining word consists only of one letter or of th.
  - If a word ends with ed, preceded by a consonant, delete the ed unless this leaves only a single letter.
  - ……
- transform words
  - if a word ends with “ies” but not “eies” or “aies” then “ies --> y.”

Distances in TD matrices

- Given a term doc matrix represetnation, now we can define distances between documents (or terms!)
- Elements of matrix can be 0,1 or term frequencies (sometimes normalized)
- Can use Euclidean or cosine distance
- Cosine distance is the angle between the two vectors
- Not intuitive, but has been proven to work well

$$d_c(D_i, D_j) = \frac{\sum_{k=1}^{T} d_{ik} d_{jk}}{\sqrt{\sum_{k=1}^{T} d_{ik}^2 \sum_{k=1}^{T} d_{jk}^2}}$$

- If docs are the same, d_c =1, if nothing in common d_c=0

Information retrieval models

- An IR model governs how a document and a query are represented and how the relevance of a document to a user query is defined.
- Main models:
  - Boolean model
  - Vector space model
  - Statistical language model
  - etc

Boolean model

- Each document or query is treated as a **"bag" of words** or **terms**. Word sequence is not considered.
- Given a collection of documents D, let $V = \{t_1, t_2, ..., t_{|V|}\}$ be the set of distinctive words/terms in the collection. V is called the vocabulary.
- A weight $w_{ij} > 0$ is associated with each term t_i of a document $\mathbf{d}_j \in D$. For a term that does not appear in document $\mathbf{d}_j$, $w_{ij} = 0$.

$$\mathbf{d}_j = (w_{1j}, w_{2j}, ..., w_{|V|j}),$$

Boolean model (contd)

- Query terms are combined logically using the Boolean operators **AND**, **OR**, and **NOT.**
  - E.g., ((*data* AND *mining*) AND (NOT *text*))
- Retrieval
  - Given a Boolean query, the system retrieves every document that makes the query logically true.
  - Called **exact match**.
- The retrieval results are usually quite poor because term frequency is not considered.

Vector space model

- Documents are also treated as a "bag" of words or terms.
- Each document is represented as a vector.
- However, the term weights are no longer 0 or 1. Each term weight is computed based on some variations of TF or TF-IDF scheme.

- Term Frequency (TF) Scheme: The weight of a term t_i in document $\mathbf{d}_j$ is the number of times that t_i appears in $\mathbf{d}_j$, denoted by f_{ij}. Normalization may also be applied.

TF-IDF term weighting scheme

- The most well known weighting scheme
  - TF: still **term frequency**
  - IDF: **inverse document frequency**.

  N: total number of docs

  df_i: the number of docs that t_i appears.
- The final TF-IDF term weight is:

$$tf_{ij} = \frac{f_{ij}}{\max\{f_{1j}, f_{2j}, \ldots, f_{|V|j}\}}$$

$$idf_i = \log \frac{N}{df_i}$$

$$w_{ij} = tf_{ij} \times idf_i.$$

Inverse Document Frequency

1. *Inverse document frequency* of a term t:

$$idf_t = log \frac{N}{df_t} \qquad \text{with } N = \text{collection size}$$

2. Rare terms have high *idf*, contrary to frequent terms
3. Example (Reuters collection):

Term t	df_t	idf_t
car	18165	1.65
auto	6723	2.08
insurance	19241	1.62
best	25235	1.5

4. In tf-idf weighting, the weight of a term is computed using both *tf* and *idf*:

$$w(t,d) = tf_{t,d} \times idf_t \qquad \text{called } tf - idf_{t,d}$$

Binary term-document incidence matrix

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calpurnia	0	1	0	0	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0

Each document is represented by a binary vector $\in \{0,1\}^{|V|}$

Term-document count matrices

- Consider the number of occurrences of a term in a document:
  - Each document is a count vector in $\mathbb{N}^{v}$: a column below

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	157	73	0	0	0	0
Brutus	4	157	0	1	0	0
Caesar	232	227	0	2	1	1
Calpurnia	0	10	0	0	0	0
Cleopatra	57	0	0	0	0	0
mercy	2	0	3	5	5	1
worser	2	0	1	1	1	0

Binary → count → weight matrix

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	5.25	3.18	0	0	0	0.35
Brutus	1.21	6.1	0	1	0	0
Caesar	8.59	2.54	0	1.51	0.25	0
Calpurnia	0	1.54	0	0	0	0
Cleopatra	2.85	0	0	0	0	0
mercy	1.51	0	1.9	0.12	5.25	0.88
worser	1.37	0	0.11	4.15	0.25	1.95

Each document is now represented by a real-valued vector of tf-idf weights $\in \mathbb{R}^{|V|}$

Retrieval in vector space model

- Query **q** is represented in the same way or slightly differently.
- Relevance of $\mathbf{d}_j$ to **q**: Compare the similarity of query **q** and document $\mathbf{d}_j$.
- Cosine similarity (the cosine of the angle between the two vectors)

$$cosine(\mathbf{d}_j, \mathbf{q}) = \frac{\langle \mathbf{d}_j \bullet \mathbf{q} \rangle}{\| \mathbf{d}_j \| \times \| \mathbf{q} \|} = \frac{\sum_{i=1}^{|V|} w_{ij} \times w_{iq}}{\sqrt{\sum_{i=1}^{|V|} w_{ij}^2} \times \sqrt{\sum_{i=1}^{|V|} w_{iq}^2}}$$

- Cosine is also commonly used in text clustering

Latent Semantic Indexing

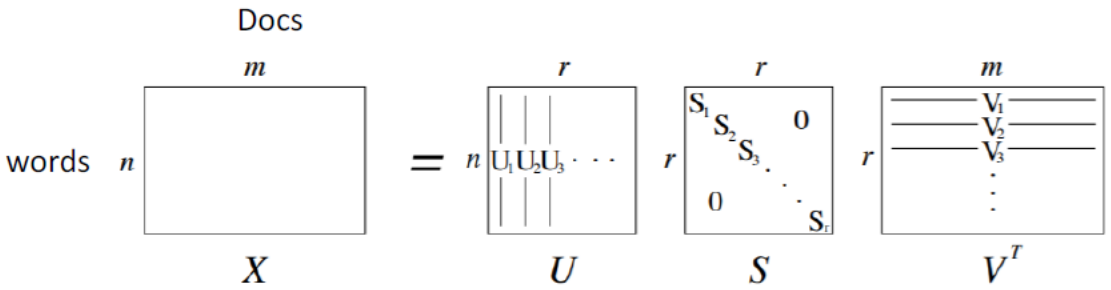
- Criticism: queries can be posed in many ways, but still mean the same
  - Data mining and knowledge discovery
  - Car and automobile
- *Semantically*, these are the same, and documents with either term are relevant.
- Using synonym lists or thesauri are solutions, but messy and difficult.
- Latent Semantic Indexing (LSI): tries to extract hidden semantic structure in the documents
- Search what I meant, not what I said!

LSI

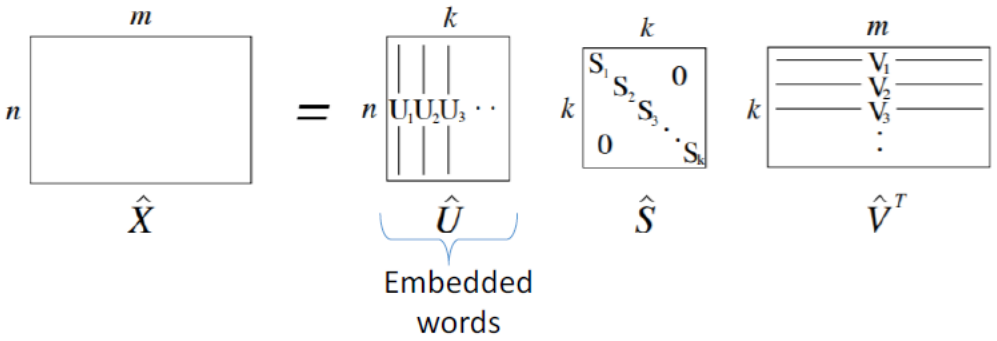
- Approximate the T-dimensional term space using principle components calculated from the TD matrix
- The first k PC directions provide the best set of k orthogonal basis vectors - these explain the most variance in the data.
  - Data is reduced to an N x k matrix, without much loss of information
- Each “direction” is a linear combination of the input terms, and define a clustering of “topics” in the data.

LSA: Dimensionality Reduction based on word-doc mat

Singular Value Decomposition of cooccurrence matrix *X*.



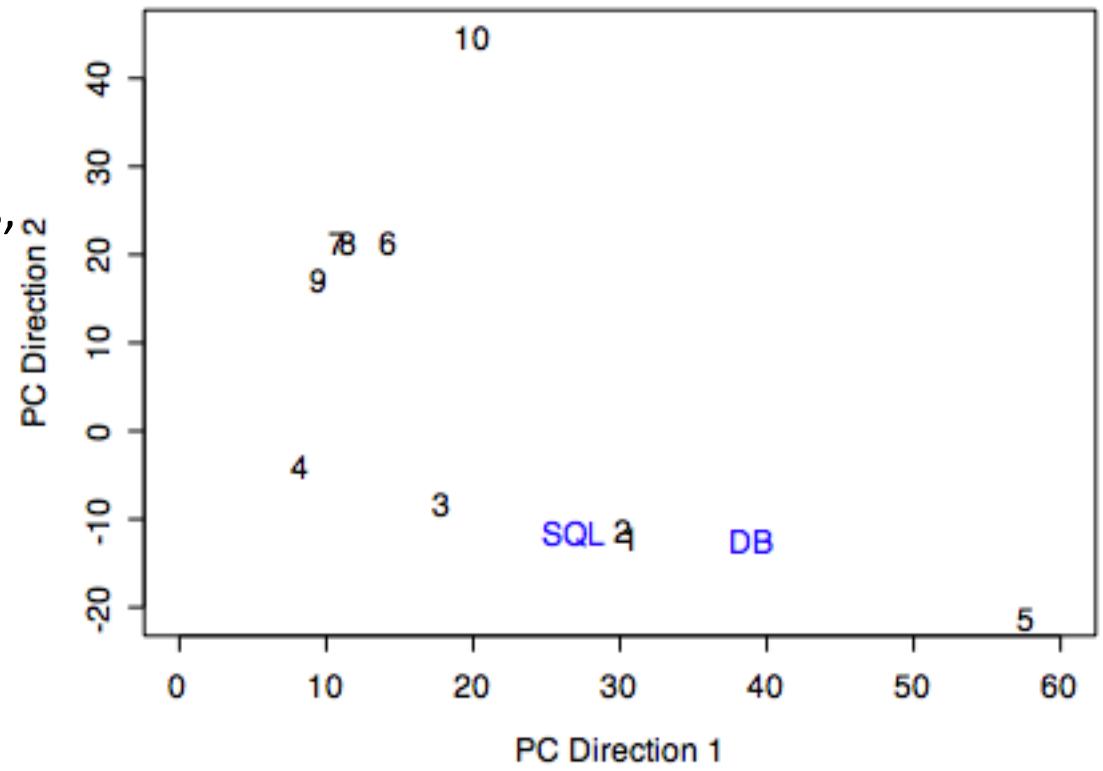
Maintaining only the k largest singular values of X



$\hat{X}$ is the best rank *k* approximation to *X* , in terms of least squares.

LSI

- Here we show the same plot, but with two new documents, one with the term "SQL" 50 times, another with the term "Databases" 50 times.
- Even though they have no phrases in common, they are close in LSI space



Textual analysis

- Once we have the data into a nice matrix representation (TD, TDxIDF, or LSI), we can throw the data mining toolbox at it:
  - Classification of documents
    - If we have training data for classes
  - Clustering of documents
    - unsupervised

Automatic document classification

- Motivation
  - Automatic classification for the tremendous number of on-line text documents (Web pages, e-mails, etc.)
  - Customer comments: Requests for info, complaints, inquiries
- A classification problem
  - Training set: Human experts generate a training data set
  - Classification: The computer system discovers the classification rules
  - Application: The discovered rules can be applied to classify new/unknown documents
- Techniques
  - Linear/logistic regression, naïve Bayes
  - Trees not so good here due to massive dimension, few interactions

Document Clustering

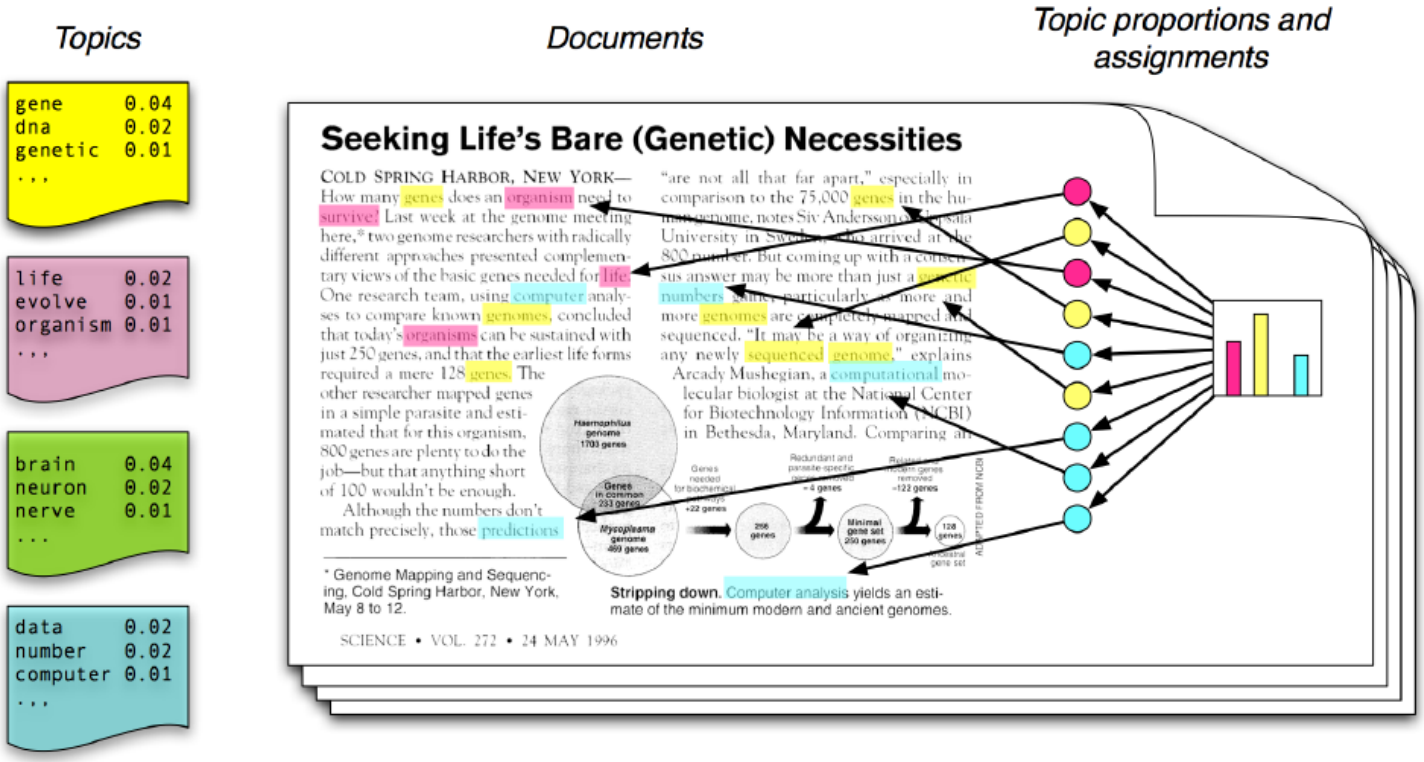
- To Cluster:
  - Latent Dirichlet Allocation (LDA)
    - LDA is a generative probabilistic model of a corpus. Documents are represented as random mixtures over latent topics, where a topic is characterized by a distribution over words.
- LDA:
  - Three concepts: words, topics, and documents
  - Documents are a collection of words and have a probability distribution over topics
  - Topics have a probability distribution over words
  - Fully Bayesian Model

- The result can be an often-useful classification of documents into topics, and a distribution of each topic across words:

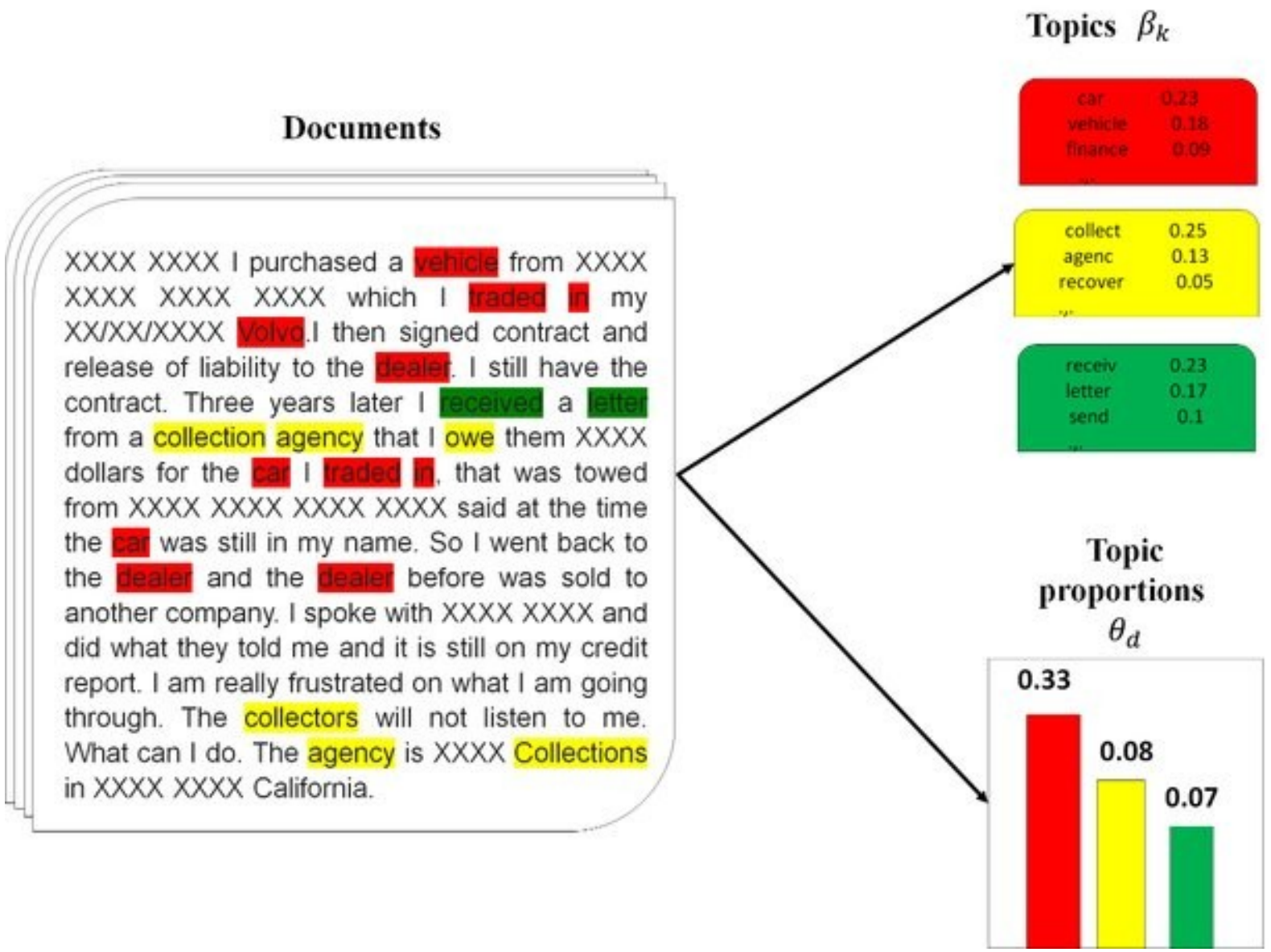
"Arts"	"Budgets"	"Children"	"Education"
NEW	MILLION	CHILDREN	SCHOOL
FILM	TAX	WOMEN	STUDENTS
SHOW	PROGRAM	PEOPLE	SCHOOLS
MUSIC	BUDGET	CHILD	EDUCATION
MOVIE	BILLION	YEARS	TEACHERS
PLAY	FEDERAL	FAMILIES	HIGH
MUSICAL	YEAR	WORK	PUBLIC
BEST	SPENDING	PARENTS	TEACHER
ACTOR	NEW	SAYS	BENNETT
FIRST	STATE	FAMILY	MANIGAT
YORK	PLAN	WELFARE	NAMPHY
OPERA	MONEY	MEN	STATE
THEATER	PROGRAMS	PERCENT	PRESIDENT
ACTRESS	GOVERNMENT	CARE	ELEMENTARY
LOVE	CONGRESS	LIFE	HAITI

The William Randolph Hearst Foundation will give $1.25 million to Lincoln Center, Metropolitan Opera Co., New York Philharmonic and Juilliard School. "Our board felt that we had a real opportunity to make a mark on the future of the performing arts with these grants an act every bit as important as our traditional areas of support in health, medical research, education and the social services," Hearst Foundation President Randolph A. Hearst said Monday in announcing the grants. Lincoln Center's share will be $200,000 for its new building, which will house young artists and provide new public facilities. The Metropolitan Opera Co. and New York Philharmonic will receive $400,000 each. The Juilliard School, where music and the performing arts are taught, will get $250,000. The Hearst Foundation, a leading supporter of the Lincoln Center Consolidated Corporate Fund, will make its usual annual $100,000 donation, too.

Topic Model: Visual Illustration

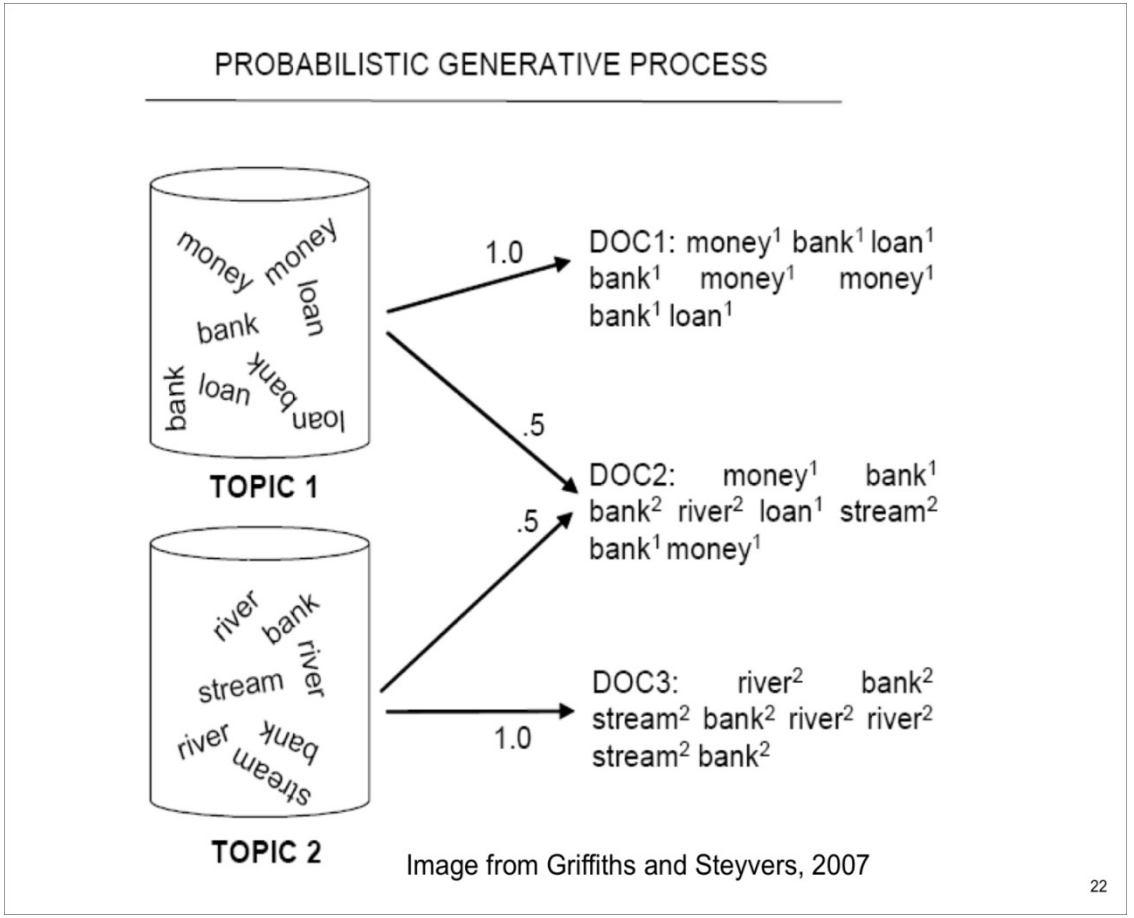


- Each **topic** is a distribution over words
- Each **document** is a mixture of corpus-wide topics
- Each **word** is drawn from one of those topics


Documents
XXXX XXXX I purchased a vehicle from XXXX XXXX XXXX XXXX which I traded in my XX/XX/XXXX Volvo.I then signed contract and release of liability to the dealer. I still have the contract. Three years later I received a letter from a collection agency that I owe them XXXX dollars for the car I traded in, that was towed from XXXX XXXX XXXX XXXX said at the time the car was still in my name. So I went back to the dealer and the dealer before was sold to another company. I spoke with XXXX XXXX and did what they told me and it is still on my credit report. I am really frustrated on what I am going through. The collectors will not listen to me. What can I do. The agency is XXXX Collections in XXXX XXXX California.
Topics β_k
car 0.23
vehicle 0.18
finance 0.09
collect 0.25
agenc 0.13
recover 0.05
receiv 0.23
letter 0.17
send 0.1
Topic proportions θ_d
0.33
0.08
0.07

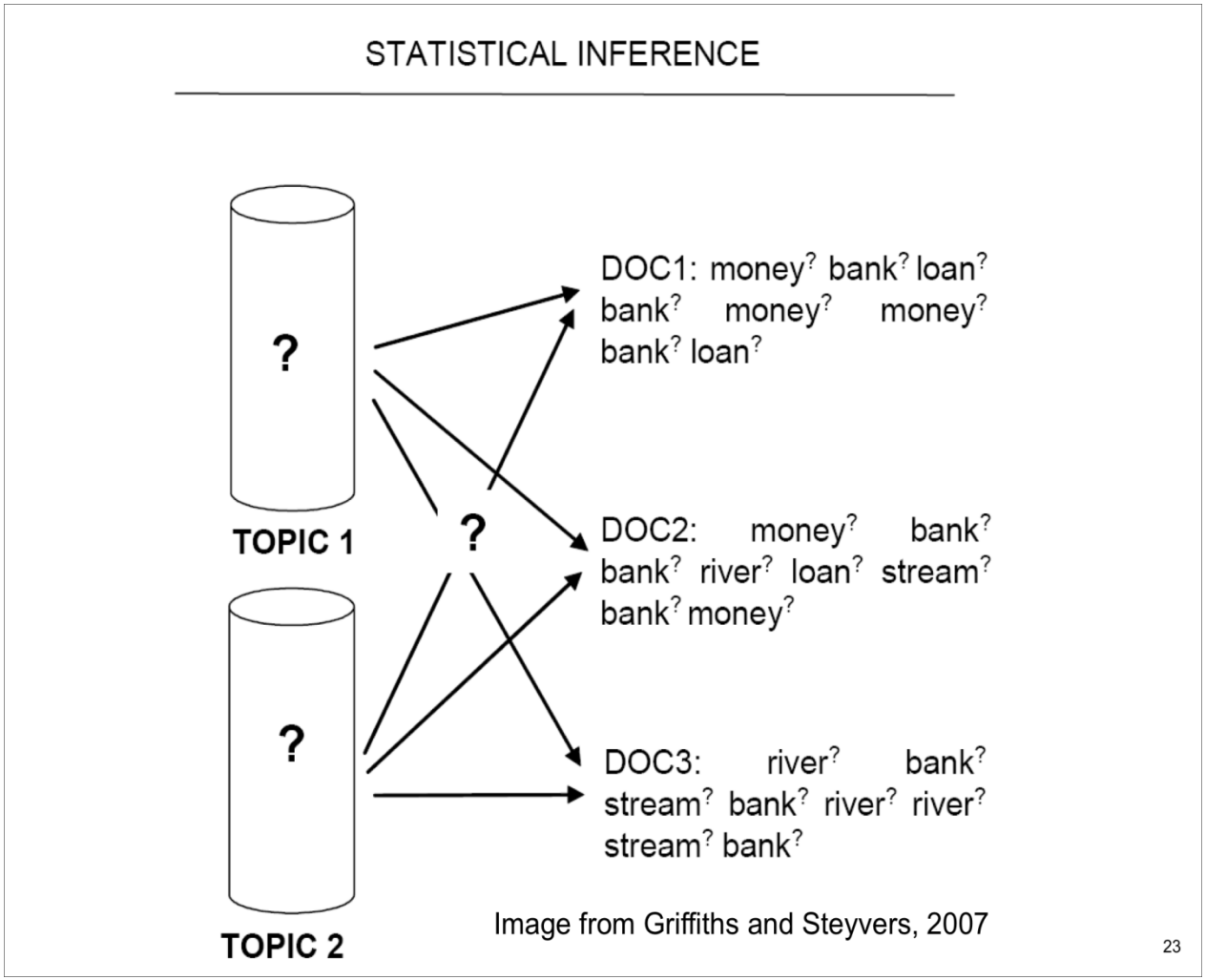
Another Look at LDA

- Model: Topics made up of words used to generate documents



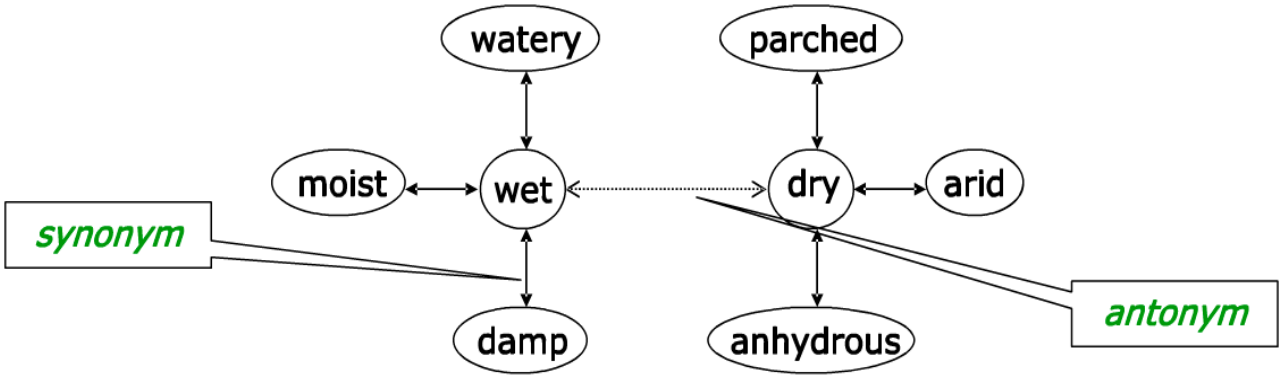
Another Look at LDA

- Reality: Documents observed, infer topics



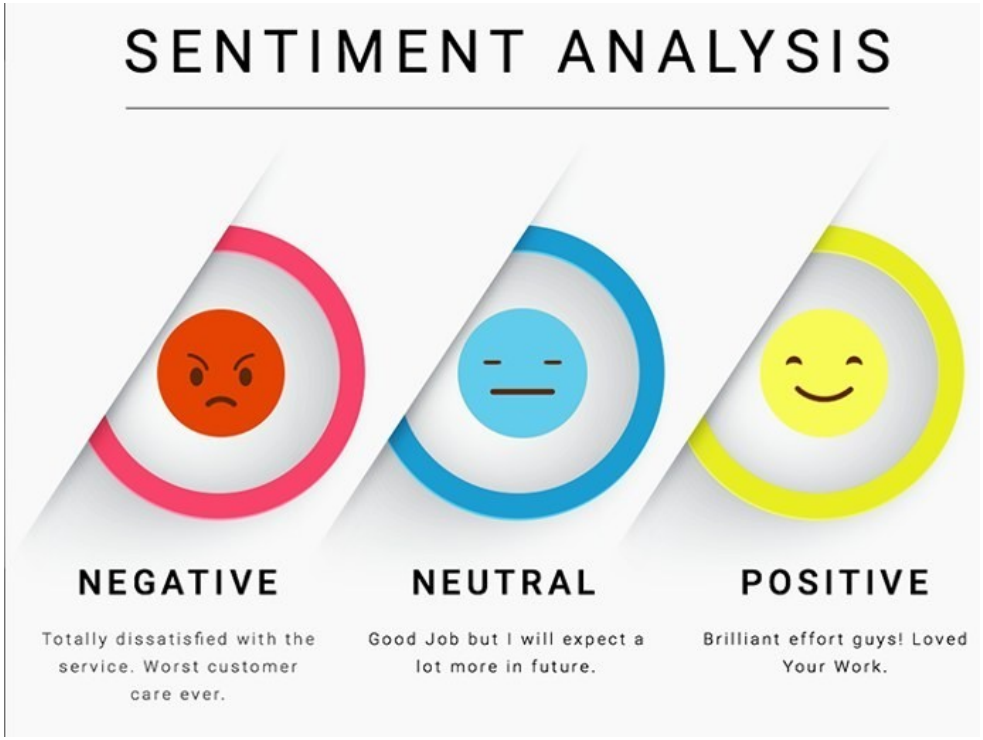
Text Mining: Helpful Data

- WordNet
  - An extensive lexical network for the English language
  - Contains over 138,838 words.
  - Several graphs, one for each part-of-speech.
  - Synsets (synonym sets), each defining a semantic sense.
  - Relationship information (antonym, hyponym, meronym …)
  - Downloadable for free (UNIX, Windows)
  - Expanding to other languages (Global WordNet Association)
  - Funded >$3 million, mainly government (translation interest)
  - Founder George Miller, National Medal of Science, 1991.



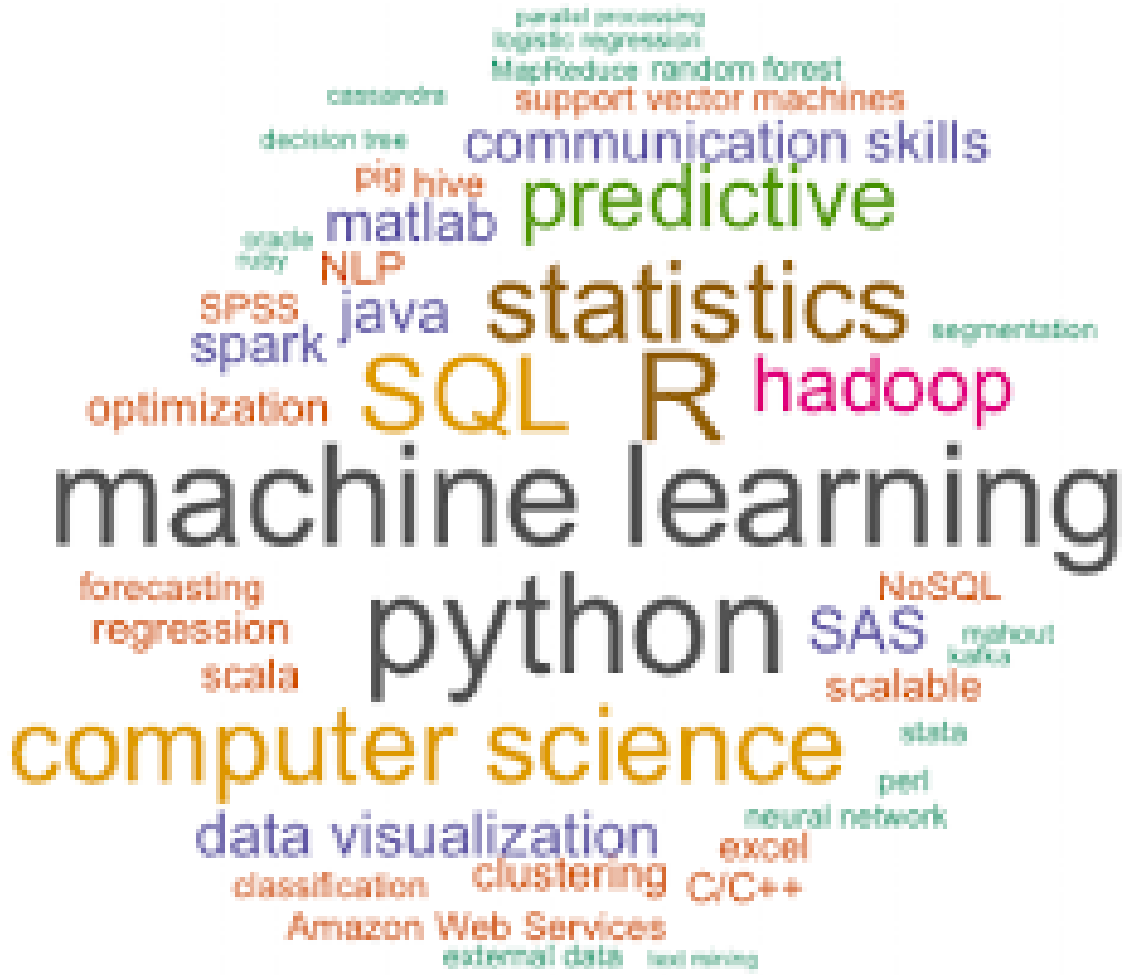
Text Mining - Other Topics

- Sentiment Analysis
  - Automatically determine tone in text: positive, negative or neutral
  - Typically uses collections of good and bad words
  - Often fit using Naïve Bayes



Text Mining - Other Topics

- Summarizing text: Word Clouds
  - Takes text as input, finds the most interesting ones, and displays them graphically


parallel processing
logistic regression
MapReduce
random forest
cassandra
support vector machines
decision tree
communication skills
pig
hive
predictive
matlab
oracle
ruby
NLP
SPSS
java
statistics
segmentation
spark
optimization
SQL
R
hadoop
machine learning
forecasting
NoSQL
regression
python
SAS
mahout
kafka
scala
scalable
computer science
stata
perl
neural network
data visualization
excel
clustering
C/C++
classification
Amazon Web Services
external data
text mining

Word Embedding

One-hot coding

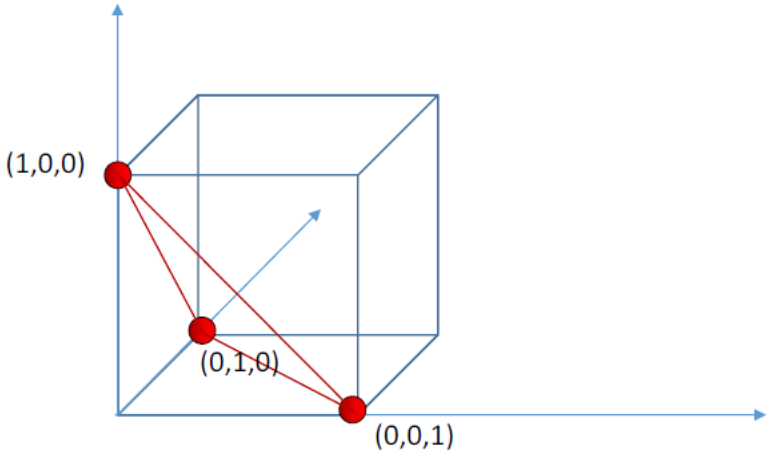
In vector space terms, this is a vector with one 1 and a lot of zeroes

[0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]

Dimensionality: 20K (speech) – 50K (PTB) – 500K (big vocab) – 13M (Google 1T)

- Represent words as one-hot vectors
  - Pre-specify a vocabulary of N words in fixed (e.g. lexical) order
  - Represent each word by an N-dimensional vector with N-1 zeros and a single 1 (in the position of the word in the ordered list of words)

Why one-hot representation



- The one-hot representation makes no assumptions about the relative importance of words
  - All word vectors are the same length
- It makes no assumptions about the relationships between words
  - The distance between every pair of words is the same

Dimensionality reduction

1. we don't need all of the dimensions that represent a word, only the most important ones.
2. There are several techniques such as
   - Principle Component Analysis (PCA): The most important dimensions contain the most variance
   - Latent Semantic Analysis (LSA): Project terms and documents into a topic space using SVD on term-document (co-occurrence) matrix.
   - Low-rank Approximation
3. Can we learn the dimensionality reduction from texts?

Word2Vec

Distributed similarity based representations

- representing a word by means of its neighbors
- "You shall know a word by the company it keeps" (J. R. Firth 1957: 11)
- One of the most successful ideas of modern statistical NLP

government debt problems turning into **banking** crises as has happened in

saying that Europe needs unified **banking** regulation to replace the hodgepodge

↖ These words will represent *banking* ↗

Word2vec algorithm

1. Proposed by Mikolov et. al. and widely used for many NLP applications (in two papers).
2. Key features
   - Uses neural networks to train word / context classifiers (feedforward neural net)
   - Uses local context windows (environment around any word in a corpus) as inputs to the NN
   - Removed hidden layer.
   - Use of additional context for training LMs.
   - Introduced newer training strategies using huge database of words efficiently.

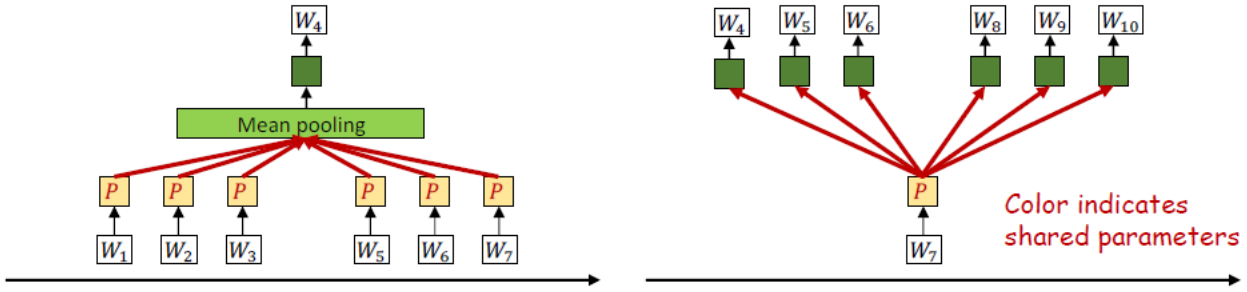
Word2vec algorithm

1. In their first paper, Mikolov et al. propose two architectures for learning word embeddings that are computationally less expensive than previous models[4].
2. In their second paper, they improve upon these models by employing additional strategies to enhance training speed and accuracy[5].

[4] Mikolov, T., Corrado, G., Chen, K., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. Proceedings of the International Conference on Learning Representations (ICLR 2013), 1-12.

[5] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. NIPS, 1-9.

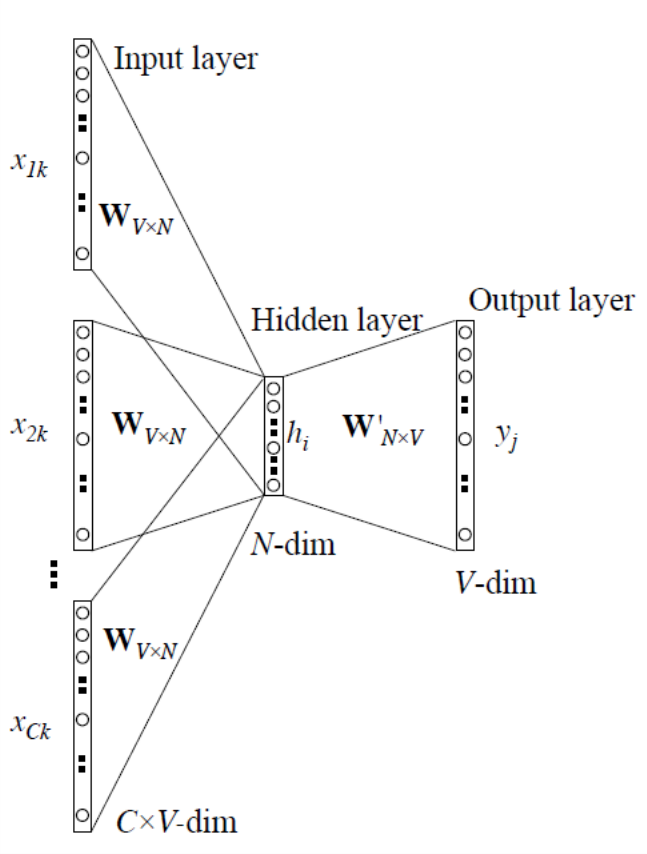
Alternative models to learn projections



- Soft bag of words: Predict word based on words in immediate context
  - Without considering specific position
- Skip-grams: Predict adjacent words based on current word

Continuous Bag-of-Words

1. Mikolov et al. thus use both the n words before and after the target word w_t to predict it.
2. They call this continuous bag-of-words (CBOW), as it uses continuous representations whose order is of no importance.



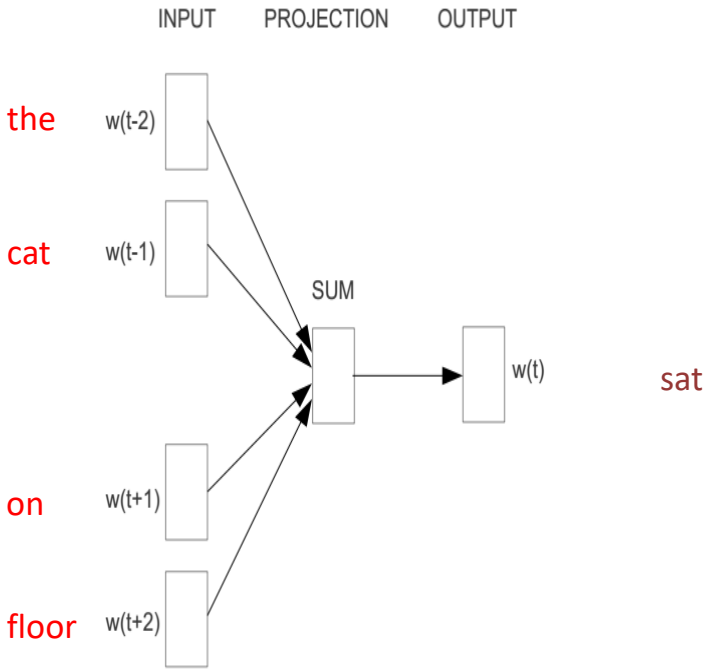
Continuous Bag-of-Words objective function

1. The objective function of CBOW in turn is

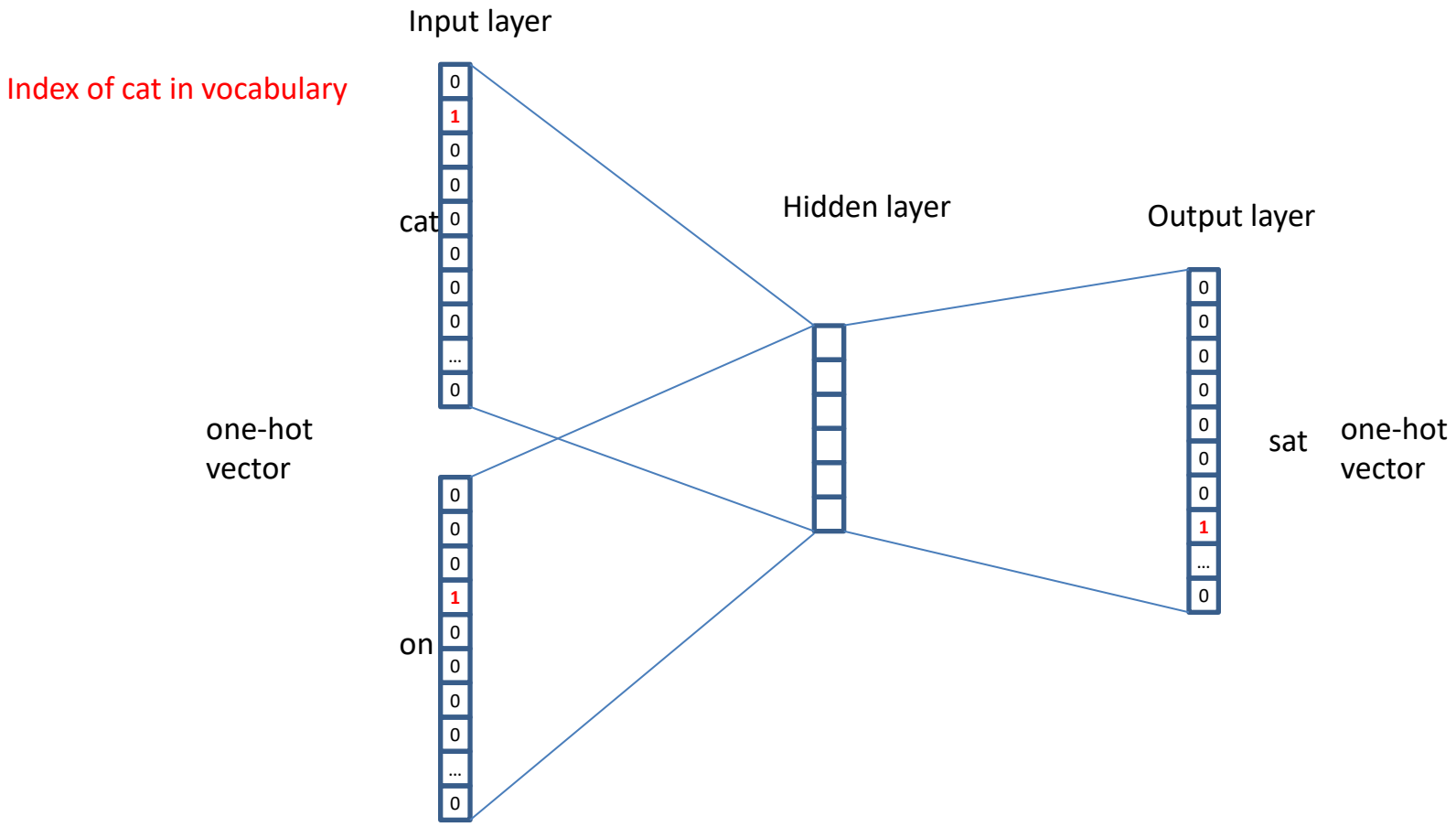
$$l(\theta) = \sum_{t \in Text} \log P(w_t | w_{t-n}, \cdots, w_{t-1}, w_{t+1}, \cdots, w_{t+n})$$

Word2vec – Continuous Bag of Word

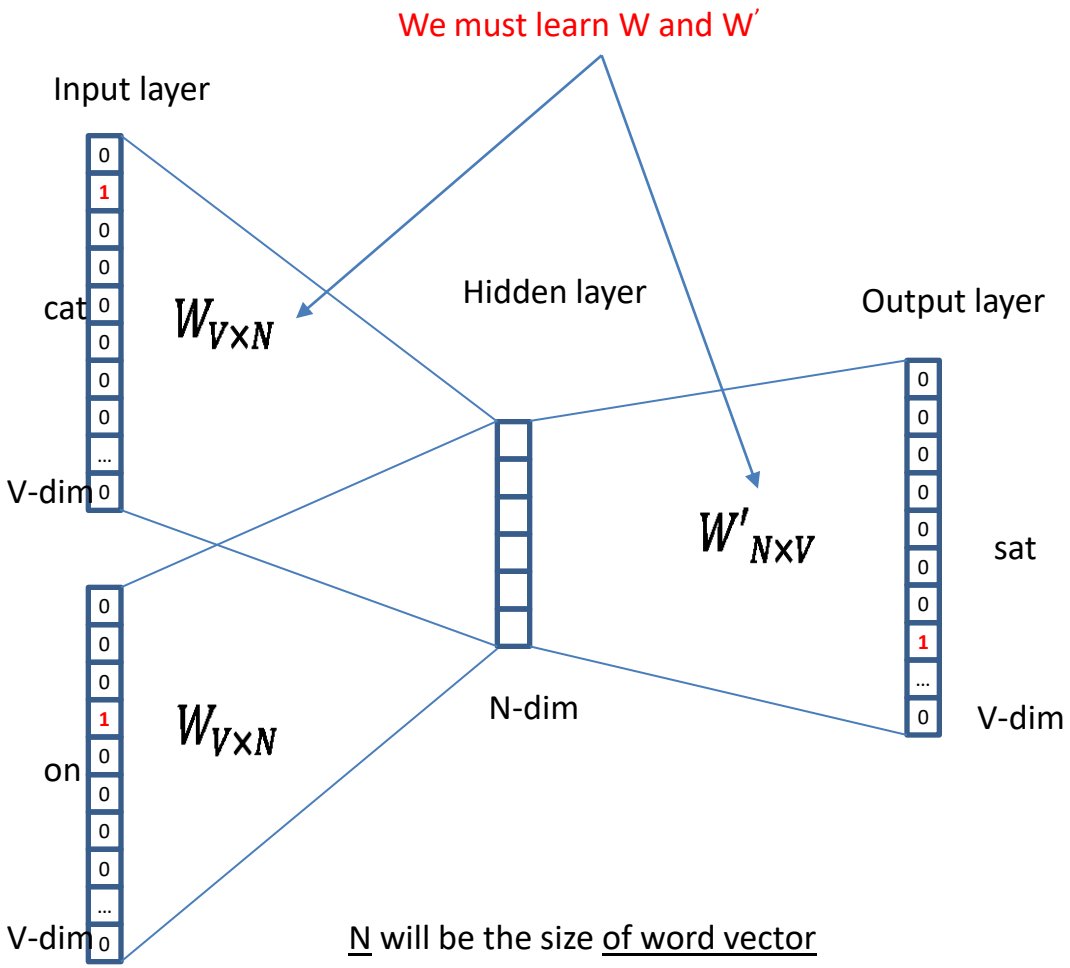
- E.g. “The cat sat on floor”
  - Window size = 2



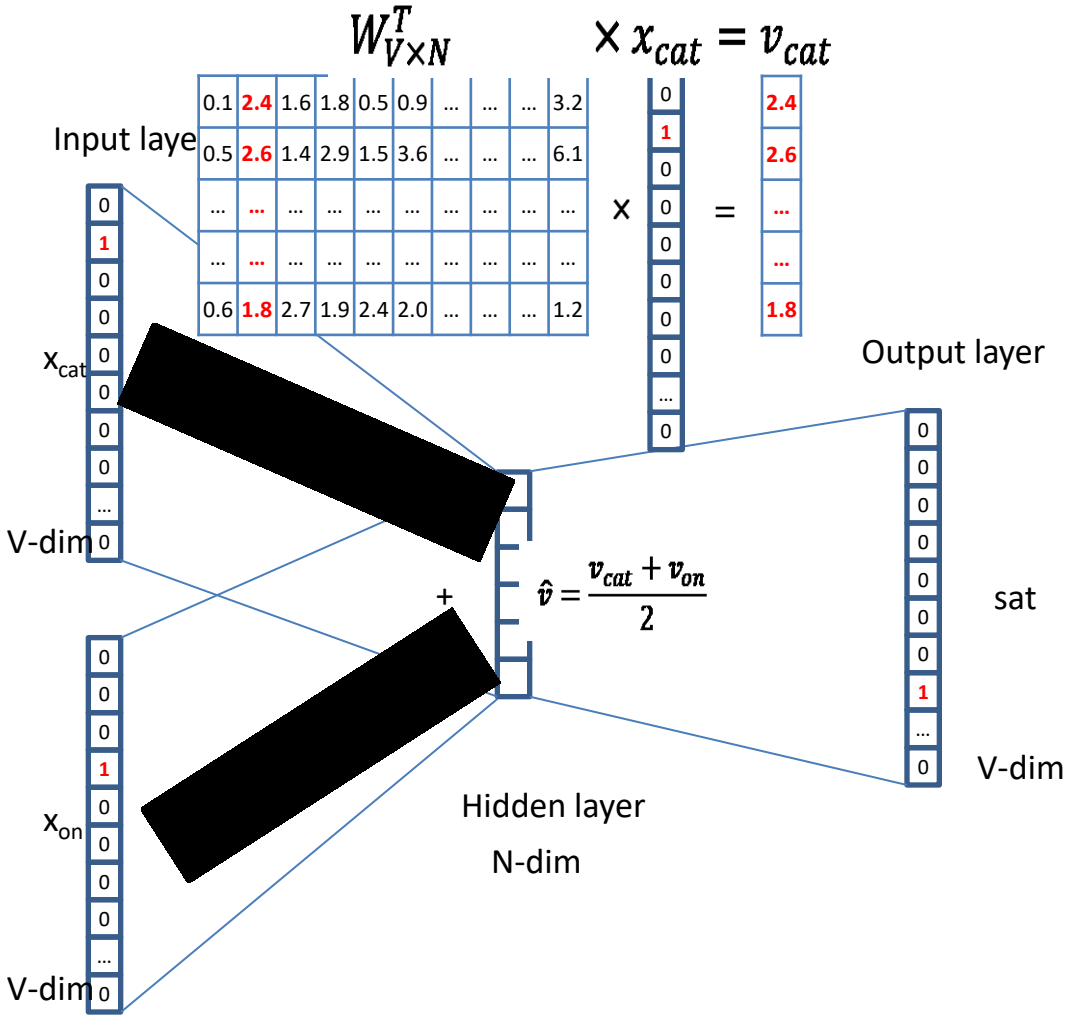
“The cat sat on floor”



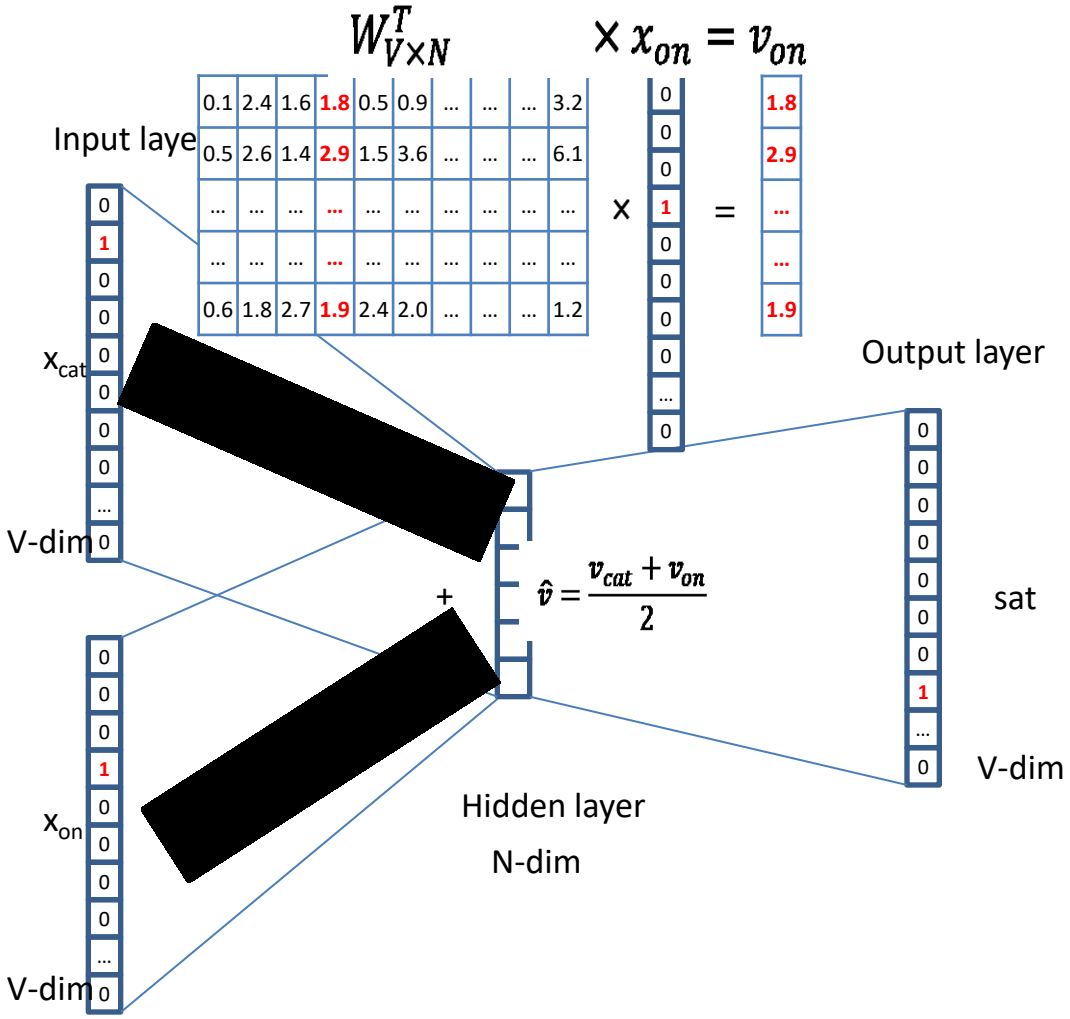
“The cat sat on floor”



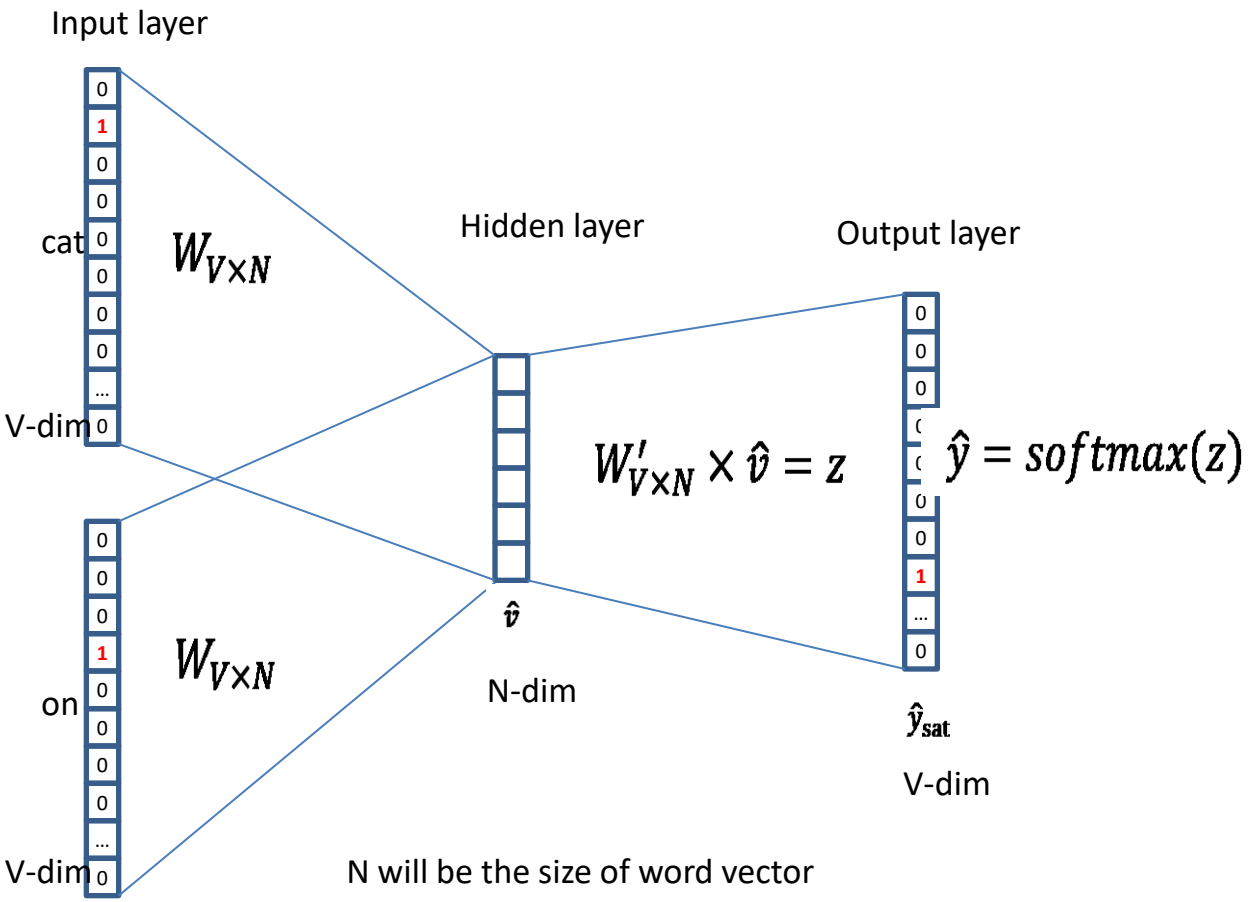
"The cat sat on floor"



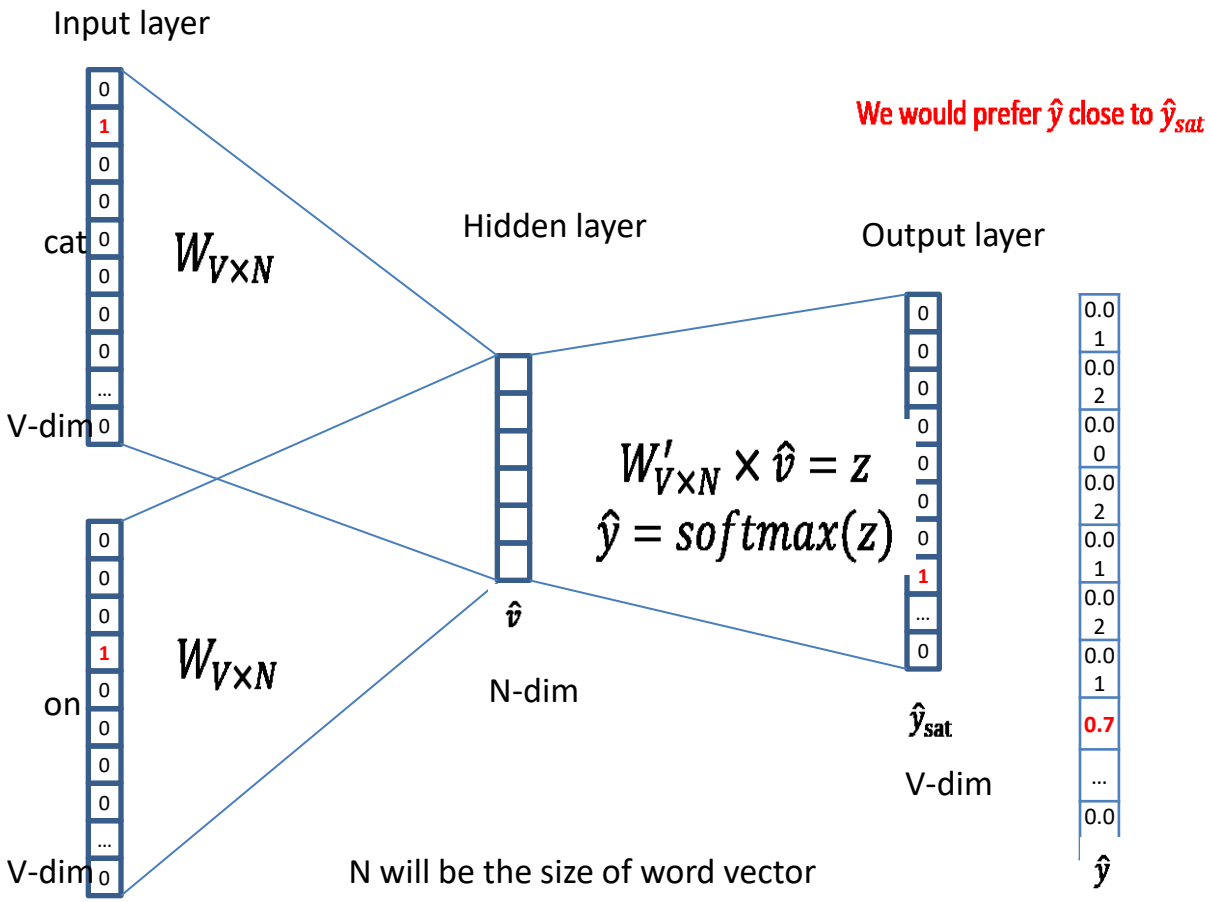
"The cat sat on floor"



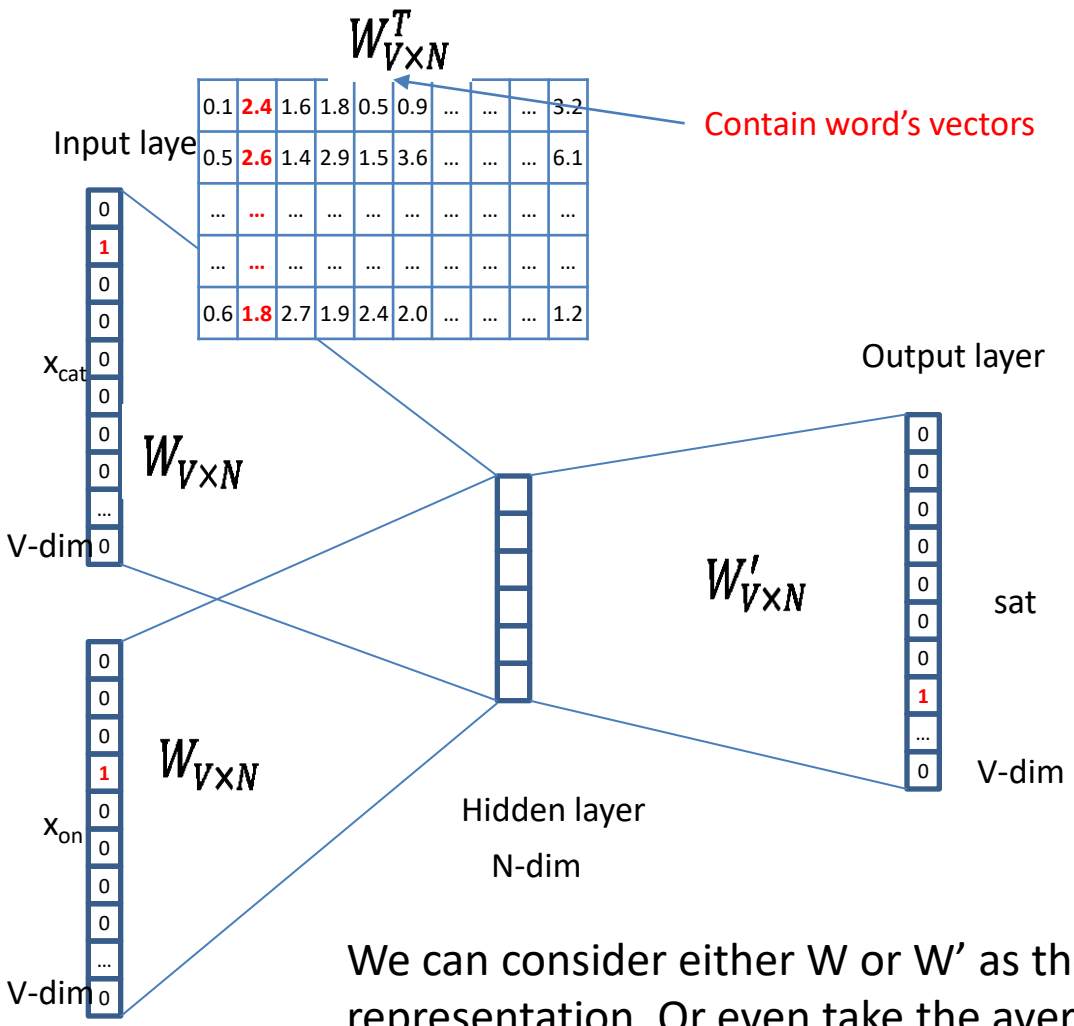
"The cat sat on floor"



"The cat sat on floor"



"The cat sat on floor"

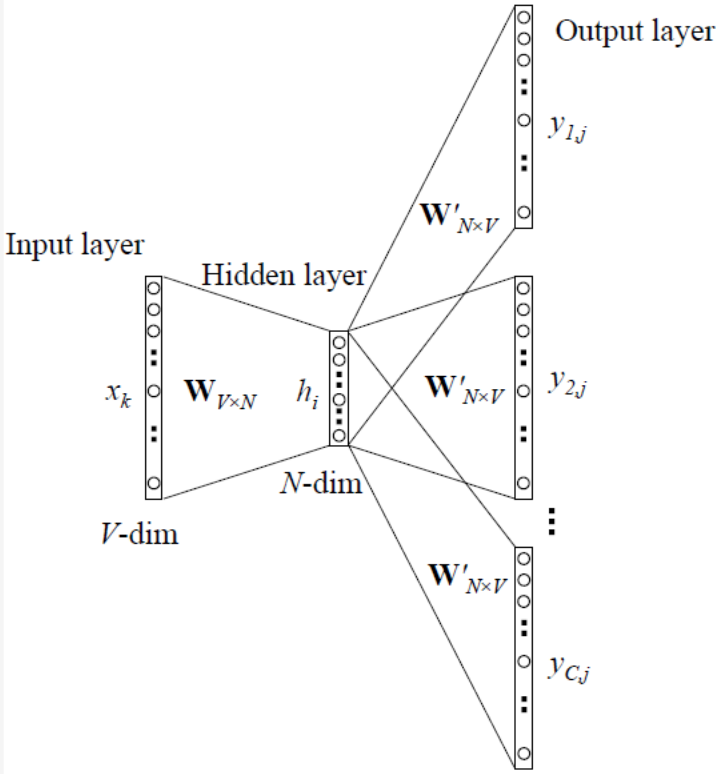


We can consider either W or W' as the word's representation. Or even take the average.

- W contains input word vectors.
- W’ contains output word vectors.
- We can consider either W or W’ as the word’s representation. Or even take the average.

Skip-gram

1. Instead of using the surrounding words to predict the center word as with CBOW, skip-gram uses the centre word to predict the surrounding words.



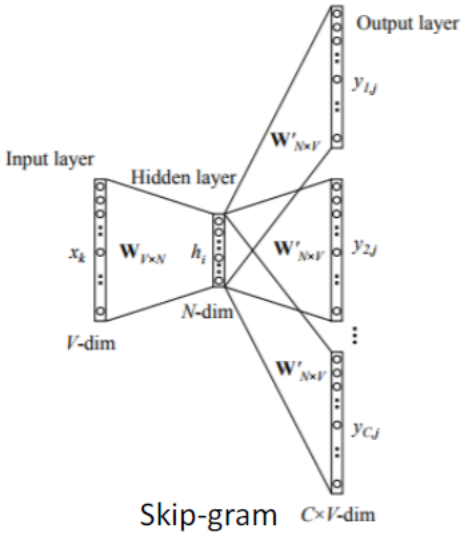
Skip-gram objective function

1. The skip-gram objective thus sums the log probabilities of the surrounding n words to the left and to the right of the target word w_t to produce the following objective function.

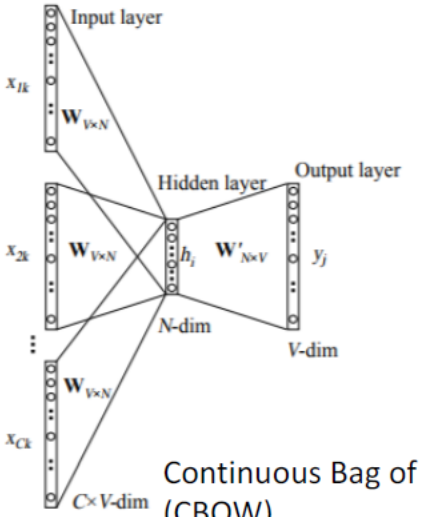
$$J(\theta) = \sum_{t \in Text} \sum_{-n \leq j \leq n, \neq 0} \log P(w_{t+j} | w_t)$$

Skip-gram vs. CBOW

- Two possible architectures:
  - given some context words, predict the center (CBOW)
    - Predict center word from sum of surrounding word vectors
  - given a center word, predict the contexts (Skip-gram)



Skip-gram



Continuous Bag of words (CBOW)

Word2vec embedding

- word2vec (Mikolov et al 2013): NN model using a two-layer network (i.e., not deep!) to perform dimensionality reduction.
  - Very computationally efficient, good all-round model (good hyperparameters already selected).
- Idea
  - Every word in a fixed vocabulary is represented by a vector
  - We have a large corpus of text
  - Go through each position *t* in the text, which has a center word *c* and context ("outside") words *o*
  - Use the similarity of the word vectors for *c* and *o* to calculate the probability of *o* given *c* (or vice versa)
  - Keep adjusting the word vectors to maximize this probability

Some interesting results

Word Analogies

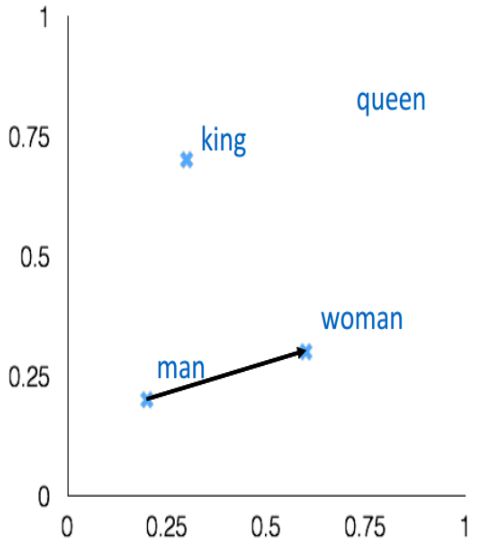
Test for linear relationships, examined by Mikolov et al. (2014)

a:b :: c:?

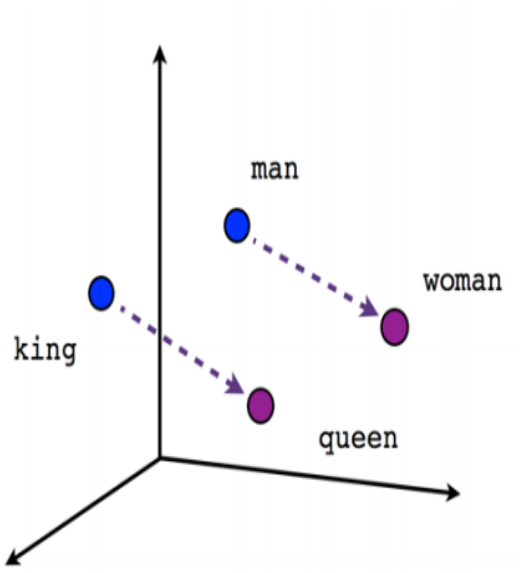
$$d = \arg\max_x \frac{(w_b - w_a + w_c)^T w_x}{||w_b - w_a + w_c||}$$

man:woman :: king:?

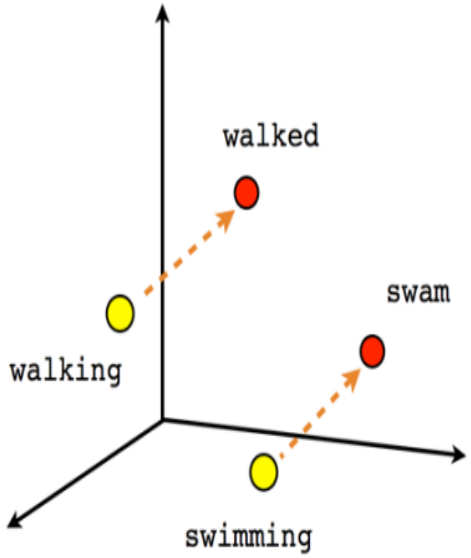
+	king	[0.30 0.70]
-	man	[0.20 0.20]
+	woman	[0.60 0.30]
	queen	[0.70 0.80]



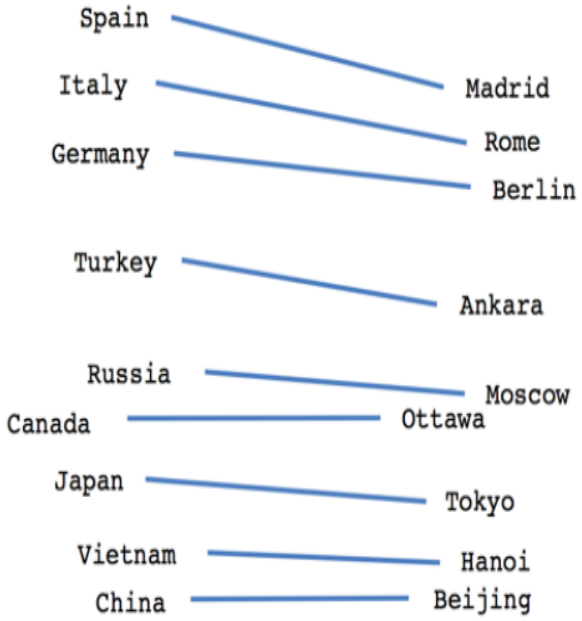
Examples



Male-Female



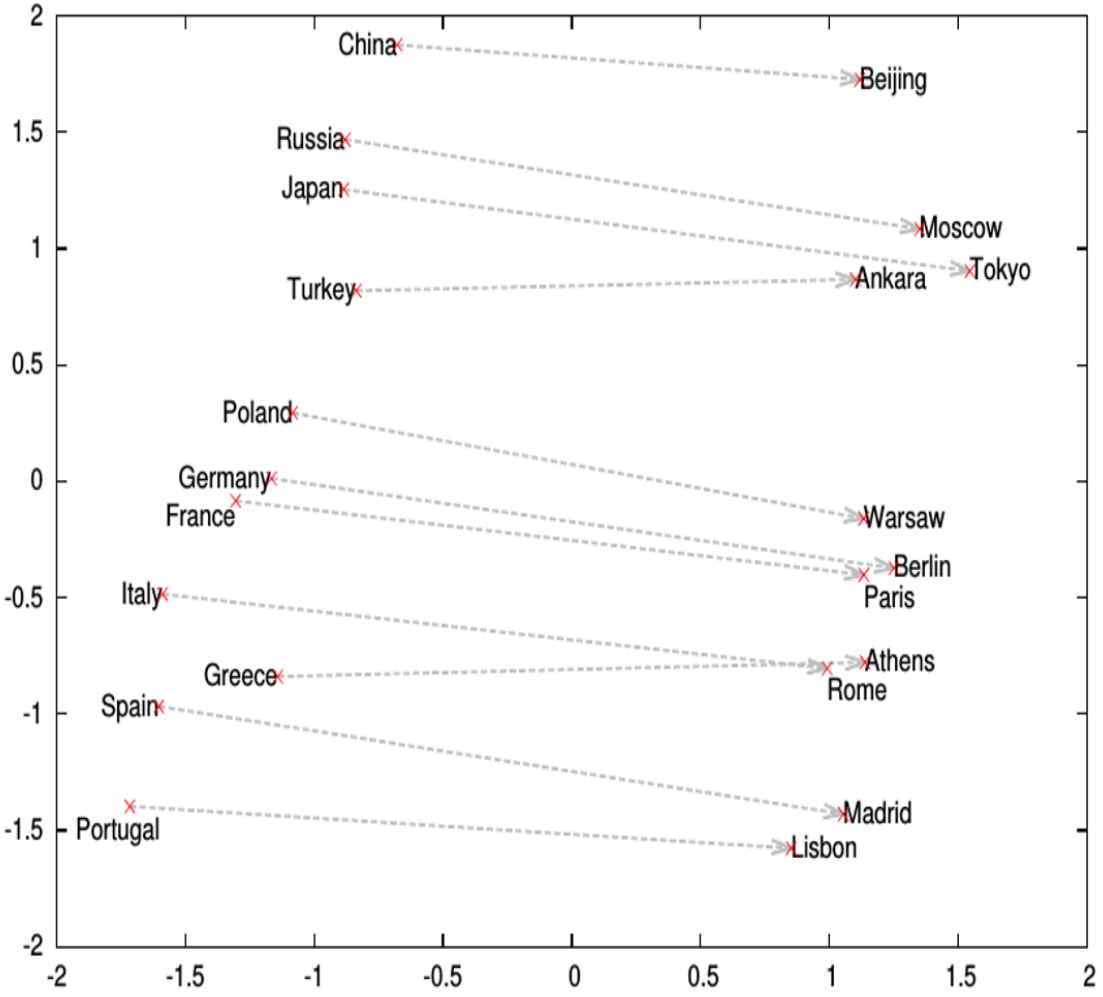
Verb tense



Country-Capital

vector[Queen] = vector[King] - vector[Man] + vector[Woman]

Word analogies



Quiz 1

Consider that our document collection S has the following documents: $D_1, ..., D_4$:

document	words
D_1	Information retrieval is an important subject.
D_2	The Johnson family has got a golden retriever.
D_3	Information theory uses plenty of theorems from mathematics.
D_4	It provides a golden opportunity for information sharing.

Our dictionary $DICT$ consists of 8 words: $\{w_1 =$ information, $w_2 =$ retrieval, $w_3 =$ subject, $w_4 =$ Johnson, $w_5 =$ golden, $w_6 =$ theory, $w_7 =$ mathematics, $w_8 =$ sharing$\}$. By stemming, "retrieval" and "retriever" are regarded as the same word, and so are "theory" and "theorem".

Problem 1. Let $tf(w, D)$ denote the term frequency of term w in a document D. Give the value of $tf(w_i, D_j)$ for all $1 \leq i \leq 8$ and $1 \leq j \leq 4$.

Problem 2. Let $idf(w)$ denote the inverse document frequency of term w. Give the value of $idf(w_i)$ for all $1 \leq i \leq 8$.

Problem 3. Convert D_1 into an 8-dimensional point according to the tf-idf model.

Problem 4. Assume that a query (which is a sequence of words) has been converted to a point $(0.415, 1, 2, 0, 0, 0, 0, 0)$. What is the score of D_1 with respect to this query according to the cosine metric?