UNIVERSITÄT ZU LÜBECK

GenAI in Education, Science, and Society

Aktuelle Themen Data Science und
KI / CS5070-KP04

Prof. Dr. Sylvia Melzer

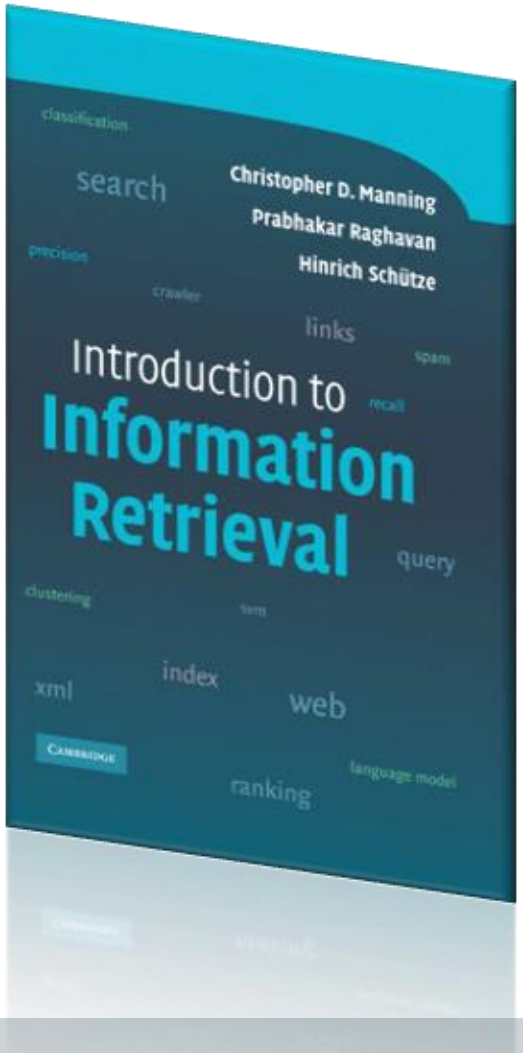
SoSe2026

Representation Learning for Sequential Structures, Embedding Spaces, word2vec, CBOW, Skip-gram

GenAI in Education, Science, and Society

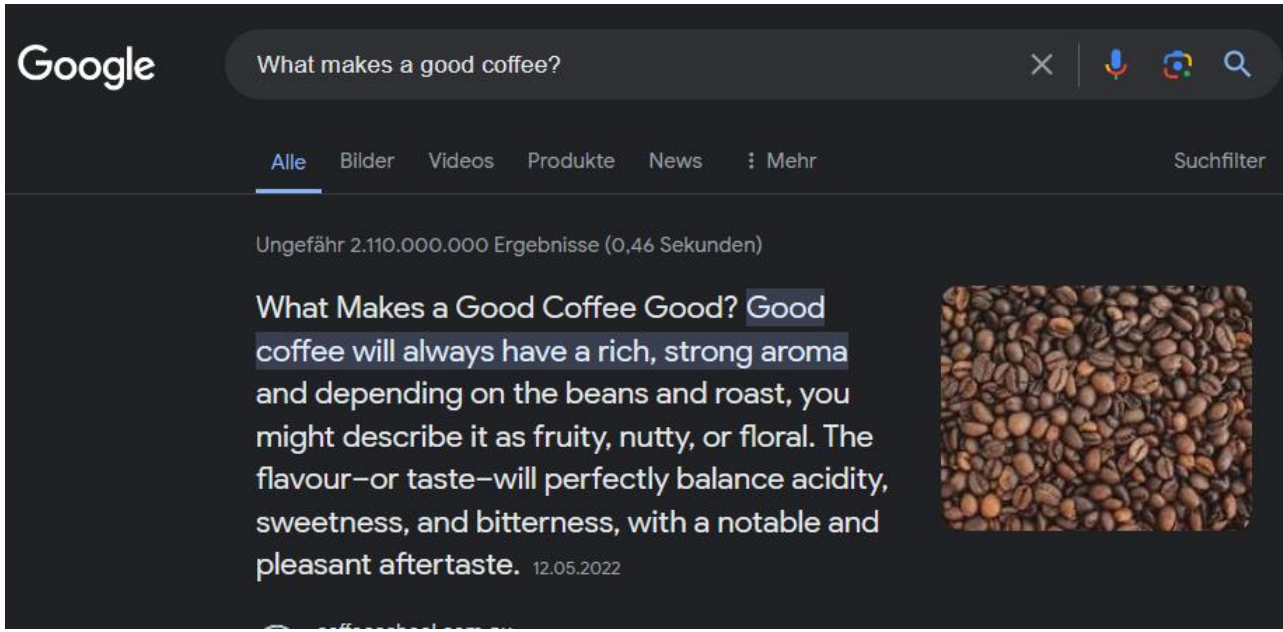
Agents for Information Retrieval

- Acknowledgement:
- These lecture notes were prepared in collaboration with Prof. Dr. habil. Ralf Möller.
- Presentations are taken from various lectures on the following book:



Agenda: IR Agents

- Agents visit accessible document repository structures (e.g., FDR[1])
- Task descriptions:
  - **Document retrieval** w.r.t. search string
  - Statistics: **Topic characterization** of documents w.r.t. reference corpus
  - GenAI: **Question answering** and information retrieval w/ and w/o **references**, **summaries** or **interpretations** (subjective context descriptions, SCDs)



[1] https://www.fdr.uni-hamburg.de

Example: Interpretative information retrieval

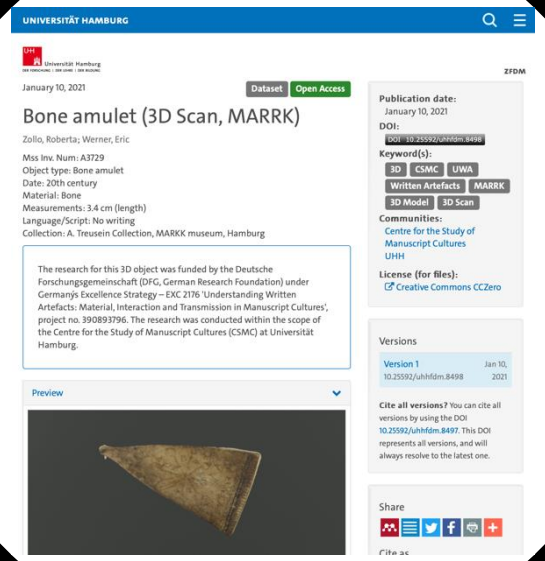
Fig. 10: Knochenamulett *sarang timah*; Katalog Nr. 29. | Bone amulet known as

provides when asked why t
bravely against the Dutch tr
sahala that gave him superna

The four objects no. 20,
once again that written artef
magic amulets. We can see
emplified here because it is
the bamboo slips but the act
their significance. Instead of
the bamboo carries combina
make much sense. This phe
tional Batak manuscript cult
it more often on bone
explained in th ection

An ex of this type
slips and tubes can b
mall half-cylinder used to b
this case, the characters we
inscription, possibly to conce
tubes forming part of the necl
catalogued as no. 32 and no.
category. Unfortunately, no
the provenance of these two
place and the date of acquisiti
it is not clear whether these s
objects or for the tourist marl

The last written artefacts
catalogue as no. 26, no. 27
tempt at classification as the
a *pagar* and an amulet. The
designed and are fascinatin
datu used: the long, tapered
several sections of text and
lengths. After being inscrib



Subjective content descriptions

Also in doubt

Counter hypothesis

in Karo-Schrift; Katalog Nr. 40. | Bone of a water buffalo with calendar and writing in Karo script

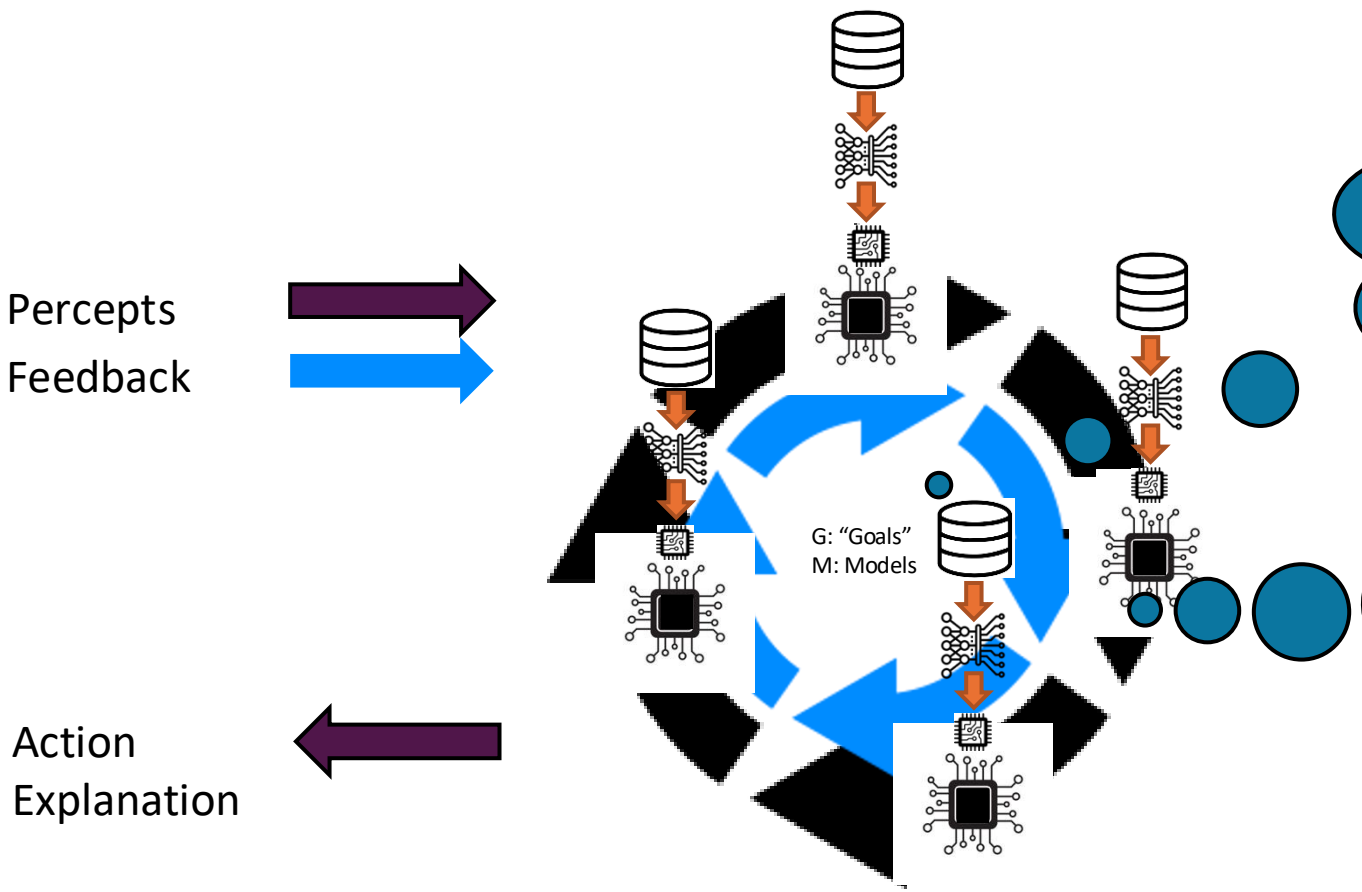
was mostly divided into numerous small boxes intended
for the texts, which were combined with several recurrent
illustrations, mainly anthropomorphic figures and diagrams
associated with divination.

One of the most common bone amulets is known by the
name of *sarang timah* and consists of a piece of buffalo's
shoulder cut in a triangular shape and serving as a protective
amulet against bullets (Fi In this catalogue, there is
only one this kin object, no. 29. It is unclear
whether it was made for local use or the tourist market. The
object is an early acquisition by the museum, having been
acquired from Adolf Tre
are poor and sloppy, and
even though these are u
As no more information
biography, it was impo
thoroughly.

The question of auth
all the bone pieces in
an inconsistent style of
nomenon of creating a bone artefact to be sold on the tourist
market is actually a practice dating back to at least the early
twentieth century. As Voorhoeve pointed out (1975, 124),

bones with Batak writing or drawings are sold as curiosities

AI Hypothesis: Agent exhibits intelligent IR behavior



How are goals represented?

How is feedback acquired, represented and handled?

Search string queries

- Rank documents by “usefulness” or “relevance” that “match” a query string (e.g., “sorting”)
- Rank score from [0,1]
  - For each document in the repository w.r.t. a certain query
- No notion of a “correct” answer
  - Only local perspectives of “match” and “relevance” as a utility measure
  - In human (principal) as well as agent

Option: Linear combination of rank measures

- First generation of rank determination with different measures for different areas/zones:
  - Example for linear combination 0.6*<"**sorting**"in Title> + 0.3*<"**sorting**" in Abstract>+ 0.05*<"**sorting**" in Body>+ 0.05*<"**sorting**" in Boldface>
  - Each expression such as <**sorting** in Title> results in a value from {0,1}
  - Then the total value is in [0,1]

In this case, the rank measure can only assume a finite set of values values. What are these?

Multiple words in search strings?

So-called
inverted indes
aka “postings lists”

What to do for
bill** OR **rights ?

- We expect the following lists for the query ***bill** AND **rights***:

Author	0.6	***bill***	1 → 2
		rights	
Title	0.3	***bill***	3 → 5 → 8
		rights	3 → 5 → 9
Body	0.1	***bill***	1 → 2 → 5 → 9
		rights	3 → 5 → 8 → 9

Determine rank for each document based on the weights 0.6,0.3,0.1

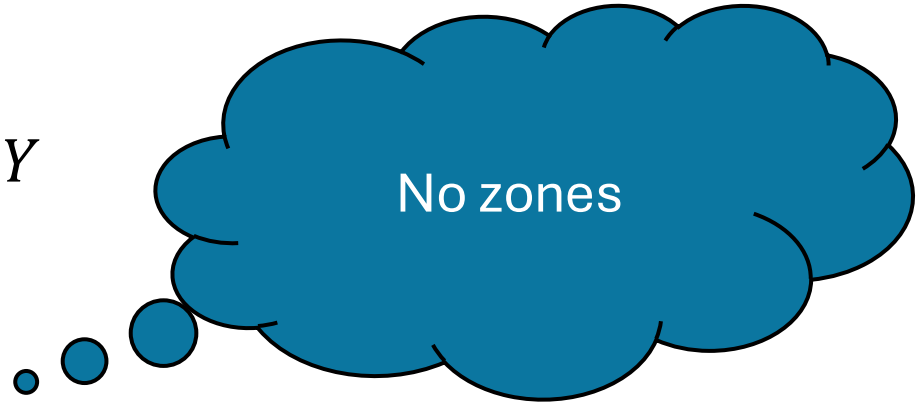
- Typically, one is only interested in the k highest-rated documents

Where do the weights come from?

- Given
  - A retrieval corpus
  - One set of test questions
  - Relevance statements/rankings for respective answers
- Determine a set of weights so that relevance information fits (better)
- Data mining for matching weights:
  - Solve an optimization problem

Incidence matrices and rank measures

- Bag of words model: Document is a set of words
- Sets = bit vectors: document as binary vector X in $\{0,1\}^v$
  - where v is the size of the vocabulary
- Query as vector Y in $\{0,1\}^v$
- Rank measure: Overlap measure: $|X \cap Y| = X \cdot Y$



Document and query representati

So-called 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
…	…	…	…	…	…	…

Example

- Query ***ides of march***:
  - All docs contain ***of***
  - Shakespeare's *Julius Caesar* has rank 3 (all words appear)
  - Other Shakespeare plays have rank 2 (for ***march***) or 1
- *Julius Caesar* comes first in ranking order

Measure of Overlap

- What is going wrong?
- Not considered:
  - Term frequency ignored in the document
  - Term rarity not observed in the collection
    - ***of*** more frequent than ***ides*** or ***march***
  - Length of documents ignored
- Normalization necessary

Overlap measure: Normalization

Two possibilities

- Jaccard measure (Sets or bit vectors):
  - $|X \cap Y| / |X \cup Y|$

X: document in repository
Y: query

Number of ones / Cardinality of the set

- „Cosine measure“ (only for bit vectors):
  - $|X \cap Y| / \sqrt{|X| \cdot |Y|}$

Normalization with product of vector lengths (max value of $X \cdot Y$)

- Discuss: Jaccard measure for query with rare terms and with frequent terms. Problems with frequent terms?
- Does the cosine measure solve this problem?

Term document counters

- Number of occurrences of a term in a document:
  - Bag of words model
  - Document is a vector in $\mathbb{N}^v$: a column in the matrix

	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

Calculation of the rank

- Addition of the measure for each individual word (possibly range weighted)
- Let's consider the query **ides of march**
  - Julius Caesar has 5 occurrences of **ides**
  - No other part mentions **ides**
  - **march** occurs in dozens of documents
  - All pieces contain **of**
- The piece with the most occurrences of **of** is ranked highest by a simple counting rank measure

Term Frequency **tf**

- Long documents contain high counters per se
- Normalization by document length
- Use of **relative counts / frequencies of occurrence** (term frequency **tf**)
- Is this already sufficient?

Weighted term frequency

- Number of occurrences with linear influence?
  - 0 vs. 1 occurrence of a term in a document
  - 1 vs. 2 occurrence of a term in a document
  - 2 vs. 3 occurrence of a term in a document
- Much helps a lot, but really much does not really help much more
  - $w_{i,d} = 0 \; if \; tf_{i,d} = 0, sonst \; 1 + \log tf_{i,d}$
- Terms that are rare, but characterize a document to a certain extent
  - 10 occurrences of **hernia** vs
  - 10 occurrences of **the**

TF.IDF (Term Frequency – Inverse Document Frequency)

- c_{ij} = number of terms t_i in document d_j

$$TF_{ij} = c_{ij} / number\ of\ terms\ overall\ in\ d_j$$

- n_i = Number of documents with term i
- N = Total number of documents

$$IDF_i = \log \frac{N}{n_i}$$

- TF.IDF measure

$$w_{ij} = TF_{ij} \cdot IDF_i$$

K. Spärck Jones. A Statistical Interpretation of Term Specificity and Its Application in Retrieval. Journal of Documentation 28: 11–21, **1972**

see Kishore Papineni, NAACL 2, **2002** for theoretical justification

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	13,1	11,4	0,0	0,0	0,0	0,0
Brutus	3,0	8,3	0,0	1,0	0,0	0,0
Caesar	2,3	2,3	0,0	0,5	0,3	0,3
Calpurnia	0,0	11,2	0,0	0,0	0,0	0,0
Cleopatra	17,7	0,0	0,0	0,0	0,0	0,0
mercy	0,5	0,0	0,7	0,9	0,9	0,3
worser	1,2	0,0	0,6	0,6	0,6	0,0

Embedding Data into Vector Spaces

- Terms as axes (20000+ dimensions, even with word stems)
- Manually defined embedding
- Later shown to be optimal in a certain sense

Documents are vectors (or points)

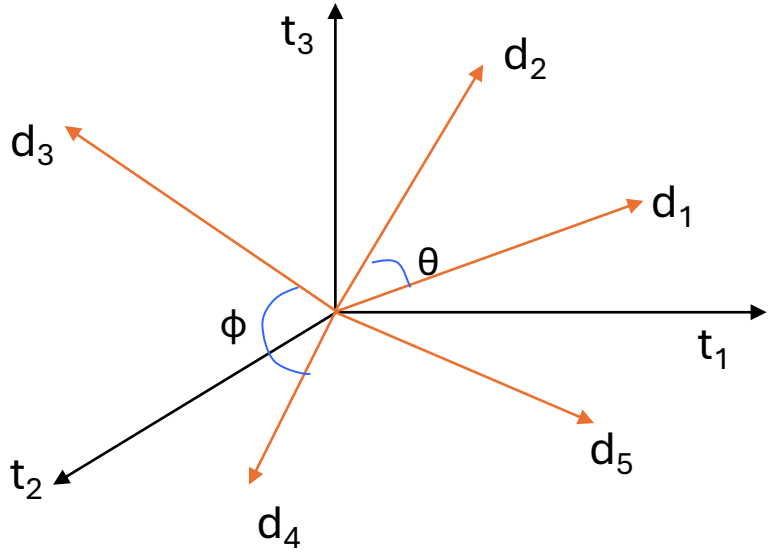
NB: measures can be >1!

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	13.1	11.4	0.0	0.0	0.0	0.0
Brutus	3.0	8.3	0.0	1.0	0.0	0.0
Caesar	2.3	2.3	0.0	0.5	0.3	0.3
Calpurnia	0.0	11.2	0.0	0.0	0.0	0.0
Cleopatra	17.7	0.0	0.0	0.0	0.0	0.0
mercy	0.5	0.0	0.7	0.9	0.9	0.3
worser	1.2	0.0	0.6	0.6	0.6	0.0

see
Kishore Papineni, NAACL 2, **2002**
for theoretical justification

Distance dissimilarity

- Embedding space operations



- Idea: Points being close, docs being „similar“, i.e., docs are about similar things
- Dissimilarity as Euclidean distance of the end points
- $dissim(d_j, d_k) = |d_j - d_k| = \sqrt{\sum_{i=1}^{n}(d_{i,j} - d_{i,k})^2}$
- The greater the distance, the more dissimilar
- Normalization of doc vecs necessary (length of vectors = 1)

Cosine similarity

- Definition 2: Similarity between documents d_1 and d_2 captured by cosine of the angle between d_1 and d_2

$$sim(d_j, d_k) = cos(\angle(d_j, d_k))$$

$$= \frac{\vec{d}_j \cdot \vec{d}_k}{\left|\vec{d}_j\right|\left|\vec{d}_k\right|} = \frac{\sum_{i=1}^{n} w_{i,j} w_{i,k}}{\sqrt{\sum_{i=1}^{n} w_{i,j}^2}\sqrt{\sum_{i=1}^{n} w_{i,k}^2}}$$

Dot product
(aka scalar product)

- Normalization would further reduce computational effort
- The more similar two objects are, the greater the similarity measure

Queries in the vector space model

- Central idea: Query is small document
- Result: Documents sorted by cosine of the angle of the assigned vectors to the query vector

- Note: Vector d_q is sparse!

$$sim(d_j, d_q) = cos(\angle(d_j, d_q))$$

Siehe auch: Salton, G., *Automatic information organization and retrieval.* New York: McGraw-Hill Book Company. **1968**.

Polysemy and context

- Word has several meanings, depending on the context
- Example: horse = animal, gymnastic apparatus, chess piece
- Vector space model does not differentiate between meanings

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

Synonymy and context

- Correspondence of different words or constructions in the same language
- Example: gift, souvenir
- Missing association in the vector space model

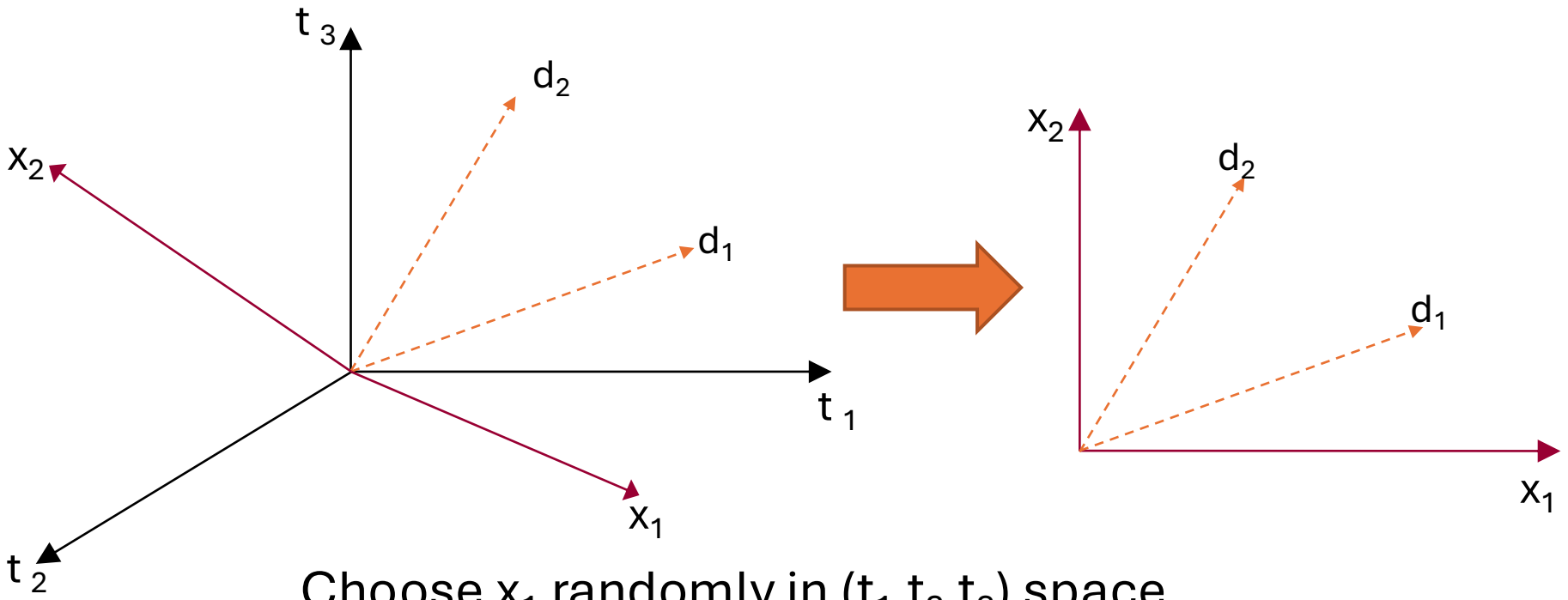
- If meanings of different words are the same, the use of words in the environment will be similar
- How can we capture this?

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

Dimensionality reduction

- Instead of using m=20000 terms for vectors,
- Reduction to approx. k=100 new dimensions, so that similarities are retained
  - Known as Latent Semantic Indexing (LSI)
- 2 Methods
  - Random projection on $k<<m$ axes
  - Singular value decomposition or principle component analysis

Example: Projection from 3 to 2 axes



Choose x_1 randomly in (t_1,t_2,t_3) space

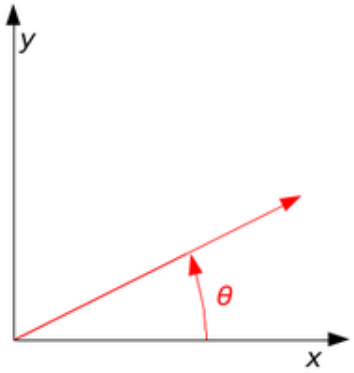
Choose x_2 randomly but orthogonal to x_1

Dot product of x_1 and x_2 equal to 0

General: Projection on *k<<m* axes

- Choose random direction x_1 in vector space
- For i from 2 to k
- Randomly choose a direction x_i orthogonal to $x_1, x_2, \ldots x_{i-1}$
- Project each vector into the subspace spanned by $\{x_1, x_2, \ldots, x_k\}$
- Relative distances are preserved with high probability
- But: Relatively complex calculations

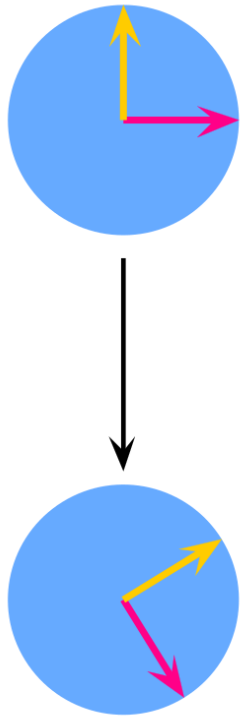
Mappings: Rotation



$$x' = x\cos\theta - y\sin\theta$$
$$y' = x\sin\theta + y\cos\theta$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

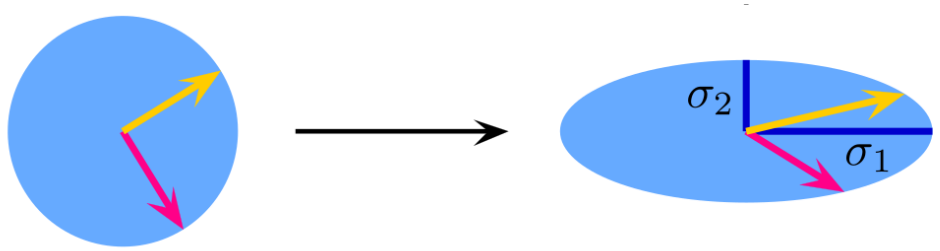


[Wikipedia]

Mappings: Scaling

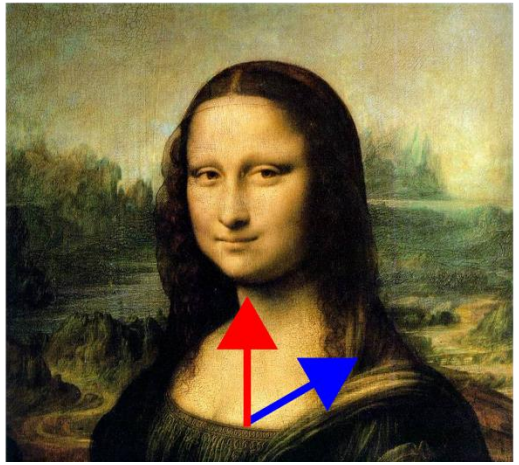
$$S_v p = \begin{bmatrix} v_x & 0 & 0 \\ 0 & v_y & 0 \\ 0 & 0 & v_z \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} = \begin{bmatrix} v_x p_x \\ v_y p_y \\ v_z p_z \end{bmatrix}$$

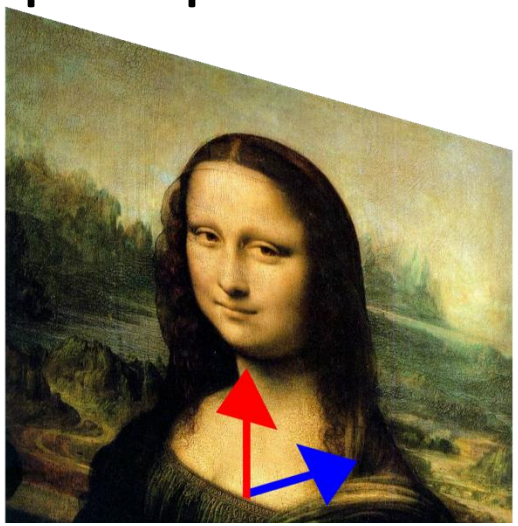
$$S_v = \begin{bmatrix} v_x & 0 & 0 \\ 0 & v_y & 0 \\ 0 & 0 & v_z \end{bmatrix}$$



[Wikipedia]

Mappings and their properties





- Example: Shear mapping
- The red arrow does not change
- Eigenvector

$$\mathbf{Sv} = \lambda\mathbf{v}$$

Horizontal shear

If the coordinates of a point are written as a column vector (a 2×1 matrix), the shear mapping can be written as multiplication by a 2×2 matrix:

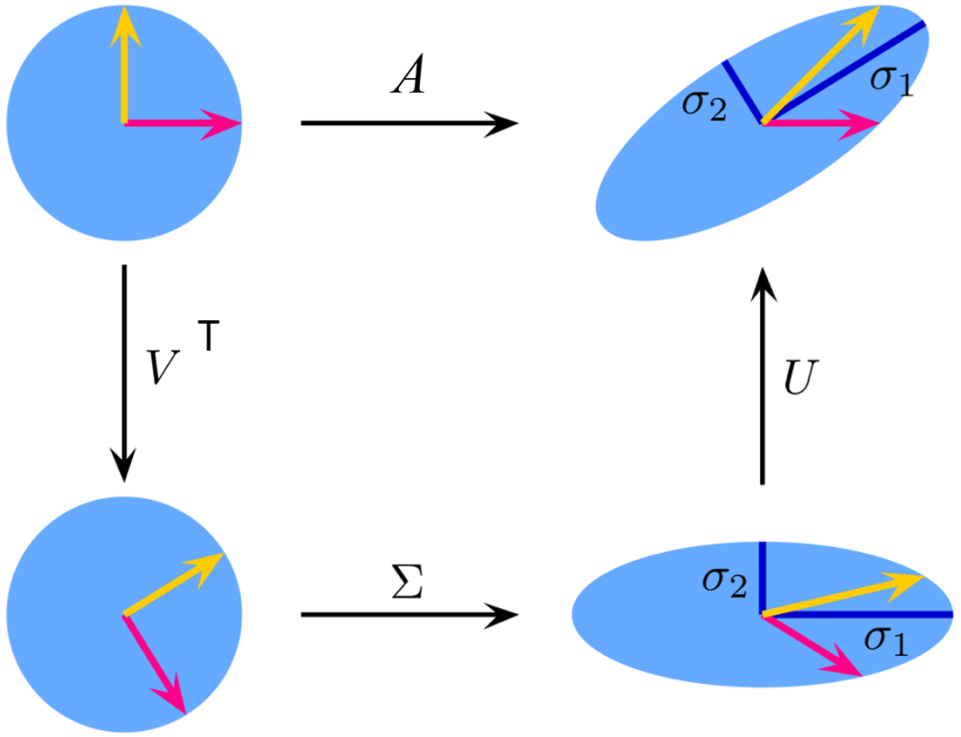
$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x + my \\ y \end{pmatrix} = \begin{pmatrix} 1 & m \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}.$$

A **vertical shear** (or shear parallel to the y-axis) of lines is similar, except that the roles of x and y are swapped. It corresponds to multiplying the coordinate vector by the transposed matrix:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x \\ mx + y \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ m & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}.$$

[Wikipedia]

Shear decomposed



$$A = U \cdot \Sigma \cdot V^{\mathsf{T}}$$

[Wikipedia]

SVD example

Be $A = \begin{bmatrix} 1 & -1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$

Number of rows: m=3
Number of columns: n=2

2D point mapped into 3D point
Dimensions of vector space: rank

The SVD is

$$\begin{bmatrix} 0 & 2/\sqrt{6} & 1/\sqrt{3} \\ 1/\sqrt{2} & -1/\sqrt{6} & 1/\sqrt{3} \\ 1/\sqrt{2} & 1/\sqrt{6} & -1/\sqrt{3} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{3} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix}$$

matrices)

Approximation by matrix with small rank

Optimization problem

k fix

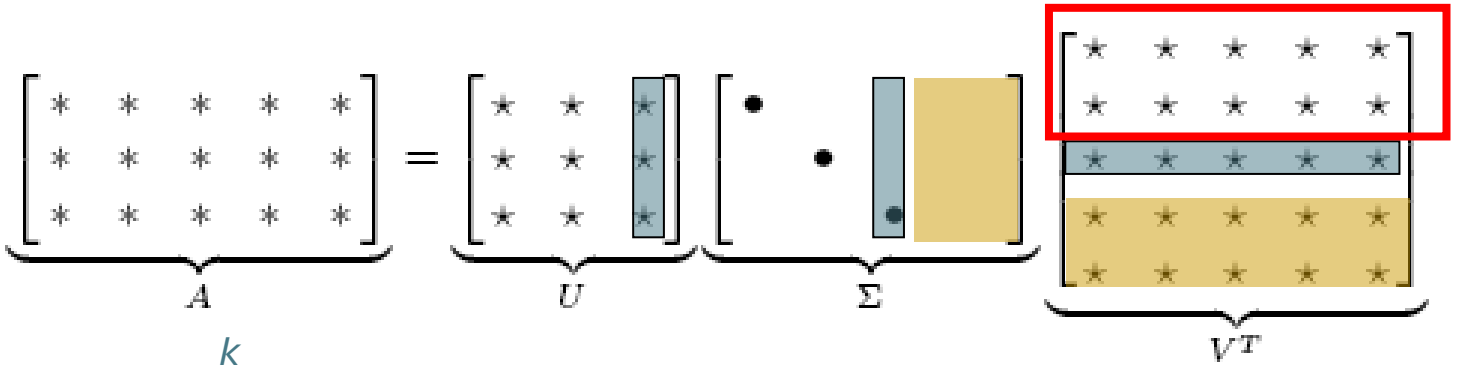
Solution using SVD

$$A_k = \underset{X:rank(X)=k}{\arg\min} \|A - X\|_F$$

$$A_k = U \cdot diag(\sigma_1, \cdots, \sigma_k, \underbrace{0, \cdots, 0}) \cdot V^T$$

Set smallest r-k
Eigenvalues to 0

new document representations

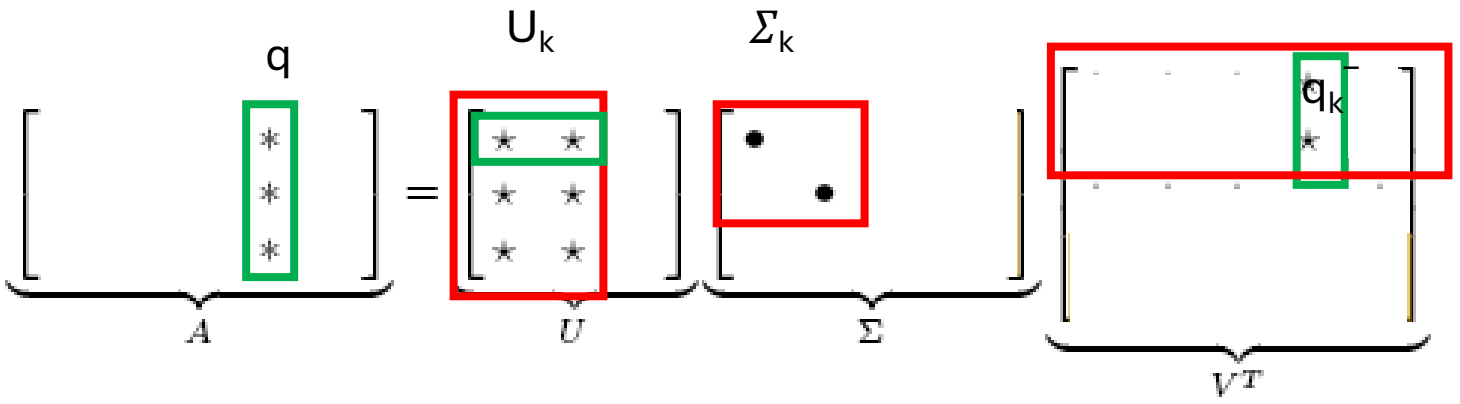


C. Eckart, G. Young, The approximation of a matrix by another of lower rank. Psychometrika, 1, 211-218, **1936**

Application for information retrieval

- Term-document matrix can have m=50000, n=10 million entries (rank close to 50000)
- We can construct an approximation A_{100} with rank 100 and smallest Frobenius norm error
  - Also known as Principle Component Analysis (PCA)
- The new matrix (see previous presentation) defines *latent* characteristics (no more comprehensible terms) for information retrieval (Latent Semantic Indexing, LSI)

How do we handle queries?



$$q_k = q^T U_k \Sigma_k^{-1} T$$

- Query q (sparsely populated)
- A query q is mapped into the LSI space as follows
- Query q_k is not sparse
- Query response via k nearest neighbors (cosine distance)
- Efficiently done in vector databases

Summary

- Significant reduction of the vector space dimensions
  - Reduction of the memory requirement
  - Faster processing
  - Reduction of "noise“
- "Semantic clustering“
  - Similar terms mapped to the same dimension
  - Synonymy and polysemy more manageable (many tests in the literature)

Scott Deerwester, Susan Dumais, George Furnas, Thomas Landauer, Richard Harshman: *Indexing by Latent Semantic Analysis.* In: *Journal of the American society for information science*. **1990**.
Landauer, Thomas; Foltz, Peter W.; Laham, Darrell. "Introduction to Latent Semantic Analysis". Discourse Processes. 25 (2–3): 259–284, **1998**.

External evaluation of retrieval results

- **Precision/Recall**

$$\text{precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|}$$

$$\text{recall} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|}$$

F-measure

The weighted harmonic mean of precision and recall, the traditional F-measure or balanced F-score is:

$$F = 2 \cdot (\text{precision} \cdot \text{recall})/(\text{precision} + \text{recall}).$$

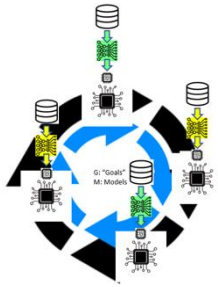
Recall also means sensitivity

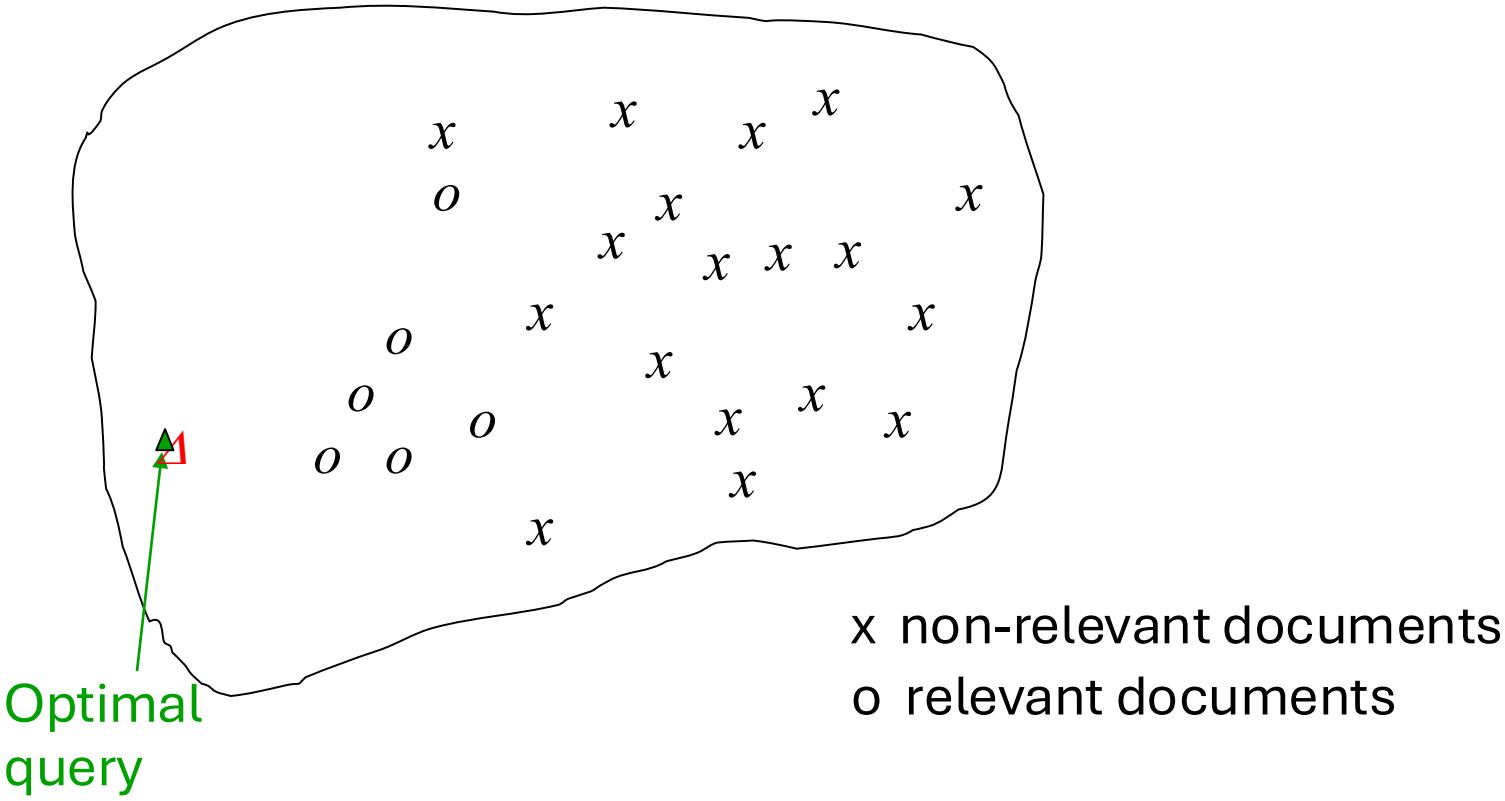
Evaluation measures: Overview

		True condition			
	Total population	Condition positive	Condition negative	Prevalence $= \frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$	
Predicted condition	Predicted condition positive	**True positive**	**False positive** (Type I error)	Positive ~~predictive value~~ (PPV), Precision $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Test outcome positive}}$	False discovery rate (FDR) $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Test outcome positive}}$
	Predicted condition negative	**False negative** (Type II error)	**True negative**	False omission rate (FOR) $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Test outcome negative}}$	Negative predictive value (NPV) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Test outcome negative}}$
	Accuracy (ACC) $= \frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$	True ~~positive rate~~ (TPR), Sensitivity, Recall $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	False positive rate (FPR), Fall-out $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$	Positive likelihood ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) $= \frac{\text{LR+}}{\text{LR−}}$
		False negative rate (FNR), Miss rate $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$	True negative rate (TNR), Specificity (SPC) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$	Negative likelihood ratio (LR–) $= \frac{\text{FNR}}{\text{TNR}}$	

[Wikipedia]

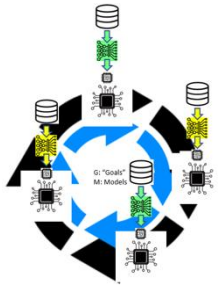
Relevance Feedback / Reinforcement

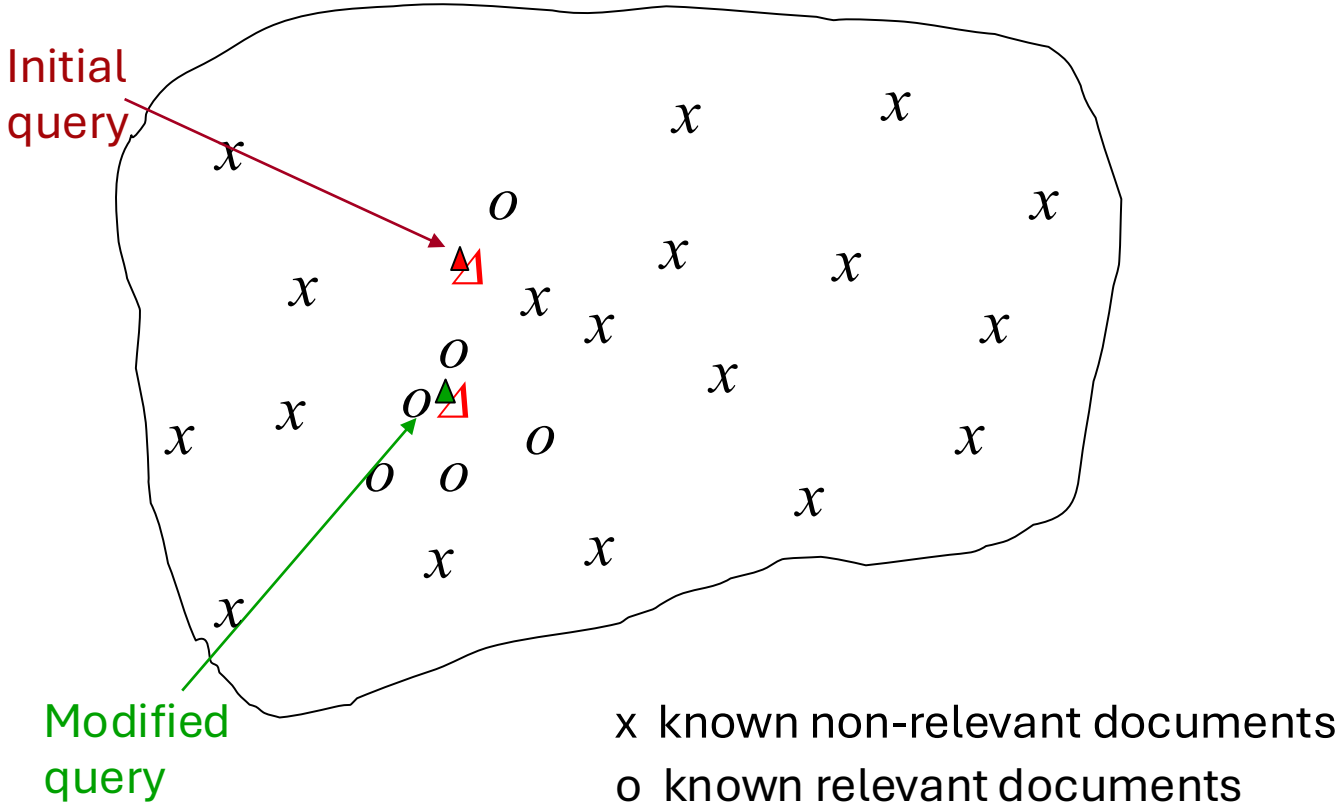




x non-relevant documents
o relevant documents

Relevance Feedback on Initial Query





x known non-relevant documents
o known relevant documents

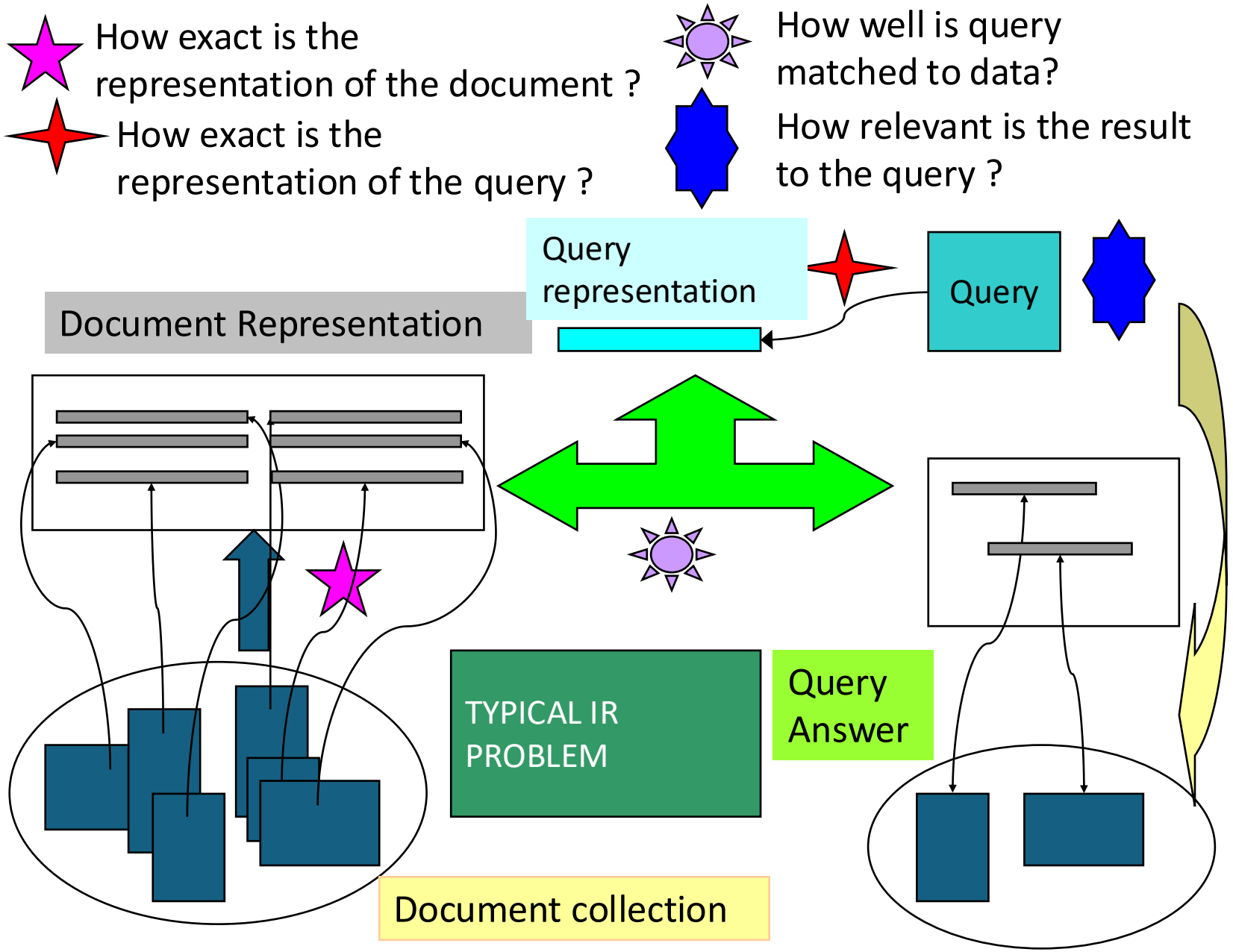
Modified Query: Rocchio Algorithm

$$\vec{q}_m = \alpha\vec{q}_0 + \beta\frac{1}{|D_r|}\sum_{\vec{d}_j \in D_r}\vec{d}_j - \gamma\frac{1}{|D_{nr}|}\sum_{\vec{d}_j \in D_{nr}}\vec{d}_j$$

q_m = modified query vector; q_0 = original query vector; α,β,γ: weights (hand-chosen or set empirically); D_r = set of known relevant doc vectors; D_{nr} = set of known irrelevant doc vectors

Wikipedia: Gerard Salton, The **SMART** (**S**ystem for the **M**echanical **A**nalysis and **R**etrieval of **T**ext or Salton's Magic Automatic Retriever of Text) Information Retrieval System is an information retrieval system, developed at Cornell University in the **1960s**. Many important concepts in information retrieval were developed as part of research on the SMART system, including the vector space model, relevance feedback, and Rocchio algorithm.

Salton, G. (Ed.). *The SMART retrieval system: Experiments in automatic document processing*. Englewood Cliffs, NJ: Prentice-Hall. 1971.


How exact is the representation of the document ?
How exact is the representation of the query ?
How well is query matched to data?
How relevant is the result to the query ?
Query representation
Query
Document Representation
TYPICAL IR PROBLEM
Query Answer
Document collection

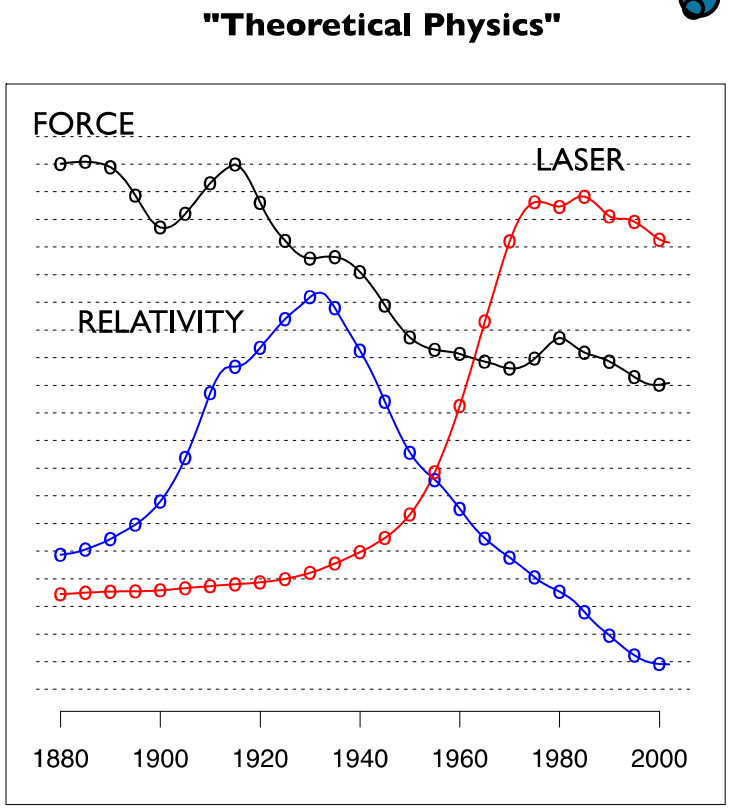
47

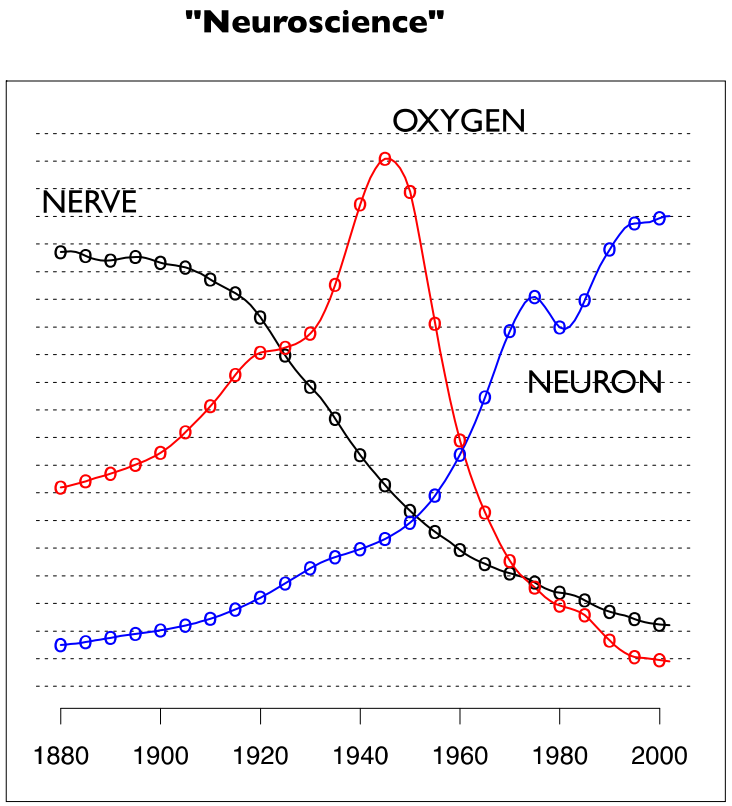
Topic Models

Statistical methods that analyze the words of texts to:

- Discover the themes that run through them (topics)
- How those themes are connected to each other
- How they change over time

Just for illustration purposes

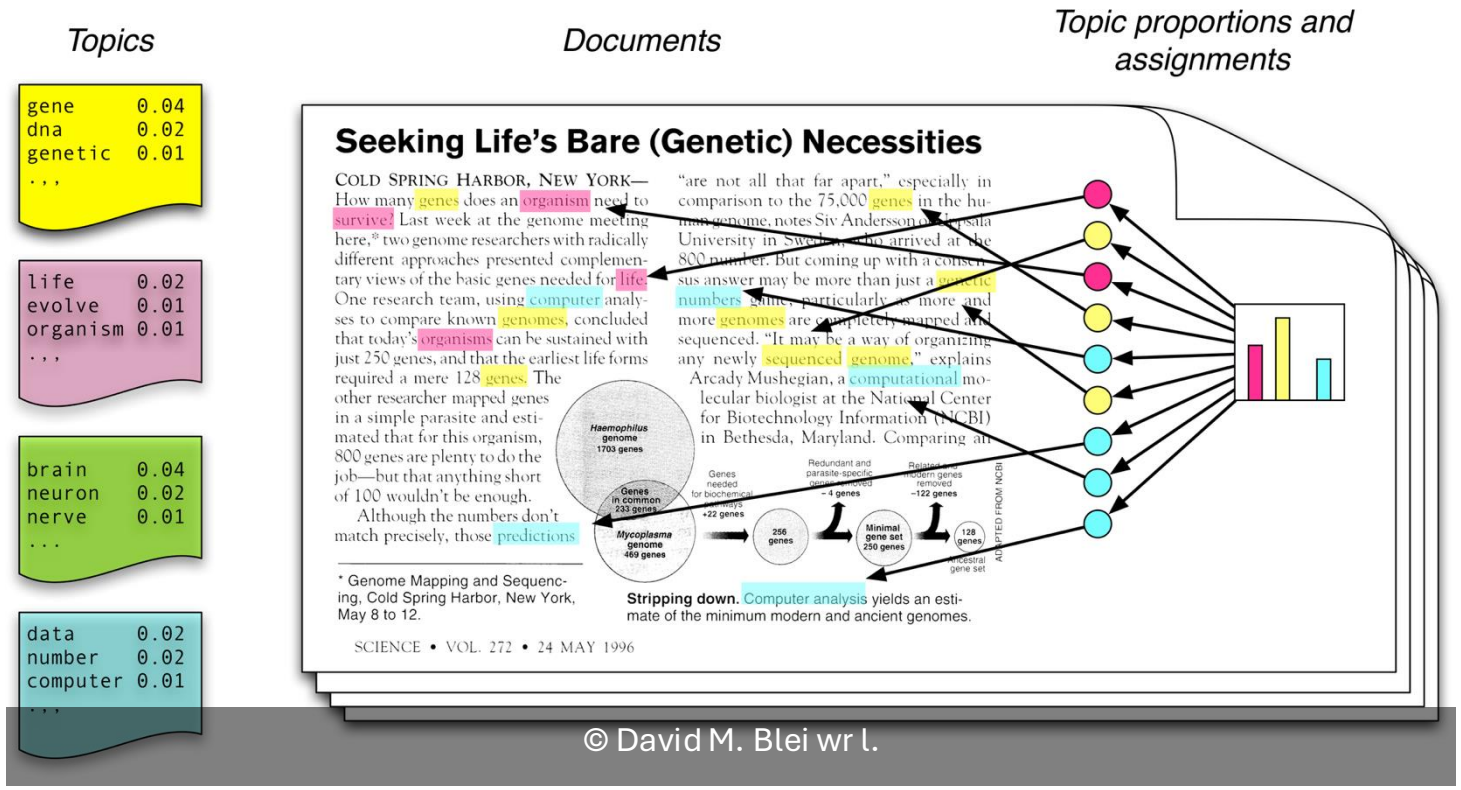




© David M. Blei wr l.

Topic Modeling Scenario

- 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

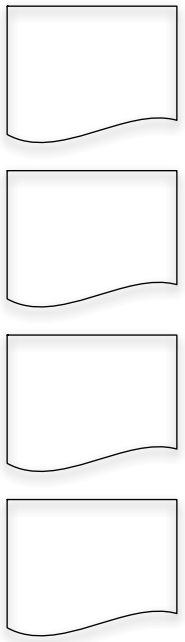


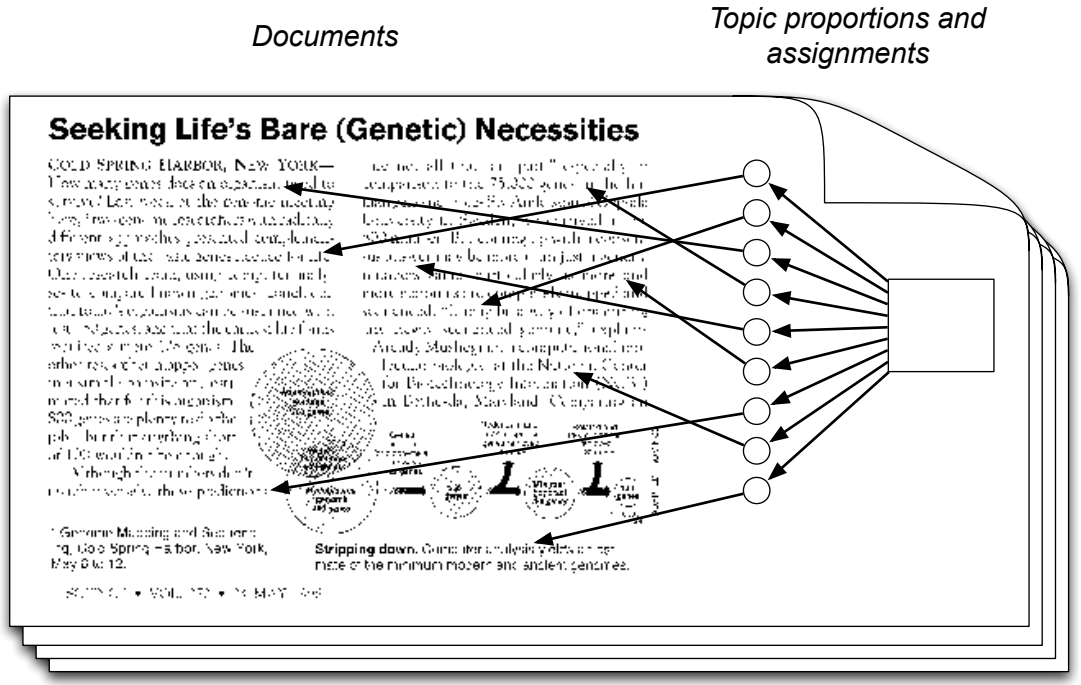
Topic Modeling: Latent Dirichlet Allocation (LDA, Blei at al.)

In reality, we only observe the documents

The other structures are hidden variables

Topic modeling algorithms infer these variables from data





© David M. Blei wr l.

Why/how does LDA work?

- Trade-off between two goals
  1. For each document, allocate its words to as few topics as pos
  2. For each topic, assign high probability to as few terms as pos
- These goals are at odds
  - Putting a document in a single topic makes #2 hard:
    All of its words must have non-negligible probability under that topic
  - Putting very few words in each topic makes #1 hard:
    To cover a document's words, it must assign many topics to it
- Trading off these goals finds groups of tightly co-occurring words

LDA Application: Reuters Data

- Setup
  - 100-topic LDA trained on a 16,000 documents corpus of news articles by Reuters
  - Some standard stop words removed
- Top-7 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

LDA Application: Reuters Data

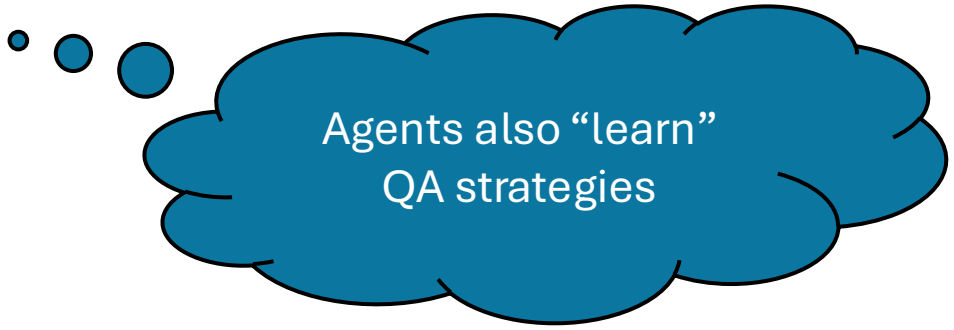
Result

Again: "Arts", "Budgets", "Children", "Education".

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.

Back to Agents

- Agents not only *use* models
- Agents *build* models that are appropriate to fulfil the agents' task descriptions …
  - … or maximize the utilities derived from preference structures and goals
- Agents need to *derive approximation algorithms* for query answering on the models they find appropriate



Approaches for Representing Word Semantics

Distributional Semantics (*Count*)

- Used since the 90's
- Sparse word-context PMI/PPMI matrix
- Decomposed with SVD

Word Embeddings (*Predict*)

- word2vec *(Mikolov et al., 2013)*
- GloVe *(Pennington et al., 2014)*

Underlying Theory: **The Distributional Hypothesis** *(Harris, '54; Firth, '57)*

"Similar words occur in similar contexts"

https://www.tensorflow.org/tutorials/word2vec

https://nlp.stanford.edu/projects/glove/

Point(wise) Mutual Information: PMI

Kenneth Ward Church and Patrick Hanks. "Word association norms, mutual information, and lexicography". Comput. Linguist. 16 (1): 22–29. **1990**.

- Measure of association used in information theory and statistics

$$\operatorname{pmi}(x;y) \equiv \log \frac{p(x,y)}{p(x)p(y)} = \log \frac{p(x|y)}{p(x)} = \log \frac{p(y|x)}{p(y)}$$

- Positive PMI: PPMI(x, y) = max(pmi(x, y), 0)
- Quantifies the discrepancy between the probability of their coincidence given their joint distribution and their individual distributions, assuming independence
- Finding collocations and associations between words
- Countings of occurrences and co-occurrences of words in a text corpus can be used to approximate the probabilities p(x) or p(y) and p(x,y) respectively

PMI – Example

- Counts of pairs of words getting the most and the least PMI scores in the first 50 millions of words in Wikipedia (dump of October 2015)
- Filtering by 1,000 or more co-occurrences.
- The frequency of each count can be obtained by dividing its value by 50,000,952.
  (Note: natural log is used to calculate the PMI values in this example, instead of log base 2)

word 1	word 2	count word 1	count word 2	count of co-occurrences	PMI
puerto	rico	1938	1311	1159	10.0349081703
hong	kong	2438	2694	2205	9.72831972408
los	angeles	3501	2808	2791	9.56067615065
carbon	dioxide	4265	1353	1032	9.09852946116
prize	laureate	5131	1676	1210	8.85870710982
san	francisco	5237	2477	1779	8.83305176711
nobel	prize	4098	5131	2498	8.68948811416
ice	hockey	5607	3002	1933	8.6555759741
star	trek	8264	1594	1489	8.63974676575
car	driver	5578	2749	1384	8.41470768304
it	the	283891	3293296	3347	-1.72037278119
are	of	234458	1761436	1019	-2.09254205335
this	the	199882	3293296	1211	-2.38612756961
is	of	565679	1761436	1562	-2.54614706831
and	of	1375396	1761436	2949	-2.79911817902
a	and	984442	1375396	1457	-2.92239510038
in	and	1187652	1375396	1537	-3.05660070757
to	and	1025659	1375396	1286	-3.08825363041
to	in	1025659	1187652	1066	-3.12911348956

[Wikipedia]

Approaches for Representing Word Semantics

Distributional Semantics (*Count*)

- Used since the 90's
- Sparse word-context PMI/PPMI matrix
- Decomposed with SVD

Word Embeddings (*Predict*)

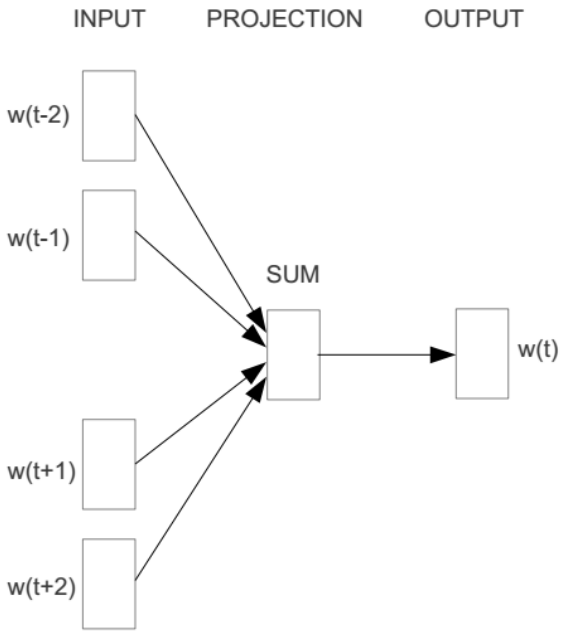
- word2vec *(Mikolov et al., 2013)*
- GloVe *(Pennington et al., 2014)*

Embedding Approaches

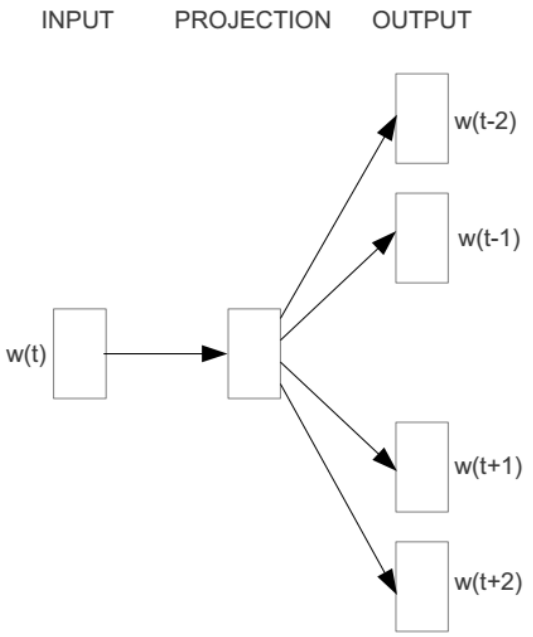
- Represent each word with a low-dimensional vector
- Word similarity = vector similarity
- Key idea: Predict surrounding words of every word
- Faster and can easily incorporate a new sentence/document or add a word to the vocabulary

Represent the meaning of a **word** – word2vec

- 2 basic models:
  - Continuous Bag of Word (CBOW): use a window to predict the middle word
  - Skip-gram (SG): use a word to predict the surrounding ones in window.



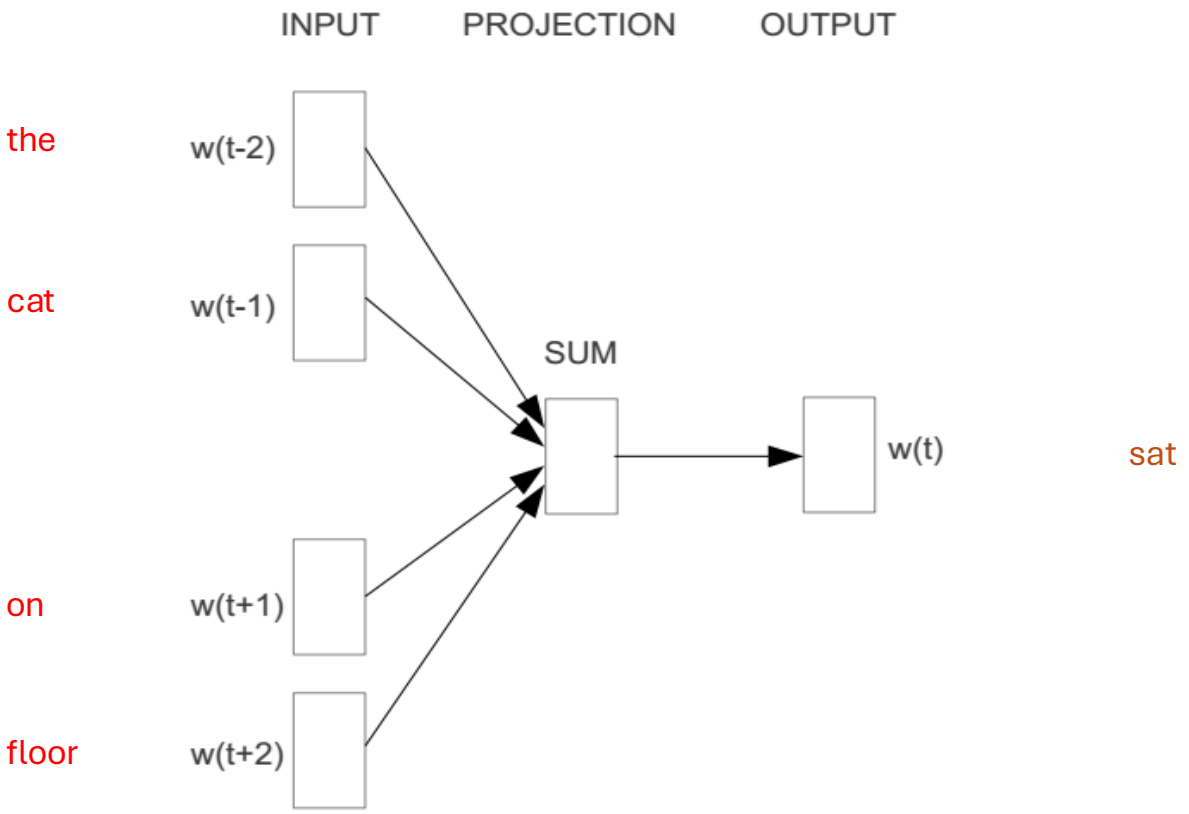
CBOW



Skip-gram

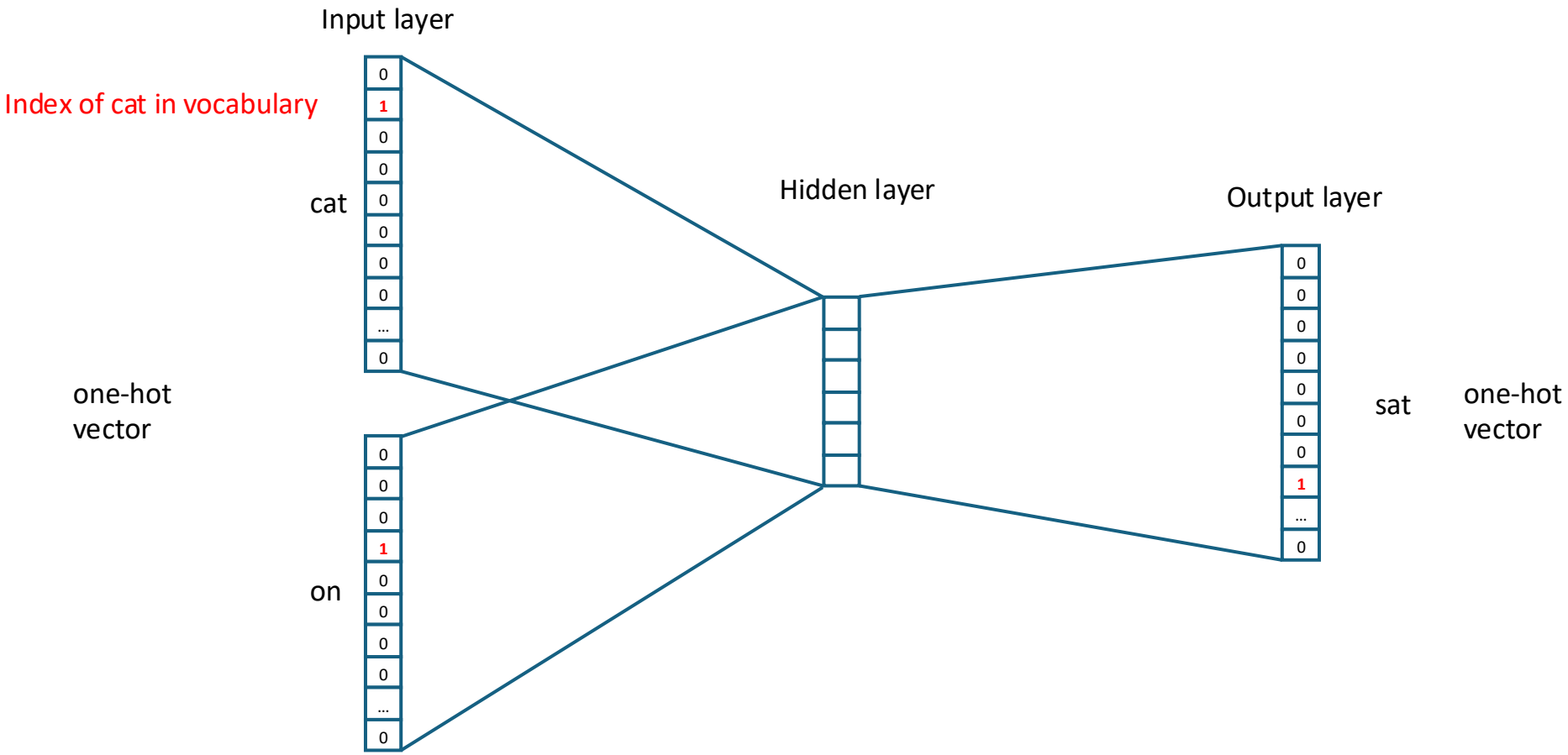
Word2vec – Continuous Bag of Words

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

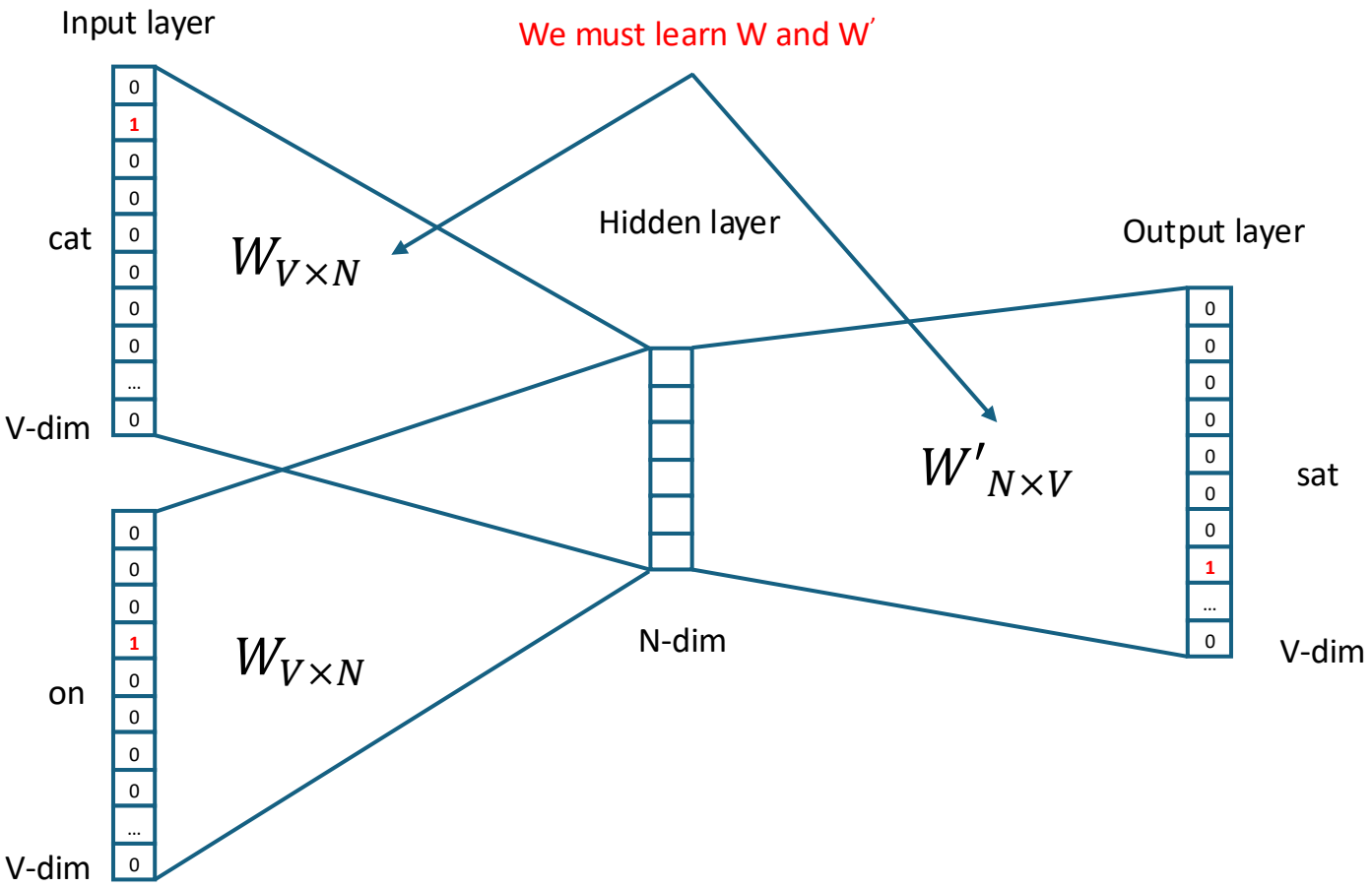


Distributed Representations of Words and Phrases and their Compositionality Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, Jeffrey Dean, NIPS **2013**

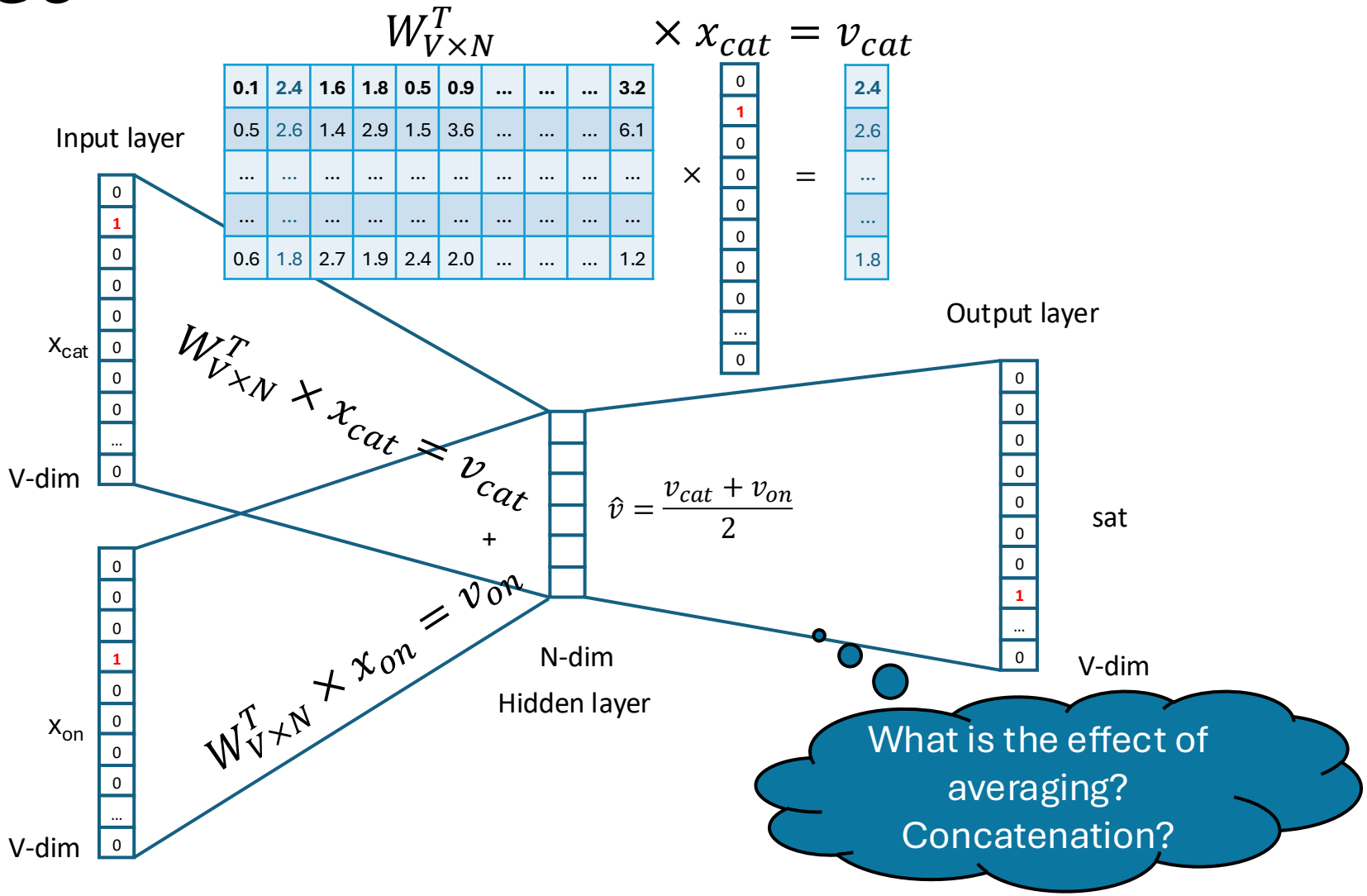
Word2vec – Continuous Bag of Words



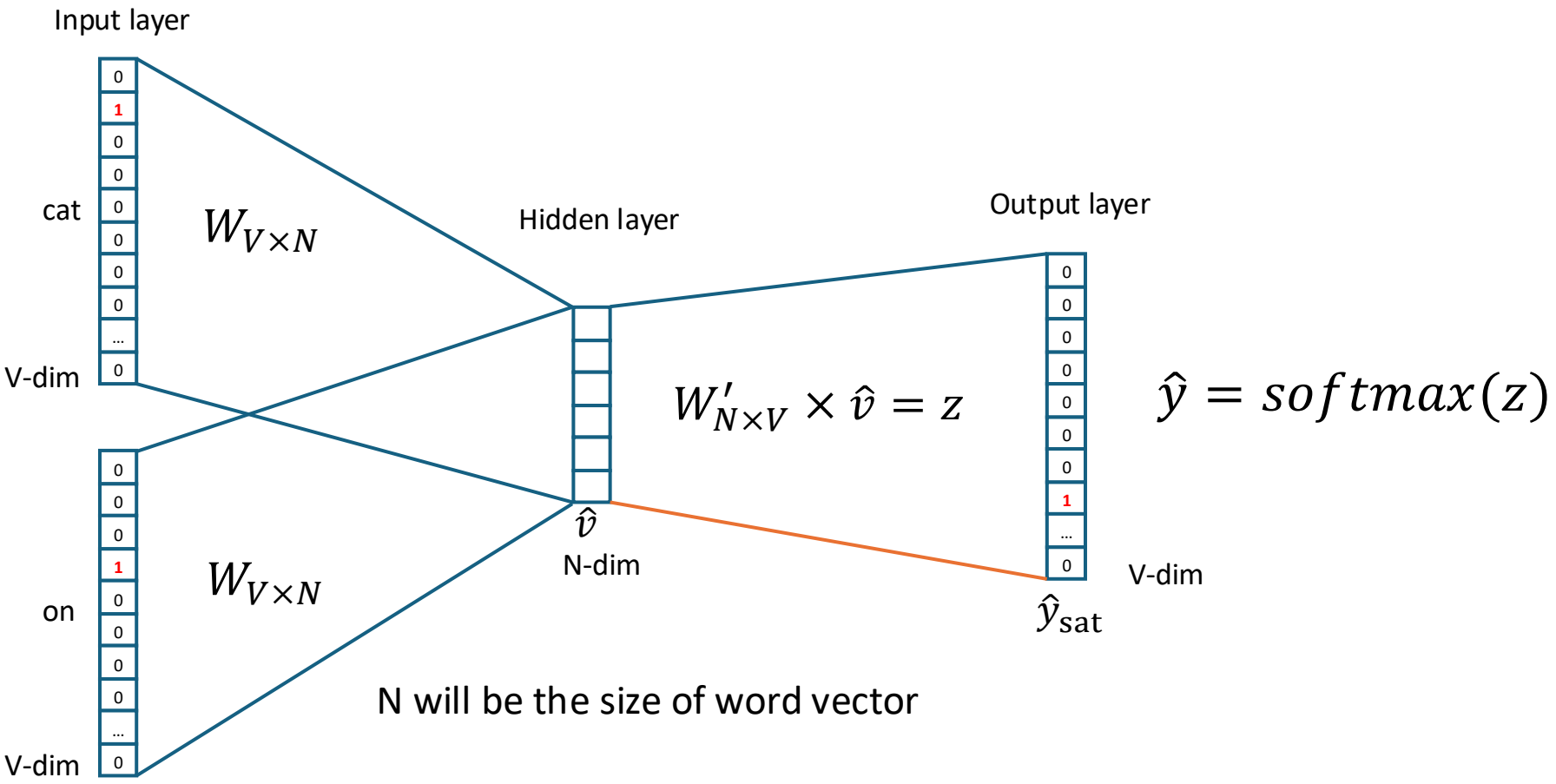
Word2vec – Continuous Bag of Words



Word2vec


$W^T_{V\times N}$
$\times x_{cat} = v_{cat}$
Input layer
Output layer
x_{cat}
V-dim
$W^T_{V\times N} \times x_{cat} = v_{cat}$
+
$W^T_{V\times N} \times x_{on} = v_{on}$
$\hat{v} = \frac{v_{cat} + v_{on}}{2}$
N-dim
Hidden layer
sat
x_{on}
V-dim
What is the effect of averaging? Concatenation?

Word2vec – Continuous Bag of Words



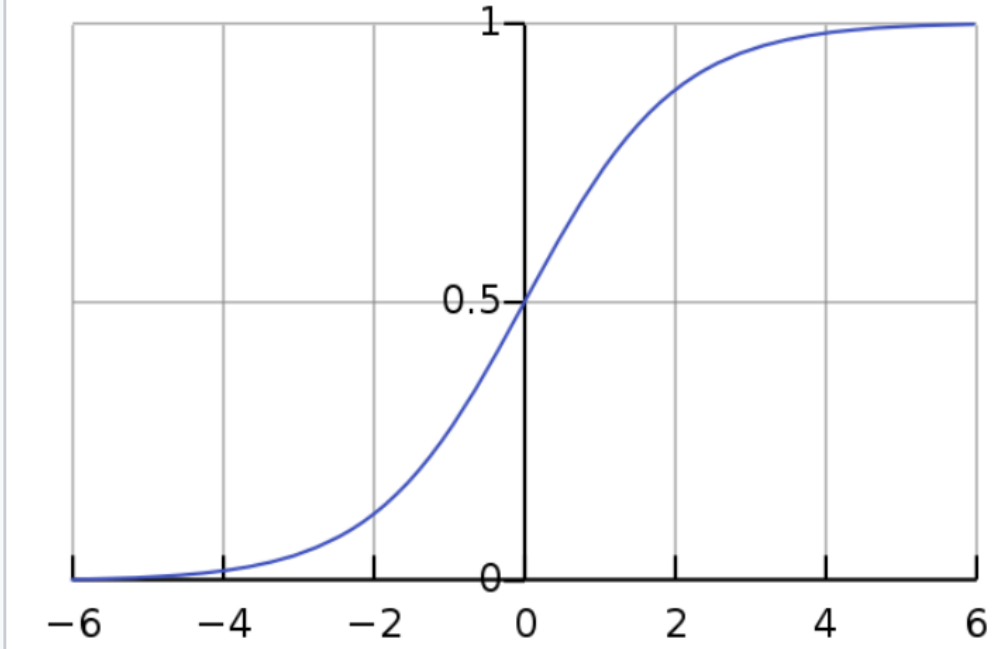
Logistic function

A **logistic function** or **logistic curve** is a common "S" shape (sigmoid curve), with equation:

$$f(x) = \frac{L}{1 + e^{-k(x-x_0)}}$$

where

- e = the natural logarithm base (also known as Euler's number),
- x_0 = the x-value of the sigmoid's midpoint,
- L = the curve's maximum value, and
- k = the steepness of the curve.[1]



Standard logistic sigmoid function i.e. $L = 1, k = 1, x_0 = 0$

[Wikipedia]

softmax(**z**)

In mathematics, the **softmax function**, or **normalized exponential function**,[1]:198 is a generalization of the logistic function that "squashes" a K-dimensional vector $\mathbf{z}$ of arbitrary real values to a K-dimensional vector $\sigma(\mathbf{z})$ of real values in the range [0, 1] that add up to 1. The function is given by

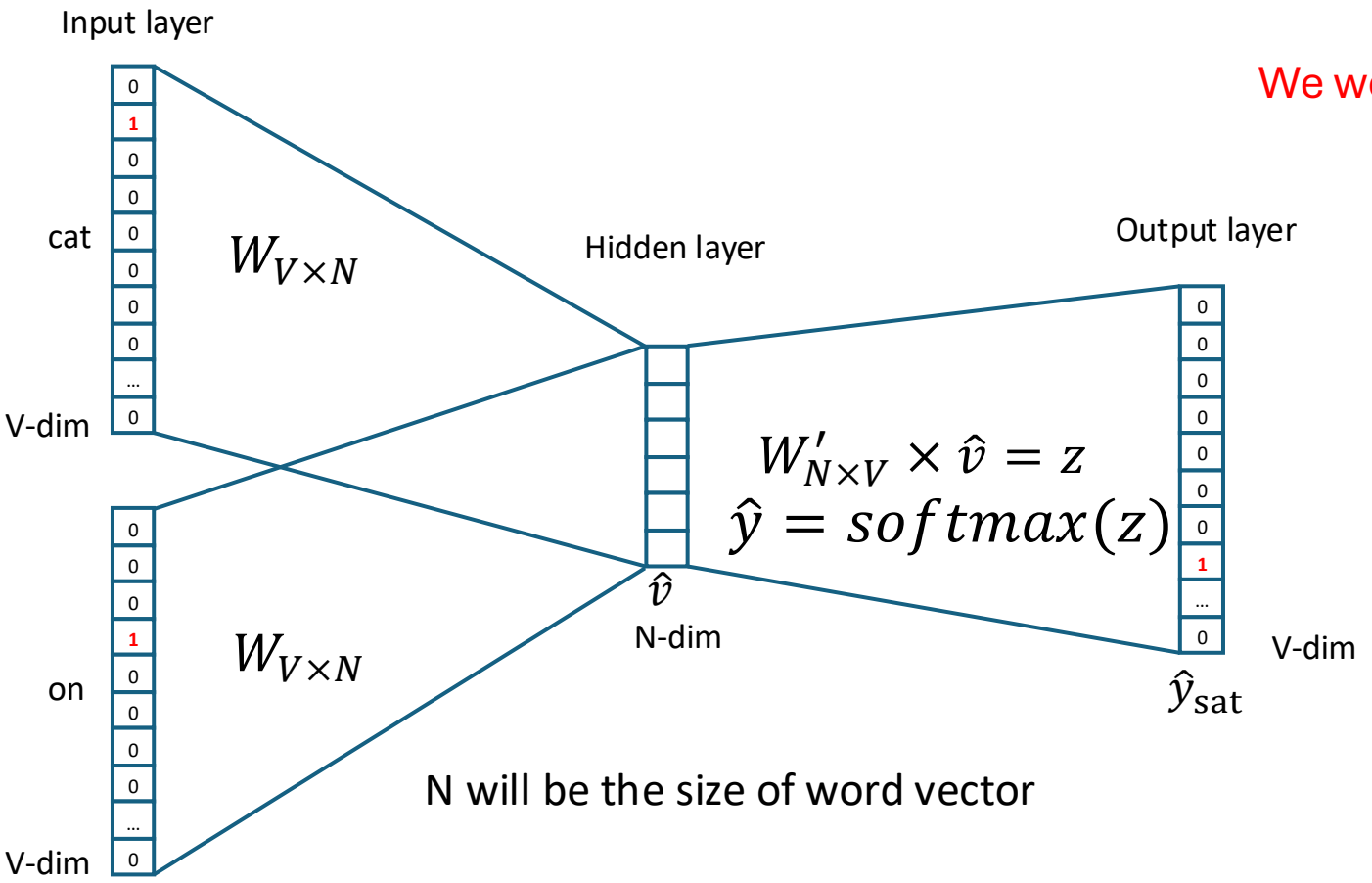
$$\sigma : \mathbb{R}^K \to [0,1]^K$$

$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \quad \text{for } j = 1, \ldots, K.$$

In probability theory, the output of the softmax function can be used to represent a categorical distribution – that is, a probability distribution over K different possible outcomes. In fact, it is the gradient-log-normalizer of the categorical probability distribution.[*further explanation needed*]

[Wikipedia]

Word2vec – Continuous Bag of Words

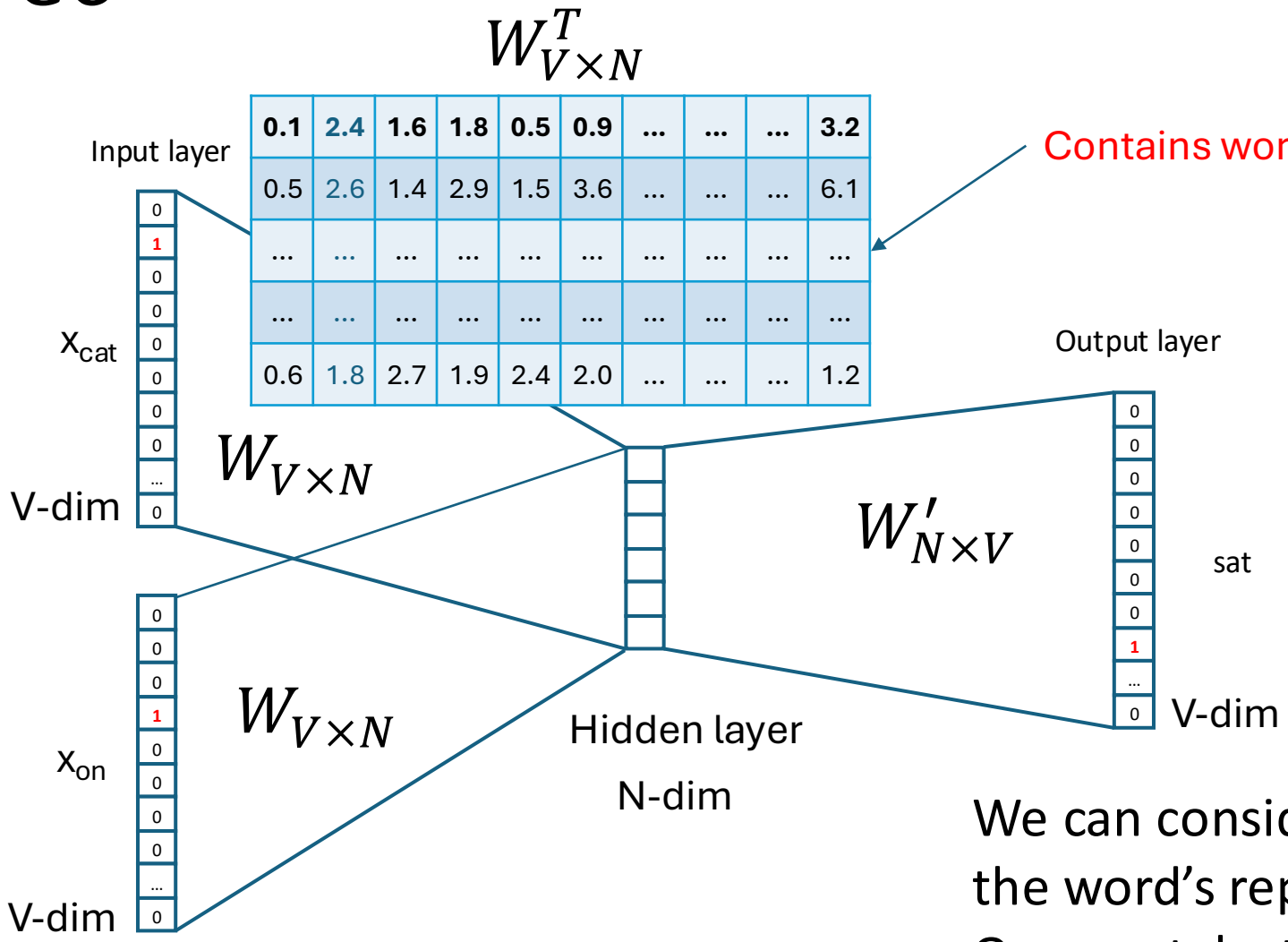


We would prefer $\hat{y}$ close to $\hat{y}_{sat}$

0.01
0.02
0.00
0.02
0.01
0.02
0.01
0.7
...
0.00

$\hat{y}$

Word2vec



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

Intrinsic Evaluation

Word Analogies

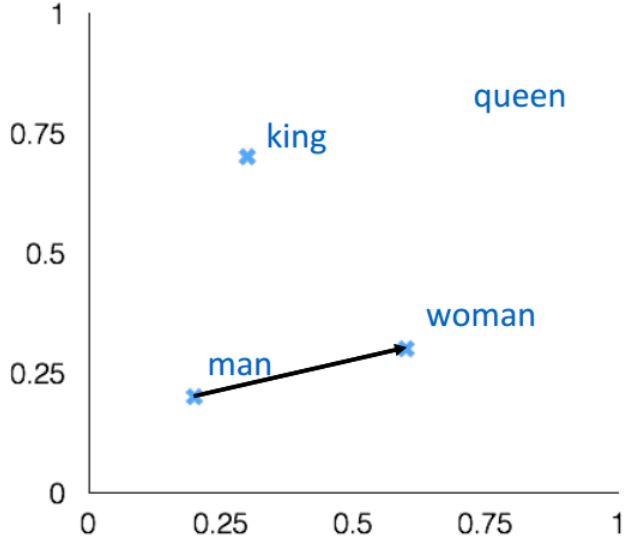
Test for linear relationships, examined by Mikolov et al. (2014)

a:b :: c:?

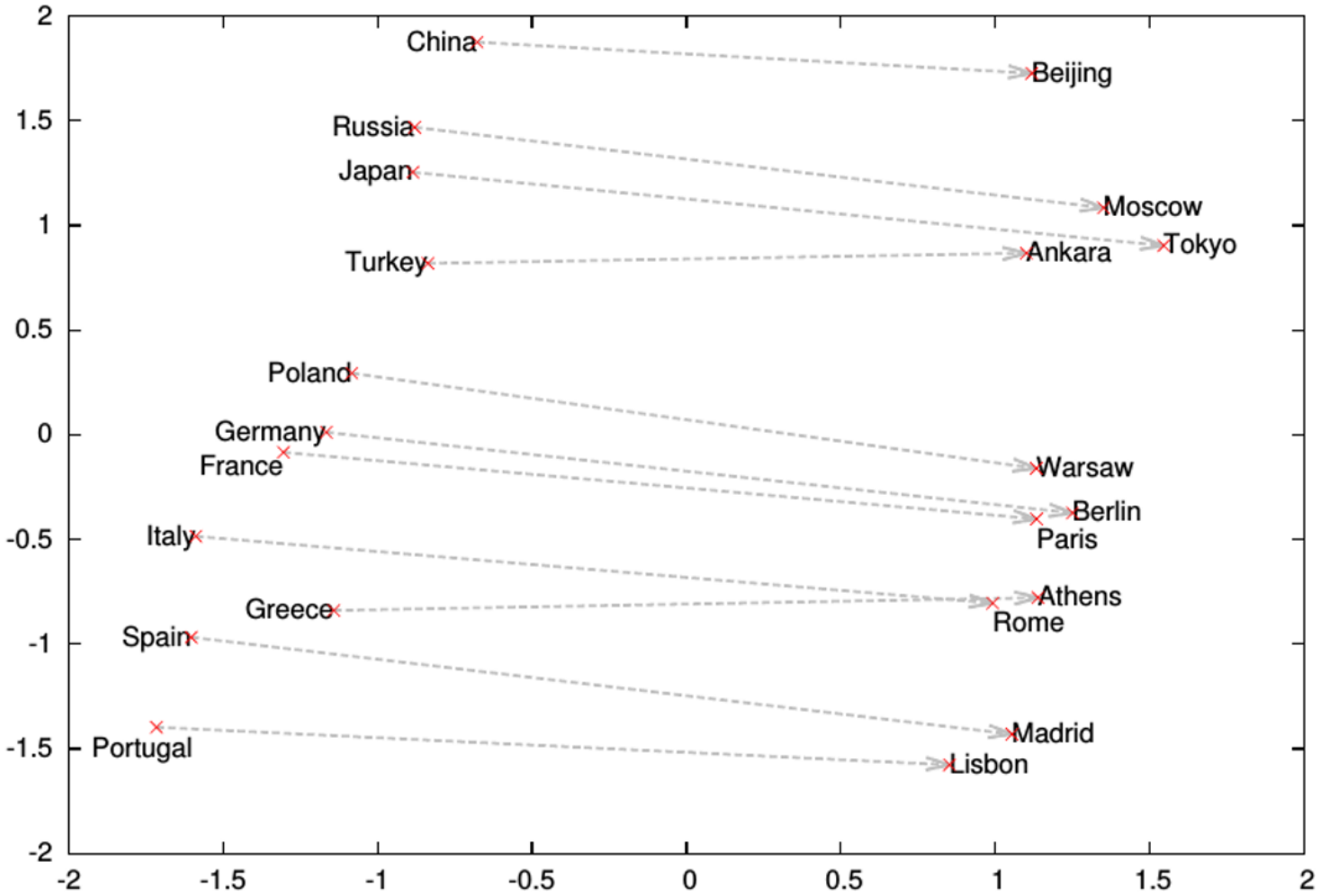
$$d = \arg\max_x \frac{(w_b - w_a + w_c)^T w_x}{||w_b - w_a + w_c||}$$

man:woman :: king:?

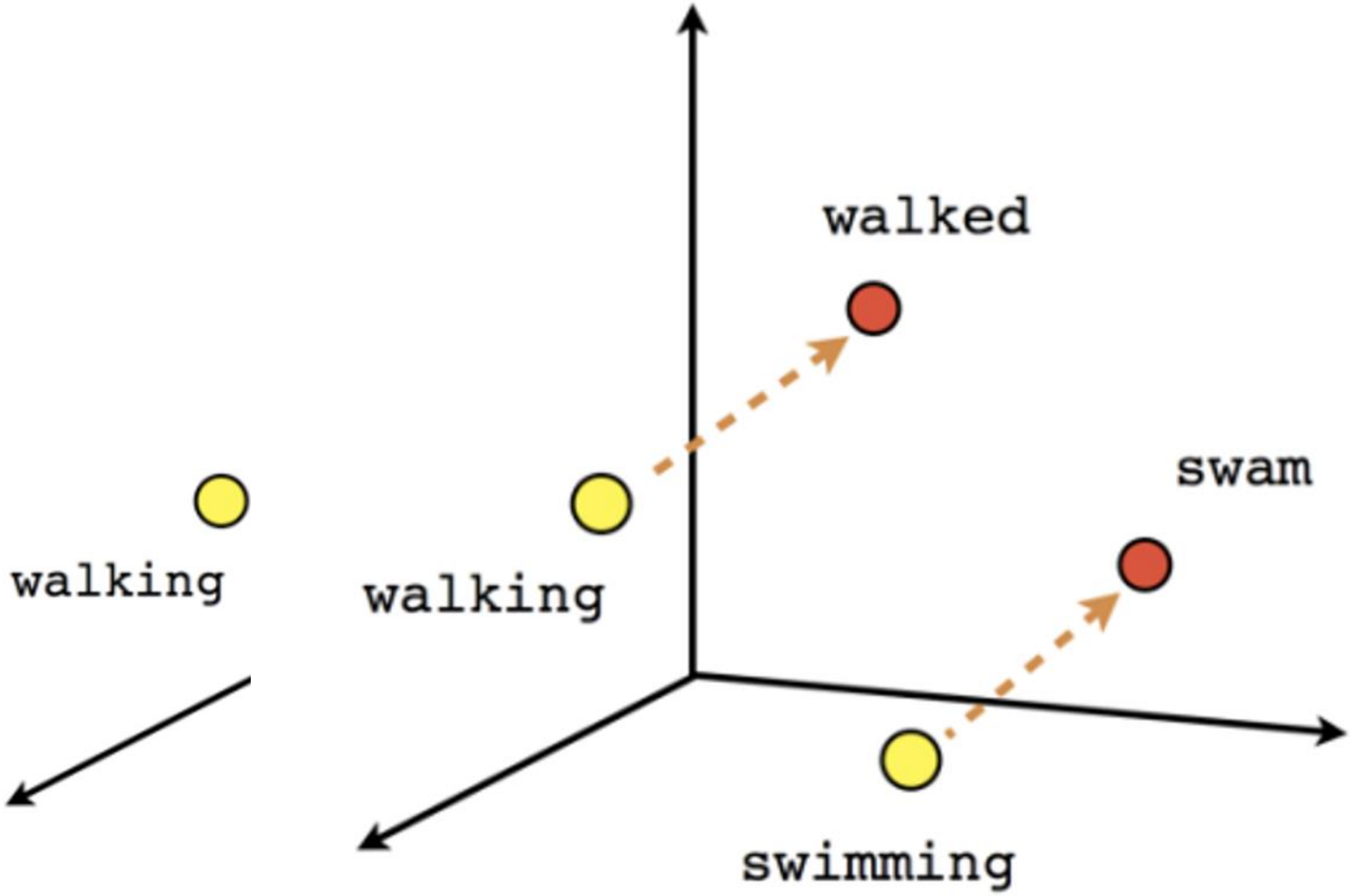
+	king	[0.30 0.70]
-	man	[0.20 0.20]
+	woman	[0.60 0.30]
	queen	[0.70 0.80]



Word analogies



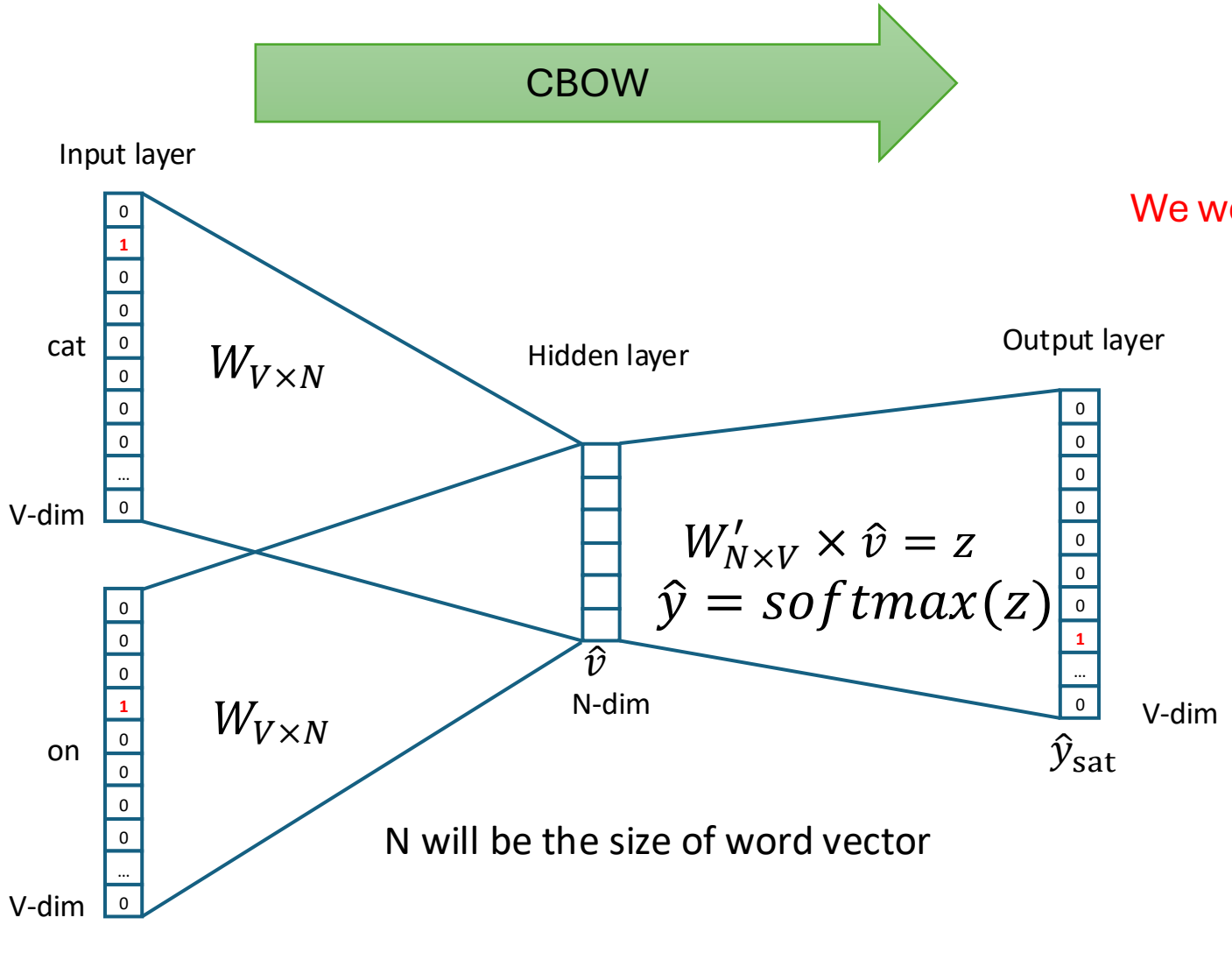
Word Analogies (Tense)



Extrinsic Evaluation

- Evaluate in applications
  - Sentiment analysis
  - …

CBOW



We would prefer $\hat{y}$ close to $\hat{y}_{sat}$

0.01
0.02
0.00
0.02
0.01
0.02
0.01
0.7
...
0.00

$\hat{y}$

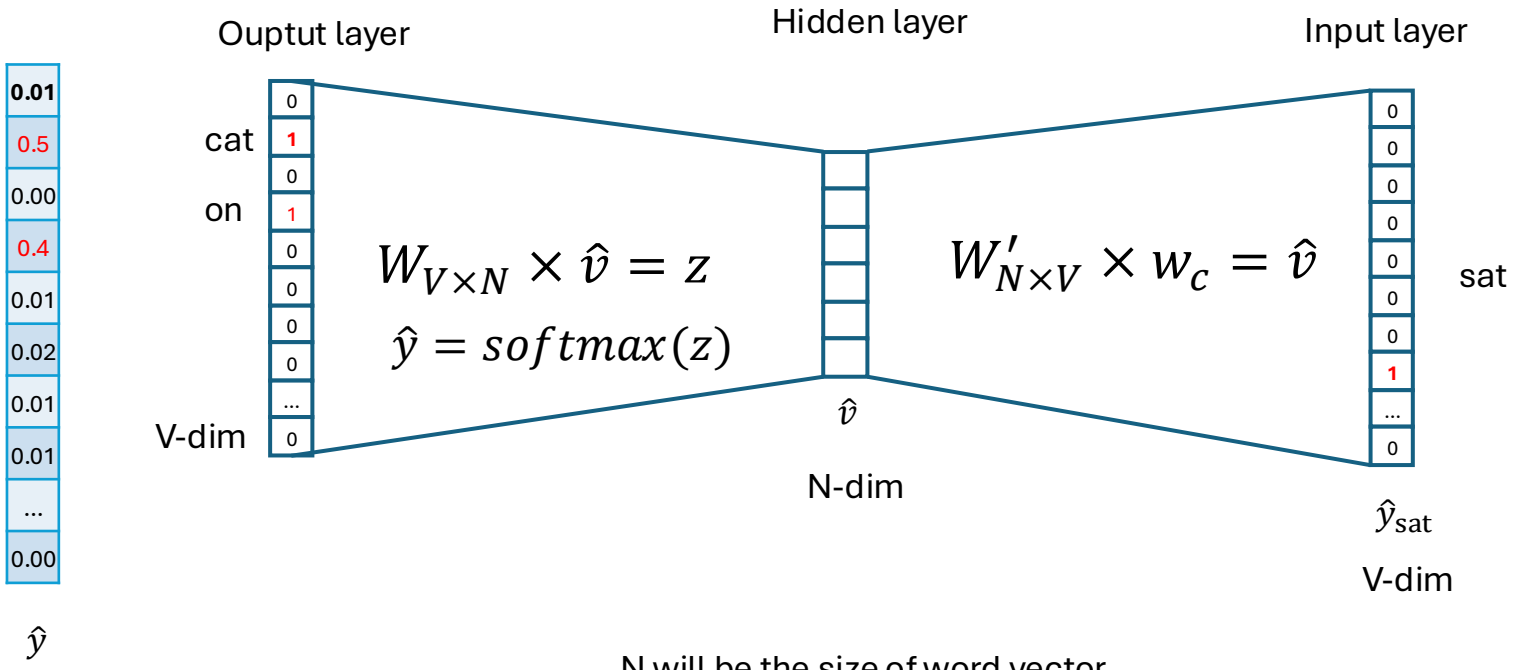
Word2vec – Continuous Bag of Words

- Used to predict a target word given its surrounding context words
- It has three key layers:
  - Input layer: This layer takes the surrounding context words, represented as one-hot vectors, as input
  - Hidden layer: This layer learns the word embeddings by transforming the one-hot input vectors into lower-dimensional vectors (the embeddings)
  - Output layer: This layer predicts the target word (the word being predicted) by applying SoftMax, which gives a probability distribution over all possible words in the vocabulary

https://medium.com/@RobuRishabh/learning-word-embeddings-with-cbow-and-skip-gram-b834bde18de4

Skip-Gram

We would prefer $\hat{y}$ close to z



N will be the size of word vector

Skip-Gram (reverse of CBOW)

- Used to predict the context words given the target word
- It tries to maximize the probability of predicting the surrounding words for a given target word
- It has three key layers:
  - Input Layer: The target word is represented as a one-hot vector
  - Hidden Layer: This learns the word embeddings (dense vectors)
  - Output Layer: The model predicts the surrounding context words for the given target word

https://medium.com/@RobuRishabh/learning-word-embeddings-with-cbow-and-skip-gram-b834bde18de4

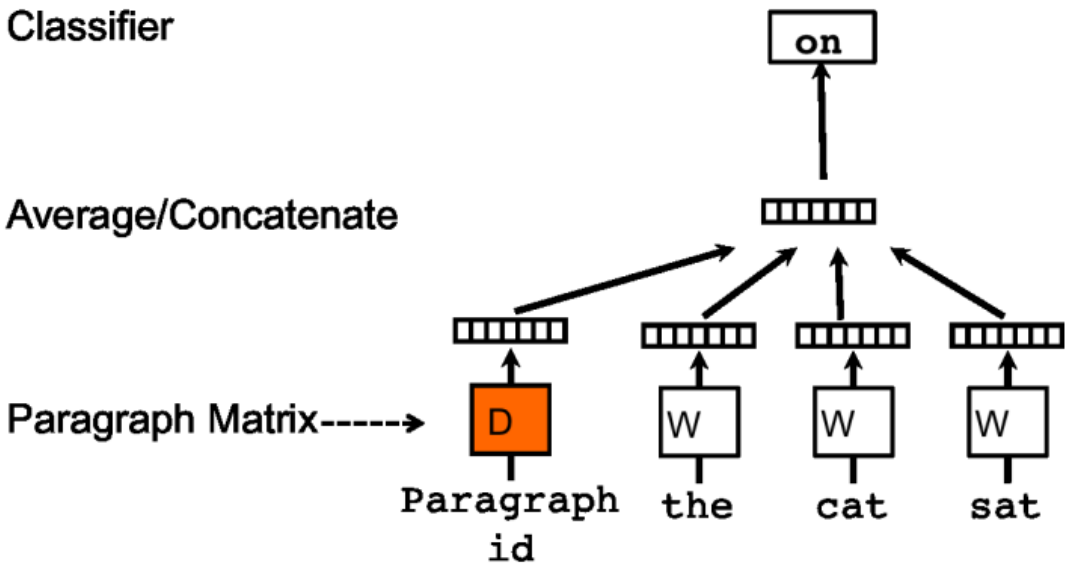
What is `word2vec`?

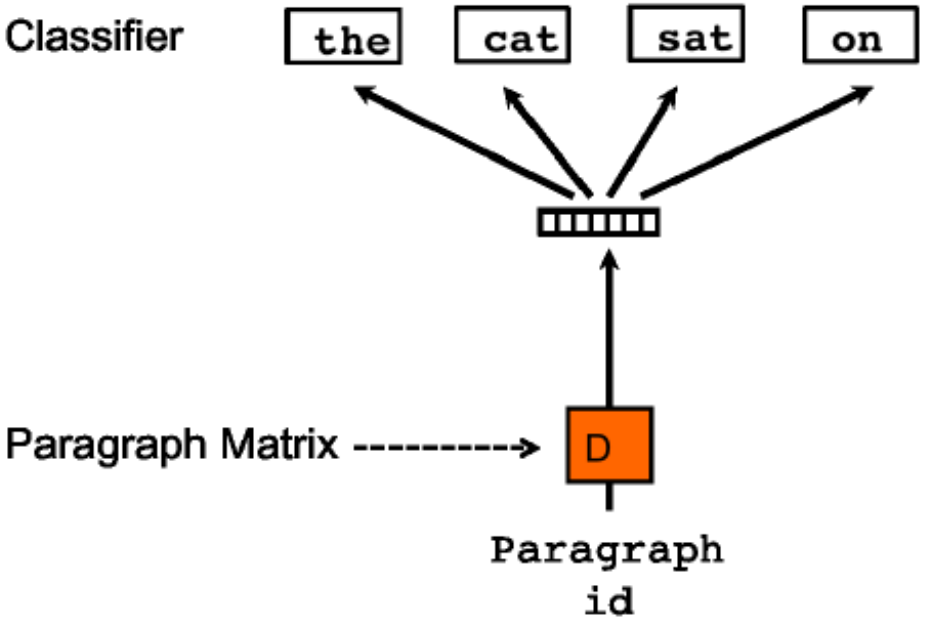
- `word2vec` is **not** a single algorithm
- It is a **software package** for representing words as vectors, containing:
  - Two distinct models
    - CBoW
    - Skip-Gram
  - Various training methods
    - Softmax is a bottleneck
  - A rich preprocessing pipeline
    - Dynamic Context Windows
    - Subsampling of Frequent Words
    - Deleting Rare Words (left out)

Represent the meaning of sentence/paragraph/doc

- Paragraph Vector (Le and Mikolov, 2014)
  - Extend word2vec to text level
  - Also two models: add paragraph vector as the input

Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In Proceedings ICML'14. **2014**.





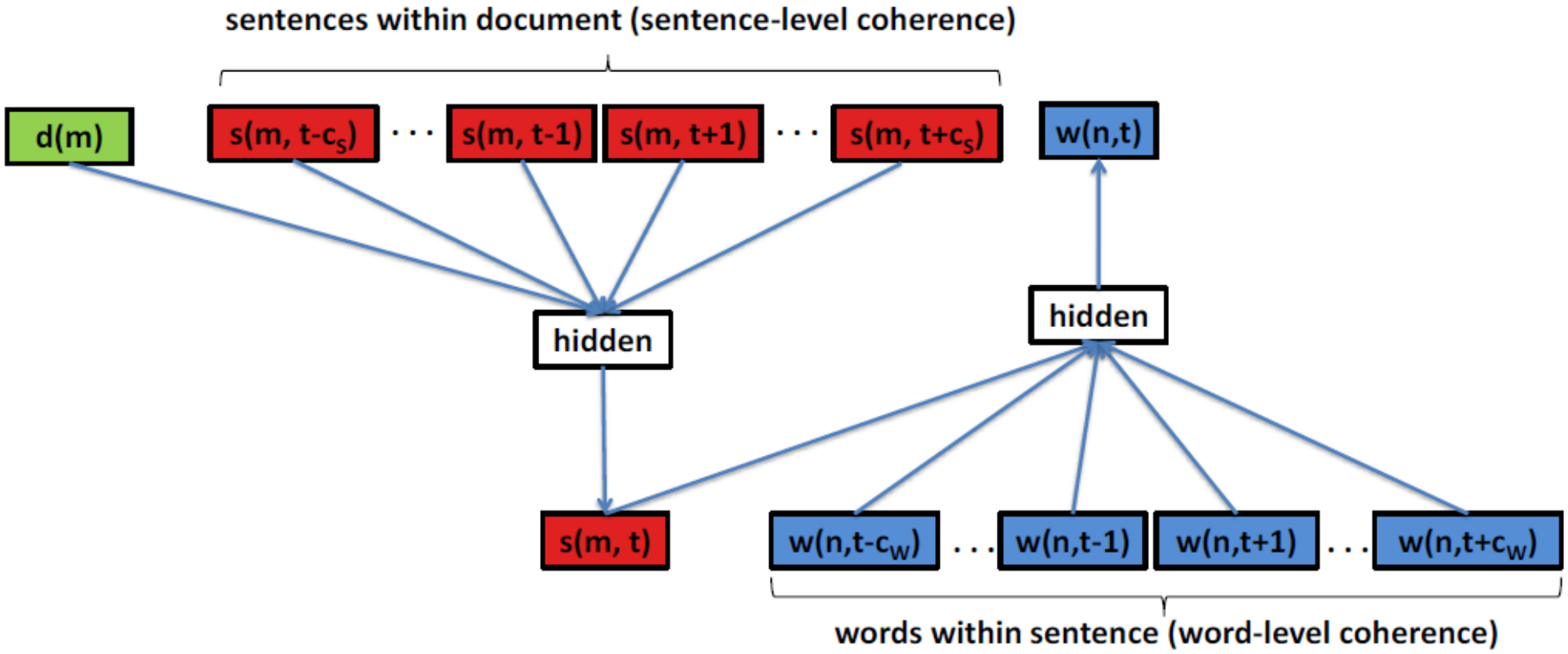
Paragraph Vector

- Learn document embedding by predicting the next word in the document using the context of the word and the ('unknown') document vector as features.
- Resulting vector captures the topic of the document.
- Update the document vectors, but not the word vectors [Le et al.]
- Update the document vectors, along with the word vectors [Dai et al.]
  - Improvement in the accuracy for document similarity tasks.

Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In Proceedings ICML'14. **2014**.

Dai, A.M., Olah, C., Le, Q.V., Corrado, G.S.: Document embedding with paragraph vectors. In: NIPS Deep Learning Workshop. **2014**

Architecture Diagram



Doc2Sent2Vec: A Novel Two-Phase Approach for Learning Document Representation, Ganesh J, Manish Gupta, Vasudeva Varma